HAL
open science

Optimization and Control of Large Systems: Fighting the Curse of Dimensionality

Nicolas Gast

To cite this version:

Nicolas Gast. Optimization and Control of Large Systems: Fighting the Curse of Dimensionality. Networking and Internet Architecture [cs.NI]. Université Grenoble Alpes, 2010. English. NNT : . tel-01875211

HAL Id: tel-01875211

https://theses.hal.science/tel-01875211v1

Submitted on 17 Sep 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Thèse

Optimisation et contrôle de systèmes à grande échelle: comment combattre l'optimisation combinatoire

English title: **Optimization and Control of Large Systems: Fighting the Curse of Dimensionality**

pour obtenir

le grade de DOCTEUR DE L'UNIVERSITÉ DE GRENOBLE
Mention INFORMATIQUE

par

Nicolas GAST

soutenue le mercredi 29 septembre 2010.

Composition du jury :

Bernard	Ycart	Président
Alain	Jean-Marie	Rapporteur
David	McDonald	Rapporteur
Francois	Baccelli	Examinateur
Jean-Yves	Le Boudec	Examinateur
Bruno	Gaujal	Directeur de thèse

Remerciements

Finir une thèse peut donner un sentiment d'accomplissement personnel mais il ne faut pas oublier que ce travail n'aurait pu être mené au bout sans la contribution de nombreuses personnes. J'espère que ces modestes paragraphes sauront leur rendre hommage.

La première personne que je tiens à remercier est mon directeur de thèse, Bruno Gaujal, pour toute l'aide et le soutien qu'il a pu m'apporter durant ces trois années. J'ai eu la chance de pouvoir travailler avec lui, tant pour ses qualités scientifiques qu'humaines. Je n'hésiterai pas une seconde si c'était à recommencer.

Je tiens aussi à remercier tous les membres de mon Jury. En premier lieu, je suis grandement reconnaissant à David McDonald et Alain Jean-Marie pour avoir accepté la lourde tâche de relire et de rapporter sur mon manuscrit. Merci à Bernard Ycart d'avoir accepter de présider le jury. Enfin, je remercie François Baccelli et Jean-Yves Le Boudec d'avoir accepté de participer à ce jury. Je ne serais sûrement pas arrivé là si François ne m'avait fait découvrir ce domaine de recherche et m'avait ensuite mis en contact avec Bruno. C'est pour moi un grand honneur de voir toutes ces grandes personnalités scientifiques participer à mon jury aujourd'hui.

Cette thèse s'est déroulée dans l'antenne de Montbonnot du laboratoire d'informatique de Grenoble. Outre la force scientifique et la bonne ambiance générale qui règne dans le bâtiment, deux éléments contribuent particulièrement à y rendre le travail agréable.

En premier lieu vient la cafet et ses discussions sans fin. Il serait trop long de citer tous les contributeurs mais je pense particulièrement aux débats animés par Bruno, Jean-Louis, Jean-Marc, Marc, Olivier ou encore Pierre C, chacun contribuant à la hauteur de sa bonne foi.

L'autre point important du laboratoire est le groupe de travail post-repas sur la théorie des jeux dans l'ensemble à 32 éléments. Il n'est pas facile de réussir à faire travailler ensemble des personnalités si différentes, entre un PF qui n'aime pas perdre et un Pierre N qui cherche le contraire, un Jean-Noël, optimiste mais aux intuitions trompeuses et des Brice, Eric, Fred ou Swann plus conservateurs ou encore un Philippe et sa philippette qui reste incompris de la majorité d'entre nous.

Je tiens aussi à remercier mes amis, dont la présence compte beaucoup pour moi. Merci en particulier à tous les eybenstoriens que j'ai la chance de connaître depuis de nombreuses années, à Mathieu D, Fabien, Mika, Clément, Thomas, Morgan et tous les autres partenaires de montagne ou d'escalade et à tous les autres que je n'ai pas cité ici.

Enfin, je tiens à remercier tout particulièrement ma famille. En premier lieu Mathilde et Elliot, pour leur présence quotidienne et les nombreux moments de joie qu'ils m'apportent. Je tiens aussi à remercier mes parents, soeur, cousins et autres grands-parents qui ont toujours été très patients et attentifs lorsque je tentais de leur expliquer sur quoi je travaillais. À la fin de la soutenance, mon grand-père m'a dit : « J'ai eu l'impression d'être dans une exposition d'art abstrait. C'était beau, ça avait l'air profond mais je n'ai rien compris ». J'espère que la lecture du manuscrit saura l'éclairer.

Contents

Résumé de la Thèse en Français ix
1. Chapitres 3 et 4 : Analyse du vol de travail . . . xi
2. Partie II : Modèles champ moyen optimaux . . . xv
3. Partie III : autres contributions . . . xxi
4. Conclusion . . . xxiv

Introduction xxvii
Organization of the Document . . . xxxi
Contributions . . . xxxiii
List of Notations . . . xxxvii

I. Foundations and First Examples 1

1. Markov Chains and Markov Decision Processes 3
1.1. Markov Processes . . . 4
1.2. Markov Decision Processes: Basic Concepts . . . 7
1.3. Optimal policies . . . 9
1.4. Algorithmic Issues: the Curse of Dimensionality . . . 15
1.5. Bibliographical notes . . . 16

2. A Survey on Mean Field Convergence 17
2.1. Mean field models . . . 18
2.2. Some Results on Point-wise Convergence . . . 21
2.3. Path-space convergence . . . 27
2.4. Concluding Remark . . . 31

3. Transient Behavior of Work Stealing: Makespan analysis 33
3.1. Introduction . . . 34
3.2. Work Stealing Model . . . 35
3.3. Principle of the Analysis and Main Theorem . . . 36
3.4. Unit Independent Tasks . . . 39
3.5. Tasks with Precedences . . . 42
3.6. Cooperation Among Thieves . . . 43
3.7. Experimental Study . . . 44
3.8. Appendix . . . 45
3.9. Bibliographical Notes . . . 48

4. Asymptotic Behavior of Work Stealing in Large-Scale Systems 51
4.1. Introduction . . . 52
4.2. Work Stealing in Grids . . . 53
4.3. Mean Field Approximation . . . 56

4.4. One Cluster Model . . . 58
4.5. Heterogeneous Clusters . . . 68
4.6. Conclusion and Future Work . . . 73

II. Optimal Mean Field **75**

5. Optimization in Discrete Time **79**
5.1. Introduction . . . 80
5.2. Notations and definitions . . . 81
5.3. Finite time convergence and optimal policy . . . 83
5.4. Application to a brokering problem . . . 91
5.5. Extensions and Counter-Examples . . . 95
5.6. Computational issues . . . 101
5.7. Conclusion and future work . . . 102
5.8. Appendix: proofs . . . 102

6. From Discrete to Continuous Optimization **115**
6.1. Introduction . . . 116
6.2. Notations and Definitions . . . 117
6.3. Mean Field Convergence . . . 120
6.4. Applications . . . 124
6.5. Appendix: proofs . . . 130

7. Non-smooth Mean-Field Models **137**
7.1. Introduction . . . 138
7.2. Description of the Model and Notations . . . 139
7.3. Convergence results . . . 141
7.4. Extension to Non-smooth Density Dependent Population Processes . . . 143
7.5. Examples . . . 145
7.6. Proofs of Theorem 7.5 and Theorem 7.7 . . . 152

III. Other Contributions **161**

8. Infinite Labeled Trees: from Rational to Sturmian Trees **163**
8.1. Introduction . . . 164
8.2. Infinite Trees . . . 165
8.3. Rational Trees . . . 170
8.4. Balanced and Mechanical Trees . . . 174
8.5. Algorithmic issues . . . 184
8.6. Glossary . . . 189

9. Distributed Delay-Power Control Algorithms in Wireless Networks **195**
9.1. Introduction . . . 196
9.2. Transmission Model . . . 197
9.3. Delay-Power Control (DPC) . . . 198
9.4. Delay-Power Control (DPC) Analysis . . . 201
9.5. DPC-Based Protocol Design . . . 210

9.6. Conclusions and Further Research 217
9.7. Appendix 217

Conclusion and Bibliography **225**

Conclusion **225**

Bibliography **229**

Abstracts **239**

Résumé de la Thèse en Français

Le but de cette thèse est de développer de nouvelles techniques pour l'analyse et le contrôle de systèmes à grande échelle, en vue d'analyser et d'optimiser les performances d'un système. Plus particulièrement, nous nous intéressons à l'étude de modèles stochastiques de grande dimension.

Les premiers modèles stochastiques de performance ont été introduits au début du XXième siècle par Erlang [60] qui étudiait le trafic téléphonique. Grâce à de nombreuses campagnes de mesures, il proposa des modèles stochastiques simples mais très proches de la réalité. Cela donna notamment naissance aux formules d'Erlang [61] qui ont été très largement utilisés aux fils des ans pour le dimensionnement de nombreux systèmes informatiques. Avec la généralisation des réseaux informatiques, les systèmes informatiques deviennent de plus en plus complexes et notre volonté de réussir à analyser les performances de tels systèmes n'en est qu'accrue. Une des principale manière d'analyser un système est de construire un modèle et d'analyser les performances de ce modèle.

Les *chaînes de Markov* sont le principal outil mathématique utilisé pour analyser les performances d'un système. Le système à modéliser est représenté sous forme d'un automate probabiliste. À un instant t donné, le système est dans un certain état $X(t)$. L'état du système peut évoluer au cours du temps en fonction de son état mais aussi d'événements extérieurs. La séquence d'événements extérieurs est modélisée par un processus aléatoire dépendant de l'état du système. Prenons l'exemple simple d'un serveur web qui reçoit des demandes extérieures de clients. À chaque instant, le serveur ne peut servir qu'un client à la fois et les autres clients sont mis en attente. L'état du serveur $X(t)$ est le nombre de clients en attente à l'instant t et évolue en fonction des arrivées ou des départs des clients. Par exemple si un client a été servi ou a quitté le système, $X(t)$ devient $X(t) - 1$. Si on considère que les clients sont traités en moyenne deux fois plus vite qu'ils n'arrivent, cela nous mène à la Figure 1 : si il y a $x \geq 1$ clients à l'instant t, à l'instant suivant, il y aura $x + 1$ clients avec probabilité $2/3$ et $x - 1$ avec probabilité $1/3$.

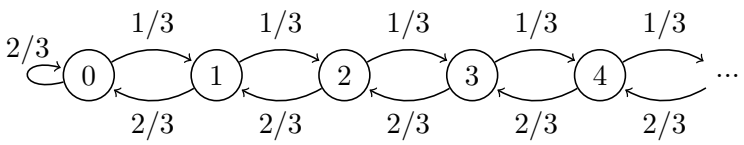


FIGURE 1. : Un modèle de file d'attente.

Les intérêts des chaînes de Markov sont nombreux. Leur pouvoir d'expression est très grand et elles permettent de modéliser de nombreux systèmes de manière assez simple. Elles permettent d'étudier de nombreuses propriétés du système, concernant le régime transitoire ou le régime permanent. D'autre part, la notion de chaîne de Markov se généralise très simplement pour modéliser des problèmes d'optimisation, à l'aide de la notion de processus de décision Markovien. Cette dernière fournit de nombreux outils

permettant de calculer les meilleures politiques de contrôle à appliquer afin d'optimiser les performances d'un système. Le Chapitre 1 présente les définitions rigoureuses de ses différentes notions ainsi que quelque unes de leur principales propriétés.

Malgré toutes leurs bonnes propriétés, la modélisation par chaînes de Markov souffre d'un problème majeur : le nombre d'états nécessaires pour décrire un système explose dès que le système à décrire devient un peu complexe. Prenons par exemple un système composé de N machines reliées en réseau. Si l'état de la machine i à l'instant t est $X_i(t)$, l'état global du système peut être décrit par le vecteur $(X_1(t) \dots X_N(t))$. Si on suppose que chaque machine peut être dans $S = 20$ états différents et qu'il y a $N = 20$ machines en tout, le nombre d'états possibles pour le système est $S^N = 20^{20} \approx 10^{26}$. Cela rend impossible toute méthode qui devrait parcourir les S^N états, même pour un système de taille réduite.

Dans la suite du document, nous explorons différentes solutions afin d'essayer de contourner ce problème d'explosion combinatoire. En particulier, nous nous intéresserons beaucoup aux *modèles champ moyen*, décrits au Chapitre 2 dans lesquels on s'intéresse au comportement du système quand le nombre N d'objets qui composent le système tend vers l'infini.

0.1. Organisation générale du document

Cette thèse est découpée en trois parties. La Partie I contient quatre chapitres qui posent les fondations de ce travail. Les deux premiers chapitres présentent un bilan de la littérature. Le Chapitre 1 rappelle quelques résultats concernant le contrôle stochastique optimal en présentant la notion de processus de décision Markovien. Le Chapitre 2 s'intéresse aux modèles champ moyen et présente en particulier différents résultats de convergence concernant le comportement asymptotique de ces modèles. Les deux derniers chapitres de la partie se concentrent sur l'étude du *vol de travail*. Le premier des deux, Chapitre 3, présente une étude du temps de complétion d'un groupe de tâches à l'aide d'une méthode de potentiel tandis que le Chapitre 4 utilise un modèle champ moyen pour faire une analyse en régime stationnaire. Ces deux chapitres sont basés sur les articles [140, 70].

Dans la Partie II, nous nous intéressons au lien entre le contrôle stochastique et les modèles champ moyen. Dans les Chapitres 5 et 6, nous montrons comment un problème d'optimisation stochastique peut se ramener à un problème d'optimisation déterministe lorsque le nombre d'objets composants le système devient grand. Le dernier chapitre de cette partie, Chapitre 7 étend les résultats champ moyen classique à des systèmes dans lesquels la dynamique est discontinue. Cette partie est basée sur les articles [69, 71, 74, 73, 72].

Dans la dernière partie de ce document, Partie III nous présentons deux problèmes de natures différentes. Dans le Chapitre 8, nous nous posons la question de comment distribuer une infinité de tâches sur un arbre infini de la manière la plus régulière possible. Nous étudions la classe des *arbres équilibrés*. Enfin le dernier Chapitre 9 présente un algorithme de contrôle de puissance pour des réseaux sans fils dont le but est de trouver un équilibre entre l'énergie consommée et le délai subi par les communications. Ces deux chapitres sont basés sur [67, 68, 10].

La Figure 2 montre le lien entre les différents chapitres. Les Chapitres 1 et 2 peuvent être lus de manières indépendante et chacun d'entre eux conduit à un coté de l'analyse du vol de travail. La Partie II repose à la fois des Chapitres 1 et 2. Les deux derniers

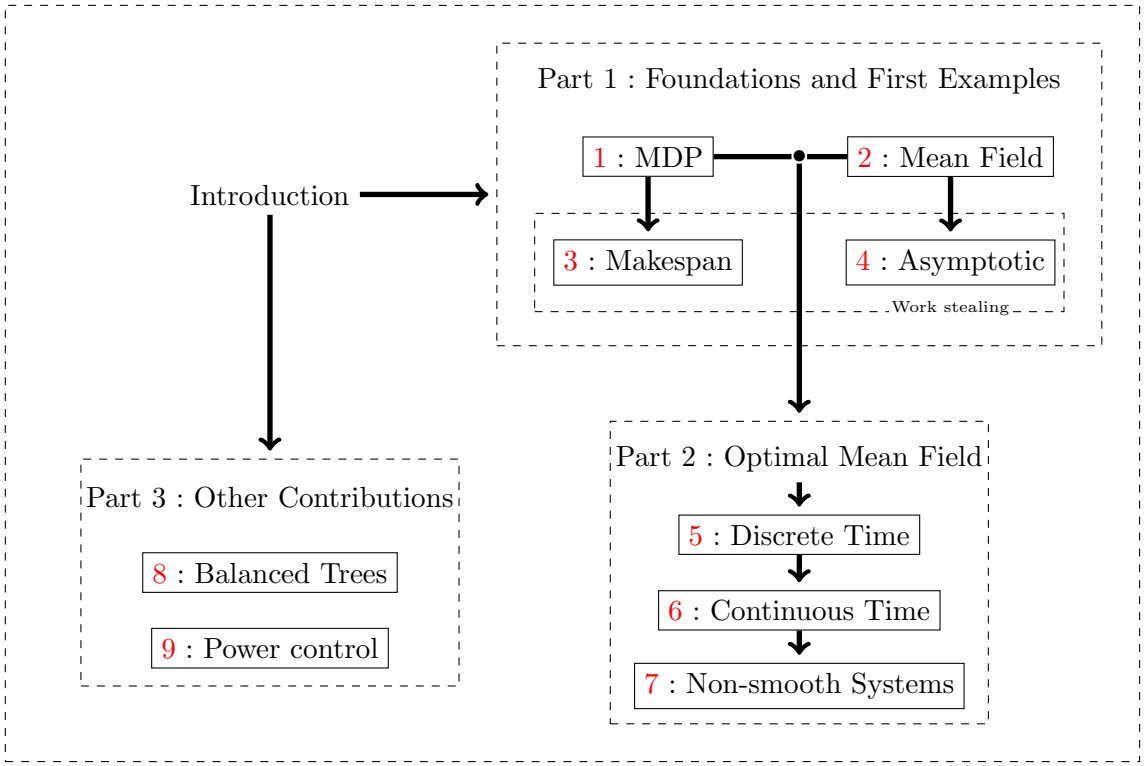


FIGURE 2. : Organisation générale du document

chapitres de la Partie III peuvent être lus de façon indépendante.

1. Chapitres 3 et 4 : Analyse du vol de travail

Actuellement, les ordinateurs deviennent de plus en plus des machines parallèles. Les ordinateurs, même personnels, disposent de deux voir quatre coeurs de calcul et ce chiffre est amené à grossir vite dans un futur proche. Un problème majeur lors de l'utilisation de machines disposant de ressources parallèles est de réussir à répartir le travail sur les différentes ressources disponibles afin d'utiliser au mieux toutes ces ressources.

Une façon simple et très populaire de nos jours pour répartir la charge entre les différentes ressources est le *vol de travail*. Le principe du vol de travail est très simple. Chaque ressource dispose d'une liste de tâches à exécuter. Tant que cette liste est non vide, la ressource travail à finir une de ses tâches. Lorsque que sa liste de tâches est vide, la ressource choisit une autre ressource au hasard et lui *vole* une partie de ses tâches. Malgré son apparente simplicité, cette technique est très efficace et est implémentée dans de nombreuses librairies comme Cilk [66], Intel TBB [99] ou KAAPI [78].

Les Chapitres 3 et 4 présentent deux approches différentes pour analyser les performances du vol de travail.

1.1. Temps de complétion total d'une application ordonnancée par vol de travail

Dans le Chapitre 3, nous étudions le temps nécessaire pour exécuter un ensemble de tâches sur une architecture parallèle utilisant le mécanisme de *vol de travail*. Nous considérons une machine parallèle composée de N processeurs. A l'instant t, $w_i(t)$ représente la quantité de travail dont dispose le processeur i. Dans le modèle le plus simple que nous étudions dans la Partie 3.4, $w_i(t)$ est le nombre de tâches dont dispose le processeur i. Tant que $w_i(t) > 0$, le processeur exécute une tâche par unité de temps. Lorsque $w_i(t)$ atteint 0, le processeur choisit un processeur cible j uniformément parmi les $N-1$ processeurs restants et lui prend la moitié de ses tâches. Si le processeur cible j a une tâche ou moins, le processeur i ne reçoit rien et retente de voler tant qu'il n'a pas de tâches à exécuter.

L'état de la machine peut être représenté par le vecteur $w(t) \stackrel{\text{def}}{=} (w_1(t) \dots w_N(t))$. Cet état évolue de façon déterministe tant qu'aucun des $w_i(t)$ n'est égal à 0. Dans ce cas l'état à l'instant $t+1$ est $w(t+1) = (w_i(t)-1, \dots w_N(t)-1)$. Lorsqu'une ou plusieurs coordonnées arrivent à 0, les processeurs correspondants tentent de voler quelqu'un d'autre. À cause du caractère aléatoire du choix de la victime, ceci introduit du hasard dans le processus.

Malgré la simplicité du modèle, le nombre d'états pour décrire le système est de l'ordre de W^N, où W est le travail total à l'instant 0 : $W = \sum_{i=1}^{N} w_i(0)$. Ceci fait que l'étude complète du processus w n'est pas envisageable. La Figure 3 montre un exemple d'exécution de 2000 tâches par un machine possédant $N = 25$ processeurs. On voit en particulier sur cette figure que les instants pendant lesquelles les machines volent ne présentent aucune régularité.

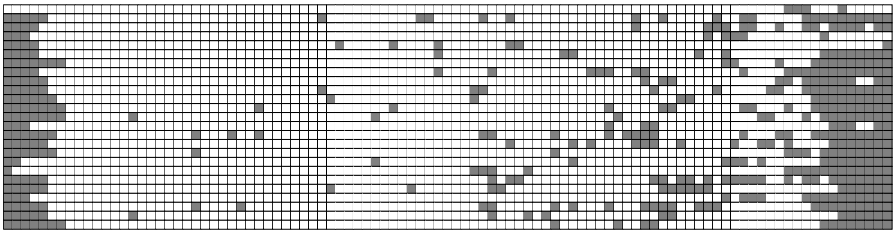


FIGURE 3. : Exemple d'une exécution de $W = 2000$ tâches indépendantes par $N = 25$ processeurs utilisant le vol de travail. L'axe des abscisses représente le temps et celui des ordonnées les différents processeurs. Une zone blanche indique un processeur actif à l'instant t. Les zones grisées sont les moments où les processeurs sont inactifs (*i.e.* $w_i(t) = 0$), qui sont aussi ceux où les processeurs font des requêtes de vol.

Afin de simplifier l'étude du problème, nous introduisons une fonction potentielle $\Phi(t)$ définie par :

$$\Phi(t) = \sum_{i=1}^{N} w_i(t)^2.$$

Cette fonction décroît en fonction des différentes tâches exécutées mais surtout en fonction du nombre de requêtes de vols. En particulier, on peut montrer qu'il existe une fonction

$h(\alpha) \in (0;1)$ telle que s'il y a α processeurs actifs à l'instant t, on a :

$$\mathbb{E}[\Phi(t) - \Phi(t+1) | \Phi(t) = \Phi, \alpha(t) = \alpha] \geq h(\alpha) \cdot \Phi. \tag{1}$$

Afin de finir l'analyse, il reste à étudier la séquence du nombre de processeurs actifs $\alpha(t)$. Si $\Phi(t) \leq 1$, il reste au maximum une tâche dans le système et l'exécution des tâches sera finie à l'instant suivant. Cela nous conduit à introduire un adversaire qui peut choisir la séquence α. Le but de cet adversaire est de maximiser le nombre de vols qui auront lieu avant que $\Phi(t) \leq 1$, partant de $\Phi(0)$ et sachant que $\Phi(t)$ respecte l'Équation (1).

Cela nous permet d'obtenir des bornes sur le nombre de vols à la fois en moyenne mais aussi sur la probabilité de s'écarter de cette moyenne à travers un théorème générique (Théorème 3.1). En particulier, dans le cas des tâches indépendantes, on peut montrer le théorème suivant :

Théorème 1 (Théorème 3.2)**.** *Soit $C_{\max}$ le temps de complétion d'un groupe de W tâches indépendantes ordonnancées par vol de travail. On a :*

(i) $$\mathbb{E}\left(C_{\max}\right) \leq \frac{W}{N} + \frac{2}{1 - \log_2(1 + \frac{1}{e})} \cdot \log_2 W + 1$$

(ii) $$\mathcal{P}\left(C_{\max} \geq \frac{W}{N} + \frac{2}{1 - \log_2(1 + \frac{1}{e})} \cdot \left(\log_2 W + \log_2 \frac{1}{\epsilon}\right) + 1\right) \leq \epsilon.$$

En particulier, cela montre que le temps nécessaire pour exécuter W tâches sur N processeurs est borné par $W/N + 3.65 \log_2 W$.

Dans le Chapitre 3, nous étendons ces résultats au cas de tâches non indépendantes (Théorème 3.4) ainsi qu'au cas du vol coopératif ou plusieurs processeurs peuvent voler en même temps un processeur cible (Théorème 3.5). De plus, nous comparons ces bornes à des simulations (Figure 3.2) qui montrent que la perte de précision liée à l'introduction de l'adversaire est de moins de 50%.

1.2. Un modèle champ moyen du vol de travail sur grille de calcul

Le Chapitre 4 présente un modèle de vol de travail dans une grille de calcul. Nous montrons que quand la taille du système grandit, le système converge vers un système d'équations différentielles qui permettent de calculer efficacement des indicateurs de performances, à la fois en moyenne mais aussi les distributions de ces fonctions.

Nous considérons un système composé de N machines. Ces machines sont groupées en C clusters et chaque cluster est composé d'un grand nombre de machines homogènes. Chaque processeur du cluster c reçoit des tâches à exécuter, à un taux λ_c et chaque tâche à une taille distribuée selon une loi exponentielle (de paramètre 1) et est exécutée à vitesse μ_c. Un processeur exécute les tâches une par une et stocke les autres dans un buffer. On note $j_n(t)$ le nombre de tâches dont le processeur n dispose à l'instant t. Lorsque $j_n(t) = 0$, le processeur choisit un cluster cible c' avec probabilité $p_{cc'}$ et choisit un processeur uniformément dans ce cluster. Un vol de c à c' prend un temps exponentiellement distribué de taux $\gamma_{cc'}$.

Encore une fois, le nombre d'états nécessaires pour décrire le système est trop gros pour permettre une étude détaillée. Pour pallier à ce problème, nous nous intéressons à la proportion de processeurs dans chaque état. On note M_{cj}^N la proportion de machines qui sont dans le cluster c et ayant j tâches. $M_{c0c'}^N$ désigne la proportion de machines du

cluster c ayant 0 tâche et en train de voler un processeur du cluster c'. On peut montrer (Proposition 4.1) que le processus $M^N \stackrel{\text{def}}{=} (M^N_{cj}, M^N_{c0c'})_{c,c' \in \mathcal{C}, j \in K}$ est une chaîne de Markov. De plus, la séquence M^N est un *density dependent population process* (DDPP), ce qui veut dire que la transition d'un état m à un état $m+\ell$ se fait à un taux $\beta_\ell(m)$. En utilisant des résultats sur les DDPP (voir Corollaire 4.2), on peut montrer que le comportement de M^N tend vers celui d'une équation différentielle quand N tend vers l'infini. Cette équation différentielle peut s'écrire facilement depuis la description du système :

$$\begin{aligned}
\dot{m}_{c0c'} &= -(\lambda_c + \gamma_{cc'})m_{c0c'} + \mu_c m_{c1} p_{cc'} + \sum_{c''} \gamma_{cc''} m_{c0c''} \frac{m_{c''0} + m_{c''1}}{m_{c''}} p_{cc'} && (2)\\
\dot{m}_{c1} &= -(\mu_c + \lambda_c)m_{c1} + \mu_c m_{c2} + \sum_{c'} \lambda_c m_{c0c'} && (3)\\
&\quad + \sum_{c'} \gamma_{c'c} m_{c'0c} m_{c2}/m_c + \sum_{c'} \gamma_{cc'} m_{c0c'}(m_{c'2} + m_{c'3})/m_{c'} && (4)\\
\dot{m}_{cj} &= -(\mu_c + \lambda_c \mathbf{1}_{j<K})m_{c,j} + \mu_c m_{c,j+1} + \lambda_c m_{c,j-1} && (5)\\
&\quad + \sum_{c'} \gamma_{c'c} m_{c'0c}(m_{c,2j} + m_{c,2j-1})/m_c && (6)\\
&\quad + \sum_{c'} \gamma_{cc'} m_{c0c'}(m_{c',2j} + m_{c',2j+1})/m_{c'} && (7)\\
&\quad - \sum_{c'} \gamma_{c'c} m_{c'0c} m_{cj}/m_c, && (8)
\end{aligned}$$

Ce résultat permet une étude exhaustive de nombreux indicateurs de performance du système. Nous présentons aussi un algorithme de simulation rapide, rappelé ci dessous, qui permet de calculer la distribution de ces indicateurs. Si J^N désigne l'état d'un processeur, dans ce cas :

Théorème 2 (Théorème 4.3)**.** *Supposons que* $\lim_{N\to\infty} J^N(0) = j(0)$ *et* $M^N(0) \to m(0)$*. Alors,* $(J^N(t), M^N(t))$ *converge faiblement vers un processus* $(J(t), m(t))$ *où* $m(t)$ *satisfait l'ODE (2-8) et* $J(t)$ *est un processus de saut non homogène de noyau* $K(m(t))$*.*

Nous nous intéressons d'abord au cas où le système est homogène (composé d'un seul cluster). En particulier, dans ce cas le régime stationnaire peut être calculé par un algorithme très simple (Algorithme 4.1), ce qui permet de calculer rapidement de nombreux indicateurs de performance. En particulier, nous montrons que les performances du vol de travail sont très bonnes, même quand la latence est élevée. Par exemple, la Figure 4 montre que même lorsque que le temps de vol est de l'ordre de 3 fois inférieur au temps nécessaire pour exécuter une tâche, les performances sont déjà grandement améliorées par rapport à une situation sans vol.

De nombreux autres résultats sont analysés dans la Partie 4.4 du Chapitre 4. En particulier, à l'aide de la simulation rapide, on peut s'apercevoir que le vol de travail améliore de beaucoup les hauts percentiles (voir Figure 4.7). Nous nous intéressons aussi à des système hétérogènes et nous étudions plusieurs stratégies de vol, montrant en particulier que la stratégie de vol optimale dépend plus de la charge des différents clusters que de la géométrie du système (Partie 4.5).

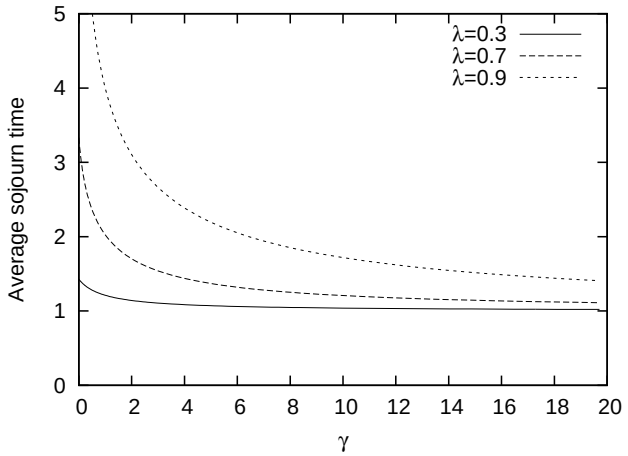


FIGURE 4. : Temps de séjour moyen en fonction du taux de vol moyen γ pour différentes valeurs de λ (.3, .7 and .9).

2. Partie II : Modèles champ moyen optimaux

La deuxième partie de la thèse étudie le lien entre modèles champ moyen et contrôle stochastique. Nous étudions le contrôle optimal dans les Chapitres 5 et 6 avant de passer aux systèmes contrôlés généraux dans le Chapitre 7.

Nous considérons des systèmes composés de N objets, avec N grand (typiquement plusieurs dizaines ou centaines). Le temps est discret et à chaque pas de temps, l'état $X_n^N(t)$ d'un objet n évolue en fonction d'un environnement commun et des états des différents autres objets. L'état global du système est décrit par le couple $(\mathcal{X}^N(t), C^N(t))$ où $\mathcal{X}(t) \stackrel{\text{def}}{=} (X_1^N(t) \dots X_N^N(t))$ représente l'état des différents objets et $C^N(t)$ est l'état du contexte – ou de l'environnement. Lorsque la dynamique du système est invariante par permutation des objets, plutôt que de considérer l'état de chaque objet, on considère la proportion d'objets dans chaque état, aussi appelée la mesure empirique à l'instant t, $M^N(t)$:

$$M^N(t) \stackrel{\text{def}}{=} \frac{1}{N} \sum_{i=1}^{N} \delta_{X_i^N(t)},$$

De tels modèles sont appelés modèles champ moyen. L'intérêt de ces modèles et que sous des hypothèses assez faibles, lorsque le nombre d'objets N devient grand, la dynamique du système converge vers une dynamique déterministe, permettant de grandement simplifier l'étude du système cible, comme au Chapitre 4. Une présentation de différents résultats de convergence est faite au Chapitre 2.

Dans cette partie, nous considérons des modèles champ moyen *contrôlés*. À chaque pas de temps, un contrôleur centralisé choisit une action a dans un ensemble prédéfini d'actions $\mathcal{A}$, qui lui permet d'influer sur la dynamique du système. En particulier, dans les Chapitres 5 et 6, nous considérons le contrôle optimal de tels systèmes. Si le système est dans l'état $(M^N(t), C^N(t))$ à l'instant t et que le contrôleur choisit une action a, il reçoit une récompense $r_t(M^N(t), C^N(t))$. Le but du contrôleur est de trouver la meilleure politique π afin d'optimiser sa récompense moyenne V_π^N entre le temps 0 et T. La quantité

V_π^N se définit par :

$$V_\pi^N(M^N(0), C^N(0)) \stackrel{\text{def}}{=} \mathbb{E}\left(\sum_{t=0}^{T-1} \mathrm{r}_t\left(M_\pi^N(t), C_\pi^N(t)\right) + \mathrm{r}_T\left(M_\pi^N(T), C_\pi^N(T)\right)\right). \tag{9}$$

Les Chapitres 5 et 6 étudient la convergence du système lorsque le contrôleur choisit la politique qui maximise la récompense moyenne (9). En particulier, on s'intéresse à la convergence de la récompense optimale V_*^N définie par :

$$V_*^N\left(M^N(0), C^N(0)\right) \stackrel{\text{def}}{=} \sup_\pi V_\pi^N\left(M^N(0), C^N(0)\right), \tag{10}$$

ainsi qu'aux politiques optimales permettant d'atteindre cet optimal.

Le Chapitre 7 s'intéresse à la convergence de la dynamique du système $M^N(t)$ dans le cas d'une dynamique non continue, ce qui est le cas en particulier lorsque l'on fixe la politique.

2.1. Optimisation en temps discret

Le Chapitre 5 étudie un cas où le problème d'optimisation initial converge vers un problème d'optimisation en temps discret. Nous montrons aussi des résultats du second ordre qui permettent d'étudier la vitesse de convergence. Les résultats du Chapitre 5 concernent trois types de systèmes. Le coeur de ce chapitre est d'étudier la convergence du système contrôlé lorsque qu'une politique est fixée. Cela nous permet ensuite d'en déduire deux types de systèmes : les systèmes pour lesquels la séquence d'actions est fixée (ce qui revient aux modèles champ moyen classiques) ainsi que les systèmes contrôlés par une politique optimale (aussi appelé champ moyen optimaux).

Le modèle stochastique est le suivant. Le système est composé de N objets $(X_1^N \dots X_N^N)$ et d'un contexte C^N. Le contexte évolue de façon déterministe en fonction du contexte à l'instant précédant $C^N(t)$, de l'action prise par le contrôleur et de la mesure empirique à l'instant $t+1$.

$$\mathrm{C}^N(t+1) = g(C^N(t), M^N(t+1), a_t),$$

L'évolution des objets est indépendante une fois le contexte C et l'action a donnée. Chaque objet évolue alors selon un noyau $K(a, C)$. Pour un objet n donné, la probabilité de passer d'un état i à un état j est :

$$\mathcal{P}\left(X_n^N(t+1) = j | X_n^N(t) = i, a_t = a, C^N(t) = C\right) = K_{i,j}(a, C). \tag{11}$$

Les hypothèses précises du modèle sont décrites à la Partie 5.2.2. En plus de l'évolution du système ainsi décrite, elles imposent à l'ensemble d'actions d'être compacte, aux fonctions r, K, g d'être continues et à la condition initiale $(M^N(0), C^N(0))$ de converger vers une valeur $(m(0), c(0))$.

Pour une séquence d'actions $a = (a_1, a_2, \dots)$, on définit un système déterministe correspondant à la moyenne des transitions (11) par :

$$\begin{aligned} m_a(t+1) &= m_a(t) K(a, c_a(t)) \\ c_a(t+1) &= g\left(c_a(t), m_a(t+1), a_t\right). \end{aligned} \tag{12}$$

Considérons une politique π fixée. À toute réalisation aléatoire du système A_π^N est associée une séquence d'actions $A_\pi^N(t) \stackrel{\text{def}}{=} \pi_t(M_\pi^N(t), C_\pi^N(t))$. La séquence A_π^N est une séquence

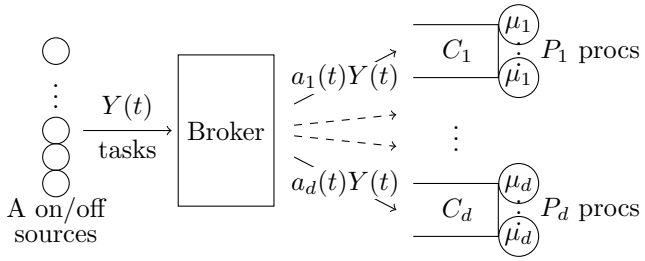


FIGURE 5. : Le problème d'allocation de ressources.

d'action aléatoire *admissible*, c'est à dire qui ne dépend pas du futur. Utilisant la définition de A_π^N et la définition de l'équation (12), on définit une approximation du système initial $(m_{A_\pi^N}(t), c_{A_\pi^N}(t))$. Plusieurs théorèmes (Théorème 5.1, Proposition 5.6) bornent l'écart entre le système stochastique et son approximation, montrant en particulier que l'écart entre $(M^N(t), C^N(t))$ et $(m_{A_\pi^N}(t), c_{A_\pi^N}(t))$ tend vers 0 à vitesse $1/\sqrt{N}$ quand N tend vers 0. Une conséquence directe pour le contrôle optimal est de montrer que la récompense optimale du système initial V_*^N converge vers la récompense optimale du système limite v_* définie par :

$$\sup_a \{v_a(m(0), c(0))\} \stackrel{\text{def}}{=} \sup_a \left\{ \sum_{t=1}^{T} \mathrm{r}_t(m_a(t), c_a(t)) \right\}.$$

La Proposition 5.4 et le Théorème 5.7 précisent ces résultats en montrant de plus que l'écart est d'ordre $1/\sqrt{N}$.

D'un point de vue pratique, les résultats permettent de résoudre de façon asymptotique des problèmes d'optimisation stochastique jusque là impossibles à résoudre. En particulier, ceci est illustré par un problème d'allocation de ressources, présenté à la Figure 5. Dans ce cas, le problème d'optimisation déterministe peut être résolu à l'aide d'un algorithme glouton. La performance de la politique optimale de la limite appliquée au système initial est comparée avec plusieurs politiques classiques à la Figure 5.2. Nous mesurons le gain asymptotique, ainsi que le seuil à partir duquel elle surpasse les politiques classiques.

Plusieurs extensions sont aussi présentées dans la Partie 5.5. Dans le cas d'un horizon infini et d'un coût actualisé, nous montrons un théorème de convergence au premier ordre ainsi qu'un théorème du second ordre si le facteur d'actualisation est suffisamment petit (Théorème 5.11 et Proposition 5.12). Dans le cas d'un horizon infini et d'un coût non actualisé, nous présentons plusieurs exemples et contre-exemples qui montrent que la convergence n'a pas toujours lieu, voir Partie 5.5.3.

2.2. D'un problème optimisation stochastique discret à un problème continu

Dans le chapitre précédant, nous avons considéré un modèle dans lequel la probabilité pour un objet donné de changer d'état ne dépend pas de N. Dans le Chapitre 6, nous nous intéressons au cas où la probabilité pour un objet de changer d'état dépend de N et tend vers 0 lorsque N tend vers l'infini. Alors que dans le cas précédant le système limite restait en temps discret, ici le système limite devient un système en temps continu.

Nous considérons un système composé de N objets dans lequel la dynamique est invariante par permutation des objets. Le temps est discret et est noté k. L'espérance de

la différence entre $M^N(k+1)$ et $M^N(k)$ sachant que le système était dans l'état m et que l'action prise est a est appelée le *drift* et notée $F^N(m,a)$:

$$F^N(m,a) \stackrel{\text{def}}{=} \mathbb{E}\left(M^N(k+1) - M^N(k) | M^N(k) = m, A^N(k) = a\right).$$

Alors que dans le chapitre précédant, ce drift était invariant en fonction de N, nous supposons qu'il existe une fonction $I(N)$ et une fonction $f(m,a)$ telles que $\lim_{N\to\infty} I(N) = 0$ et $\lim_{N\to\infty} \left|\frac{1}{I(N)} F^N(m,a) - f(m,a)\right| = 0$. Les principales hypothèses de ce modèle sont d'imposer au drift f d'être continue (plus précisément, une le drift doit être une fonction lipschitzienne) et que le nombre d'objets qui fait une transition par unité de temps est borné par $NI_0(N)$ où $I_0(N)$ tend vers 0 lorsque N grandit, voir Partie 6.2.5 pour plus de détails.

En pratique, cela veut dire que l'espérance du changement entre deux pas de temps tend vers 0 quand N tend vers l'infini. Cela conduit à considérer un processus dans lequel le temps a été accéléré. On définit $\left(\hat{M}^N(t)\right)_{t\in\mathbb{R}^+}$ le processus qui est une fonction affine prenant la valeur $M^N(k)$ en $kI(N)$. Une fonction d'action $\alpha : [0;T] \to \mathcal{A}$ est une fonction Lipschitz par morceau qui a t associe une action $\alpha(t)$. De façon similaire à la partie précédente, on définie l'approximation champ moyen de $M^N(t)$ par :

$$m(t) - m(0) = \int_0^t f(m(s), \alpha(s))ds. \tag{13}$$

Le but du chapitre est de montrer l'équivalence entre le problème d'optimisation stochastique entre 0 et $T/I(N)$ (10) et le problème d'optimisation déterministe sur une équation différentielle suivant :

$$\sup_\alpha v_\alpha(m_0) \stackrel{\text{def}}{=} \sup_\alpha \int_0^T r\left(\phi_s(m_0,\alpha), \alpha(s)\right) ds. \tag{14}$$

où $\phi_s(0,\alpha)$ désigne l'unique solution de l'équation différentielle (13).

De façon similaire au chapitre précédant, nous introduisons dans la Partie 6.3 la fonction d'action A^N_π qui est une fonction en escalier prenant la valeur $\pi_k(M^N(k))$ à l'instant $kI(N)$. Sous les hypothèses décrites à la Partie 6.2.5, on peut démontrer (Théorème 6.1) que l'écart entre $\hat{M}^N_\pi(t)$ et $\phi_t(m_0, A^N_\pi)$ tend vers 0. Cela permet en particulier de prouver le Théorème 6.4 qui montre la convergence du problème d'optimisation stochastique vers son équivalent déterministe :

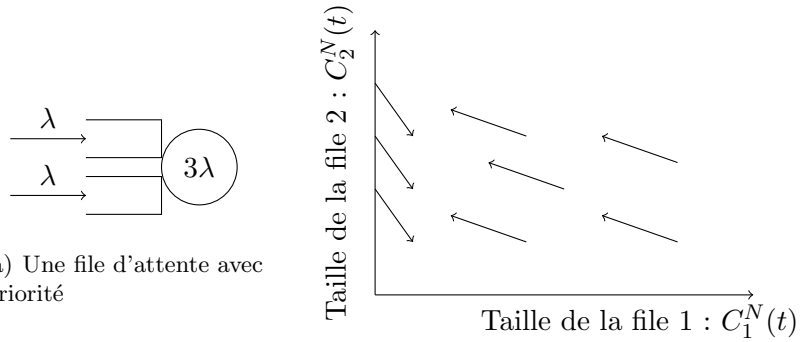
$$\lim_{N\to\infty} V^N_*\left(M^N(0)\right) = v_*\left(m_0\right).$$

Ces résultats sont illustrés Partie 6.4 par trois exemples qui concernent respectivement des stratégies d'investissement, un contrôle d'épidémie ainsi qu'un problème d'allocation de ressources. En particulier, le premier exemple présente un cas simple où le problème déterministe peut être résolu entièrement. Le deuxième considère un problème de propagation de virus. Alors que plusieurs papiers s'intéressent directement à l'optimisation sur le modèle continu, notre approche permet de justifier l'approximation ainsi faite. Enfin le dernier exemple illustre comment notre technique permet de réduire l'espace d'état à explorer en vue d'une résolution numérique.

2.3. Modèles champ moyen non réguliers et inclusions différentielles

Dans les Chapitres 5 et 6, nous avons toujours considéré que la dynamique d'évolution du problème de départ était continue. Typiquement, dans le Chapitre 6, $f(m)$ est supposée lipschitzienne. Cette hypothèse est centrale dans la plupart des modèles champ moyen. Le but du Chapitre 7 est d'étudier des cas où la dynamique du système présente des discontinuités. Nous montrons que quitte à remplacer l'équation différentielle par une inclusion différentielle, les résultats de convergence restent valables.

Considérons par exemple le modèle simple de file d'attente avec priorité présenté à la Figure 6(a). Des paquets prioritaires arrivent avec un taux λ dans le système et des paquets de priorité normale arrivent aussi avec un taux λ. Notons $C_1(t)$ le nombre de clients prioritaires à l'instant t et $C_2(t)$ le nombre de clients normaux. Un serveur sert les paquets avec un taux 3λ. Si $C_1(t) > 0$, le serveur sert uniquement les clients 1. Il ne sert les clients 2 que si $C_1(t) = 0$.



(a) Une file d'attente avec priorité

(b) Drift du système

FIGURE 6. : La file d'attente avec priorité et le drift correspondant

Notons f le drift du système, autrement dit, l'espérance de changement entre t et $t + dt$:

$$f(c_1, c_2) \stackrel{\text{def}}{=} \lim_{dt \to 0} \frac{1}{dt} \mathbb{E}\left((c_1(t+dt), c_2(t+dt)) - (c_1(t), c_2(t)) | c_1(t) = c_1, c_2(t) = c_2\right).$$

Un calcul direct montre que f est le champ de vecteurs représenté par la Figure 6(b), c'est-à-dire défini par :

$$f(c_1, c_2) = \begin{cases} (-2\lambda, \lambda) & \text{si } C_1(t) > 0 \\ (\lambda, -2\lambda) & \text{si } C_1(t) = 0. \end{cases} \tag{15}$$

En suivant la méthode classique utilisée notamment au chapitre précédant, on peut penser que la dynamique du système est proche de celle de l'équation différentielle $\dot{c}(t) = f(c(t))$. Malheureusement, à cause de la discontinuité de f quand $c_1 = 0$, cette équation différentielle n'a pas de solution. Ceci vient du fait que la notion d'équation différentielle n'est pas adaptée à l'étude de système présentant des discontinuités et que la notion d'*inclusion différentielle* est mieux adaptée à ce cas.

Une inclusion différentielle est similaire à une équation différentielle dans laquelle l'égalité $\dot{y}(t) = f(y(t))$ est remplacée par l'inclusion :

$$\dot{y}(t) \in F(y(t)), \tag{16}$$

où F est une fonction à valeur dans les parties de $\mathbb{R}^d$. Une solution de l'inclusion différentielle (16) est une fonction absolument continue telle que (16) est vérifiée presque partout. Une inclusion différentielle peut éventuellement avoir plusieurs solutions.

Il existe une méthode systématique pour passer d'une équation différentielle non continue, comme (15) à une inclusions différentielle en définissant $F(y)$ par :

$$F(y) = \bigcap_{\epsilon>0} \lim_{K\to\infty} \bigcup_{N\geq K} \overline{\text{conv}} \left\{ \frac{f^N(z)}{I(N)} : \|z-y\| \leq \epsilon \right\}. \tag{17}$$

où $\overline{\text{conv}}$ désigne la fermeture de l'enveloppe convexe d'un ensemble.

Le Chapitre 7 étudie la relation entre les modèles champ moyen non continus et l'inclusion différentielle définie par (17). De façon analogue au cas précédant, nous considérons un système composé de N objets et d'une ressource C^N qui évolue en temps discret. Si $M^N(k)$ désigne la mesure empirique à l'instant k, on note $Y^N(k) \stackrel{\text{def}}{=} (M^N(k), C^N(k))_k$, l'état du système à l'instant k. Comme dans le chapitre précédant, le drift du système est noté f^N et est défini par $f^N(y) \stackrel{\text{def}}{=} \mathbb{E}\left(Y^N(k+1) - Y^N(k) | Y^N(k) = y\right)$ et on suppose que le drift normalisé par l'intensité converge vers une fonction f : $\left\|f^N(y)/I(N) - f(y)\right\| \to 0$. Contrairement au chapitre précédent, aucune condition de régularité sur f n'est imposée.

Le principal résultat de ce chapitre est le Théorème 7.5 qui montre que le système stochastique $Y^N(t)$ converge vers l'ensemble des solutions de l'inclusion différentielle où F est définie par (17). Si $\mathscr{S}_T(y_0)$ désigne les solutions de l'inclusion différentielle (16) telle que $y(0) = y_0$, alors :

Théorème 3 (Théorème 7.5)**.** *Si le drift satisfait* (7.7) *et que :*

- *F is semi continue supérieurement et il existe $c > 0$ telle que $\|F(y)\| \leq c(1 + \|y\|)$,*
- *$Y^N(0) \xrightarrow{\mathcal{P}} y_0$.*

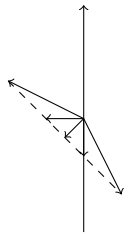
Alors, quel que soit $T > 0$:

$$\inf_{y\in\mathscr{S}_T(y_0)} \sup_{0\leq t\leq T} \left\|\bar{Y}^N(t) - y(t)\right\| \xrightarrow{\mathcal{P}} 0.$$

En particulier, dans le cas où l'inclusion différentielle a une unique solution $y(.)$, cela montre que pour tout t : $\bar{Y}^N(t)$ converge vers $y(t)$. Ce résultat peut être appliqué directement pour calculer la limite fluide de l'exemple de la file avec priorité de la Figure 6. Un calcul direct montre que le drift défini par (17) est égal à $F(c_1, c_2) = (-2\lambda, \lambda)$ si $c_1(t) > 0$ et égal à l'enveloppe convexe de $(-2\lambda, \lambda)$ et $(\lambda, -2\lambda)$ lorsque $c_1(t) = 0$, voir Figure 7(a). L'inclusion différentielle ainsi définie a une unique solution, représentée Figure 7(b). Le Théorème 7.5 montre que la dynamique du système stochastique converge bien vers celle du système décrit à la Figure 7(b).

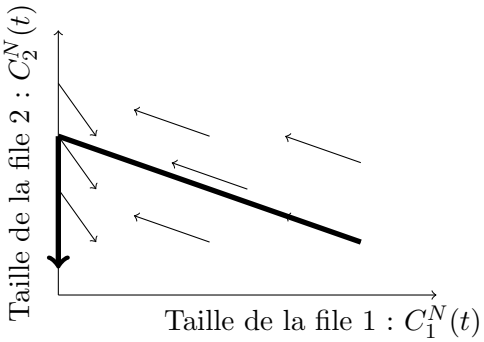
Ce théorème peut être affiné lorsque l'inclusion différentielle vérifie une propriété de one-sided Lipschitz (voir Équation (7.9) et Théorème 7.7) en montrant une convergence en probabilité avec des bornes explicites. De plus, nous présentons aussi une version en temps continu du théorème au Théorème 7.8.

Ces résultats permettent d'étendre l'applicabilité des techniques de champ moyen à des systèmes présentant des dynamiques à seuil ou d'autres formes de discontinuité. Nous présentons plusieurs exemples pour illustrer ces résultats à la Section 7.5. Nos exemples concernent des modèles de systèmes de calcul volontaire, des modèles en théorie des jeux ou encore un modèle de calcul auto-adaptatif.



$c_1 = 0$

(a) L'ensemble des valeurs du drift est l'ensemble des vecteurs en pointillés.



(b) Unique solution de l'inclusion différentielle.

FIGURE 7. : Enveloppe convexe du drift en c_1=0 et l'unique solution de l'inclusion différentielle.

3. Partie III : autres contributions

La dernière partie du document est consacrée à l'étude de deux problèmes distincts. Le Chapitre 8 s'intéresse aux propriétés des arbres infinis équilibrés tandis que le Chapitre 9 étudie un algorithme de contrôle de puissance dans les réseaux sans fils.

3.1. Arbres équilibrés

Dans ce chapitre, nous étudions des classes d'arbres non orientés étiquetés par $\{0, 1\}$. Nous introduisons les notions d'arbres rationnels et sturmiens ainsi que les définitions d'arbres (fortement) équilibrés et étudions les relations entre ces différentes classes. Cette notion généralise la notion de mots sturmiens, qui ont d'importantes propriétés d'optimalité [77].

Contrairement à d'autres papiers qui considèrent des arbres ordonnés (voir Partie 8.2.1), ce chapitre s'intéresse aux propriétés des arbres non orientés. Un arbre est un graphe infini orienté dont chaque noeud a un degré entrant 1 et un degré sortant $d \geq 2$ sauf un noeud qui a un degré entrant 0, appelé la racine. Chaque noeud est étiqueté par la lettre 0 ou 1. Le but de ce chapitre est d'étudier comment répartir les 0 et les 1 sur cet arbre et en particulier, comment répartir les 1 de manière aussi uniforme que possible.

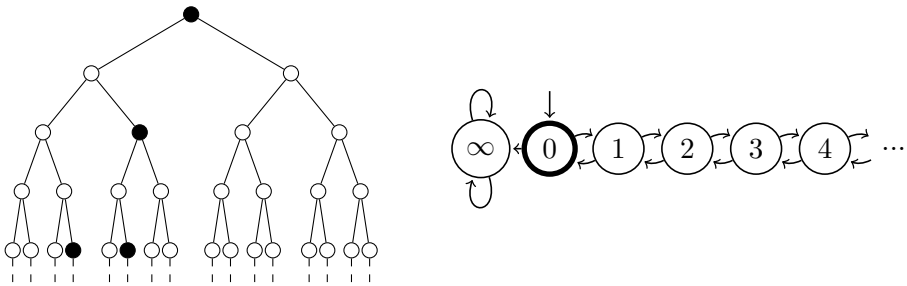


FIGURE 8. : L'arbre de Dyck et son graphe minimal. Les noeud en gras représentent les noeuds étiquetés par 1.

Nous introduisons la notion de graphe minimal associé à un arbre. Le graphe minimal d'un arbre est un multi-graphe fini ou infini. Chaque noeud est étiqueté par 0 ou 1 et possède un degré sortant égal à d mais aucune contrainte n'est imposé sur le degré entrant. À chaque multi-graphe est associé un arbre qui s'obtient en "développant" le graphe. Respectivement, on peut associer à chaque arbre un multi-graphe minimal unique. La Figure 8 donne un exemple d'arbre et de son multi-graphe minimal.

Pour un noeud v et une hauteur $n \geq 1$, on note $h_v(n)$ le nombre de noeuds étiquetés par 1 dans le sous arbre complet de racine v et de hauteur n. La proportion de 1 dans ce sous arbre est $d_v(n) = h_v(n)(d-1)/(d^n - 1)$. La densité d'un arbre est la limite (quand n tend vers l'infini) de $d_v(n)$. Si cette limite dépend de v, l'arbre est dit avoir une densité enracinée. Si cette limite n'existe pas mais que $\lim_{n\to\infty} k^{-1} \sum_{k=1}^{n} d_v(k)$ existe, la densité est dite moyennée.

Nous étudions premièrement la notion d'arbres rationnels qui sont les arbres dont le multi-graphe minimal est fini, en montrant en particulier qu'ils ont tous une densité moyennée enracinée. L'irréductibilité de leur graphe minimal assure que la densité n'est pas enracinée tandis que l'apériodicité de celui-ci implique que la densité n'a pas besoin d'être moyennée (Théorème 8.3).

Le coeur du chapitre concerne la notion d'arbres équilibrés et fortement équilibrés, présentée dans la Partie 8.4. Les arbres équilibrés sont des arbres pour lesquels les noeuds étiquetés par 1 sont dispersés de façon aussi régulière que possible. Un arbre est équilibré si pour tous noeuds v et v' et toute hauteur n, le nombre de 1 dans deux sous arbres de hauteur n diffèrent d'au plus 1 : $|h_v(n) - h_{v'}(n)| \leq 1$. Il est dit fortement équilibré si pour tout v, v', n, k : $|h_v(n) - h_v(k) - (h_{v'}(n) - h_{v'}(k))| \leq 1$. Nous montrons l'existence de ces arbres et les caractérisons en montrant qu'ils coïncident avec la notion d'arbres mécaniques (Théorème 8.9). Cela permet d'une part de les construire par un procédé algorithmique et de les caractériser. Cela nous permet aussi de décrire la forme du graphe minimal d'un arbre fortement équilibré rationnel (Théorème 8.16) et de construire un algorithme permettant de tester si un arbre rationnel est équilibré ou non (Algorithme 8.1).

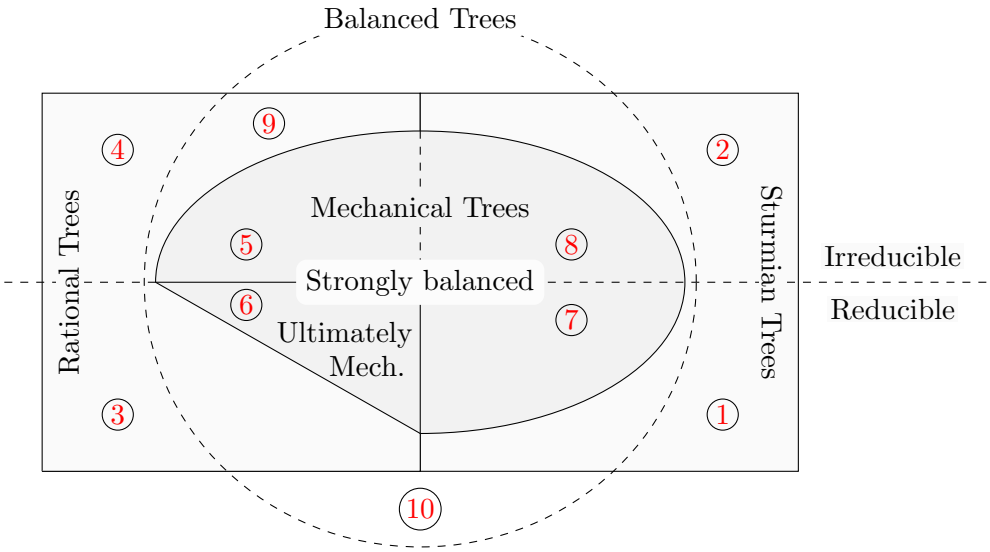


FIGURE 9. : Relation d'inclusion entre les différentes notions d'arbres. La numération des zones sur la figure indique un des exemples détaillés à la Partie 8.6.

Tous ces résultats sont illustrés par de nombreux exemples et résumés à la Figure 9. La Partie 8.6 contient de nombreux exemples qui montrent les inclusions strictes qu'il peut y avoir entre ces différentes classes. La numération des zones sur la figure indique un des exemples détaillés à la Partie 8.6. Par exemple, la zone 9 correspond à l'ensemble des arbre rationnels, irréductibles, équilibrés mais qui ne sont ni fortement équilibrés, ni mécaniques, ni sturmiens. L'exemple correspondant à tous ces critères est donné par l'arbre de la Figure 10 et détaillé au point 9 de la Partie 9.

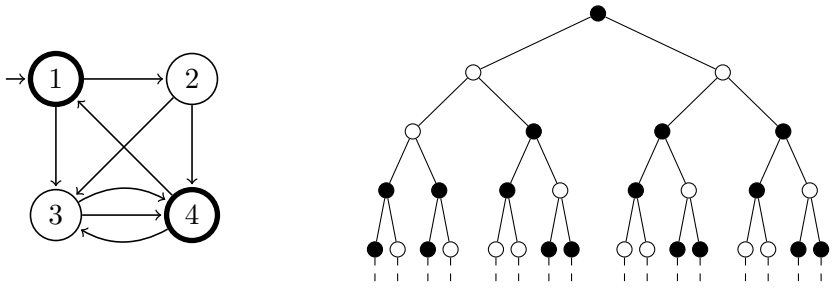


FIGURE 10. : Un arbre rationnel équilibré qui n'est pas fortement équilibré : le graphe minimal de l'arbre à gauche et les quelques premiers étages de l'arbre à droite.

3.2. Algorithme de contrôle de puissance dans les réseaux sans fils

Dans le Chapitre 9, nous présentons un algorithme de contrôle de puissance pour les réseaux sans fils. Ce mécanisme est un algorithme distribué dans lequel chaque émetteur essaie de trouver un équilibre entre la puissance consommée et le délai subi par les communications. Nous étudions la convergence de l'algorithme ainsi les propriétés de l'équilibre vers lequel il converge.

Nous considérons un modèle de réseaux sans fils composés de L liens de communications se partageant un même canal de communication. Chaque lien est composé d'un récepteur et d'un émetteur qui peut choisir sa puissance d'émission. La géométrie du système est modélisée par une matrice de gain G : si l'émetteur du lien j émet à puissance P_j, la puissance du signal reçu par le récepteur i est $G_{ij}P_j$. Chacun de ces termes contribue aux interférences reçues par le lien i. À cela s'ajoute un bruit thermique N_i. Ainsi, le rapport signal sur bruit reçu par le récepteur i est :

$$\gamma_i = \frac{G_{ii}P_i}{N_i + \sum_{j\neq i} G_{ij}P_j}.$$

Pour un rapport signal sur bruit de γ_i, le débit estimé par le noeud i est $S_i(\gamma_i)$. S_i représente typiquement la probabilité qu'un paquet soit transmis avec succès, par exemple $S_i(x) = 1 - \exp(-c \cdot x)$, mais peut aussi être une fonction plus générale, comme $S_i(x) = \log(1+c\cdot x)$. Chaque émetteur essaie d'optimiser un compromis entre la puissance dépensée par paquet $P/S(\gamma)$ et le délai estimé de ces paquets $1/S(\gamma)$. À chaque pas de temps, chaque émetteur met à jour sa puissance en minimisant la quantité $\alpha/S(\gamma)+P/S(\gamma)$ où α est facteur de son choix représentant à quel point il accorde plus d'importance au

délai ou à la puissance :

$$P_i^{t+1} = \Phi_i(b_i^t) = \arg\min_{p\in[0;\infty)} \left\{ \frac{\alpha_i}{S_i(\frac{p}{b_i^t})} + \frac{p}{S_i(\frac{p}{b_i^t})} \right\}. \tag{18}$$

où $b_i^t = N_i/G_{ii} + \sum_{j\neq i} G_{ij}P_j/G_{ii}$. Nous nous intéressons à la généralisation de (18) en incorporant des arrivées de paquets et où le délais est remplacé par $1/(S(\gamma) - \lambda)$, voir Partie 9.3.1.

Le Chapitre 9 étudie l'algorithme DPC, défini par (18). La première propriété établie (Théorème 9.12) est de montrer que sous des hypothèses faibles sur la fonction S décrites Partie 9.4.1 et si les contraintes sur le délai λ sont faisables (ce qui est le cas en particulier lorsque $\lambda = 0$, comme dans l'Équation (18)), alors DPC converge vers une assignation de puissance $P^{\alpha,N}$ indépendante du point de départ. Ce théorème se démontre par des arguments de topologie, en montrant certaines propriétés vérifiées par la fonction Φ de l'Équation (18), voir Partie 9.4.2.

Nous étudions le point d'équilibre $P^{\alpha,N}$ et les rapports $\gamma^{\alpha,N}$ correspondant, en montrant des propriétés de monotonie (Proposition 9.16) en fonction des paramètres ainsi que des propriétés d'échelles lorsque les paramètres sont multipliés par une constante (Proposition 9.17). De plus, nous montrons que la fonction $\alpha \mapsto \gamma^{\alpha,N}$ est une bijection de l'ensemble des paramètres dans l'ensemble des rapports signal sur bruit atteignables. En se basant sur DPC et sur le mécanisme de [64], nous développons en Partie 9.5 deux protocoles dont le but est d'adapter les paramètres α.

4. Conclusion

Dans ce document, nous avons vu plusieurs approches pour le contrôle et l'optimisation de grands systèmes à travers l'étude de modèles stochastiques. Un des principal problème étudié est de combattre l'explosion combinatoire liée au trop grand nombre d'états de ces modèles.

Une première solution, présentée au Chapitre 3, est d'agréger l'information contenue dans l'état du système en une seule variable, appelée *potentiel* du système. Cette méthodologie est générique. Elle est prouvée être très précise à travers l'étude du temps de complétion d'un groupe de tâches ordonnancées par vol de travail. L'autre approche présentée est l'étude des modèles champ moyen, présentés au Chapitre 2, et dont nous montrons un exemple d'application au vol de travail dans le Chapitre 4. Partant d'un modèle d'objets interagissant, il est souvent facile de construire des équations déterministes décrivant le système permettant une étude rapide du comportement du système.

Une des principales avancées de ce document est le lien établi entre les modèles champ moyen et le contrôle stochastique, présenté à la Partie II et résumé Figure 11. Nous montrons comment la résolution d'un problème de contrôle optimal stochastique peut se ramener à la résolution d'un problème d'optimisation déterministe lorsque la taille du système grandit. Ces résultats permettent à la fois de résoudre de nouveaux problèmes, comme à la Partie 5.4 ou de donner des intuitions sur les politiques asymptotiquement optimales, comme à la Partie 6.4.3. Malheureusement, d'un point de vue pratique, la résolution du problème déterministe reste souvent un problème compliqué et les méthodes numériques "brute-force" sont souvent difficiles à mettre en place. Nous pensons que les résultats sur les modèles champ moyen discontinus du Chapitre 7 peuvent nous aider dans le cas où l'on s'intéresse à l'optimalité dans une classe réduite de politiques.

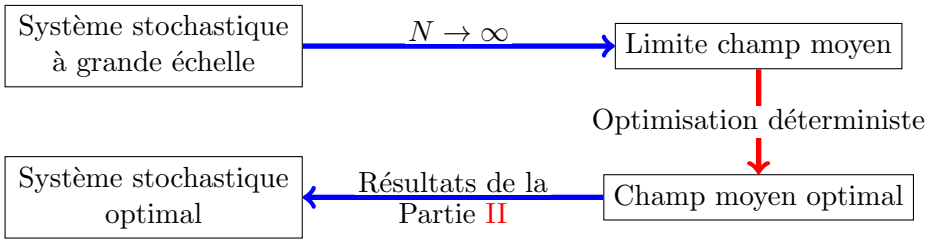


FIGURE 11. : Méthodologie générale concernant les champ moyen optimaux de la Partie II. Les flèches bleues représentent des tâches faciles à faire tandis que la flèche rouge indique un problème algorithmique.

Une généralisation de ces travaux serait de considérer le cas de coûts non additifs. Prenons par exemple un problème d'allocation de ressources où des tâches arrivent dans un système. Si $X(t)$ est le nombre de tâches présentes dans le système à l'instant t, la formule de Little nous montre qu'optimiser le temps de complétion moyen de chaque tâche revient à minimiser $\sum_t X(t)$. Néanmoins, dans le cas d'applications parallèles, les tâches arrivent dans le système par paquets de 100 ou plus et le vrai problème est d'optimiser le temps de complétion de chaque groupe de tâches et non de chaque tâches indépendamment. Un problème similaire mais sans arrivée de tâches est étudié au Chapitre 3, il serait intéressant de regarder une généralisation de ce problème en considérant un processus d'arrivée de groupe de tâches et d'étudier des mécanismes (distribués) afin d'adapter les ressources aux différentes applications en cours.

Une autre question serait d'étudier le cas d'optimisations multicritères. Dans tous les problèmes considérés, la mesure de performance a toujours été l'optimisation d'un objectif à valeur réelles, comme le temps de réponse moyen. De nos jours, l'énergie devient un enjeu capital et l'on est souvent confronté à trouver un compromis entre réactivité du système et puissance consommée. Cela amène à considérer des objectifs de la forme (D, E) : délais et énergie. Malheureusement, décider qu'un couple (D, E) est meilleur qu'un autre (D', E') n'est pas possible si $D < D'$ et $E > E'$. On est alors amené à étudier l'ensemble des points Pareto optimaux, c'est à dire l'ensemble des points (D, E) tels qu'il n'existe pas de points (D', E') avec $D' < D$ et $E' < E$. Malheureusement, décider lequel des points de Pareto choisir n'est pas quelque chose de facile.

Dans le Chapitre 9, ce problème est résolu en choisissant un paramètre α et en essayant de minimiser $\alpha D + E$. Néanmoins, les quantités D et E étant difficilement comparable, le choix du paramètre α devient un problème en soit. Pour l'instant, il n'existe pas de bonne solution à ce problème et cela est sûrement quelque chose à développer dans le futur.

Introduction

While users, administrators or developers are all interested by performance evaluation of systems, they all have different motivations. Users want to obtain a solution that provide the best performance at the lower cost. A user may have the choice between different alternatives and she wants to be able to quantify what are the gains or the losses of each solution: having the choice between a cheap solution and an expensive one would like to know how better is the expensive one. The motivation for administrators is often the dimensioning of computing systems: knowing roughly how many users will consult her website, an administrator wants to buy the smallest number of servers to satisfy this demand. The designer of a new system will try to identify the critical factors that will impact the application. In particular, knowing which piece is the bottleneck for performance is very important for someone who wants to improve the performance at a lower cost.

Model of performance started with the work of Erlang at the beginning of the twenties century who studied telephone traffic [60]. Based on measurement of the characteristics of telephone traffic and simple stochastic models, he came up with the Erlang's formula [61] that have been widely used for the dimensioning of many communication systems. Throughout the years, the systems that we want to study became more and more complex. Modern systems are now widely interconnected and one cannot neglect the interactions between. This leads to consider large stochastic networks.

The performance evaluation of a system relies on two things: the *objectives* and the *methods*. The objectives are quantified in term of performance metrics. A performance metric is a quantity that captures an interesting property of the system, concerning for example the average behavior or the probability of a particular event to appear. A precise definition of the metric depends on the problem considered. Based on this metric, the objective might either be to evaluate this particular performance metric – which is called *performance prediction* – or to find a control policy of this system in order to optimize this performance metric – in that case, we talk about *optimal control*.

The methods that can be used to achieve the objectives can be decomposed into three categories: *measurement*, *simulation* or *analytical modeling*. Each of these three approaches has its own advantages and drawbacks, illustrated in particular by the trade-off between the precision and the complexity needed to implement it, illustrated in Figure 1. The combination of the three is necessary in order to understand fully the behavior of a system.

Measurement is not an easy thing to do. It is often hard to measure a system without disturbing it and sometimes not possible, for example if the system does not exist yet. It might be hard to avoid the bias of the measuring technique [51] and measurement without modeling may lead to mistakes. Simulation is probably the most commonly used method to study a system. It is often much cheaper and faster to build a simulator than to build a real system. Moreover, the computation time to run a simulation is generally some orders of magnitude faster than the one to run the real application. However, the more complex is this model, the more time will be needed to build and to run the

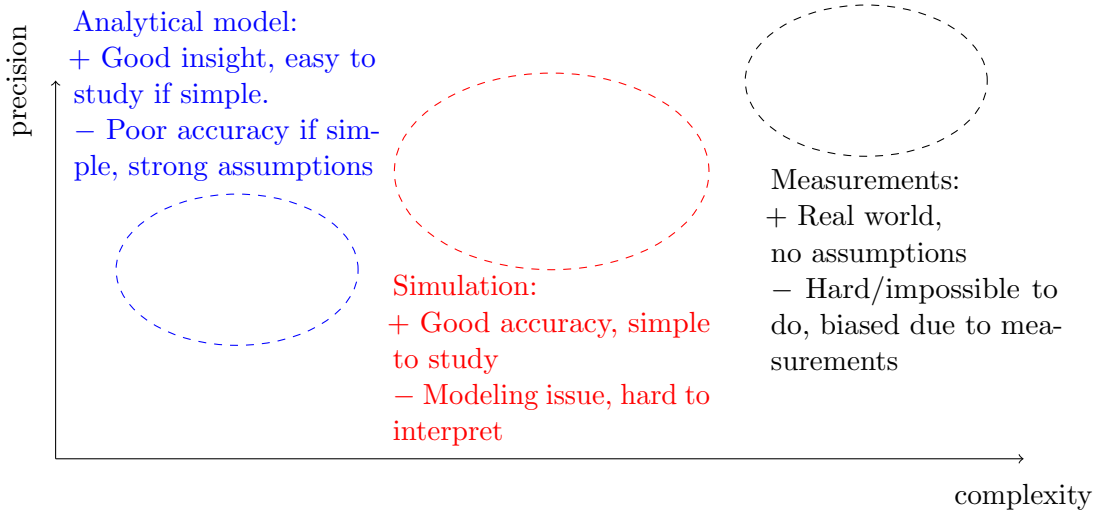


Figure 1.: The trade-off between complexity and precision for the three methods of performance evaluation illustrated by some advantages and drawbacks.

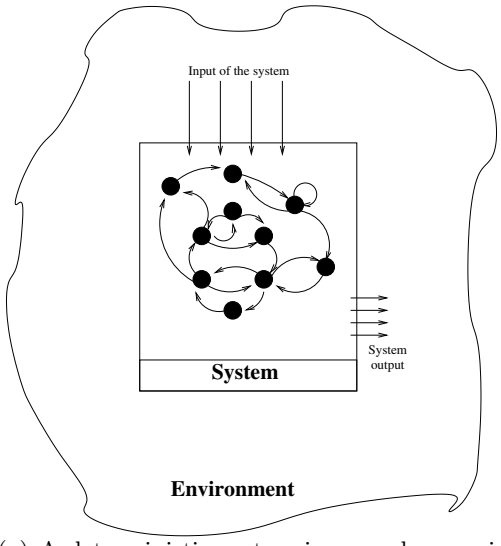
simulator. Interpreting the results of simulations raises several problems. It is often hard to distinguish if what we observe is representative of the average behavior or if it is one of many very different samplings.

The last aspect of performance evaluation and the focus of this thesis is the analytical study of mathematical models. Studying a model analytically allows one to capture structural properties of the system that cannot be measured via simulation, even if the exact resolution of the model is impossible. The main problem with model analysis is the complexity of its analysis. Without the appropriate tools, one needs wild assumptions or a dramatic simplification of the model in order to be able to say something about the model. This emphasizes the need for specific tools to study large stochastic models.

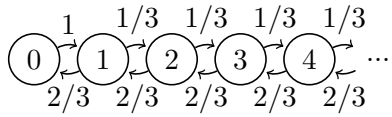
Stochastic models: the dimensionality issue

The purpose of this thesis is to develop new methods for the analysis and control of large stochastic models. In most computing systems, randomness is not inherent in the model. A software or a hardware device is totally deterministic (it is impossible to implement real randomness on a computer) but reacts to external events. The randomness will be introduced to represent the external environment in which the system evolves, as shown in Figure 2(a). For example, let us consider a simple model of a server that serves jobs sent by some customers (think about an Internet router for example). When the server has more than one job to serve, they are stored in a buffer. The quantity we are interested in is the number of jobs stored in the buffer. The possible events than can happen to this buffer are either an arrival of job in the system (denoted A) or a departure of job (denoted D) when the server has finished to serve a job. Given a sequence of events $A, A, D, A, D, A, D, D, A, D, A, A, A, A, D, D \ldots$ and if the initial state of the system is given, it is straightforward to compute what will be the state of the system at each time.

However, when one wants to study such a system, she has to know the sequence of events that occurred. This could be done by measurements on a real system and applying this sequence on our model. However, the main problem of this (apart from the difficulty



(a) A deterministic system in a random environment



(b) A Markov chain (or stochastic automaton) for the server model

Figure 2.: Deterministic systems embedded in a random environment lead to Markov chains.

of measurement) is that by doing so, one only captures one trajectory and looses a general comprehension. Instead of considering one particular sequence of events, we consider that each event occurs with some probability. This probability may depend on the current state of the system. Going back to our server model, if the arrivals occur twice more slowly than the departures, this can be represented by saying that if the number of jobs in the buffer is non zero, the next event will be a departure with probability $2/3$ and an arrival with probability $1/3$. If the buffer is empty, the next event is always an arrival (*i.e.* with probability 1). This leads to the graph of Figure 2(b).

The type of mathematical object represented in Figure 2(b) is called a *Markov chain*. A precise definition of what a Markov chain is and some of its properties is recalled in Chapter 1. Informally, a Markov chain can be seen as a graph of states. The edges of the graph represent the probability of going from one state to another. The interest of Markov chains is twofold: it is generally simple to study and has a great expressive power. Given the model of Figure 2(b), it is rather easy to study the steady-state behavior and answer questions like *what will be the average waiting time of customers entering in the system?* or *what is the probability for someone to wait more than x seconds?* It is also easy to compute quantities pertaining to the transient behavior, like the *average time to empty the buffer starting from* 10 *jobs*.

Yet another application of this stochastic modeling is to consider its controlled behavior. Let us again consider the simple model of a server but this time, the router might decide to accept or refuse incoming jobs. The more an accepted customer waits, the more it gets frustrated. After some point, the customer would have been less frustrated if it had not been accepted at all. Mixing stochastic models with controlled dynamics leads to the theory of Markov Decision Processes (MDP) that we will describe in Chapter 1. In this framework, the frustration of users is represented by a negative value (also called the cost) and the goal of the controller is to minimize the average frustration. In particular,

using this theory, one can show that under very general conditions, the acceptation policy that minimizes the average frustration has a threshold j^*: if there are less than j^* jobs in the buffer, everyone is accepted, if there are more, everyone is rejected.

The curse of dimensionality

Let us consider a situation similar to the one of Figure 2(b) but with N servers instead of one. We consider that jobs arrive in the system at rate λ and are routed to one of the servers according to some rule, also called a *policy*. All servers run independently and a job routed to a server cannot be moved after to another server. If $X_i(t)$ is the size of the queue i at time t, the state of the system is the variable $X(t) \stackrel{\text{def}}{=} (X_1(t) \dots X_N(t))$. If the routing rule is very simple, like routing a job to the queue i with some probability p_i, the behavior of $X(t)$ can be studied analytically. However, as soon as the decision rule becomes more complex, the process X becomes more and more complex and analytical methods become intractable.

A first solution to overcome this problem is to use numerical methods to solve optimization problems or compute the stationary regime. However, in general, numerical methods need at least to enumerate all of the number of possible states of the system. If there are 20 servers and if each server has $S = 20$ possible states, the possible states for the whole system is $S^N = 20^{20} \approx 10^{26}$. If we could enumerate 10^9 states per second on a single computer, it would still take more than 3500 years for a single enumeration of all the states with 10^6 computers. This makes numerical methods impossible to use for large systems and emphasizes the need for specific tools for studying large systems.

Their are multiple methods to study large systems, analytically or numerically, depending on the number N of objects.

- *Closed-form solutions*: $N \approx 3, 4$, or *product-form* systems – when the size of the system is very small or in particular cases, it is generally possible to obtain close-form solutions for most indicators. This is the best way to have insight on the behavior of a system but either this needs a very small system or specific properties leading to product form solutions (see [118] Chapter 10.6 for example).

- *Numerical methods*, $N \approx 10$ – when the state space is reasonably small, closed form solutions are no more an easy way out and one has to consider numerical methods that require to enumerate all states, also called *brute-force* algorithms.

- *Ad-hoc methods*, $N \approx 30$ – when the state space grows, brute-force solutions become intractable and one has to focus on specific solutions that use special properties of the systems. Among these methods are the two following ones:
  - *Perfect simulation* – introduced in [122] by Propp and Wilson, the perfect simulation technique provides a way to sample the steady state of a system without computing its steady-state distribution, see [93] for more details. In some cases where the system can be described by a monotone mapping, this method can be made very efficient [147]. The advantage of this approach is to give samples distributed exactly according to the stationary distribution without having to enumerate the whole state space.
  - *Stochastic automata networks* – when a system can be described by many objects with little interactions among them, it is often interesting to represent

the system by a stochastic automata network. The state space explosion is handled by a decomposition technique that reduces the complexity of their study. Specific numerical methods to study these objects are implemented for example in the PEPS software [23].

  - *Decoupling assumption and fixed-point method* – systems composed of many interacting objects are hard to study because of the dependency between objects induced by the interactions. This issue can be overpass by assuming that the objects are statistically independent, like in [134, 37]. The mean number of objects in each state is given by the fixed point of an ordinary differential equation that greatly simplifies the resolution of the system. This decoupling effect can be justify as the number of objects grows large using mean field theory (see [44] and Chapter 2) but has to be handle with care: the fact that the ODE has a unique fixed point does not guaranty the validity of the method [53].

- *Forward simulation*: $N > 50$ – when all other techniques fail, the last solution is to build a simulator of the model with all its drawbacks and advantages described earlier.

Throughout this document, we will mainly focus on another method. Instead of considering a system with $N = 10$, 20 or 50 objects and keeping track of who is in which state, we let the number of objects goes to infinity and consider the limiting behavior of this system. This technique, called *mean field*, is presented in Chapter 2. The ideas behind the mean field approach is to consider a system with a large number of objects evolving in a common environment. The behavior of these objects are not independent but are in general weakly correlated, for example at each time step, an object meets with another one at random and performs an action during this encounter. In many cases, one can show that as N grows, the behavior of the system greatly simplifies: the behavior of the system becomes deterministic and the objects become independent.

The first purpose of mean field theory is to study the behavior of such a population. We present different mean field models and the main convergence results in Chapter 2. Then these results are used to perform a study of the load balancing technique named *work-stealing* in Chapter 4. A large part of this thesis is devoted to the study of the relationship between mean field models and optimal control. These results are presented in Part II. We consider a mean field interaction system controlled by a central controller and study the limit of this controlled dynamics. There are two main questions addressed in this part. The first one pertains to the dynamics of a controlled system: given a policy, what is the limiting behavior of this controlled system. The other question is to study the limiting dynamics of the optimal controlled system: given a set of policy that can be applied, is the optimal policy of the limit also asymptotically optimal for the stochastic system? These questions are addressed in Chapters 5 to 7

Organization of the Document

This document is organized in three main parts. Part I contains the foundations of this work. The first two chapters provide an overview of the classical results that will be useful in the rest of the document. Chapter 1 presents the notion of Markov Chains and

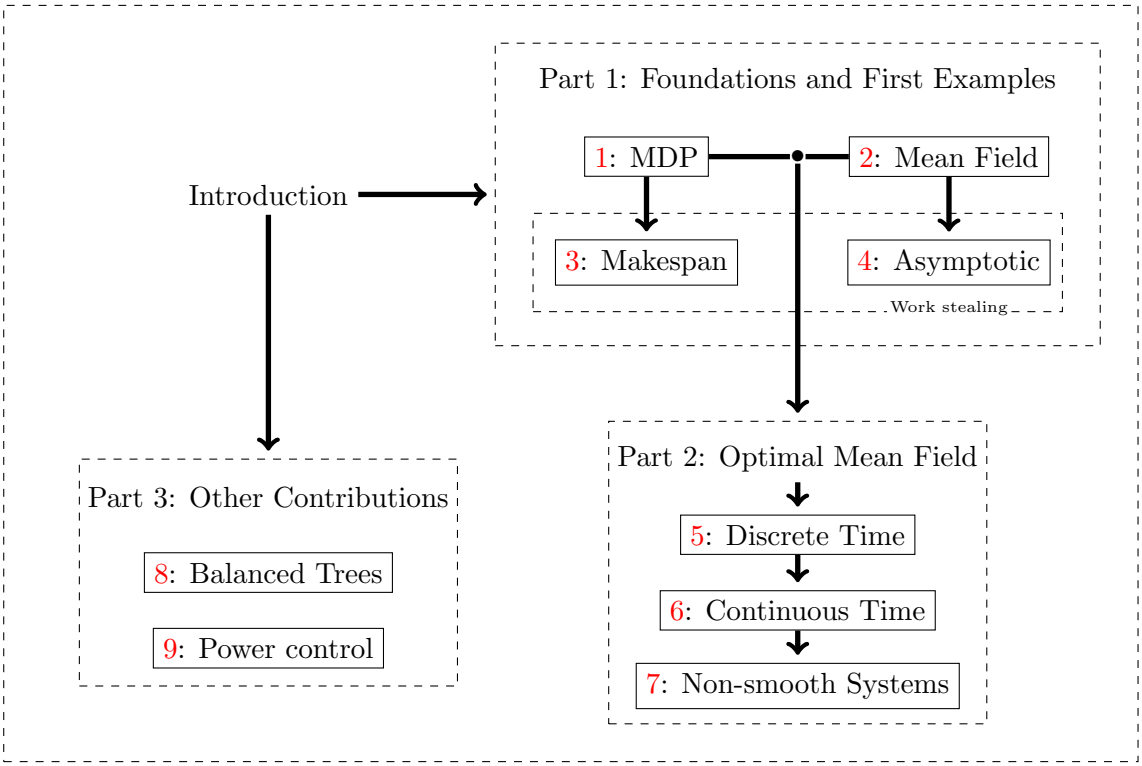


Figure 3.: Organization of the document

Markov Decision Processes and some related results. Chapter 2 is a survey on mean field models. The contributions of this thesis start with Chapters 3 and 4 in which we study a load balancing technique, namely *work stealing*. Chapter 3 is based on [140] and presents an adversary-potential method to study its transient behavior while Chapter 4, based on [70], presents a steady-state analysis.

The second part of this document focus on what we call the *optimal mean field*. We study the relationships between the optimal control of large scale systems and the optimization of deterministic systems. In Chapter 5 and Chapter 6, we show how a complicated stochastic optimization problem can be solved using its mean field limit. These two chapters are based on [69, 71, 74]. In Chapter 7, we extend the classical mean field results to systems that have some sort of discontinuity, introducing differential inclusions. This shows in particular that the behavior of any controlled system converges to its mean field controlled limit. This chapter is based on [73, 72].

Finally, Part III contains two unrelated contributions. In Chapter 8, we study how to distribute an infinite number of zeros and ones on an infinite tree as evenly as possible. We show the existence of such trees, called *balanced trees* and state their main properties. These results generalize the notion of Sturmian words that have been proved to have some optimality properties in [77, 4]. This chapter is based on [67, 68]. The last Chapter 9 presents a distributed power-control algorithm for wireless networks. We study a scenario in which each link tries to balance its delay and the power spent per packet. This work is based on [10].

Figure 3 summarizes the organization of the document and the relationship between the different chapters and provide a guide for the reader. Chapters 1 and 2 are mostly

independent. Each of these chapters leads to one side of the analysis of work-stealing. Part II depends on Chapters 1 and 2. The two chapters of Part III can be read independently.

Outline of the Document and Contributions

Chapters 1 and 2: related work

The goal of Chapters 1 and 2 is to give an overview of the mathematical background needed to read the rest of the document. The first chapter presents the concept of optimal control of Markov chains by introducing the notion of *Markov decision processes* (MDP). The theory of MDP gives the theoretical basis to study optimal control with respect to different criteria, such as expected reward over different horizons of time. We show in this chapter the main properties of optimal policies for such criteria, showing in particular that memoryless policies are dominant.

Chapter 2 contains a survey on *mean field models* and in particular of the different methods used to prove convergence to the mean field regime. The purpose of mean field models is to study systems composed by a large number N of interacting objects. In particular, we are interested in the limiting regime when the number N of objects goes to infinity. We review two different approaches. The first approach is to study the evolution of the proportion of objects in each state, which leads to systems of differential equations or discrete time dynamical system. The second approach is to study the proportion of objects that follows one or another trajectory.

Chapters 3 and 4 : performance of work stealing

A main issue when dealing with large computing systems is to distribute the load among the different resources. Work stealing is a very popular *load balancing* technique, based on the principle that when a processor has no task to execute, it steals some work from an other processors. The goal of Chapters 3 and 4 is to present two different approaches to study the performance of a system that uses *work stealing*.

In Chapter 3, we analyse the total completion time to execute a set of tasks when using *work stealing* by introducing a generic method based on an adversary-potential function. We compute an upper bound on the expected completion time as well as bounds on the deviation from the mean on a generic model (Theorem 3.1) and then apply this technique on various models (Sections 3.4 to 3.6). In particular, this technique shows that the expected completion time for executing W unit independent tasks on N processors is bounded by W/N units of time plus an additional term in $3.65 \log_2 W$. Simulations reported in Section 3.7 indicate that this bound is less than 50% greater than the actual bound.

The next Chapter 4 presents a generic model of work stealing in computational grids. Using mean field theory, we show that the analysis of its performance can be greatly simplified by considering the limiting behavior when the size of the system grows. This allows one to compute the expectation of performance functions using Corollary 4.2 as well as the distributions of these functions using a fast simulation algorithm, Theorem 4.3. Then we focus on the case where all resources are homogeneous in Section 4.4, showing in particular that work stealing is very efficient, even when the latency of steals is large. We also consider an heterogeneous case in Section 4.5: the system is made of several

clusters, and stealing within one cluster is faster than stealing between clusters. We compare different work stealing policies, showing in particular that the optimal stealing policy depends more on the load than on the geometry of the system.

Part II: mean field limits of (optimally) controlled systems.

In Chapter 2, we introduced *mean field* models and in Chapter 4, we have shown how these models can be used to simplify the performance evaluation of systems. In fact, in most of literature, mean field models have only been used to study the behavior of a system. The second part of this thesis aims at filling the gap between Chapters 1 and 2 by studying the relationships between mean field models and optimal control.

In Chapters 5 and 6, we investigate the limit behavior of Markov decision processes made of independent objects evolving in a common environment, when the number of objects N goes to infinity. We show that when the number of objects becomes large, the original stochastic optimization problem converges to a deterministic optimization problem in discrete or continuous time. The nature of the limiting regime depends on the intensity of the model $I(N)$ which is the probability for an object to change state at each time. If $I(N) = O(1)$, the limiting dynamics is in discrete time and this case is studied in Chapter 5. If $\lim_{N\to\infty} I(N) = 0$, the limiting regime is in continuous time. This is the focus of Chapter 6.

In both cases, we show that when focusing on finite-horizon reward, the optimal cost of the system converges when N grows to the optimal cost of a discrete time system that is deterministic (Theorems 5.5 and 6.4). For the discrete time limit, we further provide bounds on the speed of convergence by proving second order results that resemble central limit theorems for the cost and the state of the Markov decision process, Theorem 5.7 and 5.8. These bounds (of order $1/\sqrt{N}$) are proven to be tight in a numerical example, Section 5.4. One can even go further and get convergence of order $\sqrt{\log N}/N$ for a stochastic system made of the mean field limit and a Gaussian term. In the continuous time, speed of convergence is insured by the explicit probability bounds of Theorem 6.1. These methods are illustrated by different examples in Sections 5.4 and 6.4, like brokering problem in grid computing, investment strategies or epidemic control. For the brokering problem, several simulations with growing numbers of processors are reported in Section 5.4. They compare the performance of the optimal policy of the limit system used in the finite case with classical policies by measuring its asymptotic gain.

In the two first chapters of this part, we focus on optimal control of mean field systems. In both cases, we have shown that the optimal cost of the stochastic system converges to the optimal cost of its deterministic counterpart. However, these results do not give any insight on the convergence of the controlled stochastic system to its deterministic controlled limit. This is due to the fact that a controlled policy often shows some discontinuity and classical results on mean field convergence assume that the limiting dynamics has some regularity properties (typically a Lipschitz condition). In Chapter 7, we show that these conditions are not necessary to prove convergence to a deterministic system.

The price to pay for such general results is to introduce the notion of differential inclusion, that we recall in Section 7.2.1. Then, we show that under mild assumptions and without any condition on the smoothness of the limiting regime, the stochastic system converges to the set of solutions of a differential inclusion (Theorem 7.5). When this differential inclusion satisfies a one-sided Lipschitz condition, there exists a unique

solution of this differential inclusion and we show convergence in probability with explicit bounds (Theorem 7.7). This extends the applicability of mean field techniques to systems exhibiting threshold dynamics such as queuing systems with boundary conditions or controlled systems. These results are illustrated by several examples in Section 7.5. They also provide an easy proof for classical fluid limit regimes, as shown in Section 7.5.1.

Part III: distributing labels on trees and power-control algorithms

The last part of this document concerns two different problems.

In Chapter 8, we consider the question of how distribute an infinite number of jobs as evenly as possible on an infinite tree. The original motivation of this question was to extend Sturmian words in a multidimensional setting. Sturmian words are infinite words on a binary alphabet $\{0, 1\}$ such that the ones (and the zeros) are distributed as evenly as possible. Sturmian words are important for optimization purposes since they have been proved to have some optimality properties in [77, 4]. They have been proved to be the optimal routing sequence for a broker that can route the jobs to two servers without knowing their state [77]. These results have been extended to a situation with an arbitrary number of processors: they are no more optimal but they significantly improved classical policies [7, 130]. Chapter 8 studies an extension of Sturmian words to infinite trees with nodes labeled by $\{0, 1\}$. We introduce the notions of rational and Sturmian trees along with the definitions of (strongly) balanced trees and mechanical trees, and study the relations among them. Section 8.4 shows the existence of (strongly) balanced trees and that their coincide with mechanical trees in the irrational case (Proposition 8.8 and Theorem 8.9). Such trees also have a minimal factor complexity, hence are Sturmian (Theorem 8.15). We also give several examples illustrating the inclusion relations between these classes of trees that are summarized in Figure 8.10.

The next Chapter 9 introduces a Delay-Power Control (DPC) algorithm for wireless networking, that tries to balance out communication delay against transmitter power on each wireless link. The DPC algorithm is scalable, as each link autonomously updates its power based on the interference observed at the receiver; no cross-link communication is required. It is shown in Theorem 9.12 that DPC converges to a unique equilibrium power and some key properties are established in Section 9.4.2, concerning the nature of channel bandwidth sharing achieved by the links. Based on the DPC and FM algorithms, two protocols are developed in Section 9.5, which leverage adaptive tuning of DPC parameters. One of them is inspired by TCP and exhibits analogous behavior. The chapter primarily focuses on the theoretical underpinnings of the DPC algorithm and their practical implications for efficient protocol design. The DPC dynamics are also investigated numerically.

Publications

This thesis is based on the following papers.

- Part I, Chapters 3 and 4:

  [140] M. Tchiboukdjian, N. Gast, D. Trystram, J.-L. Roch, and J. Bernard. A tighter analysis of work stealing. *ISAAC 2010*, 2010

  [70] N. Gast and B. Gaujal. A Mean Field Model of Work Stealing in Large-Scale Systems. *SIGMETRICS'10*, 2010

- Part II, Chapters 5, 6 and 7:

  [69] N. Gast and B. Gaujal. A mean field approach for optimization in particle systems and applications. *Fourth International Conference on Performance Evaluation Methodologies and Tools, ValueTools*, 2009

  [71] N. Gast and B. Gaujal. A mean field approach for optimization in discrete time. *Accepted for publication in Discrete Event Dynamic Systems*, 2010

  [74] N. Gast, B. Gaujal, and J.Y. Le Boudec. Mean field for Markov Decision Processes: from Discrete to Continuous Optimization. *Submitted for publication*, 2010

  [73] N. Gast and B. Gaujal. Mean field limit of non-smooth systems and differential inclusions. *MAMA Workshop*, 2010

  [72] N. Gast and B. Gaujal. Mean field limit of non-smooth systems: a differential inclusion limit. *Submitted for publication*, 2010

- Part III, Chapters 8 and 9:

  [67] N. Gast and B. Gaujal. Balanced labeled trees: density, complexity and mechanicity. In *Words, 6th international conference on words*, Marseille, France, 2007

  [68] N. Gast and B. Gaujal. Infinite labeled trees: From rational to Sturmian trees. *Theoretical Computer Science*, 2009

  [10] F. Baccelli, N. Bambos, and N. Gast. Distributed delay-power control algorithms for bandwith sharing in wireless networks. *Submitted, under 2nd round of revision*, 2009

List of Notations

We recall here a list of notations that are used throughout the document.

General notations

$x \stackrel{\text{def}}{=} f(y)$ ..... Define a new notation
$\mathbb{N}, \mathbb{R}, \mathbb{Z}$ ..... set of natural numbers, real numbers, integers
$\langle x, y \rangle$ ..... Classical inner product on $\mathbb{R}^d$: $\langle x, y \rangle = \sum_{i=1}^{d} x_i y_i$
$\|x\|$ ..... Norm of x. May be L_1, L_2 or L_∞ depending on the context
$|x|$ ..... Absolute value of $x \in \mathbb{R}$
$\lfloor x \rfloor$ ..... Largest integer not greater than $x \in \mathbb{R}$
$\lceil x \rceil$ ..... Smallest integer not less than x
$\mathcal{P}(x)$ ..... Probability of an event x
$\mathcal{P}(x|y)$ ..... Probability of an event x given y
$\mathscr{P}(S)$ ..... Set of probability measures on a set S
$\mathbb{E}(X)$ ..... Expectation of a variable X
$\mathbb{E}(X \mid Y)$ ..... Expectation of a variable X given Y
$\mathrm{Var}[X]$ ..... Variance of X
$X^N \xrightarrow{\mathcal{L}} x$ ..... X^N converges in law to x
$X^N \xrightarrow{\text{a.s.}} x$ ..... X^N converges almost surely to x
$X^N \xrightarrow{\mathcal{P}} x$ ..... X^N converges in probability to x
δ_x ..... Dirac measure (gives probability 1 to x)
$\mathbf{1}_{x \geq k}$ ..... Function equal to 1 if $x \geq k$ and 0 otherwise

Mean field systems

N ..... Number of objects
$\mathcal{S}$ ..... State space of an object
t ..... Time. t might be discrete, $t \in \mathbb{N}$, or continuous, $t \in \mathbb{R}$
$X_n^N(t)$ ..... $X_n^N(t) \in \mathcal{S}$ – state of the object n at time t.
$X^N(t)$ ..... State of the system of size N at time t: $X^N(t) \stackrel{\text{def}}{=} \left(X_1^N(t) \ldots X_N^N(t)\right)$
$C^N(t)$ ..... Context for the system of size N
$M^N(t)$ ..... Empirical measure: $M^N(t) \stackrel{\text{def}}{=} \frac{1}{N} \sum_{n=1}^{N} \delta_{X_n^N(t)}$
$\mathscr{P}_N(S)$ ..... Set of possible values for $M^N(t)$
$f^N(.)$ ..... Drift of stochastic system of size N
$f(.)$ ..... Limit of the drift
$F(.)$ ..... Set-valued limit of the drift
$m(t)$ ..... (Deterministic) limit of $M^N(t)$ when N grows
$c(t)$ ..... (Deterministic) limit of $C^N(t)$ when N grows

Controlled system

$\mathcal{A}$ ..... Action set
π ..... Policy: associates an action $a \in \mathcal{A}$ to each t, M

α ..... Action function $\alpha : [0; T] \to \mathcal{A}$
π_*^N ..... Optimal policy for the system of size N
α_* ..... Optimal action function for the deterministic limit
$\mathrm{r}_t(m, c)$ or $\mathrm{r}_t(m, c, a)$ ..... Instantaneous reward at time t
$\mathbb{E}^\pi [X]$ ..... Expectation of X when the policy π is applied
$\mathcal{P}^\pi (x)$ ..... Probability of x when π is applied
T ..... Finite time horizon
$V_\pi^N(.)$ Expected reward between time 0 and T for the stochastic system under policy π
$V_*^N(.)$ ..... Optimal expected reward: $V_*^N(.) = \sup_\pi V_\pi^N(.) = V_{\pi^*}^N$
$v_\alpha(.)$ ..... Reward of the deterministic limit under action function α
$v_*(.)$ ..... Optimal reward of the deterministic limit: $v_*(.) = \sup_\alpha v_\alpha(.) = v_{\alpha^*}$

Part I.

Foundations and First Examples

Chapter 1.

Markov Chains and Markov Decision Processes

Abstract of this chapter – This chapter gives some basic definitions on Markov Chains and Markov Decision Processes. We also recall classical properties needed in the rest of the document.

We introduce the notions of discrete and continuous time Markov chains. We give classical representation and state some of their properties.

The rest of the chapter studies optimal control of Markov chains by introducing the notion of *Markov decision processes*. We define different objective functions and study the properties of optimal policies for such criteria, showing in particular that memoryless policies are dominant.

Résumé du chapitre – Ce chapitre contient une rapide présentation des notions de chaîne de Markov et de processus de décision Markovien. Nous rappelons quelques propriétés qui forment les prérequis nécessaires pour lire la suite du document.

Dans une première partie, nous rappelons brièvement la notion de chaîne de Markov, en temps continu ou discret ainsi que quelques propriétés comme leur comportement asymptotique.

Puis nous nous attardons plus longuement sur la notion de contrôle optimal de processus de Markov, à travers la notion de processus de décision Markovien. Nous étudions plusieurs objectifs. Nous nous intéressons aux propriétés des politiques optimale, et en particulier à montrer qu'il existe des politiques optimales sans mémoire pour tous ces objectifs.

1.1. Markov Processes

In all that follows, $\mathcal{S}$ will designate a set, called the *state space*. $\mathcal{S}$ may be either finite or countable. Unless specified, if $\mathcal{S}$ is finite, its elements will be denoted $\{1 \dots S\}$. We will consider random processes that takes value in $\mathcal{S}$.

1.1.1. Basic Definition

A discrete time Markov chain on a set $\mathcal{S}$ is a random process taking values in $\mathcal{S}$ such that its position at the next time step only depends on the position of the current time step. More formally, a time-homogeneous Markov chain is a random sequence $(X_0, X_1 \dots)$ such that for all time $t \in \mathbb{N}$ and all states $x, y \in \mathcal{S}$:

$$\mathcal{P}\left(X_{t+1} = y | X_0, X_1, \dots X_t = x\right) = \mathcal{P}\left(X_{t+1} = y | X_t = x\right) = P_{x,y}.$$

$P_{x,y}$ denotes the (x, y) element of a matrix P – a $S \times S$ matrix if P is finite, an infinite matrix otherwise. P is called the *transition matrix* – or *transition kernel* – of the chain. The initial state X_0 of the chain and the matrix P characterize the Markov chain.

Indeed, if the probability for the chain to be in state x at time t is denoted $\mu_t(x) = \mathcal{P}\left(X_t = x\right)$. Conditioning on the predecessors of a state $y \in \mathcal{S}$, we have that

$$\mu_{t+1} = \mu_t P = \mu_0 P^{t+1}.$$

A matrix P is a *stochastic matrix* if

1. for all $x \in \mathcal{S}$: $P_{x,y} \geq 0$.
2. for all $x, y \in \mathcal{S}$: $\sum_{y \in \mathcal{S}} P_{x,y} = 1$.

A transition matrix of a Markov chain is a stochastic matrix. Reciprocally, any stochastic matrix is a transition matrix of some Markov chain.

In the following, when we talk about a Markov chain, we will only be interested in the law of the process (*i.e.* the law of $(X_t)_{t \geq 0}$ and not a particular representation ($(X_t)_t$ itself). Again, this law is uniquely determined by the initial distribution and the transition matrix.

1.1.2. Representation

There exist multiple ways to represent Markov chains. We present here two possible representations: *random mapping* and *stochastic automata*.

A *random mapping* representation of a transition matrix P is a function $f : \mathcal{S} \times \Lambda \to \mathcal{S}$ along with a sequence of *i.i.d.* random variables Z such that:

$$\mathcal{P}\left(f(x, Z_t) = y\right) = P_{x,y}.$$

If X_o has distribution μ_0, then the sequence $(X_0, X_1 \dots)$ is defined by

$$X_{t+1} = f(X_t, Z_t)$$

is a Markov Chain with transition matrix P. It should be clear that every transition matrix P has a random mapping representation. As $\mathcal{S}$ is at most countable, it can be

numbered $\mathcal{S} = \{1, 2, 3 \ldots\}$. Let Z_t be *i.i.d.* random variable uniformly distributed on $[0; 1]$. For any $x, y \in \mathcal{S}$ and $u \in [\sum_{z<y} p_{xz}; \sum_{z \leq y} p_{xz}]$, we define

$$f(x, u) = y.$$

It should be clear that $\mathcal{P}(X_{t+1} = y | X_0, X_1, \ldots X_t = x) = \mathcal{P}(X_{t+1} = y | X_t = x) = P_{x,y}$ and that (f, Z) is a random mapping representation of the Markov chain X.

Unlike transition matrices, this representation is not unique. In some cases, choosing the best random mapping representation is very important and often non trivial (*e.g.* for perfect simulation [147]). Random mapping are crucial for simulation but can also be a convenient way to represent the chain itself.

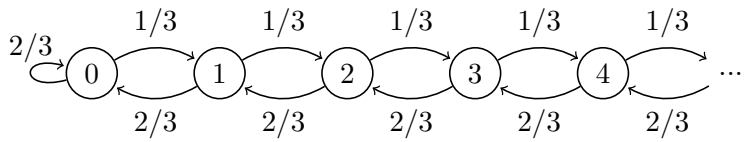


Figure 1.1.: The transition graph of a Markov chain. Vertices are labeled by the states $\mathcal{S}$. An edges (x, y) is labeled by probability p_{xy}.

Yet an other way to represent a Markov chain is to consider its transition graph. If $\mathcal{S}$ is the state space of the Markov chain, we consider a graph whose vertices are labeled by $\mathcal{S}$. There exists a edge between two vertices x and y if $P_{x,y} > 0$. In that case, the edge (x, y) is labeled by $P_{x,y}$. The graph representation is often the most simple way to represent a Markov chain. An example is drawn on Figure 1.1. It is also used to define some basic properties of Markov chains like irreducibility or aperiodicity.

1.1.3. Irreducibility, Aperiodicity and Stationary Distributions

A Markov chain with transition matrix P is said to be irreducible if for any two states $x, y \in \mathcal{S}$, there exists t such that $P^t_{x,y} > 0$. This is equivalent to say that the graph of the Markov chain is strongly connected: for any states x, y, there exists a path with positive probability from x to y.

For a state x, we define $T(x) = \{t / P^t_{x,x} > 0\}$. On the graph, $T(x)$ is the set of the cycle lengths starting in x. If the chain is irreducible, then for all x, y, the greatest common divisor (gcd) of $T(x)$ and $T(y)$ is the same. We call it the *period* of the chain: $p \stackrel{\text{def}}{=} \gcd(T(x)) = \gcd(T(y))$. If the period of the chain is 1, the chain is said to be *aperiodic*.

A measure π is said to be an invariant measure of a chain P if for all i: $\pi_i \geq 0$ and

$$\pi P = \pi. \tag{1.1}$$

If $\sum_{x \in \mathcal{S}} \pi(x) = 1$, π is a probability distribution and is called a *stationary distribution*. Irreducibility and aperiodicity plays a central role when studying the asymptotic behavior of Markov chains, as expressed the following (classical) result on Markov chain.

Theorem 1.1 (see *e.g.* [136] Section 9.8)**.** *If $\mathcal{S}$ is finite and P is irreducible, there exists a unique stationary distribution π which is the unique solution of (1.1) such that for all i: $\pi_i > 0$ and $\sum_i \pi_i = 1$.*

If $\mathcal{S}$ is infinite and P is irreducible, there exists a unique invariant measure π up to a multiplicative factor. Moreover, if there exists a solution of (1.1) such that $\sum_i \pi_i < \infty$, then there exists a unique stationary distribution.

If P is irreducible and admits a stationary distribution, then for any initial distribution μ_0:

$$\lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} \mu_0 P^t = \pi. \tag{1.2}$$

Moreover, if P is aperiodic, then:

$$\lim \mu_0 P^t = \pi. \tag{1.3}$$

Because of the convergence results (1.2) and (1.3), a Markov chain that admits a stationary distribution is said to be *stable*. In particular, if the state space $\mathcal{S}$ is finite, the system is always stable.

1.1.4. Continuous Time Markov Chains

When representing a real system, it is often more realistic to consider continuous time model rather than discrete time. For this, a standard tool is to consider *continuous time Markov chains*. Continuous time Markov chains combine both the advantages of having a large expressive power as well as being simple to study.

We consider random processes $(X_t)_{t\geq 0}$ taking values in a discrete space $\mathcal{S}$ with $t \in [0;\infty)$. A process $(X_t)_{t\geq 0}$ is called a homogeneous continuous time Markov chain if for all t, all $x \neq y \in \mathcal{S}$ and all $t_1 < t_2 < \cdots < t_n < t$ and $x_1 \ldots x_n \in \mathcal{S}$,

$$\mathcal{P}\left(X_{t+h} = y | X_t = x, \forall i : X_{t_i} = x_i\right) = \mathcal{P}\left(X_{t+h} = y | X_t = x\right) = q_{xy}h + o(h) \tag{1.4}$$

where $o(h)$ is a function such that $\lim_{h\to 0} o(h)/h = 0$. Since (1.4) represents a probability, this is positive and $q_{xy} \geq 0$.

If we define $q_{xx} \stackrel{\text{def}}{=} -\sum_{y\neq x} q_{xy}$, then:

$$\mathcal{P}\left(X_{t+h} = x | X_t = x\right) = 1 + q_{ii}h + o(h).$$

The matrix Q which (x, y) coordinate is equal to q_{xy} is called the *transition rate matrix* or *infinitesimal generator*. For $x \neq y$, q_{xy} is called the transition rate from x to y. As in the discrete time case, the transition rate matrix and the initial distribution uniquely determine the law of the process $(X_t)_{t\geq 0}$.

An equivalent construction of a continuous time Markov chain is by studying the sequence of jumps of the Markov chain. Let us define by induction a sequence of random time J_k by $J_0 \stackrel{\text{def}}{=} 0$ and $J_{k+1} \stackrel{\text{def}}{=} \min\{t > 0 : X_{J_k+t} \neq X_{J_k}\}$. J_k is the time between of the $k-1$th and the kth *jump* of (X_t). We recall the following characterization:

Theorem 1.2. *Let $(X_t)_{t\geq 0}$ be a continuous time Markov process with transition rate Q and let assume that $\sup_{x,y} Q_{x,y} < \infty$. Let us denote by J_k its sequence of jumps and $Y_k \stackrel{\text{def}}{=} X(J_0 + J_1 + \cdots + J_k)$, then*

1. *Y_k is a discrete time Markov chain of transition probability*

$$\begin{aligned} P_{xy} &= -\tfrac{q_{xy}}{q_{xx}} \text{if x} \neq \text{ y;} \\ P_{xx} &= 0. \end{aligned}$$

2. *Given* $Y_k = x$, J_{k+1} *is exponentially distributed of parameter* $q_x \stackrel{\text{def}}{=} -q_{xx} \geq 0$ *and is independent of* $Y_0 \dots Y_{k-1}$ *and* $J_1 \dots J_k$.

As for the discrete time case, it is often convenient to represent a continuous time Markov chain by its transition graph. The set of vertices is $\mathcal{S}$ and there is an edge labeled by q_{xy} between two vertices $x \neq y \in \mathcal{S}$ if $q_{xy} > 0$. There are no loops (*i.e.*, edges between a vertex x and itself). The Markov chain is irreducible if its graph is strongly connected. There is no concept of periodicity for continuous time Markov chains.

In the discrete case, a distribution is stationary of the Markov chain is a probability vector π solution of the equation $\pi P = \pi$. In the continuous case, this equation is replaced by $\pi Q = 0$. It is rather straightforward to show that a solution of $\pi Q = 0$ is indeed a stationary distribution (*i.e.*, if X_0 has law π then X_t has law π for all $t > 0$).

A basic yet fundamental example of continuous time Markov process is given by birth and death processes. The state space of a birth and death process is $\mathcal{S} \stackrel{\text{def}}{=} \{0, 1 \dots\}$. If the chain is in state k, it goes with rate λ_k in $k+1$ and with rate μ_k in $k-1$. Its transition graph is given by Figure 1.2.

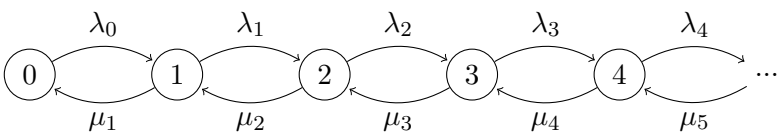


Figure 1.2.: The transition graph of a birth and death process.

Birth and death process are commonly used in queuing theory, $X_t \in \mathcal{S}$ representing the number of jobs in the system at time t. If for all k, $\lambda_k = \lambda$ and $\mu_k = \mu$ the process model an $M/M/1$ queue: jobs arrive following a Poisson process and the service time of a job is exponentially distributed of parameter μ. In that case, the system is stable (*i.e.* has a stationary distribution) *iff* $\mu < \lambda$. Queuing system based on birth and death processes have been intensively studied in the literature, see [95, 96, 11] for example. If all $\mu_k = 0$, (X_t) is increasing in t and is called a pure birth process. Pure-birth process are used in particular to model the traffic arriving in a system. In particular, if for all k, $\lambda_k = \lambda$, (X_t) is a Poisson process of intensity λ.

1.2. Markov Decision Processes: Basic Concepts

Markov processes are a simple and elegant way to represent the evolution of a complicated systems. However, in many situations, one not only wants to study the behavior of the system but also to control the system in order to improve its performance. The goal of this part on Markov decision processes is to formalize these notions from a mathematical point of view.

1.2.1. Definition

A discrete time Markov decision process is described by four elements: states, actions, transition probabilities and reward. A decision maker (also called a controller) is faced with the problem of influencing the dynamics of the stochastic system. The goal of the

decision maker is to choose a sequence of actions in order to optimize some predefined performance criterion.

At time step t, the state of the system at time t is denoted by $X_t \in \mathcal{S}$. If at time t the system is in state $x \in \mathcal{S}$, the decision maker chooses an action from a set A_x. Let $\mathcal{A} = \bigcup_{x \in \mathcal{S}} A_x$. $\mathcal{S}$ is called the *state space* and $\mathcal{A}$ is called the *action set*. In general, $\mathcal{S}$ and $\mathcal{A}$ can be arbitrary sets.

Suppose that at time t, the system is in state x and the decision maker chooses action a. Then the following events occur:

1. The decision maker receives a reward $r_t(x, a)$.

2. The state of the system at time $t + 1$ becomes $y \in \mathcal{S}$ with probability $P_{a,xy}$.

$r_t(x, a)$ is a real-valued function defined for $x \in \mathcal{S}$ and $a \in A_x$. $r_t(x, a)$ may be positive or negative. When $r_t(x, a)$ is negative, $r_t(x, a)$ is often called a *cost*. For an action a, we denote by P_a the matrix composed by all values $P_{a,xy}$. P_a is a stochastic matrix. Therefore, a Markov decision process may be viewed as a family of transition kernels P_a along with a reward function.

1.2.2. Decision Rules and Policies

A *decision rule* is a procedure that specifies the action to be selected at each time step. The decision taken at time t can not depend on the future of the process. A decision rule may depend on all the history of the process but we may also consider decision rules that only depend on the current state of the process.

Let π_t be the decision rule for the time step t. A decision rule is said to be *deterministic* if π_t is a deterministic function: it specifies exactly one action to be taken. It is said to be randomized if π_t specifies a distribution of probabilities on all possible actions. π_t is said to be *Markovian* if it only depends to the current state of the process (*i.e.* the value of X_t) and history-dependent if it depends on all the history of the process (*i.e.* the values of $X_1 \dots X_t$). A decision rule that is deterministic and Markovian is a function $\pi_t : \mathcal{S} \to \mathcal{A}$.

Deterministic Markovian decision rules are the simplest decision rules to consider. Therefore a fundamental question in the Markov decision process theory is to determine whether deterministic Markovian policies are *dominant*[1] for a given performance criterion.

A *policy* is a sequence of decision rules: $\pi = (\pi_1, \pi_2 \dots)$ that specifies the decision rules to be applied at each time step. In the case where all decision rules are equal: for all t: $\pi_t = \pi_1$, this policy is called *stationary*. Stationary policies are fundamental when considering infinite horizon criteria.

1.2.3. Objectives

There are multiple objectives that can be considered. A first issue is to choose the time horizon. For example, we may want to optimize the total reward over a finite horizon, the total reward over an infinite horizon, the average reward over an infinite horizon,... A second and big issue when choosing a reward criterion is that given some policy, the reward obtained by the decision maker is a random function that depends on

[1] A set of policies π is dominant on all policies if for every policies, there exists a policy of π that outperforms it.

the stochastic evolution of the process. Therefore, to compare two policies, one needs to decide a method to compare the two rewards: for example, is it better to compare the expectation or the probability to exceed some threshold?

In the following, we will focus on three reward criteria:

1. *Finite-horizon expected reward* – the decision maker wants to maximize the expected reward over an interval $[0; T]$. If X_t designates the state of the system at time t and A_t the action taken by the decision maker, the expected reward of a policy π is:

$$v_\pi^T(x) = \mathbb{E}^\pi \left[\sum_{t=1}^{T-1} r_t(X_t, A_t) + r_T(X_T) | X_1 = x \right],$$

   where the expectation is taken over all values of X_t and A_t with $X_1 = x$ and under policy π. The goal of the controller is to find π^* such that $v_{\pi^*}(x) = \sup_\pi v_\pi(x)$ or at least to get as close as possible to this supremum.

2. *Discounted expected reward* – in this situation, δ designates a discount factor, $0 \leq \delta < 1$. The reward at time t is multiplied by δ^t. The discount factor δ represents the deprecation of the value: an euro of today is worth more than an euro of tomorrow. In that case, we assume that the reward and the probability of transition are invariant over time and that the reward is bounded. The expected discounted of a policy π is:

$$v_\pi^\delta(x) = \mathbb{E}^\pi \left[\sum_{t=1}^{\infty} \delta^{t-1} r(X_t, A_t) | X_1 = x \right].$$

   Again, the goal of the controller is to find a policy maximizing this reward.

3. *Average reward* – For a policy π, and using the notation $v_\pi^T(x)$ to denote the reward of policy π over horizon $[O; T]$, the average reward starting from x is

$$g_\pi(x) = \lim_{T \to \infty} \frac{1}{T} v_\pi^T(x).$$

   The average reward criterion is often more complicated to study than the two previous criteria. A first issue is that the limit of its definition may not exist. In that case, one can also consider $\limsup$ or $\liminf$.

1.3. Optimal policies

In this section, we state several theorems that characterize the optimal policies for the three criteria presented before. These theorems are often constructive and provide ways to compute optimal policies. The beginning of this section shows that memoryless policies (also called Markovian policies) are dominant. Then some results for the three criteria above are presented.

1.3.1. Optimality of Markovian policies

In this section, we recall that for any history-dependent policy π and each initial state s, there exists an equivalent Markovian policy.

Let $\Pi = (\pi_1, \pi_2, \dots)$ be a history-dependent policy. A Markov decision process can be described by two variables (X_t, A_t) where X_t denotes the state at time t and A_t the action taken by the controller at time t.

Theorem 1.3 (Theorem 5.5.1 of [123])**.** *Let Π be a history-dependent policy. For each state $x \in \mathcal{S}$, there exists a Markovian policy Π' such that*

$$\mathcal{P}^{\Pi}\left(X_t = j, A_t = a | X_1 = x\right) = \mathcal{P}^{\Pi'}\left(X_t = j, A_t = a | X_1 = x\right), \tag{1.5}$$

where $\mathcal{P}^{\Pi}$ denotes the probability when applying policy Π.

This theorem is fundamental in the theory of Markov decision processes since it allows to restrict the class of interesting policies to the class of Markov policies, see Sections 5.2.1 and 6.2.2). Because of the elementary nature of the proof and the importance of this results, we give a proof of the theorem, following the one of Theorem 5.5.1 of [123].

Proof. For each $y \in \mathcal{S}$ and each $a \in \mathcal{A}$, we define $q^t_{y,a}$ by:

$$q^t_{y,a} = \mathcal{P}^{\Pi}\left(A_t = a | X_t = y, X_1 = x\right).$$

We define a randomized Markovian policy $\Pi' = (\pi'_1, \dots)$. At time step t, the probability of choosing action a if the state is y is $\mathcal{P}^{\Pi'}\left(A_t = a | X_t = y\right) = q^t_{y,a}$. We show (1.5) by induction on t. It clearly holds for $t = 1$. Assume that (1.5) holds for $t' < t$. Then:

$$\begin{aligned}
\mathcal{P}^{\Pi}\left(X_t = y | X_1 = x\right) &= \sum_{z \in \mathcal{S}} \sum_{a \in A} \mathcal{P}^{\Pi}\left(X_{t-1} = z, A_{t-1} = a | X_1 = s\right) P_{a,z,y} \\
&= \sum_{z \in \mathcal{S}} \sum_{a \in A} \mathcal{P}^{\Pi'}\left(X_{t-1} = z, A_{t-1} = a | X_1 = s\right) P_{a,z,y} \\
&= \mathcal{P}^{\Pi'}\left(X_t = y | X_1 = x\right).
\end{aligned}$$

where the second inequality comes from the induction hypothesis. It follows that

$$\begin{aligned}
\mathcal{P}^{\Pi}\left(X_t = y, A_t = a | X_1 = s\right) &= \mathcal{P}^{\Pi}\left(A_t = a | X_t = y, X_1 = s\right) \mathcal{P}^{\Pi}\left(X_t = y | X_1 = s\right) \\
&= q^t_{y,s} \mathcal{P}^{\Pi'}\left(X_t = y | X_1 = s\right) \\
&= \mathcal{P}^{\Pi'}\left(X_t = y, A_t = a | X_1 = s\right).
\end{aligned}$$

□

In particular, this shows that for any value of X_1, and any history-dependent policy Π, there exists a randomized Markovian policy Π' such that

$$\mathcal{P}^{\Pi'}\left(X_t = x, A_t = a\right) = \mathcal{P}^{\Pi}\left(X_t = x, A_t = a\right).$$

In particular, the expected reward gained at time t is the same for Π and Π'.

$$\mathbb{E}^{\Pi}\left[r_t(X_t, A_t)\right] = \mathbb{E}^{\Pi'}\left[r_t(X_t, A_t)\right].$$

Theorem 1.4 (Theorem 5.5.3 of [123])**.** *For each initial state x and each history-dependent policy Π, there exists a Markovian policy Π' such that*

- $v^T_{\pi}(x) = v^T_{\pi'}(x)$,
- $v^{\delta}_{\pi}(x) = v^{\delta}_{\pi'}(x)$,
- $g_{\pi}(x) = g_{\pi'}(x)$.

1.3.2. Finite-horizon reward

In this section, we focus on finite-horizon reward. We give some basic properties of optimal policies in that case. In particular, we give an algorithm to compute an optimal deterministic Markovian policy. For a more complete description, we refer to [123], Chapter 4.

Recall that the reward of a policy π starting from $X_1 = x$ is

$$v_\pi^T(x) = \mathbb{E}^\pi \left[\sum_{t=1}^{T-1} r_t(X_t, A_t) + r_T(X_T) | X_1 = x \right],$$

where $r_T(.)$ is a final cost that depends on where the state of the process at the end of the time-horizon.

The goal of the decision maker is to find the best history-dependent policy to maximize its reward. As we have just seen in the previous part, we may restrict to Markovian policies. The first question that arises is the evaluation of such a policy.

Let $\pi = (\pi_1 \pi_2 \dots)$ be a (randomized) Markovian policy. For a state x and an action a, we denote by $\pi_t(x, a)$ the probability for the controller to take action a knowing that the state is x. Let us call $u_\pi^{t\dots T}(x)$ the expected reward starting from $X_t = x$:

$$u_\pi^{t\dots T}(x) = \mathbb{E}^\pi \left[\sum_{s=t}^{T} r_s(X_s, A_s) + r_T(X_T) | X_t = x \right].$$

By linearity of the expectation, $u_\pi^{t\dots T}(x)$ can be decomposed in:

$$\begin{aligned} u_\pi^{t\dots T}(x) &= \mathbb{E}^\pi \left[r_t(X_t, A_t) | X_t = x \right] + \mathbb{E}^\pi \left[\sum_{s=t+1}^{T} r_s(X_s, A_s) + r_T(X_T) | X_t = x \right] \\ &= \sum_a r_t(x, a) \mathcal{P}\left(A_t = a\right) + \mathbb{E}^\pi \left[\sum_{s=t+1}^{T} r_s(X_s, A_s) + r_T(X_T) | X_{t+1} = y \right] \\ &\quad \cdot \mathcal{P}^\pi \left(X_{t+1} = y | X_t = x, A_t = a\right) \mathcal{P}^\pi \left(A_t = a | X_t = x\right) \\ &= \sum_{a \in \mathcal{A}} r_t(x, a) \pi_t(x, a) + \sum_{a \in \mathcal{A}} \sum_{y \in \mathcal{S}} u_\pi^{t+1\dots T}(y) \pi_t(x, a) P_{a,xy}. \end{aligned} \tag{1.6}$$

This provides a backward method to compute the expected reward of a policy π, implemented by Algorithm 1.1.

Require: Description of the MDP, Markovian policy π, initial state $x_1 \in \mathcal{S}$.
Ensure: $u_\pi^{0\dots T}(x_1) = v_\pi^T(x_1)$.
For all $x \in \mathcal{S}$, set $u_\pi^T(x) = r_T(x)$.
for $t = T - 1$ to 1 **do**
  for $x \in \mathcal{S}$ **do**
    $u_\pi^{t\dots T}(x) \leftarrow \sum_{a \in \mathcal{A}} r_t(x, a) \pi_t(x, a) + \sum_{a \in \mathcal{A}} \sum_{y \in \mathcal{S}} u_\pi^{t+1\dots T}(y) \pi_t(x, a) P_{a,xy}$.
  end for
end for
return $u_\pi^{1\dots T}(x_1)$.

Algorithm 1.1: Finite-horizon policy evaluation

This backward induction is the key of the resolution of finite-horizon Markov decision processes. It provides a way to compute the optimal reward as well as the optimal policy. Indeed, let $u_*^{t\dots T}(x) = \sup_\pi u_\pi^{t\dots T}(x)$. By Equation (1.6), we have:

$$\begin{aligned}
u_*^{t\dots T}(x) &= \sup_\pi u_\pi^{t\dots T}(x) \\
&= \sup_\pi \left(\sum_{a\in\mathcal{A}} r_t(x,a)\pi_t(x,a) + \sum_{a\in\mathcal{A}}\sum_{y\in\mathcal{S}} u_\pi^{t+1\dots T}(y)\pi_t(x,a)P_{a,xy} \right) \\
&= \sup_{\pi_t} \left(r_t(x,a)\pi_t(x,a) + \sum_{a\in\mathcal{A}}\sum_{y\in\mathcal{S}} \sup_{\pi=\pi_{t+1}\dots\pi_T} u_\pi^{t+1\dots T}(y)\pi_t(x,a)P_{a,xy} \right) \\
&= \sup_{\pi_t} \left(r_t(x,a)\pi_t(x,a) + \sum_{a\in\mathcal{A}}\sum_{y\in\mathcal{S}} u_*^{t+1\dots T}(y)\pi_t(x,a)P_{a,xy} \right) \\
&= \sup_a \left(r_t(x,a) + \sum_{y\in\mathcal{S}} u_*^{t+1\dots T}(y)P_{a,xy} \right) \qquad (1.7)
\end{aligned}$$

Equation (1.7) is often referred as the optimality equation. It provide an efficient algorithm to compute by a backward induction the optimal reward as well as the optimal policy of the system. In particular, if the state space and action space are finite (of cardinal S and A), then Algorithm 1.2 computes an optimal policy and an optimal cost in time $O(S \cdot A \cdot T)$. Using this equation, we directly have the following result.

Theorem 1.5. *(i) For any $\epsilon > 0$, the exists an ϵ-optimal policy that is deterministic and Markovian.*

(ii) If for all s, t, there exists $a \in \mathcal{A}$ such that the supremum in (1.7) is attained for a. Then, there exists an optimal policy that is deterministic and Markovian.

(iii) In particular, if $\mathcal{S}$ is finite or countable and $\mathcal{A}$ is compact, there exists a deterministic Markovian policy, computed by Algorithm 1.2.

The proof of this proposition is straightforward from Equation (1.7). For more details, see [123], Sections 4.4 and 4.6.

Ensure: $u_*^{1\dots T}(x) = \sup_\pi v_\pi^T(x) = v_{\pi_*}^T(x)$.
For all $x \in \mathcal{S}$, set $u_*^T(x) = r_T(x)$.
for $t = T-1$ to 1 **do**
  for $x \in \mathcal{S}$ **do**
    $u_*^{t\dots T}(x) \leftarrow \max_{a\in\mathcal{A}} \left(r_t(x,a) + \sum_{y\in\mathcal{S}} u_*^{t+1\dots T}(y)P_{a,xy} \right)$.
    $\pi_t^*(x) \leftarrow \arg\max_{a\in\mathcal{A}} \left(r_t(x,a) + \sum_{y\in\mathcal{S}} u_*^{t+1\dots T}(y)P_{a,xy} \right)$.
  end for
end for
return $u_*^{1\dots T}(x_1)$, π.

Algorithm 1.2: Finite-horizon optimal policy

1.3.3. Discounted reward

The discounted reward problem is very similar to the finite-horizon reward. Therefore, we will only emphasize the differences. Again, we focus on randomized Markovian policy because of Proposition 1.4. In all that follows, we assume that reward and probability transitions are time-homogeneous. We further assume that the reward is bounded and that the discount factor δ is in $[0;1)$. Therefore, up to adding a constant to all reward, without any loss of generality, we assume that the reward is positive.

Let $\pi = (\pi_1\pi_2\dots)$ be a Markovian policy. The expected discounted reward starting from point x is defined by:

$$v_\pi^\delta(x) = \mathbb{E}^\pi\left[\sum_{t=1}^{\infty}\delta^{t-1}r(X_t,A_t)|X_1=x\right].$$

Again, this expression can be decomposed in:

$$\begin{aligned} v_\pi^\delta(x) &= \mathbb{E}^\pi\left[r(X_1,A_1)|X_1=x\right] + \delta\mathbb{E}^\pi\left[\sum_{t=2}^{\infty}\delta^{t-2}r(X_t,A_t)|X_1=x\right] \\ &= \sum_{a\in\mathcal{A}}\Big(r(x,a)+\delta P_{a,xy}v_{\pi'}^\delta(y)\Big)\pi_1(x,a), \end{aligned} \tag{1.8}$$

where $\pi' = (\pi_2\pi_3\dots)$.

Let $\pi = (\pi_1\pi_2\dots)$ be a policy. Using a vector notation (v_π^δ designates the vector of the values $v_\pi^\delta(x)$, R_{π_1} the vector of $R(x) = \sum_{a\in\mathcal{A}} r(x,a)\pi_1(x,a)$ and $(P_{\pi_1})_{xy} = \sum_{a\in\mathcal{A}} P_{a,xy}\pi_1(x,a)$), (1.8) can be rewritten

$$v_{\pi_1\pi_2\dots}^\delta = R_{\pi_1} + \delta P_{\pi_1}v_{\pi_2\pi_3\dots}^\delta.$$

It should be clear that for any policy π, since $0\le\delta<1$ and the reward is bounded, v_π^δ is equal to:

$$v_\pi^\delta(x) = \sum_{t=1}^{\infty}\delta^{t-1}P_\pi^{t-1}R_{\pi_t}$$

where $P_\pi^0 = Id$ and $P_\pi^t = P_{\pi_t}P_\pi^{t-1}$ for $t\ge 1$.

This equation suggests an optimality equation similar to Equation (1.7) of the finite-horizon case. In the following, we will see that this is indeed the case. For a bounded value function $v:\mathcal{S}\to\mathbb{R}^+$, we define the value function Lv by

$$Lv(x) = \sup_{\pi\in\mathrm{MR}}(R_\pi+\delta P_\pi v) = \sup_{a\in\mathcal{A}}(R_a+\delta P_a v)$$

where MR designates all the possible randomized Markovian decision rules, *i.e.* functions $\pi(.,.)$ such that for all $x\in\mathcal{S}$ $\sum_{a\in\mathcal{A}}\pi(x,a)=1$ with $\pi(x,a)\ge 0$. The supremum has to be understood component-wise (*i.e.* for each x). The supremum on $\pi\in\mathrm{MR}$ is clearly equal to the supremum on $a\in\mathcal{A}$. $L: v\to Lv$ is a nonlinear operator on the set of bounded value functions.

It should be clear that since the reward is bounded, there exists a bounded value function v_*^δ such that for all x: $v_*^\delta(x)\stackrel{\text{def}}{=}\sup_\pi v_\pi^\delta(x)$. The following results link v_*^δ and the operator L. The order $\le$ has to be understood component-wise: for two functions v and v', we say that $v\le v'$ if for all x: $v(x)\le v'(x)$.

Theorem 1.6. *(i) If $v \geq Lv$, then $v \geq v_*^\delta$;*

(ii) If $v \leq Lv$, then $v \leq v_*^\delta$;

(iii) If $Lv = v$, then $v = v_*^\delta$.

The previous proposition gives basic properties that should satisfy the optimal reward of the system. The next proposition shows there exists a unique value function such that

$$v = Lv = \sup_{\pi \in \mathrm{MR}} \left(R_\pi + \delta P_\pi v\right). \tag{1.9}$$

Equation (1.9) is called the Bellman's equation. This equation also allows one to compute ϵ-optimal policy by iterating this equation.

Theorem 1.7. *(i) There exists a unique v_* satisfying Bellman's equation $v_* = Lv_*$.*

(ii) For each v_0, the iteration $v_{n+1} = Lv_n$ converges to v_*.

(iii) If the action set $\mathcal{A}$ is either finite or compact, then there exists an optimal stationary deterministic Markovian policy.

This theorem proves that the algorithm that consists at iterating Equation (1.9) converges to the solution of Bellman's equation.

1.3.4. Average reward

In this part, we assume that the reward function is time-homogeneous and bounded. The case of average reward is more complicated than the case of finite-horizon and discounted reward and there exist many different cases depending on the geometry of the problem (*i.e.* properties on the matrices P_a). Therefore, we will only present some basic results and examples.

Recall that for a policy π, the average reward is the quantity g_π where

$$g_\pi(x) = \lim_{T \to \infty} \frac{1}{T} v_\pi^T(x).$$

Since this limit may not exist, we denote

$$g_\pi^+(x) = \limsup_{T \to \infty} \frac{1}{T} v_\pi^T(x) \quad \text{and} \quad g_\pi^-(x) = \liminf_{T \to \infty} \frac{1}{T} v_\pi^T(x).$$

A policy π_* is said to be *average optimal* if for all policy π, $g_{\pi_*}^-(x) \geq g_\pi^+(x)$. π_* is *lim-sup average optimal* if $g_{\pi^*}^+(s) \geq g_\pi^+(s)$. π_* is *lim-inf average optimal* if $g_{\pi_*}^-(s) \geq g_\pi^-(s)$. Note that in general, there may not exist average optimal policies although there exist lim-inf and lim-sup average-optimal policies, [123] Example 8.1.2.

Let us fix a Markovian stationary policy π (recall that a stationary policy means that the same decision rule is applied at each time step). For the two previous criteria (finite-horizon and discounted reward), we ignored the chain structure under the policy π since we were mainly interested in the transient behavior of the process.

In the case of average reward, one can not ignore this fact and the result on Markov decision process will strongly depend on the nature of the chain. For example, if the MDP is unichain – meaning that every deterministic stationary policy generate a Markov

Chain with one single recurrent class plus a finite set of transient states – one can show that when S and A are finite, the average reward exists and does not depend on the initial state, see [123], Section 8.4.1. Moreover, there exists a unique $g \in \mathbb{R}$ such that there exists $h : \mathcal{S} \to \mathbb{R}$ with

$$h(x) = \max_{a \in \mathcal{A}} \left(r(x,a) - g + \sum_{y \in \mathcal{S}} P_{a,xy} h(y) \right).$$

In the equation above, g is the average reward. The equation above defines h up to an additive constant. There exists a constant c such that the expected reward between time 1 and t starting form x is

$$v_*^T(x) = gT + h(x) + c + o(1).$$

However, in that case, the iteration $v_{n+1} = Lv_n$ does not necessarily converges and often oscillates or diverges.

If the MDP is multi-chain – *i.e.* there exists a stationary policy that generates a Markov chain with two recurrent classes – results are more complex. In that case, the optimal reward depends on the initial state $g(s)$ and satisfies the two equations

$$g(x) = \max_{a \in \mathcal{A}} \sum_y P_{a,xy} g(y) \quad \text{and} \quad h(x) = \max_{a \in B_x} r(x,a) - g(x) + \sum_{y \in \mathcal{S}} P_{a,xy} h(y).$$

where $B_x = \{a' \in \mathcal{A} : g(x) = \sum_y P_{a,xy} g(y)\}$ (see [123], Section 9.1.1). Here, $g(x)$ represents the average reward starting from x while $h(x)$ is the asymptotic difference between $v_*^T(x)$ and $Tg(x)$. In that case, building a policy iteration to compute optimal policy is again feasible but more complex.

1.4. Algorithmic Issues: the Curse of Dimensionality

Markov models are very popular in the performance evaluation's community because they are often very simple to analyze. In particular they have two interesting features.

- There exist many theoretical tools to show the existence and uniqueness either of the transient behavior or of the stationary behavior. The conditions are often simple to verify.
- These theorems are often constructive and provide algorithms to compute performance indicators or solve optimal control problems.

However, as pointed out in the introduction, page xxx, Markov models suffer from one main drawback: the size of the state space. Consider for example that we want to model a small network of computers. Say that there are 100 computers and that each of them have 10 possible states. In that case, the total number of states of the system is 10^{100}! Therefore, an algorithm that needs to search all the states cannot be used in practice.

This problem is even more important in the case of Markov decision problems since the algorithm are often at least linear in the size of the state space and often more complicated (like PSPACE-complete [149]) . In the following of this document, we will see how to tackle this problem by studying a limiting system when the size of the state space of the system goes to infinity.

1.5. Bibliographical notes

Markov chains have been first introduced by Markov in [111] and named after him, see [14] for an history on the early development on Markov chains. For a more modern view on Markov chains and in particular the speed of convergence to its stationary regime, we refer to [108] and [3].

Markov processes play a central in the theoretical study of computer networks, in particular via the development of queuing theory. See [92] or [95, 96] for classical references on this subject or [11] for more recent work.

Markov decision processes have been introduced by Bellman in [15] which introduced the principle of dynamic programming. The mathematical foundation were established in [58] and [39]. More recent references include [35, 36]. The book [123] gives a comprehensive reference on MDP.

Chapter 2.

A Survey on Mean Field Convergence

Abstract of this chapter – Despite the great expressive power of Markov chain, Markov models often suffer from a problem of dimensionality: the state space needed to describe a complex system is too big. In this chapter, we study a way to overcome this difficulty by introducing *mean field models*.

The purpose of mean field models is to study systems composed by a large number of interacting objects. When the role of the objects exhibits some symmetry, one can show that when the number of objects goes to infinity, the system converges to a deterministic limit.

In this chapter, we review two different approaches and compare the different methods of proof. The first approach is to study the evolution of the proportion of objects in each state, which leads to dynamical systems of differential equations or discrete time dynamical system. The second approach is to study the proportion of objects that follows one or an other trajectory.

Résumé du chapitre – Dans le chapitre précédant, nous avons vu que malgré une grande puissance d'expression, les modèles reposants sur les chaînes de Markov souffrent souvent d'un problème d'explosion combinatoire. Dans ce chapitre, nous étions une façon de contourner le problème en considérant les *modèles champ moyen*.

Les modèles champs moyen s'intéressent au comportement de système composés d'un grand nombre d'objets interagissant et plus particulièrement aux propriétés de limite quand le nombre de ces objets tend vers l'infini.

Nous présentons deux approches différentes. La première est de s'intéresser à l'évolution du nombre moyens d'objets dans chaque état. La deuxième est de considérer l'espace des trajectoires et de regarder la proportion d'objets empruntant telle ou telle trajectoire.

2.1. Mean field models

In this chapter, we are interested in studying the behavior of a system composed by a large number N of objects evolving in a common environment (ofter called the *context*). These objects may be similar objects or can be decomposed in different classes of objects, each class being composed by a large number of objects. The behavior of each object is not independent of the others. It depends on the context and often also depends on the interaction with other objects.

Mean-field models arise in many situations, ranging from population dynamics to models of computing systems. As there is no commonly admitted definition of what is called a mean field model, we start by giving some examples and refer the reader to Sections 2.2 and 2.3 for a precise definitions of some mean field models.

2.1.1. Examples

We give here some examples of classical mean field models of the literature.

Epidemic models – our first example concerns the model of the evolution of an epidemic in the population, studied for example in [62] or [106]. The population is composed by a fixed number of persons N. Each person can be either susceptible, infected or recovered. A susceptible person becomes infected when meeting someone who is infected. A infected can either recover or infect a susceptible. After some time, a recovered person becomes susceptible again.

If the meeting among the different people are uniform, the probability of each event to occur only depend on the proportion of person in each state. When N is large but finite, the behavior of such system is quite complicated. However, studying the limit when N goes to infinity makes it simpler to study.

Queuing systems – let us consider a network of N processors, representing for example a cluster of computers or a computational grid. There are jobs arriving in the system that are routed according to some rules to each processor. Each processor treats one job at a given time and stores other jobs in its queue. The state of a processor is the number of jobs a processor has in backlog.

In practice, the management of such systems relies on load balancing techniques which aim at distributing the work among the different resources. Without load balancing, every job arriving in the system is rooted to one processor at random and stays with this processor until the end of its computation. A simple way to improve this situation is for every arriving job to choose a small number of processor at random and sends the jobs to the processor with the smallest load. This has been proved to be very efficient in practice [115].

The dynamics of such systems is complex because of the interactions between the different processors caused by the rooting policy. However, mean field theory can greatly reduce its complexity when studying the limit $N \to \infty$ [80, 137]. We will also see how mean field theory can be used to study another load balancing technique [114, 70], namely *work stealing*, in Chapter 4.

Medium access control – medium access control (MAC) is used in wireless or self-organizing networks to guarantee the access of a shared communication channel to a large number of users without using a central controller. Performance studies of such systems often assume independence properties among users or retransmission of packets

[37]. Studying limiting properties of the system when the number of connecting users is large allows one to overpass and justify these limitations [43].

Congestion control – Yet an other big issue when modeling communication network is to study congestion control mechanisms, such as TCP. Consider for example, a system with N computers that are all sending some packets through the same bottleneck server. As long as the queue of the bottleneck is not full, all connexions will increase linearly their sending rate. However, when its queue become full, the bottleneck starts dropping packets and the connexions decreases their sending rate exponentially. Such a mechanism is called *additive increase, multiplicative decrease* and is implemented in a lot of implementations of TCP.

In this situation, the different connexions are only dependent through the shared bottleneck which makes it a good candidate for mean field techniques. The limiting regime is then much simpler to study that the original model with a large number of connexions [13, 106]. In particular, people have shown that even when using the variant RED (random early detection), a main issue of this mechanism is that in some situation, it leads to oscillations in the system [143, 12].

2.1.2. Reducing the State Space: Point-wise and Path-space Convergence

In this section, we present the different objects of study of mean field theory that will be developed in Sections 2.2 and 2.3.

We consider a collection of N objects. Objects are numbered from 1 to N. Each object lives in a state space $\mathcal{S}$ and the state of the object i at time t is represented by a variable $X_i^N(t) \in \mathcal{S}$. In some mean field model, the state of the system is composed by the state of the objects in the system but also by the state of a shared resource, see Section 2.2.3. In this section, in order to simplify the presentation of the concept, we omit this resource and consider that the state of the system is described as:

$$\mathcal{X}^N(t) \stackrel{\text{def}}{=} \left(X_1^N(t) \dots X_N^N(t)\right) \in \mathcal{S}^N.$$

The basic assumption of our model is that the process $(\mathcal{X}^N(t))_{t\geq 0}$ is a Markov chain on the state space $\mathcal{S}^N$ (either in continuous time or discrete time). If $\mathcal{S}$ is a finite set of size S, the cardinal of $\mathcal{S}^N$ is S^N. For example, if we consider a network of 20 computers, each with 20 possible states representing the state of their memory for example, this leads to a state space of size $20^{20} \approx 10^{26}$. Just to cover the complete state space, a computer that explores 10^9 states per second would take about 3 billions years to cover the entire state space. This raises the need of a method to simplify the study of such systems.

In all that follows, we will consider systems in which the objects are only distinguishable through their state. This means that if we permute the objects $1 \dots N$, the law of the corresponding process is the permutation of the law of the original process $(\mathcal{X}^N)$. In particular, this means that the identity "i" of the ith object does not contain any information and that this information is contained in its state.

This leads to think that the process to study is not the process containing the information about which object is in which state but rather to consider the proportion of objects in each state. We define the process $M^N(t)$, called the *empirical measure* of the process $\mathcal{X}^N(t)$ by

$$M^N(t) \stackrel{\text{def}}{=} \frac{1}{N} \sum_{i=1}^{N} \delta_{X_i^N(t)},$$

where δ_x is the Dirac measure of the singleton x: δ_x is a probability measure such that $\delta(\{x\}) = 1$. $M^N(t)$ is a measure on $\mathcal{S}$ and for each $x \in \mathcal{S}$, $(M^N(t))(x)$ is the proportion of objects that are in state x:

$$(M^N(t))(x) = \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}_{\{X_i^N(t)=x\}}.$$

For each t, $M^N(t) \in \mathscr{P}(\mathcal{S})$, the set of probability measures on $\mathcal{S}$ and $t \mapsto M^N(t)$ is a càdlàg (*continue à droite, limite à gauche* which means right-continuous with left limits) random process taking values in $\mathscr{P}(\mathcal{S})$. Thus, M^N can be seen as a random variable taking values in $\mathbb{D}(\mathbb{R}^+, \mathscr{P}(\mathcal{S}))$, the Skorokhod space, see Section 2.3.1 for some definitions and properties of the Skorokhod space. A first object of study of mean field theory is to study the convergence of the random variable M^N and in particular of its law which is in $\mathscr{P}(\mathbb{D}(\mathbb{R}^+, \mathscr{P}(\mathcal{S})))$. This point of view is developed in Section 2.2.

We will see in Section 2.2 that under very general conditions, the process $M^N(.)$ converges to a deterministic process $m : \mathbb{R}^+ \to \mathscr{P}(\mathcal{S})$ and that m satisfies a set of differential equations that can be easily deduced from the original model, which simplifies greatly the study of the performance of this systems. However, when studying the dynamics of the empirical measure $(M^N(t))_t$, one loses all information about the trajectory of a particular object $(X_1^N(t))_t$. A natural question would be study the convergence of the trajectory of one object $(X_1^N(t))_t$.

Instead of studying $M^N(t)$, the empirical measure at time t, we consider the empirical measure of the trajectories Q^N, where

$$Q^N \stackrel{\text{def}}{=} \sum_{i=1}^{N} \delta_{X_i^N}.$$

A trajectory X_i^N is a càdlàg function from $\mathbb{R}^+$ to $\mathcal{S}$: $X_i^N \in \mathbb{D}(\mathbb{R}^+, \mathcal{S})$. Therefore, Q^N is a (random) measure on $\mathscr{P}(\mathbb{D}(\mathbb{R}^+, \mathcal{S}))$. Studying the convergence of the random measure $Q^N \in \mathscr{P}(\mathscr{P}(\mathbb{D}(\mathbb{R}^+, \mathcal{S})))$ is much more complicated than studying the process M^N but leads to stronger convergence results. In particular, the convergence of Q^N implies the convergence of M^N but also leads to properties of the trajectories of tuples of objects. This ideas will be developed in Section 2.3. We will introduce the notion of chaoticity of a sequence of a sequence of measure and see how this can be used to show that Q^N converges to some δ_Q where Q is the unique solution of a martingale problem.

2.1.3. Comparison of the two Approaches

Convergence results for path-space are stronger than the ones concerning point-wise processes. However, in the rest of the document and in particular in Part II, we will mainly focus on point-wise convergence. There are multiple reasons for this.

The main reason is the nature of the limit. While in the point-wise case the limit is described by a "simple" ordinary differential equation or a dynamical system, the limiting behavior of Q^N is only determined in terms of martingale problem. Therefore, the limiting behavior of $M^N(t)$ is more explicit and much simpler to study, either analytically or numerically. The limiting behavior of Q^N makes it more interesting from a theoretical point of view (for example to show asymptotic independence of objects like in [43]). We will see in Section 4.3.1 that fast simulation can be used to study the trajectory of one object in the point-wise framework.

The second reason comes from the centralized nature of the results presented in Part II where we present the links between optimal control and mean field limits. In this part, a centralized controller tries to optimize a cost that depends on $M^N(t)$. A path-space correspondence of these results would be to study a scenario where a controller would like to optimize the trajectory of each objects. This is similar to game-theoretic approaches in which each object tries to optimize its own behavior like in [142] or [105].

The last but not least reason is due to the generality of the results. Results on point-wise convergence shows that under very mild assumptions (mainly some continuity of the transition parameters and some sort of asymptotic independence) the limiting behavior can be approximated by a deterministic one. Moreover, the limiting dynamics can be described explicitly starting from the initial system. On the opposite side, path-space results need stronger assumptions and explicit results can only be obtained for very particular cases. For example, the extension to a system with more than one class of objects is not straightforward [81, 82] in the path-space case whereas there are no differences in the point-wise case.

2.2. Some Results on Point-wise Convergence

Throughout this section, we present some classical model on the convergence of the process $t \mapsto M^N(t) \stackrel{\text{def}}{=} \frac{1}{N}\sum_{i=1}^N \delta_{X_i^N(t)}$. In particular, under simple conditions, $M^N(t)$ converges to a deterministic process m, that can either be a continuous time or discrete time process. When m is a continuous time process, as in Sections 2.2.1 or 2.2.2, m satisfies a system of differential equations $\dot{m} = f(m)$. When the limit m is in discrete time, as in Section 2.2.3, m is a dynamical satisfying $m(t+1) = f(m(t))$.

2.2.1. Density Dependent Population Processes

In this section, we recall some definitions and convergence results of *density dependent population processes*. For a more complete description, the reader is referred to [103] chapter 8 or [62] chapter 11.

Let M^N be a continuous Markov chain on $\frac{1}{N}\mathbb{N}^d$ $(d \geq 1)$ for $N \geq 1$. M^N is called a density dependent population process if there exists a finite number of transitions, say $\mathcal{L} \subset \mathbb{N}^d$, such that for each $\ell \in \mathcal{L}$, the rate of transition from M^N to $M^N + \ell/N$ is $N\beta_\ell(M^N) \geq 0$, where $\beta_\ell(.)$ does not depend on N. Here, M^N is the empirical measure corresponding to a population living in a finite state space $\{1 \dots d\}$. The ith component of the vector M^N represents the proportion of persons in state i.

Let us call F the function $F(m) = \sum_{\ell \in \mathcal{L}} \ell\beta_\ell(m)$ (if the sum is well defined) and let us consider the following ordinary differential equation:

$$\begin{cases} m(0) &= m_0 \\ \dot{m}(t) &= F(m(t)) \end{cases}.$$

Theorem 2.1 below shows that the stochastic process $M^N(t)$ converges to the deterministic system $m(t)$. In all that follows, $\|.\|$ denotes the classical L_2 norm on $\mathbb{R}^d$: $\|x\| = \sqrt{\sum_{i=1}^d |x_i|^2}$.

Theorem 2.1 ([62], chapter 11)**.** *Assume that for all compact $K \subset \mathbb{R}^d$, F is Lipschitz on E and $\sum_{\ell \in \mathcal{L}} \|\ell\| \sup_{m \in K} \beta_\ell(m) < \infty$. If $\lim_{N\to\infty} M^N(0) = m_0$ in probability, then*

for all $T > 0$:

$$\lim_{N\to\infty} \sup_{t\leq T} \left\|M^N(t) - m(t)\right\| = 0 \quad \text{in probability,}$$

uniformly in the initial condition.

The proof of this result is very classical and can be extended to many other situations to obtain convergence results on M^N. The main idea is to write

$$\begin{aligned}\left\|M^N(t) - m(t)\right\| &\leq \left\|M^N(0) - m(0)\right\| + \left\|M^N(t) - M^N(0) - \int_0^t F(M^N(s))ds\right\| \\ &\quad + \left\|\int_0^t F(M^N(s))ds - \int_0^t F(m(s))ds\right\|.\end{aligned}$$

Then the first part is bounded using a martingale argument and we use Grönwall's lemma to conclude. The method used to prove this result is the basis of the proof of Chapters 6 and 7 and in particular Theorems 6.1 and 7.7. Therefore, we recall the proof here. This proof is inspired by the one of [62], chapter 11.

Proof. Following Theorem 4.1 of Chapter 6 of [62], $M^N(t)$ is the unique solution of

$$M^N(t) = M^N(0) + \sum_{\ell\in L} \ell Y_\ell \left(N \int_0^t \beta_\ell \left(M^N(s)\right) ds\right),$$

where Y_ℓ is a collection of independent Poisson processes with rate 1.

We will assume that $\sum_{\ell\in\mathcal{L}} \|\ell\| \sup_{m\in\mathbb{R}^d} \beta_\ell(m) < \infty$ and that F is lipschitz of constant L on all $\mathbb{R}^d$ and we define $\bar{\beta}_\ell = \sup_{m\in\mathbb{R}^d} \beta_\ell(m)$. To prove the general case, one has to show that $M^N(t)$ is almost surely bounded.

Let us define $\epsilon^N(T)$ by:

$$\begin{aligned}\epsilon^N(T) &\stackrel{\text{def}}{=} \sup_{t\leq T} \left\|M^N(t) - M^N(0) - \int_0^t F(M^N(s))ds\right\| \\ &\leq \sup_{t\leq T} \sum_{\ell\in L} |\ell| \left\|\frac{1}{N} Y_\ell \left(N \int_0^t \beta_\ell \left(M^N(s)\right) ds\right) - \int_0^t \beta_\ell(M^N(s))ds\right\|.\end{aligned}$$

The random process $\sum_{\ell\in L} |\ell| \left\|\frac{1}{N} Y_\ell \left(N \int_0^t \beta_\ell \left(M^N(s)\right) ds\right) - \int_0^t \beta_\ell(M^N(s))ds\right\|$ is a positive submartingale.

Therefore by Doob's Martingale inequality:

$$\begin{aligned}\mathcal{P}\left(\epsilon^N(T) \geq \epsilon\right) &\leq \frac{1}{\epsilon}\mathbb{E}\left(\sum_{\ell\in L} |\ell| \left\|\frac{1}{N} Y_\ell \left(N \int_0^T \beta_\ell \left(M^N(s)\right) ds\right) - \int_0^T \beta_\ell(M^N(s))ds\right\|\right) \\ &= \frac{1}{\epsilon}\sum_{\ell\in L} |\ell|\, \mathbb{E}\left(\left\|\frac{1}{N} Y_\ell \left(N \int_0^T \beta_\ell \left(M^N(s)\right) ds\right) - \int_0^T \beta_\ell(M^N(s))ds\right\|\right) \\ &\leq \frac{1}{\epsilon}\sum_{\ell\in L} |\ell|\, \mathbb{E}\left(\left|\frac{1}{N} Y_\ell(N\bar{\beta}_\ell T) - \bar{\beta}_\ell T\right|\right).\end{aligned}$$

Since $\mathbb{E}\left(\left|\frac{1}{N} Y_\ell(N\bar{\beta}_\ell t) - \bar{\beta}_\ell t\right|\right) \leq 2\bar{\beta}_\ell t$ and $\sum_\ell |\ell|\, \bar{\beta}_\ell < \infty$, the dominated convergence theorem shows that $\mathcal{P}\left(\epsilon^N(T) > \epsilon\right) \to 0$. Moreover, since F is Lipschitz of constant L, we have:

$$\left\|M^N(t) - m(t)\right\| \leq \left\|M^N(0) - m(0)\right\| + \epsilon^N(t) + \int_0^t L \left\|M^N(s) - m(t)\right\| ds.$$

Grönwall's lemma concludes the proof. □

This theorem does not provide any insight on the speed of convergence. In fact, if $\sum_\ell |\ell| \sqrt{\beta_l} < \infty$, the proof can be adapted to show that there exists a function f independent of N such that $\mathcal{P}\left(\sup_{0\le t\le T} \left\|M^N(t) - m(t)\right\| \ge \epsilon\right) \le \frac{1}{\sqrt{N}} f(\epsilon)$. Without this condition, one can show that for all function $b(N)$ such that $\lim_{N\to\infty} b(N) = 0$, there exists a set $\{\beta_\ell\}_{\ell\in\mathcal{L}}$ such that

$$b(N) = o\left(\mathcal{P}\left(\sup_{0\le t\le T} \left\|M^N(t) - m(t)\right\| \ge \epsilon\right)\right).$$

Moreover, Kurtz has proved second order results for the previous convergence, showing that the gap between $M^N(t)$ and $m(t)$ is of order $\sqrt{N}$. Let $V^N(t) \stackrel{\text{def}}{=} \sqrt{N}(M^N(t) - m(t))$, and $V(t) \stackrel{\text{def}}{=} V(0) + U(t) + \int_0^t \partial F(M(s))V(s)ds$, where $U(t)$ is a time inhomogeneous Brownian motion and ∂F denotes the Jacobian matrix of F. For a sequence of random variables X^N, we say that X^N converges weakly to a random variable X, denoted $X^N \xrightarrow{\text{weak}} X$ if for all continuous bounded function h, $\lim_{N\to\infty} \mathbb{E}\left(h(X^N)\right) = \mathbb{E}\left(h(X)\right)$. Using this notation, we have:

Theorem 2.2 ([62], Chapter 11)**.** *Assume that for all compact set E: F is Lipschitz on E, $\sum_\ell \left\|\ell^2\right\| \sum_{m\in E} \beta_\ell(m) < \infty$, and that F is continuously differentiable and $V^N(0) \xrightarrow{weak} V(0)$. Then for all t, $\sqrt{n}(M^N - m) \xrightarrow{weak} V$.*

This result indicates that for all fixed t, the gap between $M^N(t)$ and $x(t)$ behaves like $\frac{1}{\sqrt{N}}G$, where G is a Gaussian random variable.

Steady state convergence

Other interesting results of convergence concern the behavior of the steady state of the system. In the following, we will assume that $M^N(t)$ in bounded for all t. Assume for example that the stochastic process with N objects has an invariant measure, say π^N. A natural question is whether π^N converges (or not) to a fixed point of F and whether second order results exist in that case. In general, convergence for the steady state only holds under several restrictions.

Let $B_f(m_0)$ be the set of accumulation points of the trajectory of the differential equation starting in m_0: $B_f(m_0) \stackrel{\text{def}}{=} \text{Acc}\{m(t) : t \ge 0 \wedge \dot{m} = f(m) \wedge m(0) = m_0\}$ and $B_f \stackrel{\text{def}}{=} \overline{\cup_{m_0} B_f(m_0)}$. B_f is the closure of the set of accumulation points of the ODE $\dot{m} = F(m)$. In particular, B_F contains the invariant point of the ODE as well as the possible limit cycles. There is no general method to study B_F and it is in particular very hard to show the existence or the non-existence of limit cycles. The following proposition links the stationary distributions π^N and B_F.

Proposition 2.3. *The support of any limit point (for the weak convergence) of the stationary distributions π^N is included in the closure B_F of the accumulation points of the solutions of the equation $\dot{m} = F(m)$, for all possible initial conditions.*

The proof presented here is inspired by classical proofs for stochastic approximation (see Corollary 3.2 of [16] for example). It is based on the Poincaré recurrence theorem.

Proof. Let h be a continuous bounded function and $\delta > 0$. For a measure π^N, we denote $\int_m h(m)\pi^N(dm)$ the expectation of $h(M)$ if the distribution of M is π^N. $\mathbb{E}_m(M^N(t))$ denotes the expectation over the trajectories of $M^N(t)$ starting in $M^N(0) = m$. Moreover, the system of differential equations $\dot{y} = F(y)$ starting in $y(0) = m$ has a unique solution. Its value at time t is denoted $\Phi_t(m)$.

Let π^N be an invariant measure of M^N and π a limit point of π^N. Since π^N is an invariant measure of M^N:

$$\int_m \mathbb{E}_m(h(M^N(t)))\pi^N(dm) = \int_m h(m)\pi^N(dm).$$

Since E is compact, h is uniformly continuous and there exists $\epsilon > 0$ such that $\|m - y\| < \epsilon$ implies $\|h(m) - h(y)\| < \delta$ and

$$\begin{aligned}\mathbb{E}_m(\big|h(M^N(t)) - h(\Phi_t(M^N(0)))\big|)\\ \leq \delta + \epsilon \|h\| \mathcal{P}\left(\left\|M^N(t) - \Phi_t(M^N(0))\right\| > \epsilon\right),\end{aligned}$$

where $\|h\| \stackrel{\text{def}}{=} \sup_m \|h(m)\|$.

Proposition 2.1 shows that $\lim_{N\to\infty} \mathcal{P}\left(\left\|M^N(t) - \Phi_t(M^N(0))\right\| > \epsilon\right) = 0$ which shows that:

$$\lim_{N\to\infty} \left| \int_m \mathbb{E}_m(h(M^N(t))\pi^N(dm)) - \int_m h(\Phi_t(M^N(0)))\pi^N(dm) \right| \leq \delta.$$

Using the fact that π^N goes weakly to π and since h and δ are arbitrary, this shows that π is an invariant measure of Φ_t. Let us call H the support of π and B the set of accumulation point of Φ_t. H is a closed invariant set and π is an invariant measure for Φ. By the Poincaré recurrence theorem, $\pi(B) = 1$, which shows that $B \subset H$. □

An immediate corollary is the case where the solutions of the differential equation all converge to a single global attractor.

Corollary 2.4. *If F has a unique stationary point m^* to which all trajectories converge, then the stationary measures π^N concentrate around m^* as N goes to infinity:*

$$\lim_{N\to\infty} \pi^N \xrightarrow{\text{weak}} \delta_{m^*},$$

where δ_{m^*} is the Dirac measure in m^*.

Proof. Since M^N is bounded, the measures π^N have a compact support. Thus, the sequence π^N is tight. By Proposition 2.3, it converges to δ_{x^*}. □

In the case where a linear Lyapounov function exists, second order results for the steady state behavior exist. Norman [119] shows that under technical assumptions and if there exists a scalar product $\langle\cdot\rangle$ such that $\langle m - m^*, F(m)\rangle < 0$ and $\langle m, F'(m^*)m\rangle < 0$, then $\sqrt{N}(\pi^N - m^*)$ converges to a Gaussian variable. Unfortunately, this theorem is not very useful in practice since the construction of such a scalar product is complicated in general.

2.2.2. Stochastic Approximation Method

Stochastic models either involve continuous time or discrete time models. When modeling a real system, it may sometimes be more natural to start from a discrete or a continuous time model. In this section, we show how discrete time models can also lead to continuous time limits and in particular, how to use results on stochastic approximation algorithms to study convergence of a large class of mean field models.

A stochastic approximation algorithm is a discrete random process $x_n \in \mathbb{R}^d$ that can be written

$$x_{n+1} = x_n + a_{n+1}(F(x_n) + U_{n+1}), \tag{2.1}$$

where $F(x_n)$ is a deterministic function, U_{n+1} is a random variable of mean 0 and a_n a "small" step-size. One can think of Equation (2.1) of a Euler discretization of the ODE

$$\dot{x} = F(x), \tag{2.2}$$

plus a random error term U_{n+1}.

The theory of stochastic approximation algorithm focuses on studying the relationship between (2.1) and (2.2). The original goal of stochastic approximation was to provide numerical algorithms to compute the solutions of an equation $f(x) = 0$ [125] by building a sequence x_n that would converge to one the solution. During the years, this method has been extended to study numerous problems such as learning algorithms, adaptive control or signal processing. We refer the reader in particular to [104] that contains many examples as well convergence results and [17] that contains many theoretical results on the asymptotic behavior of (2.1) when the step-size a_n vanishes as n goes to infinity.

The stochastic approximation method is a simple and powerful way to prove mean field convergence. To each system of size N corresponds a stochastic approximation algorithm $(X_n^N)_{n\geq 0}$ whose step-size is constant: $a_n = I(N)$. In particular, the constant-step size algorithm has been studied in [18] from a stochastic algorithm point of view. Apart from mean field models, this has also applications in game-theory [21]. In the following, we show how to apply this result to mean-field models, following the presentation of [20].

We consider of model similar to the one of [20]. The system is composed of N *objects* evolving in a finite state space $\mathcal{S} = \{1 \dots S\}$. Time is discrete and the state of object n at time step k is denoted $X_n^N(k)$. In [20], the system also contains a resource but we omit this detail here in order to simplify the presentation. The state of the global system at time k is $\mathcal{X}^N(k) \stackrel{\text{def}}{=} (X_1^N(k) \dots X_N^N(k))$. We denote by $M^N(k)$ the empirical measure associated with the N objects:

$$M^N(k) \stackrel{\text{def}}{=} \frac{1}{N} \sum_{n=1}^{N} \delta_{X_n^N(k)}.$$

Since an object has S possible states, $M^N(k)$ can be represented by a vector with S components, its ith component being the proportion of objects in state i.

The system $(M^N(k))_k$ is assumed to be a Markov chain. In particular, this is true if the evolution of $(\mathcal{X}^N)$ is invariant by any permutation of the N objects.

If at time step k, the system is in state $M^N(k) = m$, the expected difference between $M^N(k+1)$ and $M^N(k)$ is called the *drift* and is denoted $f^N(m)$:

$$f^N(m) \stackrel{\text{def}}{=} \mathbb{E}\left(M^N(k+1) - M^N(k) | M^N(k) = m\right).$$

We assume that as N grows, the drift vanishes with speed $I(N)$ where $I(N) > 0$ is called the *intensity*. The intensity of the model corresponds to the length of a time step (the step-size of the stochastic approximation algorithm). The means that $\lim_{N\to\infty} I(N) = 0$ and that there exists a function f such that:

$$\lim_{N\to\infty} \frac{f^N(m)}{I(N)} = f(m) \quad \text{uniformly in } m. \tag{2.3}$$

We define the normalized process $\bar{M}^N(t)$ by for all $k \in \mathbb{N}$: $\bar{M}^N(kI(N)) = M^N(k)$ and $M^N(.)$ is linear between $kI(N)$ and $(k+1)I(N)$.

The evolution of the system $M^N(t)$ can be written:

$$M^N(k+1) = M^N(k) + I(N)\left(\frac{f^N(M^N(k))}{I(N)} + \frac{1}{I(N)}\left(M^N(k+1) - M^N(k) - f^N(M^N(k))\right)\right)$$

Using that $f^N(m)/I(N) \to f(m)$ and $\mathbb{E}\left(M^N(k+1) - M^N(k) - f^N(M^N(k))|M^N(k)\right) = 0$, this equation defines a stochastic approximation algorithm as in (2.1). Using ideas of the stochastic approximation theory, the following result is shown in [20], Theorem 1.

Proposition 2.5. *If f is Lipschitz and if the number of objects that change their state during one time step is bounded in expectation by $NI_1(N)$ and in second moment by $N^2 I(N) I_2(N)$ with $I_1(N) \to 0$ and $I_2(N) \to 0$, then for all $T > 0$ there exists constants $C_1(T)$ and $C_2(T)$ such that*

$$\mathcal{P}\left(\sup_{0\le t\le T} \left\|M^N(t) - m(t)\right\| \ge \epsilon + C_1(T)\left(1 + \left\|M^N(0) - m(0)\right\|\right)\right) \le \frac{C_2(T)}{\epsilon^2},$$

where $m(t)$ is the unique solution of the ODE $\dot{m} = f(m)$ starting in $m(0)$.

The proof of this result is very similar to the one of Proposition 2.1 for density dependent population processes, using a martingale argument to bound the stochastic part and Grönwall's lemma to conclude. Similar results pertaining to asymptotic behavior as Proposition 2.3 and 2.4 are also provided in [20].

2.2.3. Discrete Time Limits

In the first model presented in Section 2.2.1, the original system is in continuous time as well as its limiting regime. In the second model, 2.2.2, the original model is in discrete time and its limit in continuous time. In this section, we give an example of a discrete time model converging to a discrete time (deterministic) dynamical system.

We consider a similar model as the one in [106]. The system is composed of N objects. Each object evolves in a finite state space $\mathcal{S} = \{1, \dots, S\}$. The state of the nth object at time $t \in \mathbb{N}$ is denoted $X_n^N(t)$. We call $M^N(t)$ the empirical measure of the collection of objects (X_n^N). $M^N(t)$ is a vector with S components and the ith component of $M^N(t)$ is

$$M_i^N(t) \stackrel{\text{def}}{=} N^{-1} \sum_{n=1}^{N} \mathbf{1}_{X_n^N(t)=i},$$

the proportion of objects in state i. The set of possible values for M^N is the set of probability measures p on $\{1 \dots S\}$, such that $Np(i) \in \mathbb{N}$ for all $i \in \mathcal{S}$, denoted by

$\mathscr{P}_N(S)$. For each N, $\mathscr{P}_N(\mathcal{S})$ is a finite set. When N goes to infinity, it converges (for the Hausdorff metric) to $\mathscr{P}(\mathcal{S})$ the set of probability measures on $\mathcal{S}$.

The system of objects evolves depending on their common environment. We call $C^N(t) \in \mathbb{R}^d$ the context of the environment. Its evolution depends on the empirical measure $M^N(t)$, itself at the previous time slot:

$$C^N(t+1) = g(C^N(t), M^N(t+1)),$$

where $g : \mathscr{P}(\mathcal{S}) \times \mathbb{R}^d \to \mathbb{R}^d$ is a continuous deterministic function. Moreover, there exists a transition probability kernel $K(C)$ such that the probability that knowing that the context is in state C at time t, the object n goes from state i to state the j is $K_{i,j}(C)$, where

$$\mathcal{P}\left(X_n^N(t+1) = j | X_n^N(t) = i, C^N(t) = C\right) = K_{i,j}(C).$$

Moreover, we assume that the evolution of the objects are independent once C is fixed. The assumption of independence of the users is a rather common assumption in mean field models [106]. However other papers [20, 45] have shown that similar results can be obtained using asymptotic independence only. The kernel K is not assumed to be irreducible. This allows for several classes of objects interacting under context C, as in [41, 106].

This model has been studied in [106]. In particular, the authors gave results on the asymptotic behavior of $M^N(.), C^N(.)$ such as the following result.

Theorem 2.6 (Theorem 4.1 of [106])**.** *Assume that $M^N(0), C^N(0)$ converge almost surely to deterministic $m(0), c(0)$ as N goes to infinity. Then for all t, $M^N(t), C^N(t)$ converge almost surely to $m(t), c(t)$ where $m(t), c(t)$ are defined by:*

$$\begin{aligned} m(t+1) &= m(t)K(c(t)) \\ c(t+1) &= g\left(c(t), m(t+1)\right). \end{aligned}$$

Authors also give a fast-simulation algorithm that shows that the dynamics of the different objects can be decoupled as N goes to infinity.

Although this system could seem unnatural because we force the behavior of the N objects to take place during the same time step and be independent, it has many applications for example in the study of reputations systems [106] or congestion avoidance [143, 106]. The main advantage of this model is that the limiting process is very simple to describe and to study, at least numerically.

2.3. Path-space convergence

The purpose of mean field model is to study the behavior of systems composed of a large number of objects, when the number of these objects goes to infinity. In the previous section, we have seen that if $\mathcal{X}^N(t) \stackrel{\text{def}}{=} (X_1^N(t) \dots X_N^N(t))$ denotes the state of the system of particles at time t, there exist various models to study the convergence of the process $t \mapsto M^N(t) \stackrel{\text{def}}{=} \frac{1}{N}\sum_{i=1}^N \delta_{X_i^N(t)}$ which is a $\mathbb{D}(\mathbb{R}^+, \mathscr{P}(\mathcal{S}))$ random variable. In particular, we have seen that under mild conditions, the limiting behavior of $M^N(.)$ is deterministic and can be described by simple differential equations or discrete time dynamical systems.

In this section, we will focus on a more complex object, which is

$$Q^N \stackrel{\text{def}}{=} \frac{1}{N}\sum_{i=1}^{N} \delta_{X_i^N}.$$

Q^N is no longer a random process but a measure on the path space of the processes X_i^N. Since each X_i^N is a $\mathbb{D}(\mathbb{R}^+, \mathcal{S})$-valued process, Q^N is a $\mathscr{P}(\mathbb{D}(\mathbb{R}^+, \mathcal{S}))$-valued process. The behavior of Q^N is much complicated to study than the one of $(M^N(t))_t$ and convergence results on Q^N are stronger than their analogue on M^N.

In this section, we will present a method to prove convergence of Q^N to a Dirac measure δ_Q where Q is a deterministic process. This method is based on *propagation of chaos* and has been developed in [138] and further investigated in [80]. We will not go deep into the details since in the rest of the document we will focus on convergence properties of $(M^N(t))_t$ and the reader is referred to [80] for a more complete description of the method.

We first recall some basic results on the space $\mathscr{P}(\mathbb{D}(\mathbb{R}^+, \mathcal{S}))$ in Section 2.3.1. Then, we recall some results on propagation of chaos and martingale problem. Finally, we describe the generic method to prove mean-field convergence in Section 2.3.4.

2.3.1. Skorokhod Space and Weak-Convergence Topology

In this section, we recall some basic properties about the Skorokhod space and the weak-convergence topology. The reader is referred to Chapter 3 of [62], or [38].

In this section, (E, r) denotes a complete and separable metric space. For an interval I in $\mathbb{R}$, we denote by $\mathbb{D}(\mathbb{R}^+, E)$ the set of right continuous functions from $\mathbb{R}^+$ to E, *i.e.*, the set of function $x : \mathbb{R}^+ \to E$ such that:

- for all $t \in I x(t^+) \stackrel{\text{def}}{=} \lim_{h>0, h\to 0} x(t+h)$ exists and $x(t^+) = x(t)$.
- for all $t \in I$, $x(t^-) \stackrel{\text{def}}{=} \lim_{h<0, h\to 0} x(t+h)$ exists.

Functions having these properties are also called càdlàg functions (*continue à droite, limite à gauche*). A time t such that $x(t-) \neq x(t+)$ is a discontinuity point of x. A càdlàg function has at most a countable set of discontinuity points.

The topology of uniform convergence is not interesting on $\mathbb{D}(\mathbb{R}^+, E)$ since it is too restrictive. Instead, we will consider the topology induced by the Skorokhod distance. Let Λ denote the class of Lipschitz continuous functions mappings of $\mathbb{R}^+$ onto itself. If x and y are two càdlàg functions, we define a distance $d(x, y)$ by:

$$d(x,y) \stackrel{\text{def}}{=} \inf_{\lambda\in\Lambda} \left\{ \max \left\{ \sup_{s>t\geq 0} \log\left(\frac{\lambda(s)-\lambda(t)}{s-y}\right), \int_0^\infty \sup_{t\geq 0} r(x(t\wedge u), y(\lambda(t)\wedge u))du \right\} \right\}$$

where $t \wedge u \stackrel{\text{def}}{=} \min(t, u)$. $d(.,.)$ is called the *Skorokhod distance*. It is well known that $d(.,.)$ is indeed a distance and that $\mathbb{D}(\mathbb{R}^+, E)$ is complete and separable as soon as E is complete and separable (see Chapter 3.5 of [62]).

Let us now consider a distance on $\mathscr{P}(\mathcal{S})$, the space of probability measures on a metric space $(\mathcal{S}, d)$. We define the Prohorov distance ρ between two measures $\mu, \nu \in \mathscr{P}(\mathcal{S})$ by:

$$\rho(\mu,\nu) = \inf\left\{\epsilon > 0 : \mu(F) \leq \nu(F^\epsilon) + \epsilon \quad \text{for all closed set } F\right\},$$

where $F^\epsilon = \{x \in \mathcal{S} : \inf_{y\in F} d(x,y) < \epsilon\}$. Again, we refer to Chapter 3 of [62] to see that $\rho(.,.)$ is indeed a distance and that $(\mathscr{P}(\mathcal{S}), \rho)$ is separable and complete as soon as $(\mathcal{S}, d)$ is.

A collection of measures $M \subset \mathscr{P}(\mathcal{S})$ is said to be *tight* if for each $\epsilon > 0$, there exists a compact space $K \subset S$ such that $P(K) \geq 1 - \epsilon$ for all $P \in M$. Tightness of measures and convergence of measures are closely related as expressed by the next result. This characterization is important when one wants to study the convergence of a sequence of probability measures.

Theorem 2.7 (Prohorov's Theorem)**.** *Assume that $(\mathcal{S}, d)$ is a complete and separable metric space. A sequence of probability measures $M \subset \mathscr{P}(\mathcal{S})$ is tight if and only if M is relatively compact.*

Note that a set $X \in \mathscr{P}(\mathcal{S})$ is said to be relatively compact if any sequence in X has a convergent subsequence in $(\mathscr{P}(\mathcal{S}), d)$.

The Prohorov's metric is not very practical to show convergence of probability measures. For that, we define another notion of convergence, called weak convergence, which is in fact equivalent to the Prohorov's metric. A sequence of probability measures $\mu_n \in \mathscr{P}(\mathcal{S})$ is said to be *converge weakly* to a probability μ if for all bounded continuous function f on the metric space $(\mathcal{S}, d)$,

$$\lim_{n\to\infty} \int f d\mu_n = \int f d\mu. \tag{2.4}$$

If $\mathcal{S}$ is separable, then a sequence of measures μ_n converges weakly to μ if and only if μ_n converges to μ for the Prohorov's topology, *i.e.*, $\lim_{n\to\infty} \rho(\mu_n, \mu) = 0$ (Chapter 3.3 of [62]).

2.3.2. Chaotic and Exchangeable Sequences

Recall that the initial problem is to study the convergence properties of the random variable $Q^N = N^{-1} \sum_{i=1}^N \delta_{X_i^N}$ where X_i^N has sample path in $\mathbb{D}(R^+, \mathcal{S})$. Q^N is a measure on $\mathscr{P}(\mathbb{D}(\mathbb{R}^+, \mathcal{S}))$ and in general, it is not easy to check that the convergence of Equation (2.4) for all function f. Fortunately, when the original process $\mathcal{X}$ is *exchangeable*, weak convergence of Q^N is closely related to the *chaoticity* of the sequence $\mathcal{X}$.

We begin with two definitions. Let $\mathcal{X}^N = (X_1^N \dots X_N^N)$ be a sequence of tuples of random variables on some space E.

The vector $\mathcal{X}^N$ is said to be exchangeable if the law of $\mathcal{X}^N$ is invariant under any permutation of the order of the random variables. If $\mathscr{L}(X)$ denotes the law of a random variable, this means that for any permutation $\sigma : [1:N] \to [1:N]$,

$$\mathscr{L}\left(X_1^N \dots X_N^N\right) = \mathscr{L}\left(X_{\sigma(1)}^N \dots X_{\sigma(N)}^N\right).$$

If Q is a probability measure on E, the sequence $\mathcal{X}^N$ is called Q-chaotic if for any k, the marginal distribution of the first coordinate of $(X_1^N \dots X_k^N)$ is asymptotically a product of k copies of Q. This means that for any continuous function $f_1 \dots f_k$:

$$\lim_{N\to\infty} \mathbb{E}\left(f(X_1^N) f(X_2^N) \dots f(X_k^N)\right) = \prod_{i=1}^k \int f_i dQ.$$

The following theorem, due to Sznitman [138] shows that for exchangeable sequences, chaoticity is equivalent to the convergence of the empirical measure Q^N.

Theorem 2.8 (Proposition 2.2 of [138]). *Let $Q \in \mathscr{P}(E)$.*

(i) If $\mathcal{X}^N$ is exchangeable and Q-chaotic, then Q^N converges weakly to the deterministic law δ_Q.

(ii) If $\mathcal{X}^N$ is exchangeable and Q^N converges to δ_Q, then $\mathcal{X}^N$ is Q-chaotic.

(iii) The sequence of random variables $(Q^N)_N$ is tight if and only if the law of $(X_1^N)_N$ is tight.

This result, and in particular point *(iii)* greatly simplifies the proof of convergence of Q^N since verifying the tightness of the first variable (X_1^N) is much simpler than verifying the same property for Q^N.

2.3.3. Markov processes and Martingale Problems

Once Q^N is proved tight, the last step to prove mean field convergence is to characterize the limit of Q^N. This is often done by showing that the limit points of Q^N satisfy a martingale problem.

Let $(X(t))_{t\geq 0}$ be a stochastic process with value in a space E. Let $(\mathcal{F}_t)$ denotes the canonical filtration associated to $X(.)$: $\mathcal{F}_t = \sigma(X(s), s \leq t)$. X is called a *time-homogeneous Markov process* if

$$\mathcal{P}\left(X(t+s) \in \Gamma | \mathcal{F}_t\right) = \mathcal{P}\left(X(t+s) \in \Gamma | X(t)\right) = P(s, X(t), \Gamma),$$

for all $s, t \geq 0$ and Γ a measurable set. The function $P(.)$ is called the transition function of the process X. The transition function $P(.)$ and the distribution of the initial condition uniquely determine the finite dimensional distribution of X (see Chapter 4.1 of [62]) and therefore the distribution of X.

The *full generator* A of the chain X is defined by:

$$A = \left\{ (f, g) \in B(E) : \mathbb{E}\left(f(X(t)) | X(0)\right) - f(X(0)) = \int_0^t \mathbb{E}\left(g(X(s)) | X(0)\right) ds \right\}.$$

Let us consider the example of a case of a continuous time Markov chain of transition rate matrix Q – meaning that if the chain is in x, it jumps with rate q_x and selects a state y with probability p_{xy}, see Section 1.1.4. The generator of the chain contains all function (f, g) such that

$$g(x) = q_x \sum_{y \in \mathcal{S}} \left(f(y) - f(x)\right) p_{xy}.$$

By definition of the generator, if A is the full generator of a Markov process X, then for all $(g, f) \in A$,

$$M_t = f(X(t)) - \int_0^t g(X(s)) ds \tag{2.5}$$

is an $\mathcal{F}_t$-martingale.

Given A, any process such that (2.5) is a martingale for all $(f, g) \in A$ is said to be a *solution of the martingale problem associated to A*. Martingale problems are often a good way to characterize a Markov chain. In particular, if there is a unique solution to the martingale problem associated to A with sample-path in $\mathbb{D}(\mathbb{R}^+, E)$, then this solution is a time-homogeneous Markov chain.

2.3.4. Proving Mean-Field Convergence

The previous sections give us the theoretical basis that are used to prove mean field convergence. We present here a technique that can be applied to prove that Q^N converges to some deterministic measure. This technique is generic and has been used in several papers, such as [80, 43, 44, 45].

The mean-field analysis is based on three steps.

1. **Tightness** – The first step is to prove that Q^N is tight in $\mathscr{P}(\mathscr{P}(\mathbb{D}(\mathbb{R}^+, \mathbb{N})))$. Thanks to Proposition 2.8, if the objects are exchangeable, one only has to show that the law of X_i^N are tight.

2. **Characterization of the limit** – The second step of the analysis is to show that any probability measure in the support of any limit point of the sequence of Q^N satisfies a measure determining equation, for instance a martingale problem.

3. **Uniqueness of the limit** – It remains to show that the martingale problem has a unique solution Q.

Because of step 1, the law of Q^N is tight which implies the relative compactness of the measure Q^N. Together with steps 2 and 3, this implies that the law of Q^N converges to δ_Q and therefore Q^N converges to Q.

For an example of the application of this method, the reader is referred to [80], in which authors consider a system with N queues, each with a single server with rate μ and infinite buffer. Tasks arrive at rate $N\lambda$ in the system. When a task arrives in the system, it selects L queues uniformly and joins the shortest one. The state of a queue is the number of jobs in its queue: $X_i^N \in \mathbb{N}$. $\mathcal{X}^N$ has same path in $\mathbb{D}(\mathbb{R}^+, \mathbb{N}^N)$ and Q^N has values in $\mathscr{P}(\mathbb{D}(\mathbb{R}^+, \mathbb{N}))$.

The state $\mathcal{X}^N$ is clearly exchangeable. Using this property, tightness of the law of $\mathcal{X}_1^N$ is proved using the compactness criteria in $\mathscr{P}(\mathbb{D}(\mathbb{R}^+, \mathbb{N}))$ in terms of modulus of continuity (Theorem 7.2 of Chapter 1.7 of [62]). Then, authors exhibit a martingale problem in closed-form satisfied by any probability measure in the support of any limit point of Q^N and shows the uniqueness of the solution of this martingale problem. They also show that chaoticity holds at equilibrium and that stationary distribution can be characterized by the stationary distribution of the martingale problem.

2.4. Concluding Remark

In this section, we have seen mainly two different objects of study that lead to completely different techniques of proof. Although the second leads to more powerful results, in the following of the document, we will mainly focus of results of the first type, concerning $M^N(.)$. There are multiple reasons for this. Firstly the convergence results are more general and one can show convergence in probability with explicit bounds for a large class of models. This gives insights on the accuracy of the limits. Secondly, the equation satisfied by the limits of $M^N(.)$ are much simpler to study than the one defined by martingale problem. The limiting equations for $M^N(.)$ are generally easy to simulate, at least numerically while Martingale problem is only a description of the process. Moreover, the definition of optimization problems is straightforward on $M^N(.)$ and not on Q^N.

In Chapter 4, we will apply convergence results to a model of work-stealing in heterogeneous systems. In Part II, we will link results on Markov decision processes and mean

field models and show how the optimal control of a mean-field stochastic system can be simplified by studying its limit $M^N(.)$.

Chapter 3.

Transient Behavior of Work Stealing: Makespan analysis

Abstract of this chapter – A main issue when using a large computing system is to distribute the load among the processors. In this chapter, we analyse the total completion time to execute a set of tasks when using *work stealing*. Work stealing is a very popular *load balancing* technique, based on the principle that when a processor has no task to execute, it steals some work from an other processors.

The difficulty of this analysis is that it is impossible to track the exact number of tasks per processor during the time. To reduce the dimension of the problem, we present a generic method based on potential function and an adversary.

We compute expected completion time as well as bounds on the deviation from the mean. This technique shows that the expected makespan for executing W unit independent tasks on N processors is bounded by W/N unit of time plus an additional term in $3.65 \log_2 W$.

Résumé du chapitre – Un problème majeur lors de l'utilisation de grandes grilles de calcul est de réussir à distribuer le travail sur les différentes ressources disponibles. Dans ce chapitre, nous étudions le temps nécessaire pour exécuter un ensemble de tâches sur une architecture parallèle utilisant le mécanisme de *vol de travail*, dans lequel une ressource qui n'a rien n'a exécuter va *voler* des tâches à d'autres ressources.

Afin de contourner le problème d'explosion du nombre d'états du modèle, notre analyse est basée sur l'étude d'une fonction potentielle et d'un adversaire. Nous obtenons des bornes à la fois sur la moyenne et la probabilité de s'écarter de cette moyenne montrant que le temps nécessaire pour exécuter W tâches sur N processeurs est borné par $W/N + 3.65 \log_2 W$.

3.1. Introduction

List scheduling is one of the most popular technique for scheduling the tasks of a parallel program. This algorithm has been introduced by Graham [83]. Its principle is to build a list of ready tasks and schedule them as soon as there exist available resources. List schedules are low-cost (greedy) algorithms that are not too far from optimal solutions. Most proposed list algorithms always consider a centralized management of the list. However, today parallel platforms involve more and more processors. Thus, the time needed for managing such a centralized data structure can not be ignored anymore [89]. Practically, implementing such schedulers induces synchronization overheads when several processors access the list concurrently. Such overheads involve low level synchronization mechanisms.

A suitable approach to reduce the contention is to distribute the list among the processors: each processor manages its own list of tasks. When a processor becomes idle, it randomly chooses another processor and steals some work, *i.e.* it transfers some tasks from the victim's list to its own list. Such a strategy is called *work-stealing* (WS). WS has been implemented in many languages and parallel libraries including Cilk [66], Intel TBB [99] and KAAPI [78].

This scheduling policy is very easy to implement and does not require any information on the system to work efficiently. It can be made *processor oblivious* since it automatically adapts to the number and to the size of jobs as well as to the speed of the processors [28]. Several models have been proposed to study the performance of work stealing. We briefly describe some of these results at the end of this chapter, in Section 3.9.

We present a simple Markovian model of work stealing in discrete time and analyze the transient behavior of this model. In order to tackle the problem of dimensionality, our analysis is based on a potential function that aggregates the information on the state of the system. To counter the loss of information induced by the use of this potential function, we introduce an adversary that will simulate the worst possible scenario for the missing information. While in this chapter, we focus on the transient behavior, we will see in Chapter 4 how mean field models can be applied to study steady-state behavior.

Based on the analysis of the load balancing between two processors during a steal request, the expected number of steals can be deduced and a bound on the makespan is derived. The methodology is generic and it is applied to the case of independent tasks and to the WS algorithm of [8]. Our new analysis improves the classical bound on the number of steal requests of WS for scheduling precedence graphs. Constant factors are greatly reduced and are less than 50% away from values obtained by simulation. Moreover, our analysis enables us to evaluate precisely the impact of several modifications of the WS.

After presenting the model and notation in Section 3.2, we give the principle of the analysis in Section 3.3. We apply this analysis to the case of unit independent tasks and study the influence of the initial distribution of tasks in Section 3.4. Then we extend the analysis for tasks with dependencies in Section 3.5. Section 3.6 quantifies the reduction of steals when several thieves are allowed to steal the same victim simultaneously. We analyze simulation results in Section 3.7, showing in particular that the gap between simulation and bounds is small. Finally, Section 3.8 contains some technical computation used in the proofs.

3.2. Work Stealing Model

We consider a parallel platform composed of N identical processors. At time t, let $w_i(t)$ represent the amount of work on processor i. When $w_i(t) > 0$, processor i is active and executes some work: $w_i(t+1) \leq w_i(t)$. When $w_i(t) = 0$, processor i is idle and intents to steal from a random processor j. If processor j has no work, *i.e.* $w_j(t) = 0$, the steal fails and processor i will steal again at the next time slot. Otherwise, a certain amount of work is transfered from processor j to processor i: $w_i(t+1) + w_j(t+1) \leq w_j(t)$. Processor i will resume execution at time $t+1$. The execution terminates when all the processors are idle, *i.e.* $\forall i, w_i(t) = 0$. We also denote the total amount of work on all processors by $w(t) = \sum_{i=1}^{N} w_i(t)$ and the number of active processors by $\alpha(t) \in [0, N]$. Between time t and $t+1$, there are $N - \alpha(t)$ steal requests.

To model the contention on the queues, no more than one steal request per processor can succeed in the same time slot. If several requests target the same processor, a random one succeeds and all the others fail. This assumption will be relaxed in Section 3.6.

This is a high level model of a distributed list. We will show in Sections 3.4 and 3.5 how these properties accurately model the case of independent tasks and the WS algorithm of [8]. We justify here some choices of this model. There is no explicit communication cost as WS algorithms most often target shared memory platforms. A steal request is done in constant time independently of the amount of work transfered. This assumption is not restrictive: for the case of independent tasks, the description of a large number of tasks can be very short. For instance a whole subpart of an array of tasks can be represented in a compact way by the range of the corresponding indices, each cell containing the effective description of a task (a STL transform in [144]). For more general cases with dependencies, it is usually enough to transfer a task which represents a part of the graph [8].

3.2.1. The Curse of Dimensionality

The state of the system is described by the amount of work on all processors and is represented by the vector $(w_1(t) \dots w_N(t))$. At the start, the total work in the system is distributed arbitrarily among the processor. It can be distributed arbitrarily, depending on the situation. Therefore, the state of the system can be any vector $(w_1(t) \dots w_N(t))$ such that $\sum_{i=1}^{N} w_i(t) \leq W$.

Because of the mechanism of work stealing, the behavior of the system does not show any regularity, in particular, there is no monotonicity. A first idea to tackle this problem is to consider the process $(x_0(t), \dots x_W(t))$ where $x_j(t)$ represents the number of processors having a work of j at time t. But again, this state space is too large and does not contain sufficient regularity to be studied. Figure 3.1 is an example of a schedule obtained using work-stealing which shows the complex repartition of the idle times. At the beginning of the schedule, the work is concentrated on the first processor: $W_1(0) = 2000$ and for all $i \geq 2$: $W_i(0) = 0$. There are 25 processors in total. Each processor is represented by a line and the columns represents the time slots. The first processor is represented by the first line. At each time step, a processor is either working if it has one or more jobs (represented by a white area) or trying to steal (represented by a gray area).

Our approach is to introduce a potential function $\Phi(w_1(t) \dots w_N(t))$, denoted $\Phi(t)$ for simplicity. The prototype of potential function we will use is $\Phi(t) = \sum_{i=1}^{m} w_i(t)^2$ although this will vary for some situations. Then we will show that we can compute the decrease of

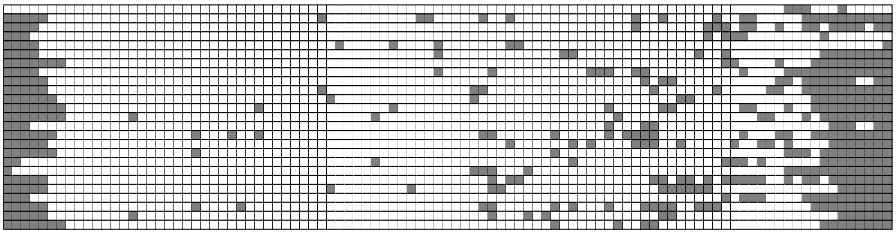


Figure 3.1.: A typical execution of $W = 2000$ unit independent tasks on $N = 25$ processors using work stealing. Grey area represents idle times (which are also stealing times). Figure taken from [141].

Φ as a function of the number of active processors $\alpha(t) \stackrel{\text{def}}{=} N - x_0(t)$. Since $\alpha(t)$ is again a complicated process, we introduce an adversary that model a worst-case scenario for the sequence of $\alpha(t)$. This adversary can be seen as the controller of a Markov Decision Process (see Chapter 1) whose goal is to maximize the completion time of the system.

3.3. Principle of the Analysis and Main Theorem

This section presents the principle of the analysis. The main result is Theorem 3.1 that gives bounds on the expectation of the number of steal requests done by the schedule as well as the probability that the number of requests exceeds this bound.

The main idea of our analysis is that we study the decrease of a potential function $\Phi(t)$, instead of studying directly the number of processors that will run out of work and become idle. The definition of $\Phi(t)$ varies depending on the scenario (see Sections 3.4 to 3.6) but Φ satisfy the following properties:

- $\Phi(t)$ is decreasing
- When $\Phi(t) \leq 1$, the schedule is finished or almost finished.
- There exists a function $h(.) \in [0; 1)$ such that $\mathbb{E}\left(\Phi(t) - \Phi(t+1) | \Phi(t) = \phi, \alpha(t) = \alpha\right) \geq h(\alpha)$.

If $T = \min\{t | \Phi(t) \leq 1\}$, the number of steal requests before $\Phi(t) \leq 1$ is given by $R = \sum_{t=0}^{T-1} N - \alpha(t)$. Since $\alpha(t)$ is a complicated random process, we tackle this problem by assuming that an adversary is choosing the number of active processors $\alpha(t)$ at each step of the schedule. At the beginning, the adversary starts with $\Phi(0)$ potential. She tries to maximize the number of steal requests, before running out of potential. This gives the optimization problem:

$$\text{maximize} \sum_{t=0}^{T-1} N - \alpha(t) \quad \text{with} \begin{cases} \Phi(0) = \Phi_0 \\ T \stackrel{\text{def}}{=} \min\{t | \Phi(t) \leq 1\} \\ \mathbb{E}\left(\Phi(t) - \Phi(t+1) | \Phi(t) = \phi, \alpha(t) = \alpha\right) \geq h(\alpha) \end{cases} . \quad (3.1)$$

The value of this maximization problem provides an upper bound on the number of stealing requests, both in expectation and in probability, as precised by Theorem 3.1.

In the real system, $\alpha(t)$ is determined by the evolution of the system and cannot be chosen at time t. The introduction of an adversary provides an upper-bound on the number of steal requests and has two main advantages. First, its simplicity makes it

applicable in several scenarios, such as the ones presented in Sections 3.4 to 3.6. Moreover, we show in Section 3.7 that the gap between the obtained bound and the value obtained by simulation is relatively small.

The analysis of the scenarios of sections 3.4 to 3.6 will be done in three steps.

1. First, we define a potential function and we compute the potential decrease $\delta_i^k(t)$ when the processor i receives k work requests.

2. Then we compute the expected decrease of the potential between step t and $t+1$, $\Delta\Phi(t) \stackrel{\text{def}}{=} \Phi(t) - \Phi(t+1)$. By linearity of expectation,

$$\mathbb{E}[\Delta\Phi(t)] = \sum_{i=1}^{N} \sum_{k=0}^{N-1} \mathbb{E}[\delta_i^k | i \text{ receives } k \text{ requests}] \mathcal{P}\left(i \text{ receives } k \text{ requests}\right),$$

   where $\mathbb{E}[X|Y]$ denotes the expectation of X knowing Y. Using the properties of $\delta_i^k(t)$, we show that there exists a function $h(\alpha) \in (0,1]$ such that

$$\mathbb{E}[\Delta\Phi(t)|\Phi(t) = \Phi, \alpha(t) = \alpha] \geq h(\alpha)\Phi.$$

3. Finally, we obtain a bound on the expected number of steal requests $\mathbb{E}[R]$ using Theorem 3.1 presented in this section. An upper bound on the expected makespan $\mathbb{E}[C_{\max}]$ can be obtained using the bound on the number of steal.

The following theorem gives an upper bound on the number of steal requests using a lower bound on the expected decrease of the potential in one step.

Theorem 3.1. *Assume that the potential function $\Phi(t)$ satisfies:*

- *There exists a constant $d > 0$ such that $d\Phi(t) \in \mathbb{N}$.*
- *$\Phi(t)$ is non-increasing.*
- *There exists a function $h(\alpha) \in (0,1]$ such that if $\alpha \in [1, N-1]$ processors are active at time t and $\Phi(t) = \Phi$, then the potential decreases on average by*

$$\mathbb{E}[\Phi(t) - \Phi(t+1)|\Phi(t) = \Phi, \alpha(t) = \alpha] \geq h(\alpha) \cdot \Phi.$$

Let $\lambda = \max_{1 \leq \alpha \leq N-1} \frac{N-\alpha}{-N \log_2(1-h(\alpha))}$ and $\Phi(0)$ be the potential at $t = 0$. Then

(i) the expected number of steal requests R until $\Phi(t) \leq 1$ is bounded by

$$\mathbb{E}[R] \leq \lambda \cdot N \cdot \log_2 \Phi(0);$$

(ii) The deviation from the mean can be bounded by:

$$\mathcal{P}\left(R \geq \lambda \cdot N \cdot \log_2 \Phi(0) + u\right) \leq 2^{-u/(\lambda N)}.$$

Proof. Without loss of generality and to simplify the notation, we assume $d = 1$.

Let T be the random variable indicating the end of the schedule: $T = \min\{t|\Phi(t) \leq 1\}$. The number of steal requests is equal to the number of idle processors at each time step. The number of steal requests *after* time t is $R(t)$:

$$R(t) = \sum_{s=t}^{T-1} N - \alpha(s).$$

The total number of steals is $R \stackrel{\text{def}}{=} R(0)$.

The sequence $\alpha(t)$ is difficult to study since it depends on the number of processors at time $t-1$ with 0 or 1 tasks, but also the successful or unsuccessful steals. Therefore, we perform the analysis assuming a worst-case scenario: at each time t, an adversary can choose $\alpha(t)$ knowing the history of the system but not the future random choices. This can be seen as a Markov decision process with total reward criteria, see [123] for more details about Markov decision processes.

We prove by induction on Φ that for all t,

$$\mathbb{E}[R(t)|\Phi(t) = \Phi] \leq \lambda N \log_2(\Phi). \tag{3.2}$$

For $\Phi = 1$, this is clearly true since in that case $T \leq t$ and $R(t) = 0$. Assume that (3.2) holds for all t and all $\phi < \Phi$. $\mathbb{E}[R(t)|\Phi(t) = \Phi]$ is equal to:

$$\begin{aligned}\mathbb{E}[R(t)|\Phi(t) = \Phi] &= \mathbb{E}[N - \alpha(t) + R(t+1)|\Phi(t) = \Phi] \\ &= N - \alpha(t) + \mathbb{E}[R(t+1)|\Phi(t) = \Phi].\end{aligned} \tag{3.3}$$

By definition of $\Delta\Phi(t)$, if $\Phi(t) = \Phi$, then $\Phi(t+1)$ is equal to $\Phi - \phi$ with probability $\mathcal{P}(\Delta\Phi(t) = \phi|\Phi(t) = \Phi)$. Since $\Phi(t)$ is non-increasing, $\Delta\Phi(t) \geq 0$. Therefore:

$$\mathbb{E}[R(t+1)|\Phi(t){=}\Phi] = \sum_{\phi=0}^{\Phi} \mathbb{E}[R(t+1)|\Phi(t+1){=}\Phi - \phi]\mathcal{P}(\Delta\Phi(t) = \phi|\Phi(t){=}\Phi).$$

Let us denote $p_0 \stackrel{\text{def}}{=} \mathcal{P}(\Delta\Phi(t) = 0|\Phi(t) = \Phi)$. Using the induction hypothesis, and the fact that $\mathbb{E}[R(t+1)|\Phi(t+1) = \Phi] = \mathbb{E}[R(t)|\Phi(t) = \Phi]$, we get from (3.3)

$$\begin{aligned}(1{-}p_0)\mathbb{E}[R(t)|\Phi(t){=}\Phi] &\leq N{-}\alpha(t){+}\sum_{\phi=1}^{\Phi} \lambda N \log_2(\Phi - \phi)\mathcal{P}(\Delta\Phi(t){=}\phi|\Phi(t){=}\Phi) \\ &= N{-}\alpha(t){+}\lambda N\mathbb{E}[\log_2(\Phi{-}\Delta\Phi(t))|\Phi(t){=}\Phi]{-}\lambda N \log_2(\Phi)p_0\end{aligned} \tag{3.4}$$

where we used the fact that $\sum_{\phi=1}^{\Phi}(\ldots) = \sum_{\phi=0}^{\Phi}(\ldots) - \lambda N \log_2(\Phi)p_0$.

Moreover, by Jensen's inequality (log is concave), we have:

$$\begin{aligned}\mathbb{E}[\log_2(\Phi - \Delta\Phi(t))|\Phi(t) = \Phi] &\leq \log_2(\Phi - \mathbb{E}[\Delta\Phi(t)|\Phi(t) = \Phi]) \\ &\leq \log_2(\Phi - h(\alpha(t))\Phi).\end{aligned} \tag{3.5}$$

Combining equations (3.4) and (3.5), we get:

$$(1 - p_0)\mathbb{E}[R(t)|\Phi(t) = \Phi] \leq (1 - p_0)\lambda N \log_2(\Phi) + N - \alpha(t) + \lambda N \log_2(1 - h(\alpha(t))).$$

If $\alpha(t) = N$, the sum of the two last terms of the equation is negative since $1 - h(\alpha) \leq 1$. If $\alpha(t) = 0$, the schedule is finished. If $0 < \alpha(t) < N$, the sum of the two last terms is negative by definition of λ (λ corresponds to the worst choice of $\alpha(t)$). Dividing on both sides by $1 - p_0$ concludes the proof of *(i)*.

The proof of *(ii)* is quite similar to the proof of *(i)*. We prove by induction on Φ that $\mathbb{E}[2^{R(t)/(\lambda N)-\log_2(\Phi)}|\Phi(t) = \Phi] \leq 1$. It clearly holds for $\Phi = 1$ since in that case it is equal to 0. $\mathbb{E}[2^{R(t)/(\lambda N)-\log_2(\Phi)}|\Phi(t) = \Phi]$ is equal to

$$\sum_{\phi=0}^{\Phi} \mathbb{E}\left(2^{R(t)/(\lambda N)-\log_2(\Phi)}|\Phi(t+1)=\Phi-\phi\right)\mathcal{P}\left(\Delta\Phi(t)=\phi|\Phi(t)=\Phi\right).$$

Using $R(t+1) = N-\alpha+R(t)$ and introducing $\log_2(\Phi-\phi)-\log_2(\Phi-\phi)$, this equals

$$\begin{aligned}
&\sum_{\phi=1}^{\Phi} 2^{\frac{N-\alpha}{\lambda N}+\log_2(1-\frac{\phi}{\Phi})}\mathbb{E}[2^{\frac{R(t+1)}{\lambda N}-\log_2(\Phi-\phi)}|\Phi(t+1)=\Phi-\phi]\mathcal{P}\left(\Delta\Phi(t)=\phi|\Phi(t)=\Phi\right)\\
&\qquad\qquad + 2^{\frac{N-\alpha}{\lambda N}}\mathbb{E}[2^{\frac{R(t+1)}{\lambda N}-\log_2(\Phi)}|\Phi(t+1)=\Phi]p_0\\
&\leq \sum_{\phi=1}^{\Phi} 2^{\frac{N-\alpha}{\lambda N}+\log_2(1-\frac{\phi}{\Phi})}\mathcal{P}\left(\Delta\Phi(t)=\phi|\Phi(t)=\Phi\right) + 2^{\frac{N-\alpha}{\lambda N}}\mathbb{E}[2^{\frac{R(t)}{\lambda N}-\log_2(\Phi)}|\Phi(t)=\Phi]p_0
\end{aligned}$$

where we used the induction hypothesis for the inequality.

Then, adding and subtracting the first term of the sum $\sum_{\phi=1}^{\Phi}$, this leads to

$$\begin{aligned}
(1 - 2^{\frac{N-\alpha}{\lambda N}}p_0)\mathbb{E}[2^{\frac{R(t)}{\lambda N}-\log_2(\Phi)}|\Phi(t)=\Phi] &\leq 2^{\frac{N-\alpha}{\lambda N}}\mathbb{E}[1 - \Delta\Phi/\Phi|\Phi(t)=\Phi] - 2^{\frac{N-\alpha}{\lambda N}}p_0\\
&\leq 2^{\frac{N-\alpha}{\lambda N}+\log_2(1-h(\alpha))} - 2^{\frac{N-\alpha}{\lambda N}}p_0\\
&\leq 1 - 2^{\frac{N-\alpha}{\lambda N}}p_0.
\end{aligned}$$

where we used the definition of λ to show that the first term is less than one. This shows that $\mathbb{E}[2^{\frac{R(t)}{\lambda N}-\log_2(\Phi)}|\Phi(t)=\Phi] \leq 1$. Therefore by Markov's inequality:

$$\mathcal{P}\left(R(t) \geq \lambda N \log_2 \Phi + u|\Phi(t)=\Phi\right) = \mathcal{P}\left(2^{\frac{R(t)}{\lambda N}-\log_2 \Phi} \geq 2^{\frac{u}{\lambda N}}|\Phi(t)=\Phi\right) \leq 2^{-\frac{u}{\lambda N}}.$$

□

3.4. Unit Independent Tasks

We apply the analysis presented in the previous section for the case of independent unit tasks. In this case, each processor i maintains a local queue Q_i of tasks to execute. At every time slot, if the local queue Q_i is not empty, processor i picks a task and executes it. When Q_i is empty, processor i sends a steal request to a random processor j. If Q_j is empty or contains only one task (currently executed by processor j), then the request fails and processor i will have to send a new request at the next slot. If Q_j contains more than one task, then i is given half of the tasks (after that the task executed at time t by processor j has been removed from Q_j). The amount of work on processor i at time t, $w_i(t)$, is the number of tasks in $Q_i(t)$. At the beginning of the execution, $w(0) = W$ and tasks can be arbitrarily spread among the queues.

Applying the method presented in Section 3.3, the first step of the analysis is to define the potential function and compute the potential decrease when a steal occurs. For this example, $\Phi(t)$ is defined by:

$$\Phi(t) = \sum_{i=1}^{N} \left(w_i(t) - \frac{w(t)}{N} \right)^2 = \sum_{i=1}^{N} w_i(t)^2 - \frac{w^2(t)}{N}.$$

This potential represents the load unbalance in the system. If all queues have the same $w_i(t) = w(t)/N$, then $\Phi(t) = 0$. $\Phi(t) \leq 1$ implies that there is at most one processor with at most one more task than the others. In that case, there will be no steal until there is just one processor with 1 task and all others idle. Moreover, the potential function is maximal when all the work is concentrated on a single queue. That is $\Phi(t) \leq w(t)^2 - w(t)^2/N \leq (1 - 1/N)w^2(t)$.

Assume that at time t, the queue i has $w_i(t) \geq 1$ tasks. If it receives one or more steal requests, it chooses a processor j among the thieves. At time $t+1$, i has executed one task and the rest of the work is split between i and j. Therefore, $w_i(t+1) = \lceil (w_i(t) - 1)/2 \rceil$ and $w_j(t+1) = \lfloor (w_i(t) - 1)/2 \rfloor$. Thus $w_i(t+1)^2 + w_j(t+1)^2 = \lceil (w_i(t)-1)/2 \rceil^2 + \lfloor (w_i(t)-1)/2 \rfloor^2 \leq w_i(t)^2/2 - w_i(t) + 1$. Therefore, this generates a difference of potential of

$$\delta_i^k(t) = \delta_i^1(t) \geq w_i(t)^2/2 + w_i(t) - 1. \tag{3.6}$$

If i receives zero steal requests, it potential goes from $w_i(t)^2$ to $(w_i(t) - 1)^2$, generating a potential decrease of $2w_i(t) - 1$. The last event contributing to the change of the potential is that $(\sum_{i=1}^{m} w_i(t))^2/N$ goes from $w(t)^2/N$ to $w(t+1)^2 = (w(t) - \alpha(t))^2/N$, generating a potential increase of $2\alpha(t)w(t)/N - \alpha(t)^2/N$.

Recall that at time t, there are $\alpha(t)$ active processors and therefore $N - \alpha(t)$ processors that send steal requests. A processor i receives zero steal requests if the $N - \alpha(t)$ thieves choose another processor. Each of these events is independent and happens with probability $(N-2)/(N-1)$. Therefore, the probability for the processor to receive one or more steal requests is $p_r(\alpha(t))$:

$$p_r(\alpha(t)) = 1 - \left(1 - \frac{1}{N-1}\right)^{N-\alpha(t)}.$$

If $\Phi(t)$=Φ and $\alpha(t)$=α, by summing the expected decrease on each active processor δ_i^1, the expected potential decrease is greater than:

$$\begin{aligned}
&\sum_{i/w_i(t)>0} \left[p_r(\alpha)\left(\frac{w_i(t)^2}{2} + w_i(t) - 1\right) + (1 - p_r(\alpha))(2w_i(t) - 1) \right] - 2w(t)\frac{\alpha}{N} + \frac{\alpha^2}{N} \\
&\quad = \frac{p_r(\alpha)}{2}\Phi + \frac{p_r(\alpha)}{2}\left(\frac{w(t)^2}{N} - 2w(t) + 2\frac{N-\alpha}{Np_r(\alpha)}(2w(t) - \alpha) \right) \\
&\quad \geq \frac{p_r(\alpha)}{2}\left(\Phi + \frac{w(t)^2}{N} - 2\frac{w(t)}{N} + 2\left(1 - \frac{1}{N}\right)(w(t) - \alpha) \right) \geq \frac{p_r(\alpha)}{2}\Phi.
\end{aligned} \tag{3.7}$$

The details of the computation of (3.7) can be found in Appendix 3.8.1.

Using Theorem 3.1 of the previous section, we conclude the analysis by the following theorem.

Theorem 3.2. *Let $C_{\max}$ be the makespan of W unit independent tasks scheduled by work stealing. Then:*

(i) $$\mathbb{E}\left(C_{\max}\right) \leq \frac{W}{N} + \frac{2}{1-\log_2(1+\frac{1}{e})} \cdot \log_2 W + 1.$$

(ii) $$\mathcal{P}\left(C_{\max} \geq \frac{W}{N} + \frac{2}{1-\log_2(1+\frac{1}{e})} \cdot \left(\log_2 W + \log_2 \frac{1}{\epsilon}\right) + 1\right) \leq \epsilon.$$

These bounds are optimal up to a constant factor in $\log_2 W$*.*

Proof. Equation (3.7) shows that $\mathbb{E}[\Delta\Phi(t)|\Phi(t) = \phi, \alpha(t) = \alpha] \leq h(\alpha)\Phi$ with $h(\alpha) = p_r(\alpha)/2$. Using Theorem 3.1 *(i)* and the fact that $\Phi(0) \leq W^2$, the expected number of steal requests before $\Phi(t) \leq 1$ is bounded by:

$$\mathbb{E}[R] \leq \lambda N \log_2(W^2) = 2\lambda N \log_2(W),$$

with $\lambda = \max_{1\leq\alpha\leq N-1}(N-\alpha)/(-N\log_2(1-h(\alpha)))$. We show in appendix 3.8.2 that $(N-\alpha)/(-N\log_2(1-h(\alpha)))$ is decreasing in α. Thus its minimum is attained for $\alpha = 1$. This shows that $\lambda \leq 1/(1-\log_2(1+\frac{1}{e}))$.

As said before, when $\Phi(t) \leq 1$, there is at most one processor with at least one more task than the others. Therefore, there will be a steal request only when this processor will have one task and the others zero. This happens only once and generates at most $N-1$ steal requests.

At each time step of the schedule, a processor is either computing one task or stealing work. This shows that $N \cdot C_{\max} = W + R$. Thus:

$$\mathbb{E}[C_{\max}] \leq \frac{W}{N} + \frac{2}{1-\log_2(1+\frac{1}{e})} \log_2 W + 1.$$

The proof of the *(i)* applies *mutatis mutandis* to prove the bound in probability *(ii)* using Theorem 3.1 *(ii)*.

We now give a lower bound for this problem. Consider $W = 2^{k+1}$ tasks and $N = 2^k$ processors, all the tasks being on the same processor at the beginning. In the best case, all steal requests target processors with highest loads. In this case the makespan is $C_{\max} = k+2$: the first $k = \log_2 N$ steps for each processor to get some work; one step where all processors are active; and one last step where only one processor is active. In that case, $C_{\max} \geq \frac{W}{N} + \log_2 W - 1$. □

This theorem shows that the factor before $\log_2 W$ is bounded by 1 and $2/(1-\log_2(1+1/e)) < 3.65$. Simulations reported in Section 3.7 seem to indicate that the factor of $\log_2(W)$ is slightly less. This shows that the constants obtained by our analysis are sharp.

Initial repartition of tasks. In practical situations, the number of tasks of an application is unknown before the beginning of the execution of this application and in the worst case, all tasks are in the same queue at the beginning of the execution. Using bounds in terms of Φ_0, one can show that a more balanced initial repartition leads to fewer steal requests on average. Suppose that we take a balls-and-bins assignment as the initial repartition: for each task, we choose a processor at random and put the task in its queue. The expected value of Φ_0 is:

$$\mathbb{E}\left(\Phi_0\right) = \sum_i \mathbb{E}\left(w_i^2\right) - \frac{W^2}{N} = \sum_i \Big(\mathrm{Var}\left[w_i\right] + \mathbb{E}\left(w_i\right)^2\Big) - \frac{W^2}{N} = \Big(1 - \frac{1}{N}\Big) \cdot W$$

as w_i follows a binomial distribution. Since the number of work requests is proportional to $\log_2 \Phi_0$, this initial distribution of tasks reduces the number of steal requests by a factor of 2 on average.

3.5. Tasks with Precedences

In this section, we show how the well known non-blocking work stealing of Arora *et al.* [8] (denoted ABP in the sequel) can be analyzed with our method which provides tighter bounds for the makespan. Following [8], a multithreaded computation is modeled as a directed acyclic graph G with a single root node; each node corresponds to a unit task and edges define precedence constraints. The out-degree of each node is either 0, 1 or 2. The critical path of G is denoted by T_∞ and W is its total number of nodes.

ABP schedules the DAG G as follows. Each process i maintains a double-ended queue (called a deque) Q_i of ready nodes (the notion of process here corresponds to our processors). At each slot, an active process i with a non-empty deque executes the node at the bottom of its deque Q_i; once its execution is completed, this node is popped from the bottom of the deque, enabling – *i.e.* making ready – 0, 1 or 2 child nodes that are pushed at the bottom of Q_i. At each top, an idle process j with an empty deque Q_j becomes a thief: it performs a steal request on another randomly chosen victim deque; if the victim deque contains ready nodes, then its top-most node is popped and pushed into the deque of one of its concurrent thieves. If j becomes active just after its steal request, the steal request is said successful. Otherwise, Q_j remains empty and the steal request fails which may occur in the three following situations: either the victim deque Q_i is empty; or, Q_i contains only one node currently in execution on i; or, due to contention, another thief performs a successful steal request on i simultaneously.

Let us first recall the definition of the *enabling tree* of [8]. If the execution of node u enables node v, then the edge (u, v) of G is an enabling edge. The sub-graph of G consisting of only enabling edges forms a rooted tree called the enabling tree. If $d(u)$ is the depth of a node u in the enabling tree, then its weight is defined as $\omega(u) = T_\infty - d(u)$. The weight of the root node is T_∞.

To represent the amount of work on each processor, we define $w_i(t) = 2^{\max\{\omega(u):u\in Q_i(t)\}}$, *i.e.* the maximum number of tasks that can be activated by a task in Q_i. We first study the repartition of the work during a steal request.

Lemma 3.3. *For any active process i, we have $w_i(t+1) \leq w_i(t)$. Moreover, after any steal request from a process j on i, $w_i(t+1) \leq \dfrac{w_i(t)}{2}$ and $w_j(t+1) \leq \dfrac{w_i(t)}{2}$.*

Proof. The proof is derived from [8], Corollary 4 in Section 3: if at t, Q_i contains the $k+1$ nodes $v_0, v_1, \ldots, v_k$ from bottom to top, then $\omega(v_0) \leq \omega(v_1) < \ldots < \omega(v_{k-1}) < \omega(v_k)$. After the execution of a node u, the maximum weight of its two enabled children is less than $\omega(u) - 1$. Thus the potential work w_i cannot increase.

We now state that the potential is halved after any steal request by distinguishing two cases. First, when a successful steal occurs on i from j, then the node v_k has

been stolen and executed by j. Thus, either $w_i(t+1) = 0$ if Q_i is empty at $t+1$; or $w_i(t+1) = 2^{\omega(v_{k-1})} \leq 2^{\omega(v_k)-1} \leq w_i(t)/2$. Besides, after execution of v_k by j, $w_j(t+1) \leq 2^{\omega(v_k)-1} \leq w_i(t)/2$. Secondly, if all steal requests that occur on i are unsuccessful, then there was only one node v_0 in Q_i whose execution was processed by i. Then, at $t+1$, $w_i(t+1) \leq 2^{\omega(v_0)-1} \leq w_i(t)/2$ and $w_j(t+1) = 0$. □

We can now state the following theorem.

Theorem 3.4. *The expected makespan of ABP work stealing verifies:*

(i) $$\mathbb{E}\left(C_{\max}\right) \leq \frac{W}{N} + \frac{2}{1-\log_2(1+1/e)} \cdot T_\infty + 1 < \frac{W}{N} + 3.65 \cdot T_\infty + 1.$$

(ii) $$\mathcal{P}\left(C_{\max} \geq \frac{W}{N} + \frac{2}{1-\log_2(1+1/e)} \cdot \left(T_\infty + \log_2 \frac{1}{\epsilon}\right) + 1\right) \leq \epsilon.$$

Proof. The proof is a direct application of Theorem 3.1 using the potential function $\Phi(t) = \sum_i w_i(t)^2$. Note that we cannot use the same potential as in Section 3.4 because the total amount of work may be reduced when a steal occurs[1]. □

Remark. In [8], the authors established the upper bounds

$$\mathbb{E}\left(C_{\max}\right) \leq \frac{W}{N} + 32 \cdot T_\infty \text{ and } \mathcal{P}\left(C_{\max} \geq \frac{W}{N} + 64 \cdot T_\infty + 16 \cdot \log_2 \frac{1}{\epsilon}\right)\right) \leq \epsilon$$

in Section 4.3, proof of Theorem 9. Our bounds greatly improve the constant factors of this previous result and are close to simulation values (cf. Section 3.7).

3.6. Cooperation Among Thieves

In this section, we modify the protocol for managing the distributed list. Previously, when $k > 1$ steal requests were sent on the same processor, only one of them could be served due to contention on the list. We now allow the k requests to be served in unit time. This model has been implemented in the middleware Kaapi [78] which allow to build parallel application scheduled by work stealing. When k steal requests target the same processor, the work is divided into $k+1$ pieces. In practice, allowing concurrent thieves increase the cost of a steal request but we neglect this additional cost here. We assume that the k concurrent steal requests can be served in unit time. We study the influence of this new protocol on the number of steal requests in the case of unit independent tasks.

We use the potential function[2] $\Phi(t) = \sum_{i=1}^{N} w_i(t)^2$. Let us first compute the decrease of the potential when processor i receives $k \geq 1$ steal requests. If $w_i(t) > 0$, it can be written $w_i(t) = (k+1)q + r + 1$ with $0 \leq r < k+1$. After one time step and k steal requests, the work will be divided in r parts with $q+1$ tasks and $k+1-r$ parts with q tasks. By a direct computation, the potential generated by these steal requests at time $t+1$ can be bounded by:

$$r(q+1)^2 + (k+1-r)q^2 = (k+1)q^2 + r(2q+1) \leq \frac{1}{k+1}\left((k+1)q + r\right)^2 \leq \frac{w_i(t)^2}{k+1}.$$

[1] This can happen when the sibling of the stolen task has only one child.

[2] The same potential function as in Section 3.4 could be used but leads to more complex computations.

If $N-\alpha$ processors send steal requests, the probability for an active processor to receive k steal requests is

$$p_k(\alpha) = \binom{N-\alpha}{k} \frac{1}{(N-1)^k} \left(\frac{N-2}{N-1}\right)^{N-\alpha-k}.$$

The expected diminution of the potential caused by the steals on processor i is equal to $\sum_{k=0}^{N-\alpha} \delta_i^k p_k(\alpha)$. By a direct computation, this is bounded by

$$\begin{aligned}\sum_{k=0}^{N-\alpha} \delta_i^k p_k(\alpha) &\geq \sum_{k=0}^{N-\alpha} \left(1 - \frac{1}{k+1}\right) w_i(t)^2 p_k(\alpha) \\ &= w_i(t)^2 \left(1 - \frac{N-1}{N-\alpha+1}\left(1 - \left(\frac{N-2}{N-1}\right)^{N-\alpha+1}\right)\right).\end{aligned}$$

This shows that $\mathbb{E}[\Delta\Phi(t)|\Phi(t)=\Phi|\alpha(t)=\alpha] \leq h(\alpha)\Phi$ where

$$h(\alpha) = 1 - \frac{N-1}{N-\alpha+1}\left(1 - \left(\frac{N-2}{N-1}\right)^{N-\alpha+1}\right).$$

Differentiating with respect to α shows that $(N-\alpha)/-\log_2(1-h(\alpha))$ is decreasing. Thus $\lambda = \max_{1\leq\alpha\leq N}(N-\alpha)/-N\log_2(1-h(\alpha)) = (N-1)/-N\log_2(1-h(1))$. A direct computation shows that $\lambda \leq 1/-\log_2(1-1/e)$. See Appendix 3.8.3 for details. Therefore we can copy *mutatis mutandis* the proof of Theorem 3.2 to show that:

Theorem 3.5. *The makespan $C_{\max}^{\text{coop}}$ of W unit independent tasks scheduled with cooperative work stealing satisfies:*

(i) $\mathbb{E}\left(C_{\max}^{\text{coop}}\right) \leq \dfrac{W}{N} + \dfrac{2}{-\log_2(1-\frac{1}{e})} \cdot \log_2 W + 1.$

(ii) $\mathcal{P}\left(C_{\max}^{\text{coop}} \geq \dfrac{W}{N} + \dfrac{2}{-\log_2(1-\frac{1}{e})} \cdot \log_2 W + 1 \geq \dfrac{2}{-\log_2(1-\frac{1}{e})} \log_2(\epsilon)\right) \leq \epsilon.$

Compared to the situation with no cooperation among thieves, the number of steal requests is reduced by a factor $\frac{1-\log_2(1+1/e)}{-\log_2(1-1/e)} \approx 1.20$. We will see in Section 3.7 that this is close to the value obtained by simulation.

3.7. Experimental Study

Theorem 3.1 provides upper bound on the expected value of the makespan for the models considered in Sections 3.4,3.5,3.6. In this section, we experimentally study the constant factor of the $\log_2 W$ term and show that it is close to the theoretical result. We focus on independent tasks as it is difficult to generate a realistic random DAG. Moreover, the DAG with the maximum number of tasks, out-degree at most 2 and critical path T_∞ is a complete binary tree of height T_∞. This is the worst case for the bound given in Section 3.5 and it is similar to the independent tasks case.

We developed a simulator that strictly follows the model of Sections 3.4 and 3.6. At the beginning, all the tasks are given to processor 0 in order to be in the worst case, *i.e.*

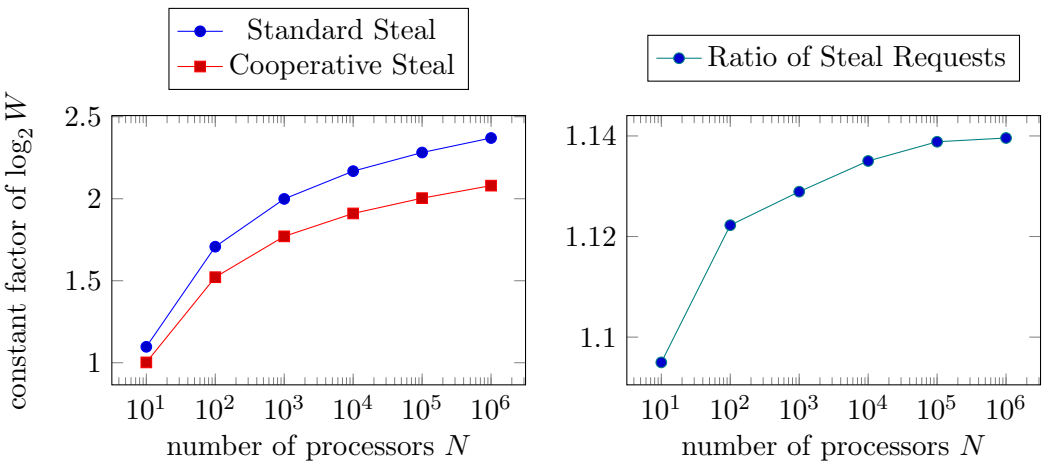


Figure 3.2.: (Left) Constant factor of $\log_2 W$ against the number of processors for the standard steal and the cooperative steal. (Right) Ratio of steal requests (standard/cooperative).

when the initial potential Φ_0 is maximum. Each pair (N,W) is simulated 10000 times to get accurate results. The empirical coefficient of variation is less than 2%.

We computed the constant factor 2λ of the $\log_2 W$ term for various number of processors and tasks. According to Theorem 3.2 and Theorem 3.5, these values are bounded by 3.65 for the standart steal and 3.02 for the cooperative steal. Figure 3.2 seems to indicate that the real values should be around $2\lambda \approx 2.37$ for unit independent tasks with standard steal and $2\lambda_{\text{coop}} \approx 2.08$ for unit independent tasks with cooperative steal. The difference between the theoretical bounds and the simulations can be explained by the use of an adversary in Theorem 3.1 which captures a worst-case scenario for the number of active processors.

Our theoretical analysis predicts an advantage of the cooperative steal with a gain of 20% over the standard steal. Surprisingly, this value is very close to the experimental gain of 14%.

3.8. Appendix

3.8.1. Proof of Inequality 3.7 of Theorem 3.2

In this section, we show that in the case of independent tasks, the expectation of the potential decrease is greater than $\Phi p_r(\alpha)/2$.

Recall that the expectation of the potential decrease is greater than:

$$\sum_{i/w_i(t)>0} \left[p_r(\alpha) \left(\frac{w_i(t)^2}{2} + w_i(t) - 1 \right) + (1 - p_r(\alpha))(2w_i(t) - 1) \right] - 2w(t)\frac{\alpha}{N} + \frac{\alpha^2}{N}$$
$$= \frac{p_r(\alpha)}{2} \left(\sum w_i(t)^2 - \frac{w(t)^2}{N} \right) + p_r(\alpha) \left(\frac{w(t)^2}{2m} + w(t) - \alpha \right)$$
$$+ (1 - p_r(\alpha))\,(2w(t) - \alpha) - 2w(t)\frac{\alpha}{N} + \frac{\alpha^2}{N}$$

where we used the fact that $\sum_{i/w_i(t)>0} w_i(t) = w(t)$ and that $\sum_{i/w_i(t)>0} 1 = \alpha$ since α is the number of active processors. A direct computation shows that this is equal to

$$\begin{aligned}
&\frac{p_r(\alpha)}{2}\Phi + p_r(\alpha)\left(\frac{w(t)^2}{2m} + w(t) - \alpha - 2w(t) + \alpha\right) + 2w(t) - \alpha - 2w(t)\frac{\alpha}{N} + \frac{\alpha^2}{N} \\
&\quad = \frac{p_r(\alpha)}{2}\Phi + \frac{p_r(\alpha)}{2}\left(\frac{w(t)^2}{N} - 2w(t) + \frac{2}{p_r(\alpha)}\left(1 - \frac{\alpha}{N}\right)(2w(t) - \alpha)\right) \\
&\quad = \frac{p_r(\alpha)}{2}\Phi + \frac{p_r(\alpha)}{2}\left(\frac{w(t)^2}{N} - 2w(t) + \frac{2}{N}\frac{N-\alpha}{p_r(\alpha)}(2w(t) - \alpha)\right). \qquad (3.8)
\end{aligned}$$

Let define f by

$$f(\alpha) \stackrel{\text{def}}{=} \frac{N-\alpha}{p_r(\alpha)} = \frac{N-\alpha}{1 - \left(1 - \frac{1}{N-1}\right)^{N-\alpha}}$$

and compute the derivative f'.

$$\begin{aligned}
\left(1 - \left(1 - \frac{1}{N-1}\right)^{N-\alpha}\right)^2 \cdot f'(\alpha) &= -\left(1 - \left(1 - \frac{1}{N-1}\right)^{N-\alpha}\right) \\
&\quad + (N-\alpha)\cdot \ln\left(1 - \frac{1}{N-1}\right)\cdot\left(1 - \frac{1}{N-1}\right)^{N-\alpha} \\
&= -1 + \left(1 - \frac{1}{N-1}\right)^{N-\alpha}\cdot\left(1 + (N-\alpha)\ln\left(1 - \frac{1}{N-1}\right)\right) \\
&\le -1 + \left(1 - \frac{1}{N-1}\right)^{N-\alpha}\cdot\left(1 + (N-\alpha)\frac{-1}{N-1}\right) \\
&\le -1 + \left(1 - \frac{1}{N-1}\right)^{N-\alpha}\cdot\frac{\alpha-1}{N-1} \\
&\le 0.
\end{aligned}$$

As f is non increasing for $1 \le \alpha \le N-1$,

$$\min_{1\le\alpha\le N-1} f(\alpha) = f(N-1) = \frac{1}{1 - \left(1 - \frac{1}{N-1}\right)^1} = N-1.$$

(3.8) is greater than:

$$\begin{aligned}
&\frac{p_r(\alpha)}{2}\Phi + \frac{p_r(\alpha)}{2}\left(\frac{w(t)^2}{N} - 2w(t) + 2\frac{N-1}{N}(2w(t) - \alpha)\right) \\
&\quad = \frac{p_r(\alpha)}{2}\Phi + \frac{p_r(\alpha)}{2}\left(\frac{w(t)^2}{N} - 2w(t) + 2w(t)\left(1 - \frac{1}{N}\right) + 2\left(1 - \frac{1}{N}\right)(w(t) - \alpha)\right) \\
&\quad = \frac{p_r(\alpha)}{2}\Phi + \frac{p_r(\alpha)}{2}\left(\frac{w(t)^2}{N} - \frac{2w(t)}{N} + 2\left(1 - \frac{1}{N}\right)(w(t) - \alpha)\right).
\end{aligned}$$

As $w(t) - \alpha(t) \ge 0$ (an active processor has at least one task) the last term is positive. Moreover, for all $w(t) > 1$, the second term is positive. Thus, this is greater than $\frac{p_r(\alpha)}{2}\Phi$ which concludes the proof of the inequality:

$$\sum_{i/w_i(t)>0}\left[p_r(\alpha)\left(\frac{w_i(t)^2}{2} + w_i(t) - 1\right) + (1 - p_r(\alpha))(2w_i(t) - 1)\right] - 2w(t)\frac{\alpha}{N} + \frac{\alpha^2}{N} \ge \frac{p_r(\alpha)}{2}\Phi.$$

3.8.2. Computation of λ for the unit tasks

In this section, we compute the constant λ for the unit tasks. We first show that the quantity $(N-\alpha)/\left(-\log_2(1-p_r(\alpha)/2)\right)$ is decreasing in α. Then we bound the value $(N-1)/\left(-\log_2(1-p_r(1)/2)\right)$ by $(N-1)/(1-\log_2(1+1/e))$.

Let $g(\alpha) \stackrel{\text{def}}{=} -\log_2(1-p_r(\alpha)/2)$ and $f(\alpha) \stackrel{\text{def}}{=} (N-\alpha)/g(\alpha)$. By definition of $p_r(\alpha)$, $g(\alpha)$ can be written:

$$g(\alpha) \quad = \quad -\log_2\left(\frac{1}{2}+\frac{1}{2}\left(1-\frac{1}{N-1}\right)^{N-\alpha}\right) = 1-\log_2\left(1+\left(1-\frac{1}{N-1}\right)^{N-\alpha}\right).$$

Denoting $p \stackrel{\text{def}}{=} 1-1/(N-1)$ and $x \stackrel{\text{def}}{=} p^{N-\alpha}$, the derivative of f with respect to α is:

$$\begin{aligned} f'(\alpha) &= \frac{\ln(1+p^{N-\alpha})-\ln 2+p^{N-\alpha}\left(\ln(1+p^{N-\alpha})-\ln 2\right)+p^{N-\alpha}\ln(p)(\alpha-N)}{(1+p^{N-\alpha})g(\alpha)^2\ln 2} \\ &= \frac{(\ln(1+x)-\ln 2)\,x+\ln(1+x)-x\ln x-\ln 2}{(1+x)g(\alpha)^2\ln 2} \\ &= \frac{(1+x)\ln(1+x)-x\ln x-(1+x)\ln 2}{(1+x)g(\alpha)^2\ln 2}. \end{aligned}$$

The derivative of $(1+x)\ln(1+x)-x\ln x-(1+x)\ln 2$ w.r.t. x is $\ln(1+x)-\ln(x)-\ln 2 = \ln(1+1/x)-\ln 2 > 0$. As $x<1$, this shows that $(1+x)\ln(1+x)-x\ln x-(1+x)\ln 2 < (1+1)\ln(1+1)-1\ln 1-(1+1)\ln 2 = 0$.

Thus, $f(\alpha)$ is decreasing and

$$\lambda = \max_{1\le\alpha\le N-1}\frac{1}{N}f(\alpha) = \frac{1}{N}f(1) \le \frac{1}{1-\log_2\left(1+\left(1-\frac{1}{N-1}\right)^{N-1}\right)}.$$

Using the fact that for all $N \ge 2$:

$$\left(1-\frac{1}{N-1}\right)^{N-1} = \exp\left((N-1)\ln(1-\frac{1}{N-1})\right) \le \exp\left(-(N-1)\frac{1}{N-1}\right) = \frac{1}{e},$$

we get that $1-\log_2\left(1+(1-1/(N-1))^{N-1}\right) \ge 1-\log_2(1+1/e)$. This shows that

$$\lambda \le \frac{1}{1-\log_2(1+1/e)}.$$

3.8.3. Computation of λ for the cooperative steal

In this section, we compute the maximum of $(N-\alpha)/-\log_2(1-h(\alpha))$ for $1\le\alpha\le N-1$ in the case of cooperative thieves. We first show this function is decreasing in α and then we bound its value in for $\alpha=1$.

Let $g(\alpha) \stackrel{\text{def}}{=} -\log_2(1-h(\alpha))$ with $h(\alpha)$ defined as in Section 3.6. $g(\alpha)$ is equal to

$$g(\alpha) = -\log_2\left(\frac{N-1}{N-\alpha+1}\left(1-\left(1-\frac{1}{N-1}\right)^{N-\alpha+1}\right)\right).$$

Let $f(\alpha) \stackrel{\text{def}}{=} (N-\alpha)/g(\alpha)$. Let $f'(\alpha)$ be the derivative of $f(\alpha)$ w.r.t. α. Denoting $p \stackrel{\text{def}}{=} 1 - 1/(N-1)$ and $n \stackrel{\text{def}}{=} N - \alpha + 1$, we define $k(p)$ by:

$$k(p) \stackrel{\text{def}}{=} f'(\alpha)g^2(\alpha)\ln 2 \quad = \quad -\ln(n(1-p)) + \ln(1+p^n) + \frac{n-1}{n}\left(1 + \frac{p^n \ln(p)n}{1-p^n}\right).$$

The derivative of $k(p)$ w.r.t. p is

$$\begin{aligned} k'(p) &= \frac{\ln(p^n)(n-1)p^{n-1}(p-1) + (p^n-1)(1-p^{n-1})}{(p-1)(p^n-1)^2} \\ &= \frac{1-p^{n-1}}{(p-1)(p^n-1)^2}\left(p^{n-1}\ln(p^n)\frac{(1-n)(p-1)}{1-p^{n-1}} + p^n - 1\right). \end{aligned}$$

Moreover, $(1-p^{n-1})/(1-p) = 1 + p + p^2 \cdots + p^{n-2} \leq n-1$ and since $\ln(p^n) < 0$ and $p^{n-1} > p^n$, we have:

$$\ln(p^n)p^{n-1}\frac{(1-n)(p-1)}{1-p^{n-1}} + p^n - 1 \quad \leq \quad p^{n-1}\ln(p^n) + p^n - 1 \leq p^n\ln(p^n) + p^n - 1.$$

Since $p^n < 1$, $p^n \ln(p^n) < 0$ and $p^n - 1 < 0$. Thus, the last part of the equation is negative. Since $1-p$ is negative, $k'(p) \geq 0$. This shows that $k'(p) \leq k(1) = 0$. Therefore $f'(\alpha) \leq 0$ and $f(\alpha) \geq f(1) = (N-1)/g(1)$ for $1 \leq \alpha \leq N-1$. We have:

$$\begin{aligned} g(1) &= -\log_2 \frac{N-1}{N} - \log_2\left(1 - \left(1 - \left(\frac{1}{N-1}\right)\right)^N\right) \\ &\geq -\log_2(1-e^{-1}). \end{aligned}$$

This shows that

$$\lambda = \frac{1}{N}f(1) \leq \frac{1}{N}\frac{N-1}{-\log_2(1-\frac{1}{e})} \leq \frac{1}{-\log_2(1-\frac{1}{e})}.$$

3.9. Bibliographical Notes

WS has been analyzed in a seminal paper of Blumofe and Leiserson [40] where they show that the expected makespan of series-parallel precedence graph with unit tasks is bounded by $\mathbb{E}(C_{\max}) \leq W/N + \mathrm{O}(T_\infty)$ where T_∞ is the critical path of the graph. This analysis has been improved in [8] using a proof based on a potential function. The case of varying processor speeds has been analyzed in [22].

A practical consideration for work stealing implementations is the stability of the algorithm. Such issues have been tackled in [25], where a system of homogeneous processors and a total arrival rate smaller than the total service rate is proved to be a positive recurrent Markov chain in a discrete time setting, the general case being still open. A steady state analysis of work stealing has been proposed in [114]. We extend this model in Chapter 4.

The analysis presented in this chapter shows some similarities with the work of Berenbrink *et al.* [26]. It is based on computing the expected decrease of a potential function. However, to simplify the analysis, we introduce an adversary that controls one parameter of the model, the number of steal requests at each time step.

Another related approach which deals with load balancing is *balls into bins* games [9, 27].

Some works have been proposed for the analysis of algorithms in data structures and combinatorial optimization (including variants of scheduling) using potential functions. Our analysis is also based on a potential function representing the load unbalance between the local queues. This technique has been successfully used for analyzing basic load balancing [26] and WS [8] which improved the result of [40].

Chapter 4.

Asymptotic Behavior of Work Stealing in Large-Scale Systems

Abstract of this chapter – In this chapter, we consider a generic model of computational grids, seen as several clusters of homogeneous processors. We present a Markovian model for performance evaluation of the load balancing technique named *work stealing* (idle processors steal work from others). Using mean field theory, we show that when the size of the system grows, it converges to a system of deterministic ordinary differential equations that allows one to compute the expectation of performance functions (such as average response times) as well as the distributions of these functions.

We first study the case where all resources are homogeneous, showing in particular that work stealing is very efficient, even when the latency of steals is large. We also consider an heterogeneous case: the system is made of several clusters, and stealing within one cluster is faster than stealing between clusters. We compare different work stealing policies.

Résumé du chapitre – Dans ce chapitre, nous présentons un modèle de grille de calcul, grace auquel nous analysons les performances du *vol de travail*, une stratégie de répartition de ressource. En utilisant la théorie du champs moyen, nous montrons que quand la taille du système grandit, le système converge vers un système d'équations différentielles qui permettent de calculer efficacement des indicateurs de performances, à la fois en moyenne mais aussi les distributions de ces fonctions.

Nous nous intéressons d'abord au cas où le système est homogène (composé d'un seul cluster), et montrons en particulier que les performance du vol de travail sont très bonnes, même quand la latence est élevé. Puis nous nous intéressons à un système plus hétérogène et nous étudions plusieurs stratégies de vol.

4.1. Introduction

A key issue when exploiting large-scale computer systems is to efficiently distribute the workload among the different resources.As described in the previous chapter, work stealing is a classical load balancing strategy that tries to tackle this problem. Its main principle is that when a resource becomes idle, it chooses another resource and *steals* part of its work. The choice of the resource to steal from depends on the implementation of work stealing. Work stealing have received a considerable amount of attention in the past. Some of these results are summarized at the end of the previous chapter, in Section 3.9.

In the previous chapter, we presented a simple model to study the total completion time of a bag of tasks scheduled by work stealing. Using a potential function, we obtained a bound on the average number of steal requests needed to complete the whole bag of tasks. However, this analysis only focuses on a worst case scenario and does not provide a analysis of the performance of work stealing when there are arrivals of tasks or when the parameters (such as the average time needed to steal tasks) change.

In [134], a continuous time Markov model of work stealing over a small number of processors is analyzed using a decoupling assumption. In [6], a numerical method based on a M/G/1 model of a work stealing system is presented. These two approaches do not scale with the number of processors and become intractable with more than 10 processors.

In this chapter, we consider the case where the number of processors N is large. This is important in practice if work stealing is to be used in computational grids. We use mean field techniques to derive a system of ordinary differential equations that is the limit of a Markovian system when N goes to infinity. Such techniques have been used for the first time in [114] where the author derives differential equations for a system of homogeneous processors who steal a single job when idle. Here, we consider the case where, at every steal, half of the jobs in the queue of the victim are stolen. This strategy is closer to what is actually implemented in available work stealing libraries and is also much more efficient, as shown in the experimental section. It also makes the resulting model (both the Markov chain and the differential limit) more complicated because it contains non-local dependencies. Another important difference with [114] is the fact that we consider the case where the geometry of the system is taken into account as well as the heterogeneity of the processors. Communication times between processors in the same cluster are homogeneous, however stealing from a remote cluster takes more time and depends on the distance between the clusters. Finally, we also provide a new numerical technique that allows one to sample the distribution of the response times of one job, rather than just compute the average response time. This is more useful in practice because it provides guaranties to the users.

The chapter is structured as follows. In Section 4.2, we describe a working stealing model in computational grids and the corresponding Markov process that falls in the category of density dependent population processes.

Section 4.3 discusses the convergence properties of the work stealing process when the number of processors goes to infinity. We provide the limit under the form of a set of deterministic ordinary differential equations (ODEs).

Section 4.4 focuses on the case where the system is made of a single cluster of homogeneous processors. We show that the ODEs have a single equilibrium point and we provide bounds on the speed of convergence to the mean field limit in extreme cases. A fast numerical method to compute the equilibrium point is given as well as a fast simulation algorithm to sample the distribution of the response times of jobs.

Finally, Section 4.5 extends the analysis to the case where processors are partitioned into several clusters. We study several scenarios (clusters with equal or unbalanced loads, master-slave cases) and we provide insight on the best stealing strategies. For example, is it worth stealing from remote clusters to balance the load at the expense of large stealing costs? So far, such issues have only been investigated experimentally in [124, 146] where real tests (with a small number of processors) have been conducted. This study shows that there is no universal answer to this question. Nevertheless, the rule of thumb is that the load factor is more critical to performance than the cost of stealing.

4.2. Work Stealing in Grids

4.2.1. Computational grids

The motivation of this work comes from computational grids. Such grids abound and have often replaced single supercomputers to provide high performance computing at a lower cost. The main architectural feature of computational grids is the fact that they are made of several clusters. Each cluster is composed of a large number of homogeneous processors, with homogeneous communications inside the cluster. However clusters are all different and the speed of communication between clusters depends on the couple of clusters considered. A typical example is Grid 5000 [49], an experimental grid deployed in Brazil, France and Luxemburg. The French part is made of 9 clusters spread all over the country, the most recent one being Digitalis (digitalis.inria.fr), a cluster of 700 Intel Nehalem processors over an InfiniBand internal network at 40 Gb/s. The clusters are inter-connected by a dedicated network at 10Gb/s. The parameters used in this chapter all come from measurements made on Grid 5000.

Users of computational grids submit jobs to a selected cluster, or to a central broker. Jobs are allocated to processors by a *batch scheduler* (oar.imag.fr for Grid 5000) whose role is to minimize the response times (also called sojourn times) of jobs *i.e.*, the time spent by the jobs in the system. The goal of this work is to analyse the performance of such systems when a work-stealing strategy is used by the scheduler.

4.2.2. Work stealing model

Let us consider a model of a computational grid made of N processors. Here, a processor represents a generic computing unit such as a single CPU core, an entire CPU (*i.e.*, all cores in the CPU), an entire GPU, or a multi-CPU/GPU server. The processors are grouped into C clusters. Each cluster is composed of a large number of homogeneous processors with homogeneous communications inside the cluster. However, clusters are heterogeneous in size and processing speeds, and communications between clusters depend on their distance as well as the network type.

We consider that each processor in each cluster c receives jobs from outside. The jobs arrive according to a Poisson process of rate λ_c. Each job has a size that is exponentially distributed with mean 1. Actually, more general distributions can be taken into account, as long as they can be represented by finite Markov processes (such as Cox distributions). We further assume that tasks are independent and cannot be divided to be executed by two different processors. The processors in cluster c have a speed μ_c: the time taken by one of these processors to serve a job of size S is S/μ_c.

If a processor has more than one job to serve, it stores them in a buffer of maximal capacity K. If a job arrives when the buffer is full, it is discarded. We denote by $j_n(t)$ the number of jobs present in the nth processor. By definition, $j_n(t) = 0$ means that the processor has no job to serve and that its buffer is empty. Otherwise $j_n(t)$ is the number of jobs in its buffer plus one (corresponding to the job it is currently serving). From a queuing theory point of view, this means that all processors can be seen as $M/M/1/K$ queues.

If the processor i has no job to serve, it tries to steal some jobs from a different processor, called the victim. For that, it selects another processor, say k, at random (according to probabilities, defined later) and asks for jobs. This operation may take some time (exponentially distributed). If after this time the processor i is still idle and if k has $j_k \geq 2$ jobs, the jobs of processor k are shared between processors i and k: $\lfloor j_k/2 \rfloor$ for i and the rest stay in k. If i is in cluster c_i and k in cluster c_k, we consider that the time to wait for the answer is exponentially distributed of mean $1/\gamma_{c_i c_k}$. The selection of the victim is not completely uniform: the processor first selects a cluster according to the probability law $p_{cc'}$ and then picks uniformly a processor in this cluster. For simplicity, we neglect the time to transfer a job and only take into account the time to get an answer. This means in particular that the time to steal jobs does not depend on the number of stolen jobs but only on the localization of the two processors (*i.e.*, the clusters they belong to).

4.2.3. A Density dependent population process

Recall from Section 2.2.1 that a sequence of Markov processes M^N on $\frac{1}{N}\mathbb{N}^d$ $(d \geq 1)$ is called a density dependent population process if there exists a finite number of transitions, say $\mathcal{L} \subset \mathbb{N}^d$, such that for each $\ell \in \mathcal{L}$, the rate of transition from M^N to $M^N + \ell/N$ is $N\beta_\ell(M^N)$, where $\beta_\ell(.)$ does not depend on N.

This model has been well studied in the literature. In particular, the work of Kurtz [62, 103] shows that as N grows, the behavior of the system goes to a deterministic limit. Moreover, this limit satisfies a set of ordinary differential equations that can be derived from the transition rates. In Chapter 2 Section 2.2.1, we presented some convergence results for this model. These results are adapted to our case in Section 4.3. Here, we will show that our system can be described by a density dependent population process.

The state of a processor is given as follows. Let $\mathcal{C}$ be set of clusters $\mathcal{C} \stackrel{\text{def}}{=} \{1, \ldots, C\}$ and $\mathcal{K}$ be the set of buffer sizes: $\mathcal{K} \stackrel{\text{def}}{=} \{0, \ldots, K\}$. If the processor p belongs to cluster c_p and has $j_p \in \mathcal{K}$ jobs in its queue, its state is (c_p, j_p). Note that the cluster c_p of the processor p is fixed by the geometry of the system. If the processor p has no job in its queue, this means that it is trying to steal some jobs from another processor, say q. If p is in cluster c_p and q in cluster c_q, then the state of p is $(c_p, 0, c_q)$. Finally, $M^N_{c_p j_p}$ denotes the proportion of processors in state (c_p, j_p) and $M^N_{c_p 0 c_q}$ the proportion of processors in state $(c_p, 0, c_q)$.

Proposition 4.1. *$(M^N_{cj}, M^N_{c0c'})_{c,c' \in \mathcal{C}, j \in \mathcal{K}}$ is a continuous time Markov process on $\mathbb{R}^{CK+C^2}$. Moreover, the sequence of Markov processes M^N is a density dependent population process.*

Proof. Let us fix N and assume that at time t, the system is in state $M^N(t) = (M_{cj}(t) \ldots M_{c0c'}(t))_{c,c' \in C, j \in \mathcal{K}}$. There are four types of events that can happen:

arrivals, departures, successful steals and unsuccessful steals. Let t' be the time of the next event and let us compute the rate at which each event occurs.

If an *arrival* occurs on a processor of cluster c that has $1 \leq j \leq K-1$ jobs, the corresponding modification of the state will be that $(M_{cj}, M_{c,j+1})$ will become $(M_{cj}(t) - 1/N, M_{c,j+1}(t) + 1/N)$. The jobs arrive according to a Poisson process of rate λ_c. This means that this transition occurs at rate $NX_{cj}(t)\lambda_c$. Similarly, an arrival on a processor of cluster c that has 0 jobs and is stealing from c' occurs at rate $NX_{c0c'}(t)\lambda_c$.

The case of *departures* is similar for processors that have $j \geq 2$ jobs. If a processor has 0 jobs, no departure is possible. When there is a departure from a processor of cluster c that has 1 job, this processor chooses a cluster c' to steal from (with probability $p_{cc'}$) and then a processor among them, uniformly. Therefore, $(M_{c1}, M_{c0c'})$ becomes $(M_{c1} - 1/N, M_{c0c'} + 1/N)$ at rate $NX_{c1}\mu_c p_{cc'}$.

Once a processor of cluster c is empty and has chosen a victim cluster, namely c', it asks for work to steal and gets its response with rate $\gamma_{cc'}$. If $M_{c'}$ is the proportion of processors in cluster c', this is equivalent to saying that the processor gets a response from each processor of cluster c' with a rate $\gamma_{cc'}/(NX_{c'})$. If the victim in cluster c' has j jobs, we distinguish two cases. If $j \geq 2$, the steal is *successful* and the processor gets $\lfloor j/2 \rfloor$ jobs. If $j = 0$ or 1, the steal is *unsuccessful* and the processor has to choose a new processor to steal from. Thus we can write the two following transitions:

- The *successful steal* of the jobs of a processor in cluster c' with j jobs from a processor in cluster c occurs with rate $N\gamma_{cc'}M_{c0c'}M_{c'j}/M_{c'}$ and changes $(M_{c0c'}, M_{c\lfloor j/2 \rfloor}, M_{c'j}, M_{c'\lceil j/2 \rceil})$ in $(M_{c0c'} - 1/N, M_{c\lfloor j/2 \rfloor} + 1/N, M_{c'j} - 1/N, M_{c'\lceil j/2 \rceil} + 1/N)$.

- The *unsuccessful steal* of a processor in cluster c trying to steal from cluster c' occurs when it steals 0 jobs. After that it chooses to steal from cluster c''. This event occurs with rate $N\gamma_{cc'}p_{cc''}M_{c0c'}(M_{c'0} + M_{c'1})/M_{c'}$ and changes $(M_{c0c'}, M_{c0c''})$ in $(M_{c0c'} - 1/N, M_{c0c''} + 1/N)$. □

4.2.4. Examples

To illustrate the power of expression of our model, we present some examples that will be studied.

- **Homogeneous cluster** – in this case, all processors are homogeneous and each processor receives jobs at the same rate. This model is studied in detail in Section 4.4. We show that the steady state can be computed by a simple algorithm and we compute the main performance indicators.

- **Two homogeneous clusters** – we consider in Section 4.5.1 the case of two homogeneous clusters: they have the same parameters λ, μ. However, the rate of steal is 10 times larger inside a cluster than between clusters: $\gamma_{ii} = 10\gamma_{ij}$ if $i \neq j$.

- **Two heterogeneous clusters** – there is again two clusters and stealing is faster inside the cluster: $\gamma_{ii} = 10\gamma_{ij}$. The clusters are homogeneous in speed but one is more loaded than the other: $\lambda_2/\mu_2 > \lambda_1/\mu_1$. Section 4.5.2 studies the optimal stealing probability p_{ij}.

- **Master-Worker** – in this case, we consider a network of homogeneous clusters but the arrivals only occur in a fraction of the processors (called the masters). This

is modeled by using two clusters per real cluster (one for the masters, one for the slaves) with the same parameters and with $\gamma_{ij} = \gamma_{ii}$ inside these two clusters. In Section 4.5.3, we study the performance of the system as a function of the fraction of masters in the system.

4.3. Mean Field Approximation

In Section 4.2.3 $(M^N_{c0c'}(t), M^N_{cj}(t))_{c,c' \in C, 1 \leq j \leq K}$ is proved to be a density dependent population process. In Chapter 2, Section 2.2.1, we have seen some convergence results of density dependent population processes when the number of objects goes to infinity. In particular, Theorem 2.1 shows that if $M^N(0)$ converges weakly to a deterministic measure $m(0)$, then $(M^N(s))_{0 \leq s \leq t}$ converges weakly to the solution of the differential equation $\dot{m}(t) = F(m)$ over $[0, t]$. In our case, the system of ordinary differential equations $\dot{m}(t) = F(m)$ has the following form.

$$\dot{m}_{c0c'} = -(\lambda_c + \gamma_{cc'})m_{c0c'} + \mu_c m_{c1} p_{cc'} + \sum_{c''} \gamma_{cc''} m_{c0c''} \frac{m_{c''0} + m_{c''1}}{m_{c''}} p_{cc'} \tag{4.1}$$

$$\dot{m}_{c1} = -(\mu_c + \lambda_c)m_{c1} + \mu_c m_{c2} + \sum_{c'} \lambda_c m_{c0c'} \tag{4.2}$$

$$+ \sum_{c'} \gamma_{c'c} m_{c'0c} m_{c2}/m_c + \sum_{c'} \gamma_{cc'} m_{c0c'}(m_{c'2} + m_{c'3})/m_{c'} \tag{4.3}$$

$$\dot{m}_{cj} = -(\mu_c + \lambda_c \mathbf{1}_{j<K})m_{c,j} + \mu_c m_{c,j+1} + \lambda_c m_{c,j-1} \tag{4.4}$$

$$+ \sum_{c'} \gamma_{c'c} m_{c'0c}(m_{c,2j} + m_{c,2j-1})/m_c \tag{4.5}$$

$$+ \sum_{c'} \gamma_{cc'} m_{c0c'}(m_{c',2j} + m_{c',2j+1})/m_{c'} \tag{4.6}$$

$$- \sum_{c'} \gamma_{c'c} m_{c'0c} m_{cj}/m_c, \tag{4.7}$$

where $\mathbf{1}_{j<K}$ equals 1 for $j < K$ and 0 for $j \geq K$. Other boundary conditions are $m_{cj} = 0$ for $j > K$.

These equations can be directly computed from the transitions described in Section 4.2.3. They can be interpreted as follows.

The first term in line 4.1 is the rate at which processors exit from state $(c, 0, c')$: This happens if an arrival occurs (λ_c) or if a steal from c' occurs $(\gamma_{c0c'})$. The second term corresponds to the rate at which processors end up in state $(c, 0, c')$. The line (4.2) of this equation represents the fraction of processors that were in cluster c trying to steal jobs from cluster c'' and that did not succeed and but decided to steal from the cluster c'. The following lines have a similar interpretation. For the last equation, line (4.6) represents the processors in cluster c that had $2j$ or $2j + 1$ jobs and have been stolen by someone else and line (4.6), the processors from cluster c that are stealing from others. The last line (4.7) represents the processors in cluster c that had j jobs and have been stolen by someone else.

The two technical conditions for applying Theorem 2.1 are clearly satisfied: The function F in the differential equation is a rational function of degree 2 and is Lipschitz on all compacts. The transition set $\mathcal{L}$ is finite and the transition rate β_ℓ are bounded

so that the second condition is also satisfied. Theorem 2.1 can be rewritten in this framework as:

Corollary 4.2. *Using, the foregoing notations, if $M^N(0)$ converges to $m(0)$ in probability, then $\sup_{0\le t\le T} |M^N - m(t)| \to 0$ in probability.*

Section 2.2.1 also recalled other results of convergence concerning the behavior of the steady state of the system. The Markov chain with N processors is clearly irreducible aperiodic and therefore the stochastic process with N processors has an invariant measure, say π^N. A natural question is whether π^N converges (or not) to a fixed point of F and whether second order results exist in that case.

In general, convergence for the steady state only holds under several restrictions (see Proposition 2.3 in Chapter 2). The interesting situation for Proposition 2.3 if when F has a unique stationary point m^* to which all trajectories converge. In that case, the stationary measures π^N concentrate around m^* as N goes to infinity:

$$\lim_{N\to\infty} \pi^N \xrightarrow{\text{weak}} \delta_{m^*},$$

see Corollary 2.4 of Chapter 2.

In the following we will show that the system of equations (4.1 - 4.7) has a unique fixed point. However, to apply Corollary 2.4, one needs to show that this point is a global attractor, which is a very difficult task for the set of equations (4.1 - 4.7) that do not admit a natural Lyapounov function. Unfortunately, we were unable to prove that this fixed point is an attractor, although this is what numerical experiments suggest.

Moreover, in the case where a linear Lyapounov function exists, second order results for the steady state behavior exist (see Section 2.2.1). Unfortunately, we have not been able to construct such a scalar product for our case. Partial results on second order results are shown in Section 4.4.2.

4.3.1. Fast simulation algorithm

The previous theorem shows that the average number of processors in each state can be approximated by a system of differential equations. Here we show that as N grows large, the behavior on one particular processor can be approximated by a simpler process.

Let us consider a system with a finite number of processors $N < \infty$. Let $J^N(t)$ be the state of one particular processor at time t. It should be clear that the process $(J^N(t), M^N(t))$ is a continuous time Markov chain. For each population value m, we define the kernel $(K_{jj'}(m))_{j,j'\in\mathcal{C}\cup\mathcal{K}}$ as follows. If the population process $M^N(t)$ is m and $j, j' \in \mathcal{K}$, then $K_{jj'}(m)$ is the rate of transition of $J^N(t)$ from (c, j) to (c, j'). The definition for $j \in \mathcal{K}$ and $j' \in \mathcal{C}$ is similar, representing the rate of transition from (c, j) to $(c, 0, j')$. The transition kernel $K(m)$ can be directly derived from the transitions written in Section 4.2.3 and is illustrated by Figure 4.1.

For finite N, the behavior of the processor $J^N(t)$ is not independent of the behavior of $M^N(t)$ as each transition of $J^N(t)$ will result in a change of $M^N(t)$. The process $J^N(t)$ is not Markovian and is very complicated. In the limit however, $J^N(t)$ goes to a non-homogeneous Markovian process.

Theorem 4.3. *Let us assume that $\lim_{N\to\infty} J^N(0) = j(0)$ and $M^N(0) \to m(0)$. Then $(J^N(t), M^N(t))$ converges weakly to a continuous time jump and drift process $(J(t), m(t))$*

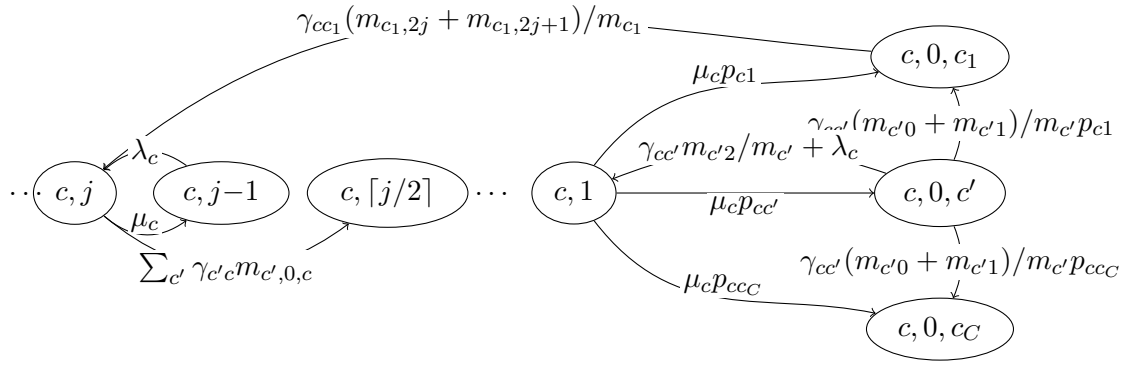


Figure 4.1.: Graph of the transition kernel of the state of one processor for the fast simulation algorithm. Due to the numerous transitions, only a fraction of the transitions are represented on this graph.

where $m(t)$ satisfies the ODE (4.1-4.7) and $J(t)$ is a non-homogeneous jump process of kernel $K(m(t))$.

Proof. (sketch) This result is similar to Theorem 3.2.1 of [142] and can be proved using similar ideas. Conditionally to M^N, J^N is a non-homogeneous Markovian process with kernel $K(M^N(t))$. Since $M^N(t)$ converges in probability to $m(t)$, $\lim_{N\to\infty} K(M^N(t)) = K(m(t))$. Finally, the convergence of J^N to J comes from Theorem 17.25 of [90]. □

From a practical point of view, this theorem is important because it allows one to use the mean field approximation to compute distributions instead of average values. Moreover, the case of steady-state is of particular interest. Assume that the system of ODE has a unique stable point m^* to which all trajectories converge and that $M^N(0)$ is chosen according to the steady state distribution. In that case, $M^N(t)$ is distributed according to the steady state distribution and $\lim_{N\to\infty} M^N(t) = m^*$ (Corollary 2.4). Theorem 4.3 shows that the behavior of one processor chosen at random converges to a continuous time Markov chain of transition kernel $K(m^*)$. In Section 4.4.5, we will see how to use this result to compute the distribution of sojourn times.

4.4. One Cluster Model

In this part, we focus on the case where all processors belong to one homogeneous cluster. The system is described by 3 parameters: the arrival rate λ, the service rate μ and the rate of stealing γ.

Algorithms 4.1 and 4.2 provide very efficient ways to perform a steady state analysis. The total time to generate all curves of this section is less than ten minutes on a desktop computer.

4.4.1. Steady state limit

In this section, we show that the differential equations (4.1-4.7) without the boundary conditions ($j \leq K$) have a unique equilibrium point. We will also show that this relaxation is justified when $\lambda < \mu$, because the number of queues with more than j jobs decreases as α^j with $\alpha < 1$. In the rest of this section, we assume that $\lambda < \mu$ to ensure stability.

An equilibrium point must satisfy the equation $\dot{m}_j(t) = 0$ for all $j \in \mathbb{N}$. With a single cluster, this can be expressed as:

$$1 = \sum_{j=0}^{\infty} m_j, \tag{4.8}$$

$$0 = -\lambda m_0 + \mu m_1 - \gamma m_0 \sum_{j=2}^{\infty} m_j, \tag{4.9}$$

$$\begin{aligned} 0 = &-(\lambda + \mu + \gamma m_0) m_j + \lambda m_{j-1} + \mu m_{j+1} \\ &+ \gamma m_0 (m_{2j-1} + 2x_{2j} + m_{2j+1}), \end{aligned} \tag{4.10}$$

where Equation (4.10) holds for all $j \geq 1$.

The variation of the number of jobs in steady state is $\sum_{j=0}^{\infty} j\dot{m}_j(t) = 0$. By a direct computation, this leads to $\lambda - \mu \sum_{j=1}^{\infty} m_j = 0$ and using (4.8), $m_0 = 1 - \lambda/\mu$. Moreover, using Equation (4.9), $m_1 = \lambda(1-\lambda)\frac{\gamma+\mu}{(1-\lambda)\gamma+\mu^2}$.

Therefore solving the whole system of equations is the same as solving a linear system with free variables $(m_j)_{j\geq 1}$. This system can be rewritten as a matrix equation $AX = Y$ where A is of the form:

$$\begin{bmatrix} -(\lambda+\mu) & \mu + 2\gamma m_0 & \gamma m_0 & & & & & \\ \lambda & -(\lambda+\mu+\gamma m_0) & \mu + \gamma m_0 & 2\gamma m_0 & \gamma m_0 & & & \\ & \lambda & -(\lambda+\mu+\gamma m_0) & \mu & \gamma m_0 & 2\gamma m_0 & \gamma m_0 & \\ & & \lambda & -(\lambda+\mu+\gamma m_0) & \mu & & \cdots & \cdots \\ \vdots & \vdots & & & \ddots & & & \end{bmatrix},$$

The main diagonal is composed of $-(\lambda+\mu+\gamma x_0)$, the first diagonal below it is composed of λ and the first diagonal above is composed of μ. To this, one has to add $2\gamma m_0$ to the cases $(j, 2j)$ and γm_0 to the cases $(j, 2j-1)$ and $(j, 2j+1)$ (thus the top left term is $-(\lambda+\mu)$. Let x be the vector defined by $x_1 = 1$ and $x_j = \frac{1}{\lambda}\sum_{i=1}^{j-1} A_{ji}x_i$. All the solutions of $A^T X = 0$ can be written $c \cdot x$ for some $c \in \mathbb{R}$. Therefore the dimension of the kernel of the matrix A is 1. This shows that there is a unique solution of the system of Equations (4.8-4.10).

So far, however, there is no guarantee that this fixed point is non-negative. To prove this, we designed an iteration algorithm over non-negative sequences that converges to the fixed point.

Require: λ, μ, γ.
$m_0 \leftarrow 1 - \lambda/\mu$
$m_1 \leftarrow \lambda(1-\lambda)\frac{\gamma+\mu}{(1-\lambda)\gamma+\mu^2}$
$\forall j \geq 2 : m_j \leftarrow 0.$
repeat
$\forall j \geq 2$
$m_j \leftarrow \frac{1}{\lambda+\mu+\gamma m_0}\Big(\lambda m_{j-1} + \mu m_{j+1} + \gamma m_0 (m_{2j-1} + 2x_{2j} + m_{2j+1})\Big)$

Algorithm 4.1: Steady-state computation

Proposition 4.4. *The successive sequences $(m_j^t)_{j\in\mathbb{N}}$ computed in Algorithm 4.1 satisfy the following:*

(i) They converge to a sequence $(m_j^\infty)_{j\in\mathbb{N}}$ *with* $m_j^\infty \geq 0$.

(ii) There exists j^* *such that* m_j^∞ *is increasing with* j *up to* $m_{j^*}^\infty$ *and is decreasing after.*

(iii) $\forall \epsilon > 0$, $\lim_{j\to\infty} m_j^\infty/(\alpha+\epsilon)^j = \lim_{j\to\infty}(\alpha-\epsilon)^j/m_j^\infty = 0$,
where $\alpha = (\lambda+\mu+\gamma m_0 - \sqrt{(\lambda+\mu+\gamma m_0)^2 - 4\mu\lambda})/(2\mu) < 1$.

(iv) m_j^∞ *is the only solution of* (4.8)- (4.10).

This implies that m_j^∞ decreases with an exponential rate: $m_j \approx_{j\to\infty} c\alpha^j$.

Proof. (i) Let $(m_j^t)_{j\in\mathbb{N}}$ be defined by $m_0^t = 1-\lambda$, $\forall t \in \mathbb{N}$, $m_j^0 = 0$ and $m_j^{t+1} = \frac{1}{\lambda+\mu+\gamma m_0}\left(\lambda m_{j-1}^t + \mu m_{j+1}^t + \gamma m_0^t(m_{2j-1}^t + 2x_{2j}^t + m_{2j+1}^t)\right)$.

By induction on t, one can show that m_j^t is positive and increasing in t for all j. Moreover, at each t, there is only a finite number of m_j^t that are non 0 (the first t) and the quantities $y^t \stackrel{\text{def}}{=} \sum_{j\geq 0} m_j^t$ and $z^t \stackrel{\text{def}}{=} \sum_{j\geq 1} j m_j^t$ are well defined and finite. The recurrence equation leads to:

$$z^{t+1}(\lambda+\mu+\gamma m_0) = (1-y^t)(\mu-\lambda) + (\lambda+\mu+\gamma m_0)\, z^t. \tag{4.11}$$

Since m^t is increasing in t, z^t is also increasing in t and $1-z^t \geq 0$. This shows that $m_j^t \leq 1$. Since m_j^t is increasing in t, it converges to some $m_j^\infty \geq 0$ that satisfies (4.9)-(4.10).

(ii) Let j^* be the minimal j such that $m_{j^*}^\infty \geq m_{j^*+1}^\infty$ (it exists since $\sum_{i=0}^\infty m_j \leq 1$ and $\lim_{j\to\infty} m_j = 0$). We define a sequence y_j^t by $y_j^0 = m_j$ for $j \leq j^*$ and $y_j^0 = 0$ for $j > j^*$ and y_j^{t+1} is updated as in Algorithm 4.1 for $j > j^*$. By a direct induction, one can show that y_j^t is increasing in t and is less than m_j and decreasing for $j \geq j^*$. Therefore it converges to m_j^∞ and m_j^∞ satisfies *(i)*.

(iii) $\mu M^2 - (\lambda+\mu+\gamma m_0)M + \lambda = 0$ has two solutions α and $\bar{\alpha}$ with $0 < \alpha < 1 < \bar{\alpha}$. Moreover, it is positive on $(\alpha; 1)$. This shows that for $\delta \in (\alpha; 1)$ and $j \geq j_\delta$ (j_δ big enough),

$$\delta^i(\lambda+\mu+\gamma m_0) \leq \lambda\delta^{i-1} + \mu\delta^{i+1} + \gamma m_0\left(\delta^{2i-1} + 2\delta^{2i} + \delta^{2i+1}\right).$$

Let us define y_j^t by $y_j^t = m_j$ for $j \leq j_\delta$ and $y_j^0 = m_j\delta^{j-j_\delta}$ for $j > j_\delta$ and y_j^{t+1} is updated as in Algorithm 4.1 for $j > j_\delta$. y_j^t is decreasing and converges to m_j^∞, showing that $\lim m_j^\infty/(\delta+\epsilon)^j = 0$ for all $\delta > \alpha$ and $\epsilon > 0$. The other limit $\lim(\delta-\epsilon)^j/m_j = 0$ can be proved similarly.

(iv) since $\lim_{j\to\infty} m_j^\infty/(\alpha+\epsilon)^j = 0$, $\sum_{j=1}^\infty m_j^\infty < \infty$. This shows that when t goes to infinity, both parts of Eq. (4.11) go to a finite limit. This shows that $\sum_{j=0}^\infty m_j^\infty = \lim_t y^t = 1$. □

The shape of the fixed point computed by Algorithm 4.1 is displayed in Figure 4.2. *A posteriori*, this can be seen as a justification to consider the ODEs without boundaries since the solution is concentrated almost entirely on small values of j. For all the numerical simulations, we used $K = 500$ and the values of m_{cK} have always been less than 10^{-10}.

In order to apply Corollary 2.4, one need to show that this unique fixed point is also a unique attractor of the ODEs. Unfortunately, we were unable to prove this fact (although this is what our numerical experiments suggest). Therefore, our only assertion is that, whenever the steady state distribution of the finite stochastic system converges to a single point, it should converge to this fixed point.

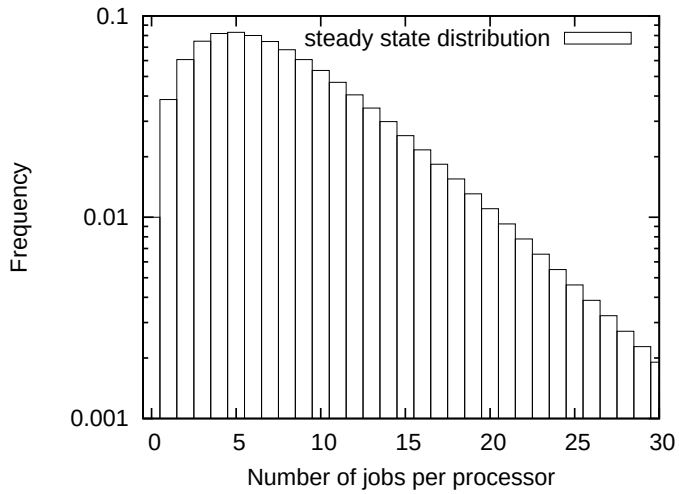


Figure 4.2.: General shape of the steady state distribution. Here for $\lambda = 0.99$, $\mu = 1$ and $\gamma = 3$. The m-axis is the number of jobs and the y-axis shows the fraction of processors with j jobs (in log-scale).

4.4.2. Gap between mean field and steady-state

In this section, we study two extreme values of γ for which we are able to compute exactly the steady state distribution and compare these results with the mean field approximation.

When $\gamma = 0$, no stealing ever occurs and the N processors behave like independent M/M/1/K queues. The steady state distribution Π_0 has a product form. In steady state, the probability of having j jobs in one M/M/1/K queue is $\pi_j = (\frac{\lambda}{\mu})^j \pi_0$ and satisfies the Equations (4.8-4.10). The steady state of the whole cluster is made of N independent variables picked according to this distribution. The law of large number shows that Π_0 converges to m almost surely and the central limit theorem shows that the speed of convergence is in $O(1/\sqrt{N})$. Also, the marginal of the distribution of the steady state for one processor is the mean field steady state.

When $\gamma = \infty$, the system can also be replaced by a simpler one. In that case, there is either no idle processor or no processor with 2 or more jobs since in that case an idle processor would instantly steal the second job. Thus the state of the system can be modeled by the total number of jobs in the system that we call $j^N(t)$. Moreover, $j^N(t)$ behaves like one $M/M/N/KN$ queue (*i.e.*, a queue with N independent processors with arrival rate $N\lambda$ and service rate μ for each processor). The probability that $j^N(t)$ is j is:

$$\begin{aligned}
\Pi_\infty(j) &= \Pi_\infty(0)\tfrac{N^j\lambda^j}{j!\mu^j} \quad (j < N),\\
\Pi_\infty(j) &= \Pi_\infty(0)\tfrac{N^j\lambda^j}{N!N^{j-N}\mu^j} \quad (KN \geq j \geq N),
\end{aligned}$$

where $\Pi_\infty(0)$ is a normalization constant.

As N grows, it can be shown that $P(j^N(t) \geq N) = o(\beta^N)$ and $\Pi_\infty(0) = \exp(-N\lambda) + o(\beta^n)$ with $(\beta = \lambda \exp(1-\lambda) \in (0;1))$. Let us compute the characteristic function $\Phi()$ of the steady state distribution of $(j^N(t) - N\lambda)/\sqrt{N}$:

$$\Phi(\xi) = \sum_{j=0}^{\infty} \Pi_\infty(j) \exp(i\xi \frac{j - N\lambda}{\sqrt{N}}) = \exp(-\lambda\xi^2 + O(\frac{1}{N})).$$

Therefore $\sqrt{N}(j^N(t)/N - \lambda/\mu)$ converges to a Gaussian law. In that case, the gap between the steady state of the system of finite N and the mean field is of order $1/\sqrt{N}$.

For both extremal cases $\gamma = 0$ and $\gamma = \infty$, the gap between the mean field approximation and the real steady state is of order $O(1/\sqrt{N})$. We conjecture that this should also be the case for all γ in between.

4.4.3. Average sojourn time

The first set of experiments given in Figure 4.3 measures the effect of the cost of stealing, $1/\gamma$, on the average sojourn time of the jobs (the average time spent by a job in the system).

Let $S_\lambda(\gamma)$ be the average sojourn time of a job in the limit system (in steady-state). By Little's formula, the number of jobs $L_\lambda(\gamma)$ verifies $L_\lambda(\gamma) = \lambda S_\lambda(\gamma)$, therefore it suffices to compute the average number of jobs in steady state. As mentioned in the previous part, the analytical computation of the steady-state seems impossible to do when N is large and we were not able to compute analytically the whole curve $S_\lambda(\gamma)$. Nevertheless we can compute two interesting points for $\gamma = 0$ and $\gamma = \infty$.

When $\gamma = 0$, the system is just a system of N independent $M/M/1/K$ queues and the average sojourn time is a classical result of queuing theory (*e.g.*, [11]). When K is large, this is approximately

$$S_\lambda(0) = \frac{1}{\mu - \lambda}. \tag{4.12}$$

The second quantity that we can compute is the limit of $S_\lambda(\gamma)$ when γ goes to infinity. In that case the steady state is composed of processors with either 0 or 1 job. A new job entering in the system either arrives directly on an empty processor or arrives in an occupied processor and is immediately stolen by an empty processor (there are empty processors with probability one), and the average sojourn time is

$$S_\lambda(\infty) = \frac{1}{\mu}, \tag{4.13}$$

and the average number of jobs in the system is $L_\lambda(\infty) = \frac{\lambda}{\mu}$.

Figure 4.3 displays the average sojourn time $S_\lambda(\gamma)$ as a function of γ for various values of λ. As expected, the average sojourn time is decreasing from $1/(1-\lambda)$ to $1/\mu = 1$. In particular, we can see that when γ is small, the average number of jobs in the system decreases drastically.

The gap between $S_\lambda(0)$ and $S_\lambda(\infty)$ is $\lambda/(1-\lambda)$. Therefore the gain obtained by work stealing is more important when the system is more congested (*i.e.*, λ is close to 1) as we can see in Figure 4.3. To provide a better estimate of the gain from using work stealing, Figure 4.4 shows the difference between the response time with a finite γ and $\gamma = \infty$ $S_\lambda(\gamma) - 1$ divided by the maximum gain $S_\lambda(0) - S_\lambda(\infty)$. One can observe that when the rate of stealing is of the same order of the service rate $\gamma = 1$ (resp. $\gamma = 4$ or $\gamma = 8$), the gain is 50% (resp. 80% or 90%) of what one could gain with a work stealing at no cost. This makes work stealing very efficient in real life systems since the time to steal a job is typically small compared to the service time of a job.

These facts can be explained by studying the behavior of m_j when γ grows. Intuitively, as γ goes to infinity, the distribution m_j concentrates in $m_0 = 1 - \lambda/\mu$ and $m_1 = \lambda/\mu$. Moreover, m_j decreases quickly. In Equation 4.10, all term in $m_{j+1}, m_{2i-1} \dots$ becomes

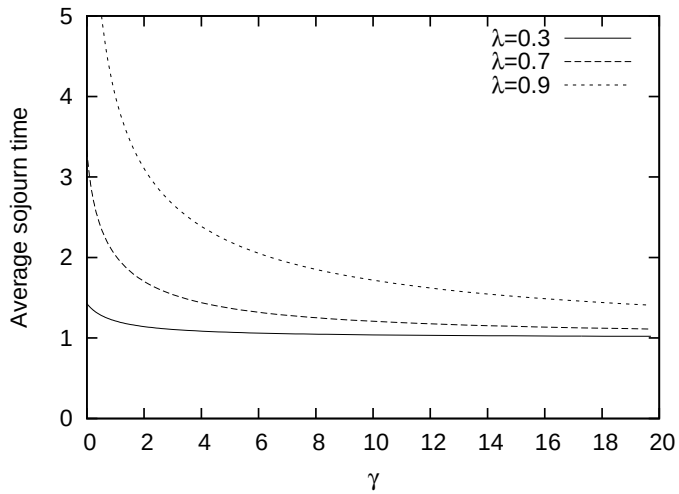


Figure 4.3.: Average sojourn time as a function of the rate of stealing γ for various values of λ (0.3, 0.7 and 0.9).

negligible compared to m_{i-1} and m_i leading to

$$0 = -\gamma m_0 m_j + \lambda m_{j-1}.$$

This gives the intuition that $m_j \approx \left(\frac{\lambda}{m_0\gamma}\right)^{j-1} m_1$ and is confirmed by the following proposition.

Proposition 4.5. *(i) For all i, $m_j = \left(\frac{\lambda}{m_0\gamma}\right)^{j-1} m_1 + O_{\gamma\to\infty}\left(\gamma^{-j}\right)$*

(ii) The mean number of jobs in the system is $\sum_{j=1}^{\infty} j m_j = \lambda/\mu + \frac{\lambda}{m_0\gamma} + O\left(\gamma^{-2}\right)$.

Proof. (i) Following the proof of Proposition 4.4-*(ii)*, one can show that for all $\epsilon > 0$, there exists γ^* such that $\gamma > \gamma^*$ implies that for all $j \geq i \geq 1$

$$m_j \leq \left(\frac{\lambda}{m_0\gamma} + \epsilon\right)^{j-i} m_i.$$

In particular, this shows that for all $i \geq j+1$, $m_i = O(\gamma^{-j})$.

If we assume that for some $j \geq 1$, $m_j = \left(\frac{\lambda}{m_0\gamma}\right)^{j-1} m_1 + O_{\gamma\to\infty}\left(\gamma^{-j}\right)$, then by Equation 4.10 and the previous remarks:

$$\begin{aligned} m_{j+1} &= \frac{\lambda}{\lambda+\mu+\gamma m_0} m_j + O(\gamma^{-j}) \\ &= \frac{\lambda}{\gamma m_0}\left(\left(\frac{\lambda}{m_0\gamma}\right)^j + O(\gamma^{-j})\right)\left(1 + O(\gamma^{-1})\right) \\ &= \left(\frac{\lambda}{m_0\gamma}\right)^{j+1} + O(\gamma^{-(j+1)}). \end{aligned}$$

(ii) Using the fact that $\sum_{j=0}^{\infty} m_j = 1$, we have $m_1 = 1 - m_0 - m_2 + O(\gamma^{-2})$. Therefore,

$$\sum_{j=0}^{\infty} j m_j = 1 - m_0 + m_2 + O(\gamma^{-2}) = \lambda/\mu + \frac{\lambda}{m_0\gamma} + O\left(\gamma^{-2}\right).$$

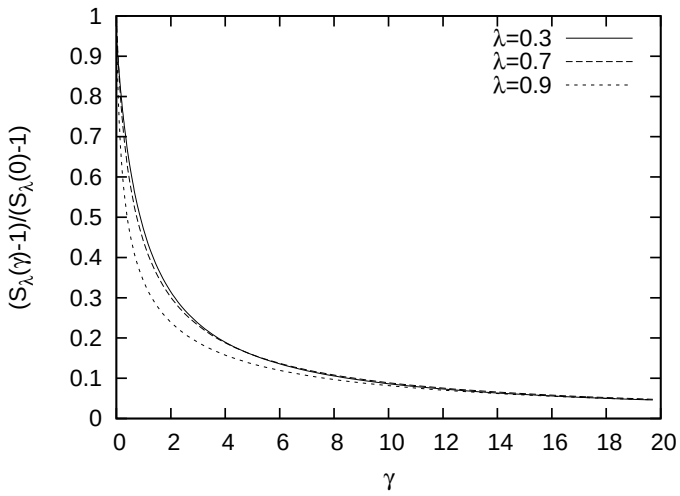


Figure 4.4.: Gain of work stealing as a function of γ.

□

4.4.4. Average number of steals

We want to study the average number of steals that a job undergoes, as a function of γ. Similar to the previous case, let $S_\lambda(\gamma)$ be the average number of steals per job. Again, it is impossible to solve the problem analytically but we can compute the two extremal values for $\gamma = 0$ and $\gamma = \infty$. When $\gamma = 0$, no job is stolen and $S_\lambda(0) = 0$. When $\gamma = \infty$, all jobs arriving in the processors with 1 job are stolen while the jobs entering in the processors with 0 jobs are not stolen. Thus $S_\lambda(\infty) = \lambda/\mu$.

For $0 < \gamma < \infty$, we can compute $S_\lambda(\gamma)$ numerically. The rate of steals from processors with j jobs is $\gamma m_0 m_j$ (where m_j corresponds to the number of processors with j jobs in steady state); each steal corresponding to $\lfloor j/2 \rfloor m_j$ jobs stolen. On average there are $\gamma m_0 \sum_{j=2}^{\infty} \lfloor j/2 \rfloor m_j$ jobs stolen per unit of time. Since the rate of arrivals is λ, we have

$$S_\lambda(\gamma) = \frac{\gamma}{\lambda} m_0 \sum_{j=2}^{\infty} \lfloor \frac{j}{2} \rfloor m_j.$$

Figure 4.5 shows that the number of steals increases when the cost of stealing decreases and converges to λ, as expected.

4.4.5. Distribution of the number of steals and sojourn time

In many practical cases, the average response time is not a good measure of performance and it is more important to improve the probability of being under a certain threshold. Using our fast simulation algorithm introduced in Section 4.3.1, we are able to sample the distribution of the number of steals and of the sojourn time in the steady state. Our simulations show that work stealing is indeed efficient at reducing the probability of having a large sojourn time.

To compute the distribution of sojourn times, we have to specify the order in which the jobs are served as well as which jobs are stolen when there is a steal. We consider

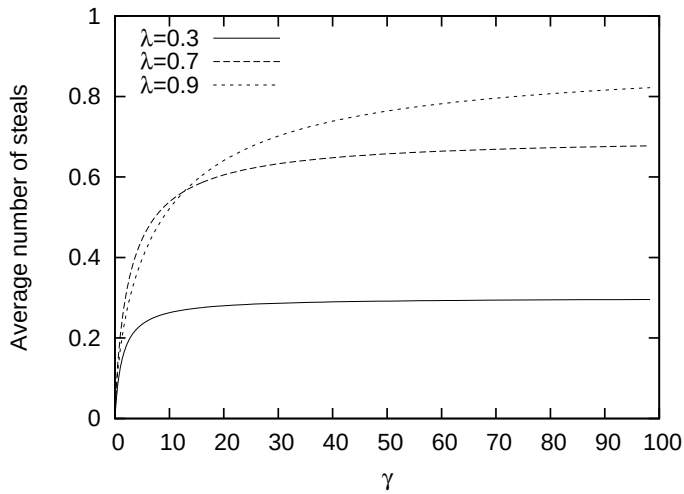


Figure 4.5.: Average number of successful steal per job $S_\lambda(\gamma)$ viewed as a function of γ for different values of λ (0.3, 0.7 and 0.9).

that the jobs are served in the FCFS order (First Come First Served). When there is a steal, the stealing processor steals the oldest jobs (except for the one that is being served) and the order of the jobs in the processor is preserved *i.e.*, if the jobs in the first processor are $\{1, 2 \dots j\}$ then after the steal the remaining jobs in the victim processor will be $\{1, 2 + \lfloor j/2 \rfloor \dots j\}$ and the jobs in the stealing processor will be $\{2 \dots \lfloor j/2 \rfloor + 1\}$.

Let us now consider a job arriving in the system. Let size_of_queue and place_in_queue be respectively the size of the queue and the place in the queue of this job, and let us consider the next event. If this event is an arrival (respectively a departure), then size_of_queue is increased by 1 (resp. both size_of_queue and place_in_queue are decreased by 1). If the event is a steal, then if place_in_queue $\in [2 \dots \lfloor \text{size_of_queue}/2 \rfloor]$ then the variable size_of_queue becomes $\lfloor \text{size_of_queue}/2 \rfloor$ and place_in_queue decreases by 1. Otherwise, size_of_queue becomes $\lceil \text{size_of_queue}/2 \rceil$ and the variable place_in_queue is decreased by $\lfloor \text{size_of_queue}/2 \rfloor$. The three events occur respectively with rates $\lambda, \mu, \gamma m_0$.

Using these considerations, the sojourn time of a job entering the system as well as the number of steals can be simulated by Algorithm 4.2.

We ran several simulations for various values of λ and γ. The percentage of jobs that undergo two steals or more is displayed in Figure 4.6 as a function of γ. Notice that in all cases, the distribution of the number of steals is mostly concentrated in 0 or 1 (less than 2% of the jobs are subjected to more than two steals). Moreover, the shape of the curve is the same for all λ: it starts from 0 when $\gamma = 0$; quickly reaches its maximum and then decreases slowly as γ goes to infinity. The main consequence of this is that there are few useless steals (if a job is stolen twice, then the first steal was actually useless).

Another interesting measure is the sojourn time distribution. When $\gamma = 0$ (*i.e.*, in the M/M/1/K case), the sojourn time distribution is exponential of parameter $\mu(1 - \lambda/\mu)$ when K goes to infinity. When γ goes to infinity, the sojourn time distribution follows an exponential law of parameter μ. Using our fast simulation algorithm, we can also sample the distribution of the sojourn time.

The sojourn time of a job in the system is denoted $T_\lambda(\gamma)$. When $\gamma = 0$ or $\gamma = \infty$, the distribution of $T_\lambda(\gamma)$ is an exponential distribution. Therefore for these values of γ, the

Require: λ, μ, γ
  Pick size_of_queue according to the steady state distribution.
  $m_0 \leftarrow 1 - \lambda/\mu$.
  size_of_queue $\leftarrow$ size_of_queue $+ 1$
  place_in_queue $\leftarrow$ size_of_queue.
  soj_time $\leftarrow 0$.
  while place_in_queue > 0 **do**
    soj_time $\leftarrow$ soj_time $+ \mathrm{Exp}(\lambda + \mu + \gamma m_0)$
    Pick an event $e \in \{\text{arrival}, \text{departure}, \text{steal}\}$ with probabilities proportional to $\{\lambda, \mu, \gamma m_0\}$ respectively.
    Modify place_in_queue and size_of_queue according to the event e.
  end while
  Return soj_time.

Algorithm 4.2: Sojourn time simulation

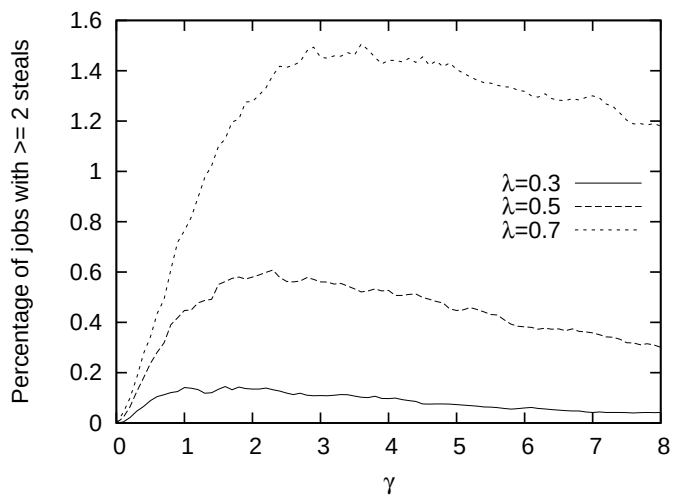


Figure 4.6.: Fraction of jobs that are stolen twice or more as a function of γ.

99th percentile is $\log(100)S_\lambda(\gamma)$. Figure 4.7 reports the empirical 99th percentile of the distribution of $T_\lambda(\gamma)$ as well as the 99th percentile of an exponential variable of the same mean $\log(100)S_\lambda(\gamma)$. For intermediate values of γ, the percentile is strictly less than for exponential distributions.

4.4.6. Fraction of stolen jobs

In all of our analysis, we choose to consider that if there were $j \geq 2$ jobs in a queue, the processor that is stealing would steal $\lfloor j/2 \rfloor$ jobs. This is the most natural strategy in the sense that it optimizes the balance between the processors. Works such as [114] study the case where every steal concerns only one job. This has a negative impact on performance, as shown in the following experiments. In this section, we further show the advantage of stealing half of the jobs instead of stealing a smaller fraction of the work.

When an empty processor steals from a processor with $j \geq 2$ jobs, then it steals $\max(1, \lfloor \delta j \rfloor)$, *i.e.*, a fraction $0 \leq \delta < 1$ of the total work or a single job if this fraction

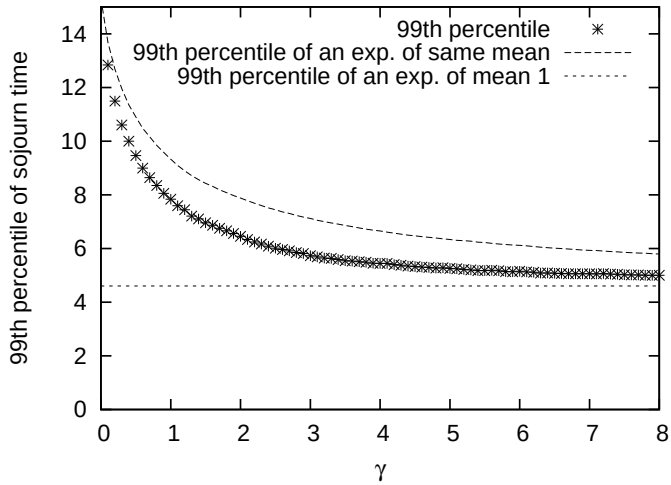


Figure 4.7.: 99 percentiles of the distribution of sojourn time and of an exponential variable of the same mean as functions of γ, for $\lambda = 0.7$.

is less than one. The case $\delta = 0$ corresponds exactly to the steal of one job while the condition $\delta < 1$ insures that we always leave at least one job in the processor (since $j - \lfloor \delta j \rfloor) \geq 1$).

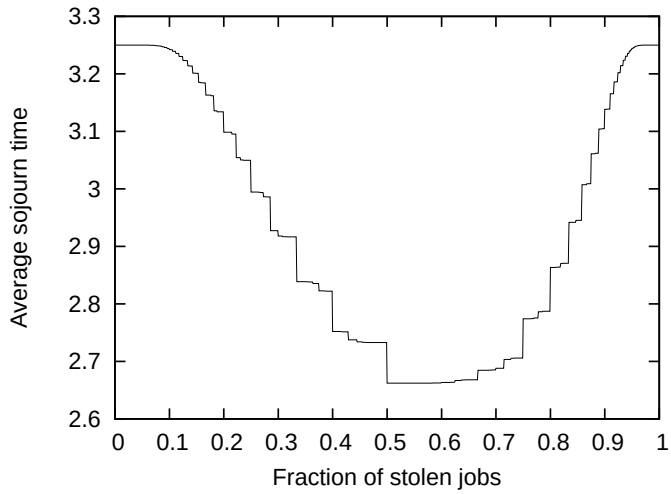


Figure 4.8.: Average sojourn time as a function of the fraction of jobs stolen at each time for $\lambda = 0.9$ and $\gamma = 3$.

Algorithm 4.1 can be modified to compute the steady-state in that case. Figure 4.8 shows the average sojourn time as a function of δ for $\lambda = 0.9$, $\mu = 1$ and $\gamma = 3$.

Figure 4.8 highlights two interesting properties. The first one is that the optimal fraction to steal is one half and the difference in speed is approximately 25%. The second interesting property is that this curve is not symmetric in 1/2 and it is not continuous in δ. Indeed, in such systems the processors generally have a small number of jobs and the function $\lfloor \delta j \rfloor$ is not continuous in j. In particular, the figure shows discontinuities at p/q for small values of q $(1/2, 1/3, 2/3 \ldots)$.

4.4.7. Batch arrivals

In this section, we consider that jobs arrive in batches of b jobs according to a Poisson point process of rate λ/b (the rate is divided by b to keep an arrival rate λ). We will see that work stealing is a very efficient way to diminish the effect of these batch arrivals.

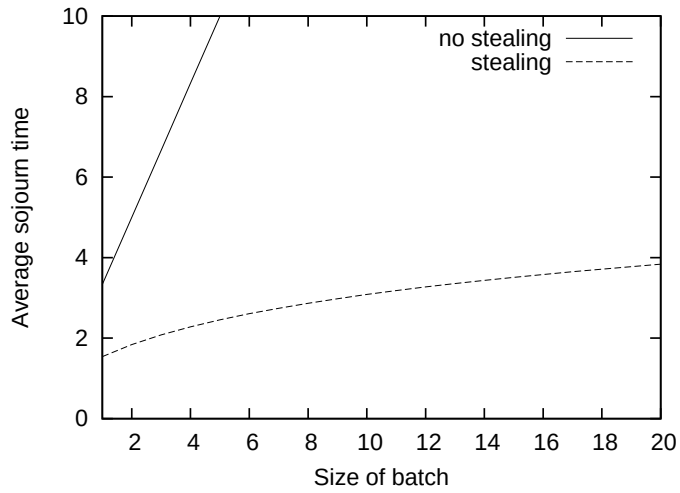


Figure 4.9.: Average sojourn time as a function of the batch size for $\lambda = 0.7$. The higher curve represents a system without work stealing while the bottom one shows the results for $\gamma = 3$.

In a system with no load balancing mechanism, when the size b of the batches increases, it can be shown that the average sojourn time grows linearly in b [117].

Algorithm 4.1 can be easily modified to take into account the batch arrivals. Using this, we can compute the average sojourn time for various values of the size of batches (from 1 to 30) for $\lambda = 0.7$ and $\gamma = 3$. In Figure 4.9 we compare the average sojourn time in the system with work stealing to the system without work stealing. Once again, work stealing has a tremendous impact on the performance. In fact when computing the average sojourn time for large values of b, it seems to grow as the logarithm of b. This is in agreement with known results on the performance of work stealing in transient cases (*i.e.*, over a finite number of tasks), like the ones of the previous chapter.

4.5. Heterogeneous Clusters

As mentioned in the introduction, we are interested in evaluating the performance of work stealing in a system made of several clusters. Each cluster is made of homogeneous processors with the same processing rate. Also the time to steal between two clusters depends on their "distance" and is much larger than the time to steal within one cluster. The main problem addressed here is to come up with a stealing strategy (namely, values for the stealing probabilities p_{ij}) that minimizes the response times for jobs.

If the system is heterogeneous, the natural work stealing algorithm which is to steal uniformly at random among all the resources may suffer from communication cost since it is much faster to steal inside a cluster. Several modifications of this algorithm have been proposed to take into account the geometry of the system. Many of the

proposed algorithms rely on a master-worker paradigm: some resources, called masters, are dedicated to balance the load between clusters while the others try to balance the work insider a cluster. Other strategies are based on tuning the preferences between internal and external steals. All these strategies are experimentally compared in [146, 124] and the latter proved more efficient. In our framework, this corresponds to tuning the probabilities $p_{cc'}$ for choosing a victim.

In the following, we focus on the *average sojourn time* as the performance indicator to compare different strategies. We used a numerical algorithm to compute the fixed point of the system of ODEs (4.1-4.7). The time to generate one curve is less than ten minutes, allowing us to explore many scenarios.

4.5.1. Two homogeneous clusters

A first case of interest is the case with two homogeneous clusters: $\mu_0 = \mu_1$, $\lambda_0 = \lambda_1$ and $\gamma_{00} = \gamma_{11}$. Each cluster can be viewed as a network of closely interconnected processors. The two clusters are connected by a slow network. Generally, communication between two hierarchies (intra-processor, intra-cluster, inter-cluster) is about 10 to 100 times slower (as in Grid 5000 [49]). For our simulations, $\gamma_{ij}{=}\gamma_{ii}/10$ for $i{\neq}j$.

Let p_{ij} be the probability for a processor in cluster i to choose to steal from cluster j. As the system is symmetric, we choose $p_{00} = p_{11}$ and $p_{01} = p_{10} = 1 - p_{00}$. We want to study the effect of this probability p_{00} on the performance.

The loads of the two clusters are the same and communications are much slower if the two processors are in two different clusters. Therefore the optimal p_{00} is 1: it is always better to steal inside one's own cluster. Figure 4.10 displays the average sojourn time of a job entering the system as a function of p_{00}, called the probability of *self-stealing*. The parameters of the system displayed in the figure are $\lambda = 0.7$, $\mu = 1$ and $\gamma_{ii} = 10, \gamma_{ij} = 3$ and the clusters have the same size; the results are very similar for other values.

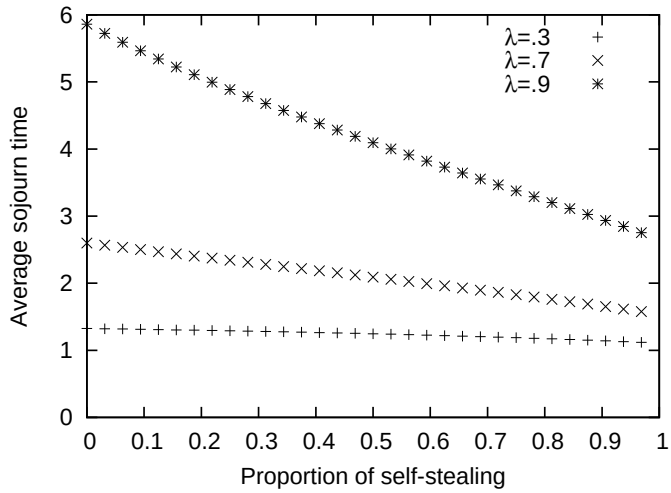


Figure 4.10.: Average sojourn time in a system with two homogeneous clusters as a function of the probability for a processor to steal inside its cluster.

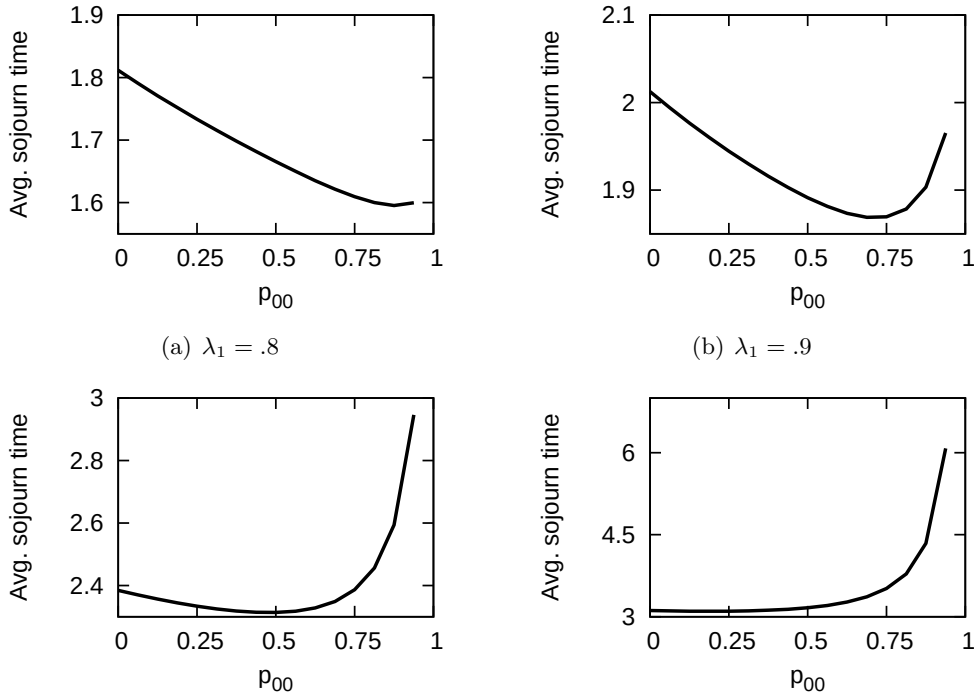
This figure exhibits the two main features of such systems. First, as expected in this case, it is much more efficient to pick $p_{00} = 1$ rather than a uniform stealing probability,

$p_{00} = 1/2$. Moreover, the dependence of the sojourn time on p_{00} is almost linear, showing that this probability has a real effect. Computing the same curve for other values of λ, shows that when the load is low (for example $\lambda = 0.3$), then the curve is slightly concave while when the load is very high ($\lambda > 0.9$), the curve is slightly convex but in all cases the dependence is almost linear.

4.5.2. Two heterogeneous clusters

A direct extension of the previous model is to consider the case where the two clusters are heterogeneous. We set $\lambda_0 < \lambda_1$, $\mu_0 = \mu_1$ and $\gamma_{ij} = \gamma_{ii}/10$. As the load of cluster 1 is higher and since it is faster to steal inside the cluster, we consider that the processors of cluster 1 only steal inside their cluster: p_{11}=1. We want to study the effect of the probability for the processors in cluster 0 to steal from cluster 1.

Figure 4.11 displays the average sojourn time as a function of p_{00} for different values of λ_1. In all cases, the load of cluster 0 is low: $\lambda_0 = .5$ and the load of the other cluster varies. In all cases, we can see that there exists an optimal p_{00} that is neither 0 nor 1 that minimizes the average sojourn time. In the top left figure, cluster 1 is just slightly more loaded than the cluster 0 ($\lambda_1 = .8$) and in that case the optimal p_{00} is close to .85. As the load of the cluster 1 grows, the optimal probability gets closer to 0.



(a) $\lambda_1 = .8$ (b) $\lambda_1 = .9$

(c) $\lambda_1 = 1$ (d) $\lambda_1 = 1.1$

Figure 4.11.: Average sojourn time as a function of p_{00} for the two heterogeneous model. The first cluster is lightly loaded ($\lambda_0 = 0.5$). The load of the second cluster is λ_1 (varying from 0.8 to 1.1).

This shows that the optimal probability strongly depends on the load of the different clusters which is an unknown variable in many cases. However, we also see that the

average sojourn time does not vary that much around the optimal p_{00} which shows that a rough estimation of the load is enough to make a good choice for p_{00}.

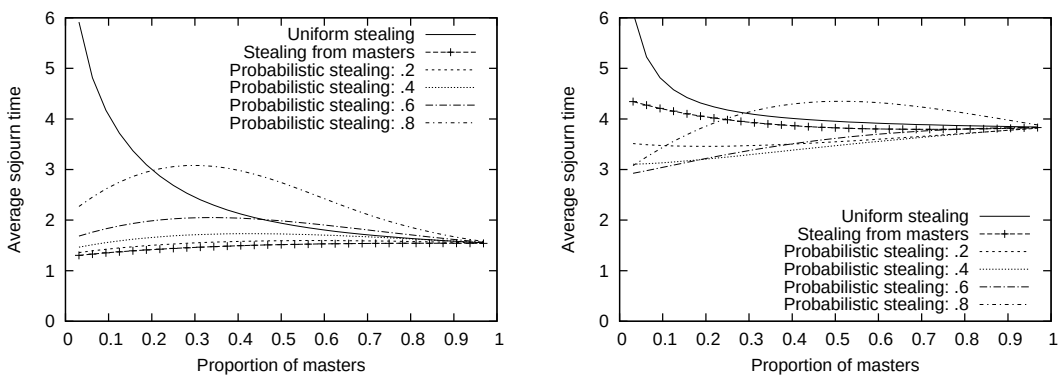
4.5.3. Hierarchical work stealing: master worker paradigm

Many work stealing algorithms rely on a master/worker paradigm. Here we show that this approach is indeed valid, but tuning its parameters is not easy.

We consider a network of homogeneous clusters, with a rate of steal 10 times greater inside a cluster than between two clusters. In each cluster, we set a fraction f_0 of the resources to be "masters" while the rest of the resources are "workers". We consider that the masters receive all the work which means that there is an arrival rate of λ/f_0 for every master (so that the total arrival rate of the system remains constant equal to λ). A master will only steal work from other masters while a worker can steal both from a master and from other workers.

Master-worker in one cluster

Let us first consider a network composed of only one cluster and let us study the effect of the fraction of masters on the performance of the system. We compare three different strategies: *probabilistic stealing* with various parameters, *uniform stealing* and *steal from masters*. This situation can be described in our model by setting the number of clusters to 2, cluster 0 representing the masters and cluster 1 representing the workers. We set $\gamma_{ij} = \gamma_{ii}$, $\lambda_0 = \lambda/f_0$, $\lambda_1 = 0$ and $p_{00} = 1$. For the probabilistic stealing strategy, we study different probabilities for a worker to steal a master $p_{10} = .2, .4, .6, .8$. The *uniform stealing* (resp. *steal only from masters*) corresponds to $p_{10} = m_0$ (resp. $p_{10} = 1$). We also compare the case where the jobs arrive one by one and the case of batch arrivals with a batch size of 20.



(a) Average sojourn time when the batch size is 1. (b) Average sojourn time when the batch size is 20.

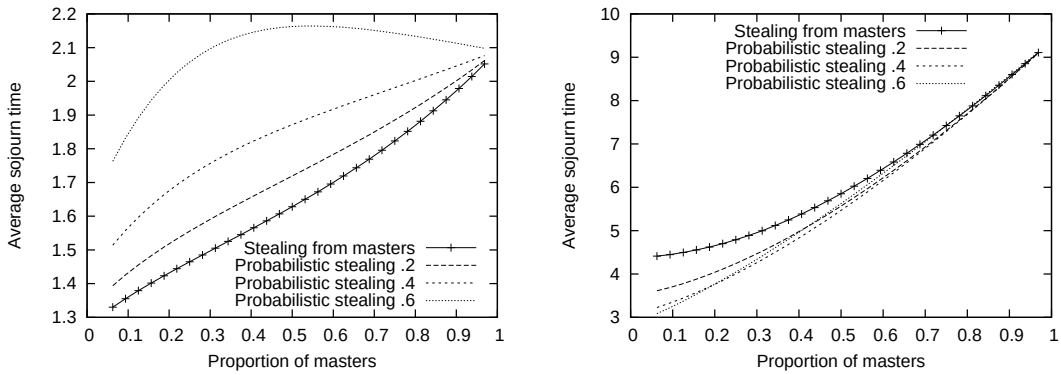
Figure 4.12.: Comparison of the average sojourn time in the Master-Worker setting with one cluster.

The average sojourn time for all strategies as a function of the fraction of masters is shown in Figure 4.12. As expected, in all cases, the uniform stealing is the worst strategy when the number of masters is low and when the proportion of masters grows, things get better. When the proportion of masters is close to 1, all strategies coincide. When the batches are of size 1, the optimal strategy is always to steal from the masters. When

the batches are of size 20, the optimal strategy is to steal about 50% from the masters (more precisely, the optimal is about 60% for low proportions of masters (< 0.3) and 40% above).

The most interesting property shown in this figure is that in all cases, having all the arrivals concentrated on a few resources improves the performance of the system if we tune the probability of stealing correctly.

Master-worker in two clusters



(a) Average sojourn time when the batch size is 1. (b) Average sojourn time when the batch size is 20.

Figure 4.13.: Average sojourn time in the Master-Worker setting with two clusters for $\lambda = 0.7$.

The behavior observed in the two cluster model is similar to the behavior with a single cluster. We consider a network of two clusters with the same proportion f_0 of masters in each cluster. All the arrivals are concentrated on the masters. The masters are only stealing from other masters and we study different strategies of stealing for the workers. As in the one cluster case, the uniform policy is not efficient. Therefore, we only focus on the policies *stealing from master* and *probabilistic stealing* with a probability of .2, .4 and .6.

The results are shown in Figure 4.13, comparing a case with arrivals by batches of size 20 and a case without batch arrivals. Once again we see that in the case where the batches are of size 1, the most efficient strategy is always to steal from the masters while if the size of the batches is larger (20 in this example), the most efficient strategy is to steal from workers with a non zero probability. Moreover in both cases, a good choice of the probabilities makes the performance of the system better when the proportion of masters is low.

When the proportion of masters is low, the good performance can be explained in part by the fact that we neglect the congestion that might occur when a single resource has to deal with too arrivals or steal requests. In practical applications, one would have to take this into account.

4.6. Conclusion and Future Work

In this chapter we presented a mean field approximation of the work stealing algorithm on a large number of processors and show convergence for finite time as well as for steady state. This allows one to run a rather exhaustive evaluation for several performance measures such as average response times, average number of steals per job. The distribution of response times can also be sampled using fast simulation, providing more meaningful performance indexes (variance, percentiles, tail behavior). This also allows one to evaluate work stealing under several scenarios: all processors are in one homogeneous cluster, or partitioned into several clusters where the time to steal from one cluster to another depends on the distance between the clusters. The scenario where work is allocated to a few master processors and stolen by workers is also evaluated and numerical evidences show that it performs really well with appropriate stealing parameters.

We are currently considering two extensions of this work. The first one is to model the case with a finite number of jobs using a mean approach, and then tune the parameters to minimize the total completion time. The second one is to conduct a thorough comparison with push and pull policies currently used in computational grids.

Part II.

Optimal Mean Field

Outline of this part

We presented the notion of *mean field* models in Chapter 2. Mean field models are generally used to study the dynamics of large stochastic systems. This is illustrated by Chapter 4 in which we used mean field theory to simplify the performance evaluation of a system using work stealing. The goal of this part is to fill the gap between Markov decision processes, presented in Chapter 1, and mean field models. We study the relationships between mean field models and optimal control.

In the first two Chapters 5 and 6, we investigate the limiting behavior of Markov decision processes made of independent objects evolving in a common environment, when the number of objects (N) goes to infinity. We show that in both cases, when the number of objects becomes large, the original stochastic optimization problem converges to a deterministic optimization problem in discrete or continuous time. The nature of the limiting regime depends on the intensity of the model $I(N)$ which is the probability for an object to change state at each time. If $I(N) = O(1)$, the limiting dynamics is in discrete time and this case is studied in Chapter 5. If $\lim_{N\to\infty} I(N) = 0$, the limiting regime is in continuous time. This is the focus of Chapter 6.

More precisely, we show that the optimal cost on a finite-horizon of the stochastic system converges to the optimal cost of a deterministic dynamical system as the number of objects grows (Theorems 5.5 and 6.4). For the discrete time limit, we further provide bounds on the speed of convergence by proving second (Theorem 5.7) and third order results (Theorem 5.9). In the continuous time, speed of convergence is insured by the explicit probability bounds of Theorem 6.1. These methods are illustrated by different examples in Sections 5.4 and 6.4, like brokering problem in grid computing, investment strategies or epidemic control.

The focus of Chapter 7 is slightly different. We study the limiting behavior of a mean field model when the dynamics of the system exhibits some discontinuity. We show that without any condition on the smoothness of the limiting regime, the stochastic system converges to a deterministic limit (Theorem 7.5). Contrary to the smooth case, the limiting regime is characterized by the set of solutions of a differential inclusions, which definition is recalled in Section 7.2.1). This extends the applicability of mean field techniques to systems exhibiting threshold dynamics. In particular, when considering a controlled system, the controlled policy often presents some discontinuity. These results are illustrated by several examples in 7.5. They also provide an easy proof for classical fluid limit regimes, as shown in Section 7.5.1.

Organization of this part

The two first chapters of this part focus on optimal control of mean field models – in Chapter 5 when the limit is in discrete time and in Chapter 6 when the limit is in continuous time. Chapter 7 shows the convergence of mean field systems whose dynamics is discontinuous, which is in particular the case of controlled systems.

Each chapter of this part is self-contained. To improve the readability of the different sections, the most technical proofs have been moved to the end of each chapter.

Chapter 5.

Optimization in Discrete Time

Abstract of this chapter – This chapter investigates the limit behavior of Markov decision processes made of independent objects evolving in a common environment, when the number of objects (N) goes to infinity.

In the finite horizon case, we show that when the number of objects becomes large, the optimal cost of the system converges to the optimal cost of a discrete time system that is deterministic. We further provide bounds on the speed of convergence by that the gap between the deterministic system and its limit is in $1/\sqrt{N}$. One can even go further and get convergence of order $\sqrt{\log N}/N$ to a stochastic system made of the mean field limit and a Gaussian term. Several extensions are also discussed. In particular, for infinite horizon cases with discounted costs, we show that first order limits hold and that second order results also hold as long as the discount factor is small enough.

Our framework is applied to a brokering problem in grid computing. Several simulations with growing numbers of processors are reported. They compare the performance of the optimal policy of the limit system used in the finite case with classical policies by measuring its asymptotic gain.

Résumé du chapitre – Ce chapitre examine le comportement limite de processus de décision Markovien constitués de particules indépendantes évoluant dans un environnement commun, lorsque le nombre de particules tend vers l'infini.

Dans le cas où on s'intéresse à un coût à horizon fini, nous montrons que lorsque le nombre de particules devient grand, le coût optimal du système converge vers le coût optimal du système déterministe. De plus, nous donnons un aperçu de la vitesse de convergence en prouvant que l'écart entre le système stochastique et sa limite déterministe est en $1/\sqrt{N}$. Cette convergence peut être raffinée en considérant l'écart avec une limite déterministe plus un bruit gaussien. Dans ce cas on atteint une convergence en $\sqrt{\log(N)}/N$.

Le modèle est appliqué à un problème de gestionnaire de ressources dans des grilles de calcul. Nous comparons les performances de la politique optimale de la limite appliquée au système initiale avec plusieurs politiques classiques. Nous mesurons le gain asymptotique, ainsi que le seuil à partir duquel elle surpasse les politiques classiques.

5.1. Introduction

The general context of this chapter is the optimization of the behavior of controlled Markovian systems, namely Markov Decision Processes composed by a large number of objects evolving in a common environment.

Consider a discrete time system made of N objects, N being large, that evolve randomly and independently (according to a transition probability kernel K). At each step, the state of each object changes according to a probability kernel, depending on the environment. The evolution of the environment only depends on the number of objects in each state. Furthermore, at each step, a central controller makes a decision that changes the transition probability kernel. The problem addressed in this chapter is to study the limit behavior of such systems when N becomes large and the speed of convergence to the limit.

The seminal work of Kurtz (see for example [102] and the related work in Chapter 2) initiated a long stream of work on the use of mean field techniques in performance evaluation. Several papers [20, 45] study the limit behavior of Markovian systems in the case of vanishing intensity (the expected number of transitions per time slot is $o(N)$). In these cases, the system converges to a system in continuous time, see Section 2.2.2 of Chapter 2. The control and the optimization of systems with an intensity that goes to zero are investigated in [70]. In the present chapter, the intensity is bounded away from zero so that time remains discrete at the limit. This requires a different approach to construct the limit.

In Section 2.2.3 of Chapter 2, such discrete time systems are presented. In [106], authors show that under certain conditions, as N grows large, a Markovian system made of N objects converges to a deterministic system. Since a Markov decision process can be seen as a family of Markovian kernels, the class of systems studied in [106] corresponds to the case where this family is reduced to a unique kernel and no decision can be made. Here, we show that under similar conditions as in [106], a Markov decision process also converges to a deterministic one. More precisely, we show that the optimal costs (as well as the corresponding states) converge almost surely to the optimal costs (resp. the corresponding states) of a deterministic system (the "optimal mean field"). These first order results are very similar to the results proved independently in [42]. Additionally, the quality of the deterministic approximation and the speed of convergence can also be estimated. For that, we provide second order results giving bounds on the speed of convergence under the form of central limit theorems for the state of the system as well as for the cost function.

Actually, the contributions of this chapter concern three types of systems. The first one is the class of Markov decision processes where the sequence of actions is fixed. In some sense, their study could boil down to considering classical Markovian systems. The second one is the class of MDPs where the policy is fixed. These *controlled* systems constitute the main object of this chapter. The last type of results concerns MDP under optimal control. This is the part where optimization plays a role. The second type (controlled systems) plays a central role indeed. The results on systems with fixed sequences of actions are simple corollaries of theorems on controlled systems by considering constant policies. The results on optimal control are obtained by taking the supremum over all policies in controlled system.

On a practical point of view, all this allows one to compute the optimal policy in a deterministic system which can often be done very efficiently, and then to use this policy

in the original random system as a good approximation of the optimal policy, which cannot be computed efficiently because of the curse of dimensionality. This is illustrated by an application of our framework to optimal brokering in computational grids. We consider a set of multi-processor clusters – forming a computational grid, like EGEE[1] – and a set of users submitting tasks to be executed. A central broker assigns the tasks to the clusters (where tasks are buffered and served in a FIFO order) and tries to minimize the average processing time of all tasks. Computing the optimal policy (solving the associated MDP) is known to be hard [121]. Numerical computations can only be carried out up to a total of 10 processors and two users. However, our approach shows that when the number of processors per cluster and the number of users submitting tasks grow, the system converges to a mean field deterministic system. For this deterministic mean field system, the optimal brokering policy can be explicitly computed. Simulations reported in Section 5.4 show that, using this policy over a grid with a growing number of processors, makes performance converge to the optimal sojourn time in a deterministic system, as expected. Also, simulations show that this deterministic static policy outperforms classical dynamic policies such as Join the Shortest Queue, as soon as the total number of processors and users is over 50.

5.2. Notations and definitions

The model studied in this chapter is a generalization of the discrete time mean field model of [106], also described in Section 2.2.3, to which has been a controller that can change the dynamics of the system. More details on the model can be found in Section 2.2.3.

The system is composed of N objects. Each object evolves in a finite state space $\mathcal{S} = \{1, \ldots, S\}$. Time is discrete and the state of the nth object at time $t \in \mathbb{N}$ is denoted $X_n^N(t)$. We denote $M^N(t)$ the empirical measure of the collection of objects (X_n^N). $M^N(t)$ is a vector with S components and the ith component of $M^N(t)$ is the proportion of objects in state i,

$$M_i^N(t) \stackrel{\text{def}}{=} N^{-1} \sum_{n=1}^{N} \mathbf{1}_{X_n^N(t)=i}.$$

The system of objects evolves depending on their common environment. We call $C(t) \in \mathbb{R}^d$ the context of the environment. Its evolution depends on the empirical measure $M^N(t)$, itself at the previous time slot and the action $a \in \mathcal{A}$ chosen by the controller (see below):

$$\mathrm{C}^N(t+1) = g(C^N(t), M^N(t+1), a_t),$$

where $g : \mathbb{R}^d \times \mathscr{P}_N(\mathcal{S}) \times \mathcal{A} \to \mathbb{R}^d$ is a continuous function.

For an action $a \in \mathcal{A}$ and a context $C \in \mathbb{R}^d$, we have a transition probability kernel $K(a, \mathrm{C})$ such that the probability that for any n, a object goes from state i to state the j is $K_{i,j}(a, \mathrm{C})$:

$$\mathcal{P}\left(X_n^N(t+1) = j | X_n^N(t) = i, a_t = a, \mathrm{C}^N(t) = \mathrm{C}\right) = K_{i,j}(a, \mathrm{C}).$$

The evolutions of objects are supposed to be independent once C and a are given. Moreover, we assume that $K_{i,j}(a, \mathrm{C})$ is continuous in a and C.

[1] EGEE: Enabling Grids for E-sciencE, http://www.eu-egee.org

5.2.1. Actions, policies and reward functions

We consider a Markov decision process corresponding to this system. At each time t, a decision maker chooses an action a from the set of possible actions $\mathcal{A}$. $\mathcal{A}$ is assumed to be a compact set (finite or infinite). In practical examples, $\mathcal{A}$ is often finite or a compact subset of $\mathbb{R}^k$. The action determines how the system will evolve by modifying the transition function $g(a,.)$ and $K(a,.)$.

To each possible state $(M(t), C(t))$ of the system at time t, we associate a reward $\mathrm{r}_t(M, C)$. The reward is assumed to be continuous in M and C. This function can be either seen as a reward – in that case the controller wants to maximize the reward–, or as a cost – in that case the goal of the controller is to minimize this cost. In this chapter, we will mainly focus on finite-horizon reward. Extensions to infinite-horizon reward (discounted and average reward) are discussed in Section 5.5.

The focus of Markov Decision Processes theory is the computation of optimal policies. A policy $\pi = (\pi_1\pi_2\dots)$ specifies the decision rules to be used at each time slot. A decision rule π_t is a procedure that provides an action at time t. As seen in Section 1.3.1 of Chapter 1, if the state space is finite and the action space is compact, then deterministic Markovian policies (*i.e.* that only depends deterministically on the current state) are dominant, *i.e.* are as good as general policies. In what follows, we will only focus on them and a policy π will represent a sequence of functions $(\pi_t)_{t\geq 0}$ where each function $\pi_t : \mathscr{P}(\mathcal{S})\times\mathbb{R}^d \to \mathcal{A}$ is deterministic. For any policy π, the variables $M_\pi^N(t), C_\pi^N(t)$ will denote the state of the system at time t when the controller applies the policy π. $(M_\pi^N(t), C_\pi^N(t))_{t\geq 0}$ is a sequence of random variables on $\mathscr{P}_N(\mathcal{S}) \times \mathbb{R}^d$.

5.2.2. List of assumptions

Here is the list of the assumptions under which all our results will hold, together with some comments on their tightness and their degree of generality and applicability.

Throughout the document, for all $(m, c) \in \mathscr{P}(\mathcal{S})\times\mathbb{R}^d$, $\|(m, c)\|$ denotes the L^∞ norm of the vector $(m, c) \in \mathbb{R}^{S+d}$: $\|(m, c)\| = \max(|m_1| \dots |m_s|, |c_1| \dots |c_d|)$.

(A1) **Independence of the users, Markov system** – If at time t if the environment is C and the action is a, then the behavior of each object is independent of other objects and its evolution is Markovian with a kernel $K(a, \mathrm{C})$.

(A2) **Compact action set** – The set of action $\mathcal{A}$ is a compact metric space.

(A3) **Continuity of** K, g, r – the mappings $(\mathrm{C}, a) \mapsto K(a, \mathrm{C})$, $(\mathrm{C}, \mathrm{M}, a) \mapsto g(\mathrm{C}, \mathrm{M}, a)$ and $(\mathrm{M}, \mathrm{C}) \mapsto \mathrm{r}_t(\mathrm{M}, \mathrm{C})$ are continuous, Lipschitz continuous on all compact set.

(A4) **Almost sure initial state** – Almost surely, the initial measure $\mathrm{M}^N(0), \mathrm{C}^N(0)$ converges to a deterministic value $m(0), c(0)$. Moreover, there exists $B < \infty$ such that almost surely $\left\|\mathrm{C}^N(0)\right\| \leq B$ where $\|C\| = \sup_i |C_i|$.

To simplify the notations, we choose the functions K and g not to depend on time. However as the proofs will be done for each time step, they also hold if the functions are time-dependent (in the finite horizon case).

Also, K, g and r do not to depend on N, while this is the case in most practical cases. Adding a uniform continuity assumption on these functions for all N will make all the proofs work the same.

Here are some comments on the uniform bound B on the initial condition (A4). In fact, as $\mathrm{C}^N(0)$ converges almost surely, $\mathrm{C}^N(0)$ is almost surely bounded. Here we had a bound B which is uniform on all events in order to be sure that the variable $\mathrm{C}^N(0)$ is dominated by an integrable function. As g is continuous and the sets $\mathcal{A}$ and $\mathscr{P}(\mathcal{S})$ are compact, this shows that for all t, there exists $B_t < \infty$ such that

$$\left\|\mathrm{C}^N(t)\right\| \leq B_t. \tag{5.1}$$

Finally, in the Markov decision process literature, the reward function often depends on the action taken. To simplify the notations, we choose to take the reward independent of the action but again the proofs are the same in that case.

5.3. Finite time convergence and optimal policy

In this section, we focus on optimizing the finite horizon reward. T is fixed throughout all this section and the aim of the controller is to find a policy to maximize:

$$V_\pi^N(M^N, C^N) = \mathbb{E}\left(\sum_{t=1}^{T} \mathrm{r}\left(M_\pi^N(t), C_\pi^N(t)\right)\right).$$

The infinite horizon case will be treated in Section 5.5.2.

This section contains the main results of this chapter. There are four main results. Theorem 5.1 states the convergence of the controlled system to a deterministic limit. Next, we show that the optimal reward for the limit is asymptotically optimal as the size of the system grows (Theorem 5.5) and we characterize the speed of convergence towards this limit (Theorem 5.7) which is basically of order $1/\sqrt{N}$. Finally (Theorem 5.9) shows that a Gaussian approximation of the deterministic limit system leads to a better error of order $N^{-1}\sqrt{\log(N)}$.

Because of Equation 5.1, $(M^N(t), C^N(t))$ always stays in a compact space when $t \in \{0 \dots T\}$. By assumption (A3), this implies that K, g and r are Lipschitz continuous and we denote by L_K, L_g and L_r their Lipschitz constants.

5.3.1. Controlled mean field

Let $a = a_0, a_1 \dots$ be a sequence of actions. We define the deterministic variables $m_a(t)$ and $c_a(t)$ starting in $m_a(0), c_a(0) \stackrel{\text{def}}{=} m(0), c(0) \in \mathscr{P}(\mathcal{S}) \times \mathbb{R}^d$ by induction on t:

$$\begin{aligned} m_a(t+1) &= m_a(t)K(a_t, c_a(t)) \\ c_a(t+1) &= g\left(c_a(t), m_a(t+1), a_t\right). \end{aligned} \tag{5.2}$$

Here, $(m_a(t), c_a(t))$ corresponds to a deterministic approximation of the stochastic system (M^N, C^N) assuming that instead of having a probability K_{ij} for an object to go from state i to state j, there is a proportion K_{ij} of objects in state i that moves to state j.

Let π be a policy and consider a realization of the sequence $(M^N(t), C^N(t))$. At time t, a controller that applies the policy π, will apply the action $A_\pi^N(t) \stackrel{\text{def}}{=} \pi_t(M_\pi^N(t), C_\pi^N(t))$. The actions $A_\pi^N(t)$ form a random sequence depending on the sequence $(M_\pi^N(t), C_\pi^N(t))$. To this random sequence, corresponds a deterministic approximation of M^N, C^N, namely

$m_{A_\pi^N}(t)$ defined by Equation (5.2). The quantity $m_{A_\pi^N}(t)$ is a random variable depending on the sequence A_π^N (and is deterministic once A_π^N is fixed).

The following theorem is the main result of convergence, showing that as N grows, the gap between the stochastic system M_π^N, C_π^N and its deterministic limit $m_{A_\pi^N}, c_{A_\pi^N}$ vanishes (in probability) with a bound only depending on the initial condition.

Theorem 5.1 (Controlled mean field)**.** *Under assumptions (A1,A3), and if the controller applies the policy π, then there exists a sequence of functions $\mathcal{E}_t(\epsilon, x)$ such that $\lim_{\epsilon\to 0, x\to 0} \mathcal{E}_t(\epsilon, x) = 0$ and for all t:*

$$\mathcal{P}\left(\sup_{s\leq t}\left\|(M_\pi^N(s), C_\pi^N(s)) - (m_{A_\pi^N}(s), c_{A_\pi^N}(s))\right\| \geq \mathcal{E}_t(\epsilon, \epsilon_0^N)\right) \leq 2tS^2\exp(-2N\epsilon^2),$$

where

$$\begin{aligned}
\epsilon_0^N &\stackrel{\text{def}}{=} \left\|(M^N(0), C^N(0)) - (m(0), c(0))\right\|; \\
\mathcal{E}_0(\epsilon, \delta) &\stackrel{\text{def}}{=} \delta; \\
\mathcal{E}_{t+1}(\epsilon, \delta) &\stackrel{\text{def}}{=} \left(S\epsilon + (2 + L_K)\,\mathcal{E}_t(\epsilon, \delta) + L_K\mathcal{E}_t(\epsilon, \delta)^2\right)\max(1, L_g).
\end{aligned}$$

Proof. The proof is done by induction on t. We show that at each time step, we stay close to the deterministic approximation with high probability. A detailed proof is given in Appendix 5.8.1. □

Assuming that the initial condition converges almost surely to $m(0), c(0)$, we can refined the convergence in law into an almost sure convergence:

Corollary 5.2. *Under assumptions (A1,A3,A4),*

$$\left\|(M_\pi^N(t), C_\pi^N(t)) - (m_{A_\pi^N}(t), c_{A_\pi^N}(t))\right\| \xrightarrow{\text{a.s.}} 0.$$

Proof. This proof is a direct corollary of Theorem 5.1 and the Borel-Cantelli Lemma. □

5.3.2. Optimal mean field

Using the same notation and hypothesis as in the previous section, we define the reward of the deterministic system starting at $m(0), c(0)$ under the sequence of action a:

$$v_a(m(0), c(0)) \stackrel{\text{def}}{=} \sum_{t=1}^{T} \mathrm{r}_t(m_a(t), c_a(t)).$$

If for any t, the action taken at instant t is fixed equal to a_t, we say that the controller applies the policy a. a can be viewed as a policy independent of the state M^N, C^N and $M_a^N(t), C^N(t)$ denotes the state of the system when applying the policy a. According to Corollary 5.2, the stochastic system $M_a^N(t), C_a^N(t)$ converges almost surely to $m_a(t), c_a(t)$. Since the reward at time t is continuous, this means that the finite-horizon expected reward converges as N grows large:

Lemma 5.3. (Convergence of the reward) *Under assumptions (A1,A3,A4), if the controller takes actions $a = (a_0, a_1 \dots)$, the finite-horizon expected reward of the stochastic system converges to the finite-horizon reward of the deterministic system when initial conditions converge. If $(M^N(0), C^N(0)) \to (m(0), c(0))$ a.s. then*

$$\lim_{N\to\infty} V_a^N\left(M^N(0), C^N(0)\right) = v_a(m(0), c(0)) \quad \text{a.s.}$$

Proof. For all t, $(M_a^N(t), C_a^N(t))$ converges in probability to $(m_a(t), c_a(t))$. Since the reward at time t is continuous in (M, C), then $\mathrm{r}_t(M_a^N(t), C_a^N(t))$ converges in probability to $\mathrm{r}_t(m_a(t), c_a(t))$. Moreover, as (M, C) are bounded (see Equation (5.1)), the $\mathbb{E}\left(\mathrm{r}_t(M_a^N(t), C_a^N(t))\right)$ goes to $\mathrm{r}_t(m_a(t), c_a(t))$ which concludes the proof. □

The previous lemma can also be deduced from the following proposition.

Proposition 5.4 (Uniform convergence of reward)**.** *Under assumptions (A1,A2, A3,A4), there exists a function $\mathcal{E}(N, \epsilon)$ such that:*

- $\lim_{N\to\infty,\epsilon\to 0} \mathcal{E}(N, \epsilon) = 0$,
- *for all policy π:*

$$\left|V_\pi^N\left(M^N(0), C^N(0)\right) - \mathbb{E}\left(v_{A_\pi^N}\left(m_{A_\pi^N}(0), c_{A_\pi^N}(0)\right)\right)\right| \leq \mathcal{E}(N, \epsilon_0^N),$$

where $\epsilon_0^N \stackrel{\text{def}}{=} \left\|(M^N(0), C^N(0)) - (m(0), c(0))\right\|$ and the expectation is taken on all possible values of A_π^N.

Proof.

$$\begin{aligned}
&\left|V_\pi^N\left(M^N(0), C^N(0)\right) - \mathbb{E}\left(v_{A_\pi^N}\left(m_{A_\pi^N}(0), c_{A_\pi^N}(0)\right)\right)\right| \\
&= \left|\mathbb{E}\left(\sum_{t=1}^T \mathrm{r}_t(M_\pi^N(t), C_\pi^N(t)) - \mathrm{r}_t(m_{A_\pi^N}(0), c_{A_\pi^N}(0))\right)\right| \\
&\leq L_\mathrm{r}\mathbb{E}\left(\max_{t\leq T}\left\|(M_\pi^N(t), C_\pi^N(t)) - (m_{A_\pi^N}(0), c_{A_\pi^N}(0))\right\|\right)
\end{aligned} \tag{5.3}$$

where L_r is a Lipschitz constant of the function r.

According to Theorem 5.1, $\left\|(M_\pi^N(t), C_\pi^N(t)) - (m_{A_\pi^N}(t), c_{A_\pi^N}(t))\right\| \geq \mathcal{E}_t(\epsilon, \epsilon_0^N)$ with probability at most $T^2S^2\exp(-2\epsilon N)$, where $\epsilon_0^N \stackrel{\text{def}}{=} \left\|(M^N(0), C^N(0)) - (m(0), c(0))\right\|$. Computing the expectation on the events such that this is verified and the others, we get:

$$(5.3) \leq L_\mathrm{r}\max_{t\leq T}\mathcal{E}_t(\epsilon, \epsilon_0^N)\left(1 - T^2S^2\exp(-2\epsilon N)\right) + DT^2S^2\exp(-2\epsilon N),$$

where $D \stackrel{\text{def}}{=} \sup_{x\in B}\|x\|$ with B a bounded set such that $\mathcal{P}\left((M_\pi^N(t), C_\pi^N(t)) \in B\right) = 1$ (see the remark above Equation (5.1)).

Let us define $\mathcal{E}(N, \epsilon_0^N) \stackrel{\text{def}}{=} \inf_{\epsilon>0} L_\mathrm{r}\max_{t\leq T}\mathcal{E}_t(\epsilon, \epsilon_0^N)\left(1 - T^2S^2\exp(-2\epsilon N)\right) + DT^2S^2\exp(-2\epsilon N)$. The function $\mathcal{E}(.,.)$ satisfies the two requirements of the theorem. □

Now, let us consider the problem of convergence of the reward under the optimal strategy of the controller. First, it should be clear that the optimal strategy exists for the limit system. Indeed, the limit system being deterministic, starting at state $(m(0), c(0))$, one only needs to know the actions to take for all $(m(t), c(t))$ to compute the reward. The optimal policy is deterministic and $v_*(m(0), c(0)) \stackrel{\text{def}}{=} \sup_{a\in\mathcal{A}^T}\{v_a(m(0), c(0))\}$. Since the action set is compact, this supremum is a maximum: there exists a sequence of actions $a^* = a_0^*a_1^*\ldots$ – depending on $m(0), c(0)$ – such that $v_*(m(0), c(0)) = v_{a^*}(m(0), c(0))$. Such a sequence is not unique and in many cases there are multiple optimal action sequences. In the following, a^* design one of them and will be called the sequence of *optimal limit actions*.

Theorem 5.5. (Convergence of the optimal reward) *Under assumptions (A1, A2, A3, A4), if* $\left\|(M^N(0), C^N(0)) - (m(0), c(0))\right\|$ *goes to* 0 *when* N *goes to infinity, the optimal reward of the stochastic system converges to the optimal reward of the deterministic limit system:*

$$\lim_{N\to\infty} V_*^N\left(M^N(0), C^N(0)\right) = \lim_{N\to\infty} V_{a^*}^N\left(M^N(0), C^N(0)\right) = v_*(m(0), c(0)), \quad a.s.$$

In words, this theorem states two important results. Firstly, as N goes to infinity, the reward of the stochastic system goes to the reward of its deterministic limit. Secondly, the reward of the optimal policy under full information $V_*^N\left(M^N(0), C^N(0)\right)$ is asymptotically the same as the reward obtained when applying to the stochastic system a sequence of optimal actions of the deterministic limit, both being equal to the optimal reward of the limit deterministic system, $v_T^*(m(0), c(0))$.

Proof. Let a^* be a sequence of optimal actions for the deterministic limit starting at $m(0), c(0)$. Lemma 5.3 shows that $\lim_{N\to\infty} V_{a^*}^N\left(M^N(0), C^N(0)\right) = v_{a^*}(m(0), c(0)) = v_*(m(0), c(0))$. This shows that

$$\liminf_{N\to\infty} V_*^N\left(M^N(0), C^N(0)\right) \geq \liminf_{N\to\infty} V_{a^*}^N\left(M^N(0), C^N(0)\right) = v_*(m(0), c(0))$$

Conversely, let π_*^N be an optimal policy for the stochastic system and $A_{\pi_*^N}^N$ the (random) sequence of action $A_{\pi_*^N}^N(t) \stackrel{\text{def}}{=} \pi_*^N(M^N(t), C^N(t))$. This policy is suboptimal for the deterministic limit: $v_*(m(0), c(0)) \geq v_{A_{\pi_*^N}^N}(m(0), c(0))$. Using Proposition 5.4,

$$\begin{aligned} V_*^N\left(M^N(0), C^N(0)\right) = V_{\pi_*^N}^N\left(M^N(0), C^N(0)\right) &\leq v_{A_{\pi_*^N}^N}(m(0), c(0)) + \mathcal{E}(N, \epsilon_0^N) \\ &\leq v_*(m(0), c(0)) + \mathcal{E}(N, \epsilon_0^N) \end{aligned}$$

where $\mathcal{E}(.,.)$ is defined as in Proposition 5.4 and $\epsilon_0^N \stackrel{\text{def}}{=} \left\|(M^N(0), C^N(0)) - (m(0), c(0))\right\|$. Since, $\lim_{N\to\infty} \mathcal{E}(N, \epsilon_0^N) = 0$, $\limsup_{N\to\infty} V_*^N\left(M^N(0), C^N(0)\right) \leq v_*(m(0), c(0))$. □

This result has several practical consequences. Recall that the sequence of actions $a_0^* \dots a_{T-1}^*$ is a sequence of optimal actions in the limit case, *i.e.* such that $v_{a^*}(m, c) = v_*(m, c)$. This result proves that as N grows, the reward of the constant policy $a_0^*, \dots, a_{t-1}^*$ converges to the optimal reward. This implies that the difference between the reward of the best complete information policies and the best incomplete information policies vanishes. However, the state $(M^N(t), C^N(t))$ is not deterministic and on one trajectory of the system, it could be quite far from its deterministic limit $(m(t), c(t))$. Let us also define the policy $\mu_t^*(m(t), c(t))$ which is optimal for the deterministic system starting at time t in state $m(t), c(t)$. The least we can say is that this strategy is also asymptotically optimal, that is for any initial state $M^N(0), C^N(0)$:

$$\lim_{N\to\infty} V_{\mu^*}^N\Big(M^N(0), C^N(0)\Big) = \lim_{N\to\infty} V_{a^*}^N\Big(M^N(0), C^N(0)\Big) = \lim_{N\to\infty} v_*(m(0), c(0)). \tag{5.4}$$

In practical situations, using this policy in the original system will decrease the risk of being far from the optimal state. On the other hand, using this policy has some drawbacks. The first one is that the complexity of computing the optimal policy for all states can be much larger than the complexity of computing a^*. Moreover, the system

becomes very sensitive to random perturbations and therefore harder to analyze: the policy μ^* is not necessarily continuous and M_μ^N, C_μ^N may not have a limit. In Section 5.4, a comparison between the performances of a^* and μ^* is provided over an example and we see that the performance of μ^* is much better, especially for small values of N.

5.3.3. Second order results

In this part we give bounds on the gap between the stochastic system and its deterministic limit. This result provides estimates on the speed of convergence to the mean field limit. These theorems have a flavor of central limit theorems in the sense that the convergence speed towards the limit is of order $1/\sqrt{N}$. This section contains two main results:

The first one is that when the control action sequence is fixed, the gap to the mean field limit decreases as the inverse square root of the number of objects. The second result states that the gap between the optimal reward for the finite system and the optimal reward for the limit system also decreases as fast as $1/\sqrt{N}$.

Proposition 5.6. *Under assumptions (A1,A2,A3,A4), there exist constants β_t, β'_t and a sequence of constants e_t^N only dependent on the parameters of the system such that:*

$$\sqrt{N}\mathbb{E}\left(\left\|(M_\pi^N(t), C_\pi^N(t)) - (m_\pi(t), c_\pi(t))\right\| : \epsilon_0^N\right) \leq \beta_t + \beta'_t\sqrt{N}\epsilon_0^N + e_t^N \tag{5.5}$$

where :

- $\mathbb{E}\left(\cdot : \epsilon_0^N\right)$ *designates the expectation knowing* ϵ_0^N.
- $\epsilon_0^N \stackrel{\text{def}}{=} \left\|\left(M^N(0), C^N(0)\right) - (m(0), c(0))\right\|$;
- β_t, β'_t *are defined by* $\beta_0 = 0$, $\beta'_0 = 1$ *and for all* $t \geq 0$:

$$\begin{aligned}\beta_{t+1} &= \max\{1, L_g\}\left((S + L_K + 1)\,\beta_t + \frac{S}{2}\right);\\ \beta'_{t+1} &= \max\{1, L_g\}\,(S + L_K + 1)\,\beta'_t;\end{aligned}$$

- *There exists a constant* $C > 0$ *such that* $e_0^N = 0$ *and*

$$e_{t+1}^N = \max\{1, L_g\}\left((S + L_K + 1)\,e_t^N + C\sqrt{\frac{\log(N)}{N}}\right).$$

In particular, for all t: $\lim_{N\to\infty} e_t^N = 0$.

Proof. See appendix 5.8.2. □

An almost direct consequence of the previous result is the next theorem.

Theorem 5.7. *Under assumptions (A1,A2,A3,A4), there exist constants γ and γ' such that if* $\epsilon_0^N \stackrel{\text{def}}{=} \left\|M^N(0), C^N(0) - m(0), c(0)\right\|$

- *For any policy* π:

$$\sqrt{N}\left|V_\pi^N\left(M^N(0), C^N(0)\right) - \mathbb{E}\left(v_{A_\pi^n}^N\left(m(0), c(0)\right)\right)\right| \leq \gamma + \gamma'\epsilon_0^N.$$

- $$\sqrt{N}\left|V_*^N\left(M^N(0),C^N(0)\right)-v_*^N\left(m(0),c(0)\right)\right|\leq\gamma+\gamma'\epsilon_0^N.$$

This theorem is the main result of this section. The previous result (Theorem 5.5) says that $\limsup_{N\to\infty}V_T^{*N}(\mathrm{M}^N(0),\mathrm{C}^N(0))=\limsup_{N\to\infty}V^N_{a_0^*\dots a_{T-1}^*}(\mathrm{M}^N(0),\mathrm{C}^N(0))=v_*(m(0),c(0))$. This new theorem says that both the gap between the cost under the any policy for the original and the limit system and the gap between the optimal costs for both systems are two random variables that decrease to 0 with speed $\sqrt{N}$. When the parameters of the system are not Lipschitz but differentiable, this results can be improved by showing that the term in $1/\sqrt{N}$ has a Gaussian law (see Theorem 5.8).

Proof. For any policy π, the expected reward of the stochastic system and the expected reward of the deterministic limit under actions A_π^N are:

$$V_\pi^N\left(M^N(0),C^N(0)\right)=\sum_{t=1}^T\mathbb{E}\left(\mathrm{r}\left(M_\pi^N(t),C_\pi^N(t)\right)\right)$$
$$\mathbb{E}\left(v^N_{A^n_\pi}\left(m(0),c(0)\right)\right)=\sum_{t=1}^T\mathbb{E}\left(\mathrm{r}\left(m_{A^N_\pi}(t),c_{A^N_\pi}(t)\right)\right).$$

The first part of the theorem comes directly corollary of Proposition 5.6 and the fact that if X and Y are two stochastic variables,and f is a real function Lipschitz of constant L then $\mathbb{E}\left(|f(X)-f(Y)|\leq L\left\|X-y\right\|\right)$.

The second part is proved as Theorem 5.5 by bounding both part of the inequality (for readability, the following equation is written suppressing the dependence in $M^N(0),C^N(0)$ and $m(0),c(0)$).

$$V_*^N=\sup V_\pi^N=V^N_{\pi_*^N}\leq\mathbb{E}\left(v_{A^N_{\pi_*^N}}\right)+(\gamma+\gamma'\epsilon_0^N)/\sqrt{N}\leq v_*+(\gamma+\gamma'\epsilon_0^N)/\sqrt{N},$$

where $\epsilon_0^N\stackrel{\text{def}}{=}\left\|M^N(0),C^N(0)-m(0),c(0)\right\|$ and π_*^N is the optimal policy of the stochastic system of size N. The first inequality comes from the first part of this theorem, the second from the fact that v_* is the optimal reward of the deterministic system.

The other inequality is similar:

$$v_*=v_{a^*}\leq V^N_{a^*}+(\gamma+\gamma'\epsilon_0^N)/\sqrt{N}\leq V_*^N+(\gamma+\gamma'\epsilon_0^N)/\sqrt{N},$$

where a^* is the sequence of optimal actions of the deterministic system. □

In the case where the parameter are differentiable and not just only Lipschitz, the Proposition 5.6 can be refined into Theorem 5.8 which is a central limit theorem for the states.

(A4-bis) **Initial Gaussian variable** – There exists a Gaussian vector G_0 of mean 0 with covariance Γ_0 such that the vector $\sqrt{N}((M^N(0),C^N(0))-(m(0),c(0)))$ (with $S+d$ components) converges almost surely to G_0.

(A5) **Continuous differentiability** – For all t and all $i,j\in\mathcal{S}$, all functions g, K_{ij} and r_t are continuously differentiable.

This differentiability condition is slightly stronger than the Lipschitz condition and is indeed false in many cases because of boundary conditions. The initial state condition is slightly stronger that (A4) but remains very natural. For example, if $C^N(0)$ if fixed to some $c(0)$ and if the initial states $X_1^N \dots X_N^N$ of all objects are independent and identically distributed (*i.i.d.*), then $\sqrt{N}((M^N(0), C^N(0)) - (m(0), c(0)))$ converges in law to a Gaussian variable G of the same covariance as X_1^N – this is just the multidimensional central limit theorem, see for example Theorem 9.6 of Chapter 2 of [59]. The fact that we assumed that the convergence holds almost surely rather that in law is just a technical matter: we can replace the variables $M^N(0), C^N(0)$ by random variables with the same law that converge almost surely.

Theorem 5.8. (Mean field central limit theorem) *Under assumption (A1,A2, A3,A4bis,A5), if the actions taken by the controller are $a_0 \dots a_{T-1}$, there exist Gaussian vectors of mean 0, $G_1 \dots G_{T-1}$ such that for every t:*

$$\sqrt{N}((\mathrm{M}^N(0), \mathrm{C}^N(0)) - (m(0), c(0)), \dots, (\mathrm{M}^N(t), \mathrm{C}^N(t)) - (m(t), c(t))) \xrightarrow{\mathcal{L}} G_0, \dots, G_t. \tag{5.6}$$

Moreover if Γ_t is the covariance matrix of G_t, then:

$$\Gamma_{t+1} = \left[\begin{array}{c|c} P_t & F_t \\ \hline Q_t & H_t \end{array}\right]^T \Gamma_t \left[\begin{array}{c|c} P_t & F_t \\ \hline Q_t & H_t \end{array}\right] + \left[\begin{array}{c|c} D_t & 0 \\ \hline 0 & 0 \end{array}\right], \tag{5.7}$$

where for all $1 \le i, j \le S$ and $1 \le k, \ell \le d$: $(P_t)_{ij} = K_{ij}(a_t, c(t))$, $(Q_t)_{kj} = \sum_{i=1}^S m_i \frac{\partial K_{ij}}{\partial c_k}(a_t, c(t))$, $(F_t)_{ik} = \frac{\partial g_k}{\partial m_i}(m_{t+1}, c(t))$, $(H_t)_{k\ell} = \frac{\partial g_k}{\partial c_\ell}(m(t), c(t))$, $(D_t)_{jj} = \sum_{i=1}^n m_i (P_t)_{ij}(1 - (P_t)_{ij})$ and $(D_t)_{jk} = -\sum_{i=1}^n m_i (P_t)_{ij}(P_t)_{ik}$ $(j \neq k)$.

Proof. The proof is done by induction on t. We show that each time step:

- a new Gaussian error independent of the past is created by the Markovian evolution of the objects.
- Since all of the evolution parameters are differentiable, the Gaussian error of time t is scaled by a linear transformation.

The proof is detailed in Appendix 5.8.3. □

5.3.4. Beyond square-root convergence

So far, we have proved that as N grows, the system gets closer to a deterministic one: if at time t the system is in state $M^N(t)$, then at time $t+1$, the state of the system is close to $M^N(t)K(t)$. Moreover, we have shown that the optimal policy for the deterministic limit is asymptotically optimal for the stochastic system as well and we give bounds for the speed of convergence. The mean field central limit theorem (Theorem 5.8) shows that $M^N(t+1) \approx M^N(t)K(t) + \frac{1}{\sqrt{N}}G(t)$. This should be an even better approximation of the initial system. The purpose of this part is to show that this approximation is indeed better than mean field, in the sense that it leads to an error on the reward of order $\frac{\sqrt{\log N}}{N}$ instead of $\frac{1}{\sqrt{N}}$.

For any policy π and any initial condition $M^N(0), C^N(0)$ of the original process, let us define a coupled process $\widetilde{M}_\pi^N(t), \widetilde{C}_\pi^N(t)$ in $\mathbb{R}^S \times \mathbb{R}^d$ as follows:

- $(\widetilde{M}_\pi^N(0), \widetilde{C}_\pi^N(0)) \stackrel{\text{def}}{=} (M^N(0), C^N(0))$
- for $t \geq 0$:

$$\begin{aligned} \widetilde{M}_\pi^N(t+1) &\stackrel{\text{def}}{=} \widetilde{M}_\pi^N(t) K(A^N(t), \widetilde{C}_\pi^N(t)) + G_t(A^N(t), \widetilde{C}_\pi^N(t)) \\ \widetilde{C}_\pi^N(t+1) &\stackrel{\text{def}}{=} g(\widetilde{C}_\pi^N(t), \widetilde{M}_\pi^N(t+1), A^N(t)) \end{aligned}$$

where $A^N(t) \stackrel{\text{def}}{=} \pi_t(\widetilde{M}_\pi^N(t), \widetilde{C}_\pi^N(t))$ and $G_t(a, \widetilde{C}_\pi^N(t))$ is a sequence of *i.i.d.* Gaussian random variables independent of all $\widetilde{M}_\pi^N(t'), \widetilde{C}_\pi^N(t')$ for $t' < t$, corresponding to the error added by the random evolution of the objects between time t and time $t+1$. The covariance of $G_t(a, C)$ is a $S \times S$ matrix $D(a, C)$ where if we denote $P_{ij} \stackrel{\text{def}}{=} K_{ij}(a, C)$, then for all $j \neq k$:

$$D_{jj}(a, C) = \sum_{i=1}^{n} m_i P_{ij}(1 - P_{ij}) \quad \text{and} \quad D_{jk}(a, C) = -\sum_{i=1}^{n} m_i P_{ij} P_{ik}$$

Notice that $\widetilde{M}^N$ is not a positive measure anymore, but a signed element of $\mathbb{R}^S$. The Lipschitz functions r_t and g are originally only defined for positive vectors but can be extended to $\mathbb{R}^S \times \mathbb{R}^d$ (theorem of Kirszbraun, [132]) with the same Lipschitz constants.

In the following, the process $(\widetilde{M}_\pi^N(t), \widetilde{C}_\pi^N(t))$ is called the *mean field Gaussian approximation* of $(M_\pi^N(t), C_\pi^N(t))$. As for the definition of $V_\pi^N\left(M^N(0), C^N(0)\right)$, we define the expected reward of the mean field Gaussian approximation by:

$$W_\pi^N\left(M^N(0), C^N(0)\right) = \mathbb{E}\left(\sum_{t=1}^{T} \mathrm{r}_t\left(\widetilde{M}_\pi^N(t), \widetilde{C}_\pi^N(t)\right)\right).$$

The optimal cost of the mean field Gaussian approximation starting from the point $\left(M^N(0), C^N(0)\right)$ is $W_*^N\left(M^N(0), C^N(0)\right) = \sup_\pi W_\pi^N\left(M^N(0), C^N(0)\right)$. The following result shows that the Gaussian approximation is indeed a very accurate approximation of the original system.

Theorem 5.9. *Under assumptions (A1,A2,A3,A4), there exists a constant H independent of M^N, C^N such that*

(i) for all sequence of actions $a = a_1 \dots a_T$:

$$\left|V_a^N(M^N, C^N) - W_a^N(M^N, C^N)\right| \leq H \frac{\sqrt{\log(N)}}{N}.$$

(ii)

$$\left|V_*^N(M^N, C^N) - W_*^N(M^N, C^N)\right| \leq H \frac{\sqrt{\log(N)}}{N}.$$

Proof. The proof is detailed in Appendix 5.8.4. □

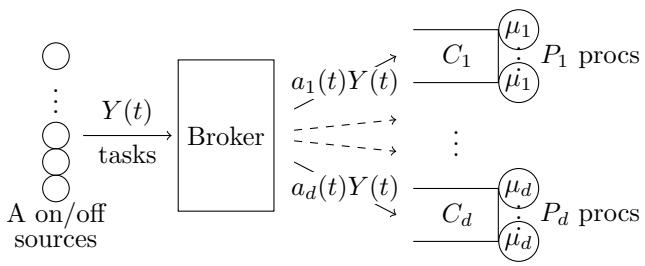


Figure 5.1.: The routing system

5.4. Application to a brokering problem

To illustrate the usefulness of our framework, let us consider the following model of a brokering problem in computational grids. There are A application sources that send tasks into a grid system and a central broker routes all theses tasks into d clusters (seen as multi-queues) and tries to minimize the total waiting time of the tasks. A similar queuing model of a grid broker was used in [120, 32, 33].

Here, time is discrete and the A sources follow a discrete on/off model: for each source $j \in \{1 \dots A\}$, let $(Y_j(t)) \stackrel{\text{def}}{=} 1$ if the source is on (*i.e.* it sends a task between t and $t+1$) and 0 if it is off. The total number of tasks sent between t and $t+1$ is $Y(t) \stackrel{\text{def}}{=} \sum_j Y_j(t)$. Each queue $i \in \{1 \dots d\}$ is composed of P_i processors, and all of them work at speed μ_i when available. Each processor $j \in \{1 \dots P_i\}$ of the queue i can be either *available* (in that case we set $X_{ij}(t) \stackrel{\text{def}}{=} 1$) or *broken* (in that case $X_{ij}(t) \stackrel{\text{def}}{=} 0$). The total number of processors available in the queue i between t and $t+1$ is $X_i(t) \stackrel{\text{def}}{=} \sum_j X_{ij}(t)$ and we define $B_i(t)$ to be the total number of tasks waiting in the queue i at time t. At each time slot t, the broker (or controller) allocates the $Y(t)$ tasks to the d queues: it chooses an action $a(t) \in \mathscr{P}(\{1 \dots Y_t\}^d)$ and routes each of the $Y(t)$ tasks to queue i with probability $a_i(t)$. The system is represented figure 5.1. The number of tasks in the queue i (buffer size) evolves according to the following relation:

$$B_i(t+1) = \Big(B_i(t) - \mu_i X_i(t) + a_i(t)Y(t)\Big)^+. \tag{5.8}$$

The cost that we want to minimize is the sum of the waiting times of the tasks. Between t and $t+1$, there are $\sum_i B_i(t)$ tasks waiting in the queue, therefore the cost at time t is $r_t(B) \stackrel{\text{def}}{=} \sum_i B_i(t)$. As we consider a finite horizon, we should decide a cost for the remaining tasks in the queue. In our simulations, we choose $r_T(B) \stackrel{\text{def}}{=} \sum_i B_i(T)$.

This problem can be viewed as a multidimensional restless bandit problem where computing the optimal policy for the broker is known to be a hard problem [149]. Here, indexability may help to compute near optimal policies by solving one MDP for each queue [149, 148]. However the complexity remains high when the number of processors in all the queues and the number of sources are large.

5.4.1. Mean field limit

This system can be modeled using the framework of objects evolving in a common environment.

- There are $N \stackrel{\text{def}}{=} A + \sum_{i=1}^{d} P_i$ "objects". Each object can either be a source (of type s) or a server (belonging to one of the queues, $q_1 \cdots q_d$), and can either be "on" or "off". Therefore, the possible states of one object is an element of $\mathcal{S} = \big\{(x,e) | x \in \{s, q_1, \cdots, q_d\}, e \in \{\text{on}, \text{off}\}\big\}$. The population mix M is the proportion of sources in state on and the proportion of servers in state on, for each queue.

- The action of the controller are the routing choices of the broker: $a_d(t)$ is the probability that a task is sent to queue d at time t.

- The environment of the system depends on the vector $B(t) = (B_1(t) \dots B_d(t))$, giving the number of tasks in queues $q_1, \dots q_d$ at time t. The time evolution of the i-th component is

$$B_i(t+1) = g_i(B(t), M^N(t+1), a(t)) \stackrel{\text{def}}{=} \Big(B_i(t) - \mu_i X_i(t) + a_i(t) Y(t)\Big)^+.$$

  The shared environment is represented by the context $C^N(t) \stackrel{\text{def}}{=} (\frac{B_1(t)}{N} \dots \frac{B_d(t)}{N})$.

- Here, the transition kernel can be time dependent but is independent of a and C. The probability of a object to go from a state $(x,e) \in \mathcal{S}$ to $(y,f) \in \mathcal{S}$ is 0 if $x \neq y$ (a source cannot become a server and vice-versa). If $x = y$ then $K_{(x,\text{on}),(x,\text{off})}(a,C)(t)$ as well as $K_{(x,\text{off}),(x,\text{on})}(a,C)(t)$ are arbitrary probabilities.

Here is how a system of size N is defined. A preliminary number of sources A_0 as well as a preliminary number P_i of servers per queue is given, totaling in N_0 objects. For any N, a system with N objects is composed of $\lfloor A_0 N/N_0 \rfloor$ (resp. $\lfloor P_i N/N_0 \rfloor$) objects that are sources (resp. servers in queue i). The remaining objects (to reach a total of N) are allocated randomly with a probability proportional to the fractional part of A/N_0 and $P_i N/N_0$ so that the mean number of objects that are sources is A/N_0 and the mean number of objects that are servers in queue i is $P_i N/N_0$. Then, each of these objects changes state over time according to the probabilities $K_{u,v}(a,C)(t)$. At time $t = 0$, a object is in state "on" with probability one half.

One can easily check that this system satisfies Assumptions (A1) to (A4) and therefore one can apply the convergence Theorem 5.5 that shows that if using the policies a^* or μ^*, when N goes to infinity the system converges to a deterministic system with optimal cost. An explicit computation of the policies a^* and μ^* is possible here and is postponed to Section 5.4.2. Also note that Assumption (A4-bis) on the convergence of the initial condition to a Gaussian variable is true since the random part of the initial state is bounded by $\frac{N_0}{N}$ and $\sqrt{N}\frac{N_0}{N}$ goes to 0 as N grows.

5.4.2. Optimal policy for the deterministic limit

As the evolution of the sources and of the processors does not depend on the environment, for all i, t, the quantities $\mu_i X_i(t)$ and $Y(t)$ converge almost surely to deterministic values that we call $x_i(t)$ and $y(t)$. If $y_i(t)$ is the number of tasks distributed to the ith queue at time t, $c_i(t+1) = (c_i(t) + y_i(t) - x_i(t))^+$. The deterministic optimization problem is to compute

$$\min_{y^1(1)\dots y^d(T)} \left\{ \sum_{t=1}^{T} \sum_{i=1}^{d} c_i(t) \text{ with } \left\{ \begin{array}{l} c_i(t+1) = (c_i(t) + y_i(t) - x_i(t))^+ \\ \sum_i y_i(t) = y(t) \end{array} \right. \right\}. \tag{5.9}$$

Let us call $w_i(t)$ the work done by the queue i at time t: $w_i(t) = c_i(t) - c_i(t-1) + y_i(t-1)$. The sum of the size of the queues at time t does not depend on with queue did the job but only on the quantity of work done:

$$\sum_{i=1}^{d} c_i(t) = \sum_{i=1}^{d} c_i(0) - \sum_{u \leq t, i} w_i(t).$$

Therefore to minimize the total cost, we have to maximize the total work done by the queues. Using this fact, the optimal strategy can be computed by iteration of a greedy algorithm. See [69] for more details.

The principle of the algorithm is the following.

1. The processors in all queues, which are "on" at time t with a speed μ are seen as slots of size μ.

2. At each time t, $y(t)$ units of tasks have to be allocated. This is done in a greedy fashion by filling up the empty slots starting from time t. Once all slots at time t are full, slots at time $t+1$ are considered and are filled up with the remaining volume of tasks, and so forth up to time T.

3. The remaining tasks that do not fit in the slots before T are allocated in an arbitrary fashion.

5.4.3. Numerical examples

We consider a simple instance of the resource allocation problem with 5 queues. Initially, they have respectively $1, 2, 2, 3$ and 3 processors running at speed $.5, .1, .2, .3$ and $.4$ respectively. There are 3 initial sources and the time horizon T equals 40. The transition matrices are time dependent and are chosen randomly before the execution of the algorithm – that is they are known for the computation of the optimal policy and are the same for all experiments. We ran some simulations to compute the expected cost of different policies for various sizes of the system. We compare different policies:

1. Deterministic policy a^* – to obtain this curve, the optimal actions $a_0^* \dots a_{T-1}^*$ that the controller must take for the deterministic system have been computed. At time t, action a_t^* is used regardless of the currently state, and the cost up to time T is displayed.

2. Limit policy μ^* – here, the optimal policy μ^* for the deterministic case was first computed. When the stochastic system is in state $(\mathrm{M}^N(t), \mathrm{C}^N(t))$ at time t, we apply the action $\mu_t^*(\mathrm{M}^N(t), \mathrm{C}^N(t))$ and the corresponding cost up to time T is reported.

3. Join the Shortest Queue (JSQ) and Weighted Join the Shortest Queue (W-JSQ) – for JSQ, each task is routed (deterministically) in the shortest queue. In W-JSQ, a task is routed in the queue whose weighted queue size $B_i/(\mu_i X_i)$ is the smallest.

The results are reported in Figures 5.2 and 5.3.

A series of several simulations for with different values of N was run. The reported values in the figures are the mean values of the waiting time over 10000 simulations for small values of N and around 200 simulations for big values of N. Over the whole range

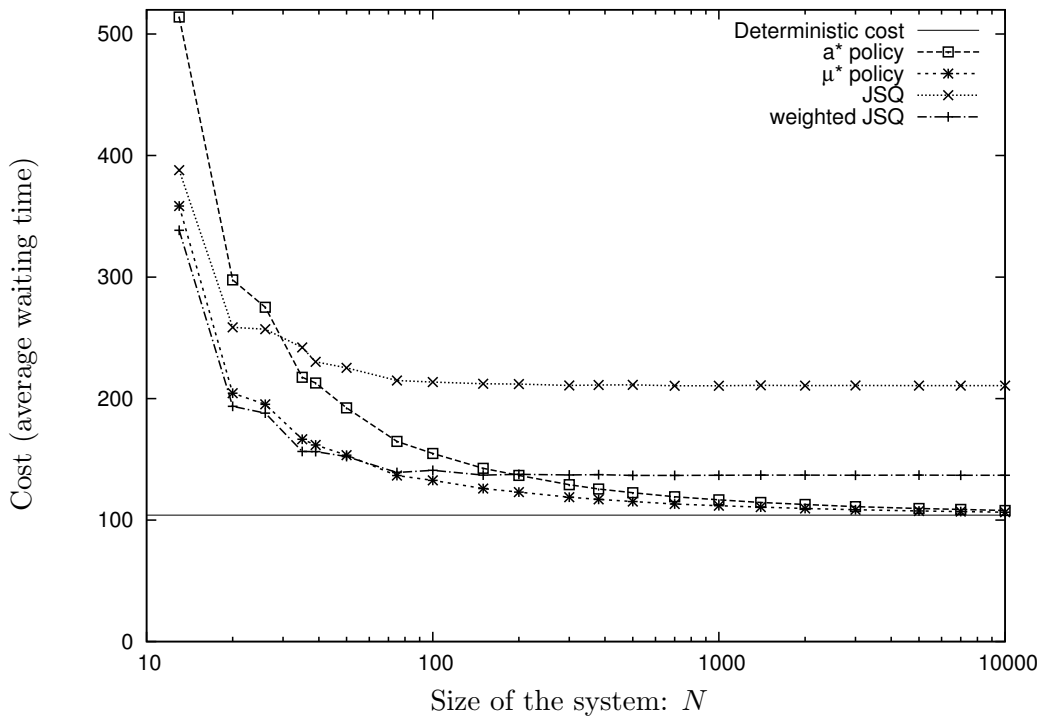


Figure 5.2.: Expected cost of the policies a^*, μ^*, JSQ and W-JSQ for different values of N.

for N, the 95% confidence interval is less than 0.1% for the expected cost – Figure 5.2 – and less than 5% for the central limit theorem – Figure 5.3.

Figure 5.2 shows the average waiting time of the stochastic system when we apply the different policies. The horizontal line represents the optimal cost of the deterministic system $v^*(m_0, c_0)$ which is probably less than $V_*^N(\mathrm{M}(0), \mathrm{C}(0))$. This figure illustrates Theorem 5.5: if we apply a^* or μ^*, the cost converges to $v_*(m(0), c(0))$.

In Figure 5.2, one can see that for low values of N, all the curves are not smooth. This behavior comes from the fact that when N is not very large with respect to N_0, there are at least $\lfloor \frac{N}{N_0} A \rfloor$ (resp. $\lfloor \frac{N}{N_0} P_i \rfloor$) objects that are sources (resp. processors in queue i) and the remaining objects are distributed randomly. The random choice of the remaining states are chosen so that $\mathbb{E}\left(A^N\right) = \frac{N}{N_0} A$, but the difference $A^N - N N_0 A$ may be large. Therefore, for some N the load of the system is much higher than the average load, leading to larger costs. As N grows, the proportion of remaining objects decreases and the phenomena becomes negligible.

A second feature that shows in Figure 5.2, is the fact that on all curves, the expected waiting times are decreasing when N grows. This behavior is certainly related to Ross conjecture [126] that says that for a given load, the average queue length decreases when the arrival and service processes are more deterministic.

Finally, the most important information on this figure is the fact that the optimal deterministic policy and the optimal deterministic actions perform better than JSQ and weighted JSQ as soon as the total number of elements in the system is over 200 and 50 respectively. The performance of the deterministic policy a^* is quite far from W-JSQ and

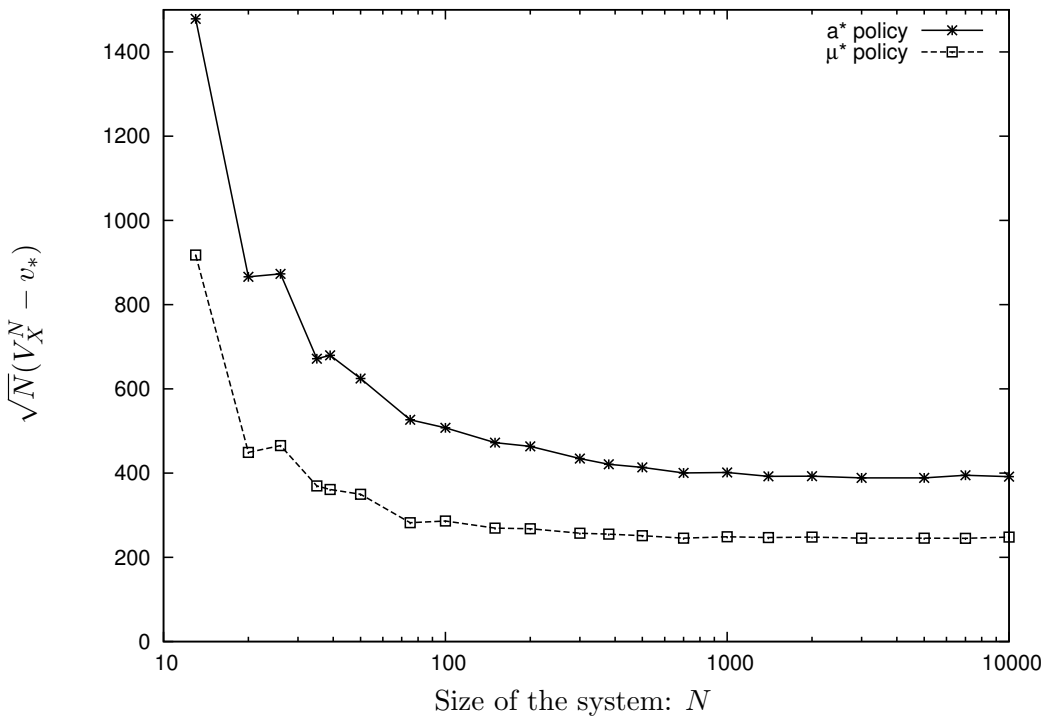


Figure 5.3.: Speed of convergence of the policies $X = a^*$ or μ^* for different values of N.

JSQ for small values of N, and it rapidly becomes better than JSQ ($N \geq 30$) and W-JSQ ($N \geq 200$). Meanwhile the behavior of μ^* is uniformly good even for small values of N.

Figure 5.3 illustrates Theorem 5.7 which says that the speed of convergence towards the limit is of order $\sqrt{N}$. On the y-axis, $\sqrt{N}$ times the average cost of the system minus the optimal deterministic cost is plotted. One can see that the gap between the expected cost of the policy μ^* (resp. a^*) and the deterministic cost $v_*(m(0), c(0))$ is about $250/\sqrt{N}$ (resp. $400/\sqrt{N}$) when N is large. This shows that the speed of convergence of $1/\sqrt{N}$ is a tight bound. This should be an upper bound on the constant δ defined in Equation (5.7).

Besides comparing a^* and μ^* to other heuristics, it would be interesting to compare it to the optimal policy of the stochastic system, whose cost is $V_*^N(\mathrm{M}(0), \mathrm{C}(0))$. One way to compute this optimum would be by using Bellman fixed point equation. However to do so, one needs to solve it for all possible values of M and C. In this example, C can be as large as the length of the five queues and each object's state can vary in {on,off}. Therefore even with $N = 10$ and if we only compute the cost for queues of size less than 10, this leads to $2^N 10^5 \approx 10^8$ states which is hard to handle even with powerful computers.

5.5. Extensions and Counter-Examples

This part is devoted to several extensions of the previous results as well as to some counter-examples showing the limitations of the mean field model.

5.5.1. Object-Dependent Actions

Up to now, we have assumed that the controller takes the same action for all objects. Here, we show that our framework can also be used in the case where the controller can take a different action for each object, depending on its state but also on the object itself.

For that, we consider the following new system. The state of the system is the states of the N objects $\mathcal{X}^N(t) = (X_1^N(t) \dots X_N^N(t))$ and the state of the context. At each time step, the controller chooses an N-uple of actions $a_1 \dots a_N \in \mathcal{A}$ and uses the action a_i for the ith object. We assume that $\mathcal{A}$ is finite. The reward and the evolution of the context is defined as before and we call $V_{od*}^N(\mathcal{X}^N(0), \mathrm{C}^N(0))$ the optimal reward of this system over a finite horizon $[0; T]$ where *od* stands for "Object-Dependent"-actions.

As before, $\mathrm{M}^N(0) \stackrel{\text{def}}{=} N^{-1} \sum_{n=1}^N \delta_{X_n^N(0)}$ is the empirical measure of $\mathcal{X}^N(0)$. We consider our original problem in which we replace the action set $\mathcal{A}$ by $\mathscr{P}(\mathcal{A})^{\mathcal{S}}$. An action is a $\mathcal{S}$-uple $(p_1 \dots p_S)$. If the controller takes the action p, then an object in state i will choose an action a according to the distribution p and evolves independently according to $K(a, \mathrm{C})$.

Compared to the problem in which the action sent is object-dependent, the action of the controller in the latter case is more constrained since it can not choose which object or even the exact number of objects receiving a particular action. However, as we see in Proposition 5.10, this difference collapses as N grows. Other results, such as second order results, also hold.

Proposition 5.10. *If $g, K, \mathcal{A}, \mathrm{M}^N(0), \mathrm{C}^N(0)$ satisfy assumptions (A1,A2,A3,A4), then the object-dependent reward V_{od*}^N converges to the deterministic limit:*

$$\lim_{N\to\infty} V_{od*}^N(\mathcal{X}^N(0), \mathrm{C}^N(0)) = \lim_{N\to\infty} V_*^N(\mathrm{M}^N(0), \mathrm{C}^N(0)) = v_*(m(0), c(0))$$

where the deterministic limit has an action set $\mathscr{P}(\mathcal{A})$.

Sketch of proof. To each N-uple $a = a_1 \dots a_N \in \mathcal{A}^N$ and each vector $\mathcal{X}^N \in \mathcal{S}^N$, we associate a $\mathcal{S}$-uple of probability measures on the set $\mathcal{A}$ defined by

$$(p_{a,\mathcal{X}^N})_i \stackrel{\text{def}}{=} \frac{1}{\sum_{n=1}^N \mathbf{1}_{X_n^N = i}} \sum_{n=1}^N \mathbf{1}_{X_n^N = i} \delta_{a_n}.$$

For $b \in \mathcal{A}$, $(p_{a,\mathcal{X}^N})_i(b)$ represents the average number of objects in state i that received the action b.

One can show that starting from $\mathcal{X}^N(t), \mathrm{C}^N(t)$ and applying action a or $p_{a,\mathcal{X}^N(t)}$ both lead to the same almost sure limit for $(\mathrm{M}^N(t+1), \mathrm{C}^N(t+1))$. Then one can show by induction on the time-horizon T that the reward under a fixed sequence of action $V_{od*}^N(\mathcal{X}^N(0), \mathrm{C}^N(0))$ is asymptotically equal to $V_*^N(\mathrm{M}^N(0), \mathrm{C}^N(0))$.

Then remarking that $(\mathscr{P}(\mathcal{A}))^{\mathcal{S}}$ also satisfies hypothesis (A2) and (A3) – it is compact and the mappings g, K and r are uniformly continuous if $p_a \in \mathscr{P}(\mathcal{A})$ – we can apply the rest of the results and show that the reward converges to the deterministic counterpart. □

5.5.2. Infinite horizon discounted reward

In this section, we prove first and second order results for infinite-horizon discounted Markov decision processes. As in the finite case, we will show that when N grows large,

the maximal expected discounted reward converges to the one of the deterministic system and the optimal policy is also asymptotically optimal. To do this, we need the following new assumptions:

(A6) **Homogeneity in time** – The reward r_t and the probability kernel K_t do not depend on time: there exists r, K such that, for all $\mathrm{M}, \mathrm{C}, a$ $\mathrm{r}_t(\mathrm{M}, \mathrm{C}) = \mathrm{r}(\mathrm{M}, \mathrm{C})$ and $K_t(a, \mathrm{C}) = K(a, \mathrm{C})$.

(A7) **Bounded reward** – $\sup_{\mathrm{M},\mathrm{C}} \mathrm{r}(\mathrm{M}, \mathrm{C}) \leq K < \infty$.

The homogeneity in time is clearly necessary as we are interested in infinite-time behavior. Assuming that the cost is bounded might seem strong but it is in fact very classical and holds in many situation, for example when C is bounded. The rewards are discounted according to a discount factor $0 \leq \delta < 1$: if the policy is π, the expected total discounted reward of π is (δ is omitted in the notation):

$$V_\pi^N(\mathrm{M}^N(0), \mathrm{C}^N(0)) \stackrel{\text{def}}{=} \mathbb{E}^\pi(\sum_{t=1}^{\infty} \delta^{t-1} \mathrm{r}(\mathrm{M}^N(t), \mathrm{C}^N(t))).$$

Notice that Assumption (A7) implies that this sum remains finite. The optimal total discounted reward V_*^N is the supremum on all policies. For $T \in \mathbb{N}$, the optimal discounted finite-time reward until T is

$$V_{T*}^N(\mathrm{M}(0), \mathrm{C}(0)) \stackrel{\text{def}}{=} \sup_\pi \mathbb{E}^\pi(\sum_{t=1}^{T} \delta^{t-1} \mathrm{r}(\mathrm{M}(t), \mathrm{C}(t))).$$

As r is bounded by $K < \infty$, the gap between V_{T*}^N and V_*^N can be bounded independently of $N, \mathrm{M}, \mathrm{C}$:

$$\left|V_{T*}^N(\mathrm{M}, \mathrm{C}) - V_*^N(\mathrm{M}, \mathrm{C})\right| \leq K \sum_{t=T+1}^{\infty} \delta^t = K \frac{\delta^{T+1}}{1-\delta}. \tag{5.10}$$

In particular, this shows that V_{T*}^N converges uniformly in (M, C) and N to V_*^N as T goes to infinity:

$$\lim_{T\to\infty} \sup_{N,\mathrm{M},\mathrm{C}} \left|V_{T*}^N(\mathrm{M}, \mathrm{C}) - {V_*}^N(\mathrm{M}, \mathrm{C})\right| = 0.$$

Equation (5.10) is the key of the following analysis. Using this fact, we can prove the convergence when N grows large for fixed T and then let T go to infinity. Therefore with very few changes in the proofs of Section 5.3.2, we have the following result:

Theorem 5.11. (Optimal discounted case) *Under assumptions (A1,A2,A3,A4,A6, A7), as N grows large, the optimal discounted reward of the stochastic system converges to the optimal discounted reward of the deterministic system:*

$$\lim_{N\to\infty} {V_*}^N(\mathrm{M}^N, \mathrm{C}^N) =_{a.s} v_*(m, c),$$

where $v_*(m, c)$ satisfies the Bellman equation for the deterministic system:

$$v_*(m, c) = \mathrm{r}(m, c) + \delta \sup_{a\in\mathcal{A}} \Big\{ v_*(\Phi_a(m, c)) \Big\}.$$

The first order of convergence for the discounted cost is a direct consequence of the finite time behavior convergence. However, when considering second order results, similar difficulties as in the infinite horizon case arise and the convergence rate depends on the behavior of the system when T goes to infinity.

Proposition 5.12. *Under assumptions (A1,A2,A3,A4,A6,A7) and if the functions $c \mapsto K(a,c)$, $(m,c) \mapsto g(c,m,a)$ and $(m,c) \mapsto r(m,c)$ are Lipschitz with constants L_K,L_g and L_r satisfying $\max(1, L_g)(S + L_K + 1)\delta < 1$, the constants $H \stackrel{\text{def}}{=} L_r \sum_t \delta^t \beta_t$ and $H' \stackrel{\text{def}}{=} L_r \sum_t \delta^t \beta'_t$ yield*

$$\lim_{N\to\infty} \sqrt{N} \left\| V_*^N(\mathrm{M}^N(0), \mathrm{C}^N(0)) - v_*(m(0), c(0)) \right\| \leq H + H'\sqrt{N}\epsilon_0^N$$

where $\epsilon_0^N \stackrel{\text{def}}{=} \left\| \left(\mathrm{M}^N(0), \mathrm{C}^N(0)\right) - (m(0), c(0)) \right\|$ and β_t and β'_t are defined in Proposition 5.6.

Proof. (sketch) This result is a direct consequence of Proposition 5.6 and can be proved similarly to Theorem 5.7. In particular, it uses the fact that in Equation (5.5) of Proposition 5.6,

$$\sqrt{N}\mathbb{E}\left(\left\| \left(M_\pi^N(t), C_\pi^N(t)\right) - \left(m_{A_\pi^N}(t), c_{A_\pi^N}(t)\right) \right\|\right) \leq \beta_t + \beta'_t\sqrt{N}\epsilon_0^N + e_t^N, \tag{5.11}$$

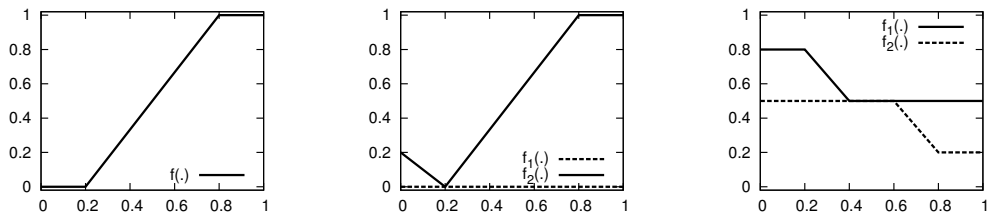
the growth of the constants β_t and β'_t and e_t^N is bounded by a factor $\max(1, L_g)(S + L_K + 1)$. □

Example 1. *This example is a system without control. We show that even in this simple case, $\sum_{t=0} \delta^t \sqrt{N} \left(\mathrm{r}(M^N(t)) - \mathrm{r}(m(t))\right)$ does not converge if δ does not satisfy the assumptions of Proposition 5.12. The system is defined as follows:*

- *The state space is $\mathcal{S} = \{0, 1\}$ and the context is $\mathrm{C}(t) \stackrel{\text{def}}{=} \mathrm{M}_0^N(t)$ (C is the mean number of particles in state 0). Therefore the interesting process is $\mathrm{C}(t)$.*
- *For any object, the probability of going to state 0 is $f(\mathrm{C})$ (independent of the state) where f is a piecewise linear function with $f(0) = f(\alpha) = 0$ and $f(1-\alpha) = f(1) = 1$ for a number $\alpha < \frac{1}{2}$. The transition function is depicted on the left of Figure 5.4(a).*
- *The starting point is $\mathrm{M}_0^N(0) = \mathrm{C}(0) = \frac{1}{2}$.*
- *The reward function is $\mathrm{r}(\mathrm{M}, \mathrm{C}) = \left|\mathrm{C} - \frac{1}{2}\right|$.*

The deterministic limit of M^N starting in $\frac{1}{2}$ is constant equal to $\frac{1}{2}$. Therefore, the gap between the discounted reward of the stochastic system and the discounted reward of the limit normalized by $\sqrt{N}$ is $\sum_{t=0}^{\infty} \delta^t \sqrt{N} \left|\mathrm{C}^N(t) - \frac{1}{2}\right|$.

The process $\mathrm{C}^N(t) - \frac{1}{2}$ is quite complicated. However for any finite horizon $[0; T]$, if N is large enough, $\mathrm{C}^N(t) - \frac{1}{2}$ is close 0 with high probability. During this time, $\sqrt{N}(\mathrm{C}^N(t) - \frac{1}{2})$ is close to the process $(Y^N(t))_{t\in\mathbb{N}}$ where $Y^N(t)$ is defined by $Y^N(0) = 0$ and $X_{t+1}^N = LX_t^N + G_t$ where L is the Lipschitz constant of f, $L \stackrel{\text{def}}{=} \frac{1}{1-2\alpha}$, and G_t are i.i.d *Gaussian variables of mean 0 and variance $1/4$. As sum of* i.i.d *Gaussian variables, X_t^N is a Gaussian variable of variance $\sum_{i=0}^{t-1} L^{2i} = L^{2t}\frac{1-L^{-2t}}{L^2-1}$. Therefore, if $\delta \geq 1/L$, $\mathbb{E}\left(\sum_{i=0}^{t-1} \delta^t \left|X_t^N\right|\right)$ goes to $+\infty$ as t goes to infinity. By choosing T large enough, this shows that $\sum_{t=0}^{\infty} \delta^t \sqrt{N} \left|\mathrm{C}^N(t) - \frac{1}{2}\right|$ is not bounded as N grows.*



(a) Example 1 (b) Example 2 (c) Example 3

Figure 5.4.: The transitions functions of Examples 1, 2 and 3 (from left to right). On each figure is draw the probability for an object to go in state 1 as a function M_0^N for the different actions.

5.5.3. Average Reward

The discounted problem is very similar to the finite case because the total reward mostly depends on the rewards during a finite amount of time. Here, we consider another infinite-horizon criterion, namely the average reward. The optimal average reward is (if it exists[1])

$$V_{av*}^N = \lim_{T\to\infty} \frac{1}{T} V_{T*}(\mathrm{M}(0), \mathrm{C}(0)).$$

This raises the problem of the exchange of the limits $N \to \infty$ and $T \to \infty$. Consider a case without control with two states $\mathcal{S}=\{0;1\}$ and $\mathrm{C}(t)$ is the mean number of objects in state 1 ($\mathrm{C}(t) = (\mathrm{M}(t))_1$) and with a function $f{:}[0;1]{\to}[0;1]$ such that the transition kernel K is $K_{i1}(\mathrm{C}) = f(\mathrm{C})$ for $i \in \mathcal{S}$. If $\mathrm{M}_0^N(0) \xrightarrow{\text{a.s.}} m_0$ then for any fixed t, $\mathrm{M}^N(t)$ converges to $f(f(\dots f(m(0))\dots))$. However, in general we may have $\lim_{t\to\infty}\lim_{N\to\infty}\mathrm{M}^N(t) \neq \lim_{N\to\infty}\lim_{t\to\infty}\mathrm{M}^N(t)$. For example if $f(x) = x$, the deterministic system is constant while the stochastic system converges almost surely to a random variable (as a bounded Martingale) that only takes values in $\{0;1\}$. In some situations, such as Example 2, the optimal policy of the deterministic limit differs from the optimal policy for the stochastic system.

Example 2. *We consider a similar example as Example 1: there are two states and $C^N = M_0^N$ (the proportion of objects in state 0). We consider two possible actions, say 1 and 2, corresponding to a probability to go from any state to 0 equal to $f_1(C)$ and $f_2(C)$ respectively, defined by (see Figure 5.4(b)).*

- $f_1(C) = 0$.
- *f_2 is piecewise linear with $f_2(0) = 0.2, f_2(0.1) = 0, f_2(0.8) = 1, f_2(1) = 1$.*

The reward function is $\mathrm{r}(C) = |C - 0.1|$.

For the deterministic system, applying action 1 always makes the system converge to 0 while applying action 2 makes the system converge to .2 if we start in $[0;0.5)$ and 1 in $(0.5;1]$. Therefore, if we start in $[0;.5)$, the average reward of the deterministic system is maximized under action 1 (it gives 0.1).

For the stochastic system, applying action 1 makes the system converge to 0. However, applying action 2 makes the system converge to 1: there is a small but positive probability

[1] If it does not exist, one may replace this limit by lim sup or lim inf.

that M^N goes to something greater than 0.8 at each step which makes the system go to 1. Therefore, if we start in $[0; .5)$, it is better to apply the action 2, which is different from the optimal policy of the limit.

In the case without control, Proposition 5.13 gives the condition under which the ergodic behavior of the system with N finite converges to the asymptotic behavior of its deterministic limit. This result is similar to the classical results of stochastic approximation theory concerning differential equations limit (see [47] for example). However, no general results for the controlled problem is presented here since the condition to apply theses results are too restrictive to be applied in practical situations (see Example 3) for an example where many assumptions are verified but where we cannot exchange the limits.

Let us assume that the context C is bounded (from above and from below). This implies that the couple (M, C) lives in a compact set $B \subset \mathbb{R}^{S+d}$. Let $f_a : B \to B$ denote the deterministic function corresponding to one step of the evolution of the deterministic limit under action a. The definition of f_a is given by Equation (5.2):

$$f_a(m, c) = (m', c') \text{ with } \begin{cases} m' &= m \cdot K(a, c) \\ c' &= g(c, m', a). \end{cases}$$

We say that a set H is an attractor of the function f_a if

$$\lim_{t\to\infty} \sup_{x\in B} d(f_a^t(x), H) = 0,$$

where $d(x, H)$ denotes the distance between a point x and a set H and $f_a^t(x)$ denotes t iterations of f_a applied to x: $f_a^t(x) = f_a(f_a(\dots f_a(x)))$.

The following proposition shows that as t goes to infinity and N goes to infinity, $(M^N(t), C^N(t))$ concentrates around the attractors of f_a.

Proposition 5.13. *Under assumptions (A1,A2,A3), if the controller always chooses action a then for any attractor H of f_a and for all $\epsilon > 0$:*

$$\lim_{N\to\infty} \limsup_{t\to\infty} \mathcal{P}\left(d\left(\left(M_a^N(t), C^N(t)\right), H\right) \geq \epsilon\right) = 0$$

Proof. Let $\epsilon > 0$. Since H is an attractor, there exists T such that

$$\sup_{x\in H} d(f_a^T(x), H) \leq \epsilon/2.$$

For all $t \in \mathbb{N}$, using the triangular inequality, we have

$$d(X^N(t+T), H) \leq \left\|X^N(t+T) - f_a^T(X^N(t))\right\| + d(f_a^T(X^N(t)), H).$$

By Theorem 5.1, the first part of this inequality is less than $\mathcal{E}_T(\delta, 0)$ with probability greater than $\exp(-2N\delta^2)$. Moreover, $\mathcal{E}_T(\delta, 0)$ converges to 0 as δ goes to 0. Therefore, there exists δ such that $\mathcal{E}_T(\delta, 0) < \epsilon/2$. This implies that for such δ and all $t \geq 0$,

$$\mathcal{P}\left(d(X^N(t+T), H) \geq \epsilon\right) \leq 2TS^2 \exp(-2N\delta^2),$$

which goes to 0 as N goes to infinity. □

We say that a point x is an attractor of f_a if $\{x\}$ is an attractor of f_a. As a direct corollary of Proposition 5.13, we have:

Corollary 5.14. *If the function f_a has a unique attractor (m_∞, c_∞), then*

$$\limsup_{t\to\infty} \left\| \left(\mathrm{M}^N(t), \mathrm{C}^N(t)\right) - (m_\infty, c_\infty) \right\| \to 0 \quad \text{in probability.}$$

In the controlled case, there is no simple positive result under assumptions that are easy to check in practice. In particular, assuming that all f_a have the same attraction point, does not ensure that the average reward converges to its deterministic counterpart as Example 3 shows.

Example 3. *As in the example 1, we consider a system with 2 states where $\mathrm{C}^N = \mathrm{M}_0^N$ is the proportion of objects in state 0. The only difference here is that there are two possible actions 1 and 2, corresponding to a probability of transition from any state to 0 of $f_1(\mathrm{C})$ and $f_2(\mathrm{C})$. Both f_1 and f_2 are piecewise linear functions taking the values:*

- *$f_1(x) = 0.8$ for $x \leq 0.2$, 0.5 for $x > 0.4$;*
- *$f_2(x) = 0.5$ for $x \leq 0.6$, 0.2 for $x > 0.6$.*

Figure 5.4(c) shows the transition functions. The reward is set to $|\mathrm{C}^N - 1/2|$.

Both f_1 and f_2 have the same attractor, equal to $\{1/2\}$. Moreover, one can prove that under any policy, $\lim_{N\to\infty} \lim_{t\to\infty} \mathrm{M}_\pi^N(t)$ will converge to 0.5, leading to an average reward of 0 regardless of the initial condition. However, if the deterministic limit starts from the point $\mathrm{C}^N(0) = .2$, then by choosing the sequence of actions $1, 2, 1, 2 \ldots$ the system will oscillate between 0.2 and 0.8, leading to an average reward of 0.3.

This is caused by the fact that even if f_1 and f_2 have the same unique attractor, $f_1 \circ f_2$ has 3 accumulation points: $0.2, 0.5$ and 0.8.

5.6. Computational issues

Throughout the chapter, we have shown that if the controller uses the optimal policy μ^* of the deterministic limit of the finite real system, the expected cost will be close to the optimal one (Theorem 5.5). Moreover, Theorem 5.7 gives a bound on the error that we make. However to apply these results in practice, a question remains: how difficult is it to compute the optimal limit policy?

The first answer comes straight from the example. In many cases, even if the stochastic system is extremely hard to solve, the deterministic limit is often much simpler. The best case of course is, as in the example of Section 5.4, when one can compute the optimal policy. If one can not compute it, there might also exist approximation policies with bounded error (see [87] for a review on the subject). Imagine that a 2-approximation algorithm exists for the deterministic system, then, Theorem 5.5 proves that for all ε, this algorithm will be a $(2+\varepsilon)$-approximation for the stochastic system if N is large enough. Finally, heuristics for the deterministic system can also be applied to the stochastic version of the system.

If none of this works properly, one can also compute the optimal deterministic policy by "brute-force" computations using the equation

$$v_{t\ldots T*}(m, c) = \mathrm{r}_t(m, c) + \sup_a v_{t+1\ldots T*}(\Phi_a(m, c)),$$

where $v_{t\ldots T*}$ denotes the optimal reward of the deterministic limit over finite horizon $\{t\ldots T\}$. In that case, an approximation of the optimal policy is obtained by discretizing the state space and by solving the equation backward (from $t=T$ to $t=0$), to obtain the optimal policy for all states. The brute force approach can also be applied directly on the stochastic equation using:

$$V^N_{t\ldots T*}(\mathrm{M},\mathrm{C}) = \mathrm{r}_t(\mathrm{M},\mathrm{C}) + \sup_{a\in\mathcal{A}} \mathbb{E}\left(V^N_{t+1\ldots T*}\left(\Phi^N_a(\mathrm{M},\mathrm{C})\right)\right).$$

However, solving the deterministic system has three key advantages. The first one is that the size of the discretized deterministic system may have nothing to do with the size of the original state space for N objects: it depends mostly on the smoothness of functions g and ϕ rather than on N. The second one is the suppression of the expectation which might reduce the computational time by a polynomial factor[1] by replacing the $|\mathscr{P}_N(\mathcal{S})|$ possible values of M^N_{t+1} by 1. The last one is that the suppression of this expectation allows one to carry the computation going forward rather than backward. This latter point is particularly useful when the action set and the time horizon are small.

5.7. Conclusion and future work

In this chapter, we have shown how the mean field framework can be used in an optimization context: the results known for Markov chains can be transposed almost unchanged to Markov decision processes. We further show that the convergence to the mean field limit in both cases (Markovian and Markovian with controlled variables) satisfies a central limit theorem, providing insight on the speed of convergence. In the next chapter, we consider the case of stochastic systems where the event rate depends on N. in that case, the deterministic regime is given by differential equations.

5.8. Appendix: proofs

5.8.1. Proof of Theorem 5.1 (controlled mean field)

Let $U^N_{i,n}(t)$ be a collection of *i.i.d.* random variables uniformly distributed on $[0;1]$. Let $\pi=\{\pi_t:\mathscr{P}(\mathcal{S})\times\mathbb{R}^d\mapsto a\}$ be a policy. The evolution of the process $(M^N(t),C^N(t))$ can be defined as follows:

$$\begin{aligned} M^N_j(t+1) &= \frac{1}{N}\sum_{i=1}^{S}\sum_{n=1}^{NM^N_i(t)} \mathbf{1}_{H_{ij}(A^N_\pi(t),C^N(t))\le U^N_{i,n}(t)\le H_{ij+1}(A^N_\pi(t),C^N(t))} \\ C^N(t+1) &= g(C^N(t),M^N(t+1),A^N_\pi(t)) \end{aligned}$$

where $H_{ij}(a,C)\stackrel{\text{def}}{=}\sum_{\ell=1}^{j-1}K_{i\ell}(a,C)$ and $A^N_\pi(t)\stackrel{\text{def}}{=}\pi_t(M^N(t),C^N(t))$.

Let $B^N_{inj}\stackrel{\text{def}}{=}\mathbf{1}_{H_{ij}(A^N_\pi(t),C^N(t))\le U^N_{i,n}(t)\le H_{ij+1}(A^N_\pi(t),C^N(t))}$. $(B^N_{inj})_{i,n,N}$ are *i.i.d.* Bernoulli random variable with mean $\mathbb{E}\left(B^N_{inj}|A^N_\pi(t)=a,C^N(t)=c\right)=K_{ij}(a,c)$. Therefore, by

[1] The size of $\mathscr{P}_N(\mathcal{S})$ is the binomial coefficient $(N+1+S,S)\sim_{N\to\infty}\frac{N^S}{S!}$

Hoeffding's inequality (Inequality 2.3 of [88]), we have:

$$\mathcal{P}\left(\left|\sum_{n=1}^{NM_i^N(t)} B_{inj}^N(t) - NM_i^N(t)K_{ij}(A_\pi^N(t), C^N(t))\right| \geq N\epsilon\right) \leq 2\exp(-2\frac{N}{M_i^N(t)}\epsilon^2)$$
$$\leq 2\exp(-2N\epsilon^2) \qquad (5.12)$$

Therefore, the quantity $\left|\sum_{n=1}^{NM_i^N(t)} B_{inj}^N(t) - NM_i^N(t)K_{ij}(A_\pi^N(t), C^N(t))\right|$ is less than $N\epsilon$ for all i, j with probability greater than $1 - 2S^2\exp(-2N\epsilon^2)$. If this holds for all i, j and if $\left\|(M^N(t), C^N(t)) - (m(t), c(t))\right\| \leq \epsilon_t$, then for all $1 \leq j \leq S$, the gap at time $t+1$ between the jth component of M^N ($M_j^N(t+1)$) and m ($m_j(t+1)$) is:

$$\begin{aligned}
\left|M_j^N(t{+}1){-}m_j(t{+}1)\right| &= \left|\sum_{i=1}^{S}\frac{1}{N}\sum_{n=1}^{NM_i^N(t)} B_{inj}^N(t) - m_i(t)K_{ij}(A_\pi^N(t), c(t))\right| \\
&\leq \sum_{i=1}^{S}\frac{1}{N}\left|\sum_{n=1}^{NM_i^N(t)} B_{inj}^N(t) - NM_i^N(t)K_{ij}(A_\pi^N(t), C^N(t))\right| \\
&\quad + \sum_{i=1}^{S}\left|\left(M_i^N(t) - m_i(t)\right)\right| K_{ij}(A_\pi^N(t), C^N(t)) \\
&\quad + \sum_{i=1}^{S} m_i(t)\left|K_{ij}(A_\pi^N(t), C^N(t)) - K_{ij}(A_\pi^N(t), c(t))\right| \\
&\quad + \left|\sum_{i=1}^{S}\left(M_i^N(t){-}m_i(t)\right)\left(K_{ij}(A_\pi^N(t), C^N(t)){-}K_{ij}(A_\pi^N(t), c(t))\right)\right| \\
&\leq S\epsilon + S\epsilon_t + L_K\epsilon_t + SL_K\epsilon_t^2. \qquad (5.13)
\end{aligned}$$

where we use (5.12) for the first part of the inequality, the fact that $K_{ij} \leq 1$ for the second and the fact that $\sum_{i=1}^{S} m_i(t) = 1$ and that K is Lipschitz with constant L_K for the third one.

Moreover, using the fact that g is Lipschitz with constant L_g ($\|g(c, m, a) - g(c', m', a)\| \leq L_g\|(c, m) - (c', m')\|$), we have

$$\begin{aligned}
\left\|C^N(t+1) - c(t+1)\right\| &= \left\|g(C^N(t), M^N(t), a) - g(c(t), m(t), a)\right\| \\
&\leq L_g\max(\epsilon_t, S\epsilon + S\epsilon_t + L_K\epsilon_t + SL_K\epsilon_t^2) \\
&\leq L_g(S\epsilon + S\epsilon_t + L_K\epsilon_t + SL_K\epsilon_t^2).
\end{aligned}$$

This implies that

$$\begin{aligned}
\left\|(M^N(t{+}1), C^N(t{+}1)) - (c(t{+}1), m(t{+}1))\right\| &\leq (S\epsilon + S\epsilon_t + L_K\epsilon_t + SL_K\epsilon_t^2)\max(L_g, 1) \\
&\leq \mathcal{E}_{t+1}\left(\epsilon, \left\|M^N(0), C^N(0) - m(0), c(0)\right\|\right),
\end{aligned}$$

where

$$\mathcal{E}_{t+1}(\epsilon, \delta) = \left(S\epsilon + (2 + L_K)\,\mathcal{E}_t(\epsilon, \delta) + L_K\mathcal{E}_t(\epsilon, \delta)^2\right)\max(1, L_g).$$

By a direct induction on t, this holds with probability greater than $2(t{+}1)S^2\exp(-2N\epsilon^2)$.

5.8.2. Proof of Theorem 5.6 (second order result)

We are interested in comparing the behavior of $(M_\pi^N(t), C_\pi^N(t))$ and $(m_{A_\pi^N}(t), c_{A_\pi^N}(t))$ where A_π^N is the sequence of actions taken by the controller under policy π (A_π^N is a random variable depending on the values of $(M_\pi^N(t), C_\pi^N(t))$) and $(m_{A^N}(t), c_{A_\pi^N}(t))$ corresponds to the value of the deterministic limit when the controller apply the sequence of action A_π^N. In order to improve the readability, we suppress the indexes π and A_π^N. Variables $M_\pi^N(t), C_\pi^N(t), m_{A_\pi^N}(t), c_{A_\pi^N}(t)$ will be denoted $M^N(t), C^N(t), m(t), c(t)$, respectively.

We will show Equation (5.5) by induction on t. Let us first recall that Equation (5.5) is:

$$\sqrt{N}\mathbb{E}\left(\left\|\left(M^N(t), C^N(t)\right) - (m(t), c(t))\right\|\right) \leq \beta_t + \beta_t'\sqrt{N}\epsilon_0^N + 1 + e_t^N,$$

where $\epsilon_0^N \stackrel{\text{def}}{=} \mathbb{E}\left(\left\|\left(M^N(0), C^N(0)\right) - (m(0), c(0))\right\|\right)$.

For $t = 0$, it is satisfied by taking $\beta_t = 0$, $\beta_t' = 1$ and $\epsilon^N(0) = 0$.

Assume now that Equation (5.5) is true for some $t \geq 0$. Let $P^N(t) \stackrel{\text{def}}{=} K(A^N(t), C^N(t))$ and $p(t) \stackrel{\text{def}}{=} K(A^N(t), c(t))$. $P^N(t)$ corresponds to the transition matrix at time t of the objects in the system of size N, $p(t)$ is its deterministic counterpart. $\left\|M^N(t+1) - m(t+1)\right\|$ can be decomposed in:

$$\begin{aligned}\left\|M^N(t{+}1) - m(t{+}1)\right\| &\leq \left\|M^N(t{+}1) - M^N(t)P^N(t)\right\| \\ &\quad + \left\|\left(M^N(t) - m(t)\right)P^N(t)\right\| \\ &\quad + \left\|m(t)\left(P^N(t) - p(t)\right)\right\|\end{aligned}$$

The central limit theorem shows that $\sqrt{N}(M^N(t+1) - M^N(t)P^N(t))$ converges in law to a Gaussian vector of mean 0 and covariance D where D is defined in Equation (5.15). Moreover, by Lemma 5.18, there exists a constant $\alpha_2 > 0$ such that

$$\begin{aligned}\sqrt{N}\mathbb{E}\left(\left\|M^N(t{+}1) - M^N(t)P^N(t)\right\|\right) &\leq \sum_i^S \sqrt{N}\mathbb{E}\left(\left|M^N(t{+}1)_i - (M^N(t)P^N(t))_i\right|\right) \\ &\quad + \alpha_2\sqrt{\frac{\log(N)}{N}} \\ &= \sum_{i=1}^S \sqrt{m_i p_{ij}(1-p_{ij})} + \alpha_2\sqrt{\frac{\log(N)}{N}} \\ &\leq \frac{S}{2} + \alpha_2\sqrt{\frac{\log(N)}{N}}. \qquad (5.14)\end{aligned}$$

The other terms can be bounded using similar ideas as for proving Equation (5.13) in the proof of Theorem 5.1. Since $p_{ij} \leq 1$ and $\sum_{i=1}^S m_i(t) = 1$, we have for all $1 \leq j \leq S$:

$$\left|\left(M^N(t) - m(t)\right)P^N(t)\right|_j \leq \sum_{i=1}^S \left|M^N(t) - m(t)\right|_i P_{ij}^N(t) \leq S\left\|M^N(t) - m(t)\right\|; \quad (5.15)$$

$$\left|m(t)\left(P^N(t) - p(t)\right)\right|_j \leq \sum_{j=1}^S m_j(t)\left|P_{ij}^N(t) - p_{ij}(t)\right| \leq L_K\left\|C^N(t) - c(t)\right\|. \quad (5.16)$$

Combining Equations (5.14), (5.15) and (5.16), we get

$$\sqrt{N}\mathbb{E}\left(\left\|M^N(t+1)-m(t+1)\right\|\right) \leq \frac{S}{2} + (S+L_K)\left\|(M^N(t),C^N(t))-(m(t),c(t))\right\| + \alpha_2\sqrt{\frac{\log(N)}{N}}.$$

Since $C^N(t+1) = g(C^N(t), M^N(t), a)$ where $(c,m) \mapsto g(c,m,a)$ is a deterministic Lipschitz function with constant L_g, we have:

$$\begin{aligned}\left\|C^N(t+1)-c(t+1)\right\| &\leq L_g \max\left(\left\|M^N(t+1)-m(t+1)\right\|, \left\|C^N(t)-c(t)\right\|\right)\\ &\leq L_g\left(\left\|M^N(t+1)-m(t+1)\right\| + \left\|C^N(t)-c(t)\right\|\right)\\ &\leq L_g\Bigg(\frac{S}{2} + (S+L_K+1)\left\|\left(M^N(t),C^N(t)\right)-(m(t),c(t))\right\| \\ &\qquad +\alpha_2\sqrt{\frac{\log(N)}{N}}\Bigg).\end{aligned}$$

Using the induction hypothesis, this implies that

$$\begin{aligned}&\sqrt{N}\mathbb{E}\left(\left\|\left(M^N(t+1),C^N(t+1)\right)-(m(t+1),c(t+1))\right\|\right)\\ &\qquad \leq \max\{1,L_g\}\left(\frac{S}{2} + (S+L_K+1)\left(\beta_t+\beta'_t\epsilon_0^N+e_t^N\right)+\alpha_2\sqrt{\frac{\log(N)}{N}}\right).\end{aligned}$$

5.8.3. Proof of Theorem 5.8 (mean field central limit theorem)

We first start by a technical lemma.

Lemma 5.15. *Let* M^N *be a sequence of random measures on* $\{1,\dots,S\}$ *and* P^N *a sequence of random stochastic matrices on* $\{1,\dots,S\}$ *such that* $(\mathrm{M}^N,P^N) \xrightarrow{\text{a.s.}} (m,p)$. *Let* $(U_{ik})_{1\leq i\leq S, k\geq 1}$ *be a collection of* i.i.d. *random variables following the uniform distribution on* $[0;1]$ *and independent of* P^N *and* M^N *and let us define* Y^N *such that for all* $1\leq j\leq S$,

$$Y_j^N \stackrel{\text{def}}{=} \frac{1}{N}\sum_{i=1}^{S}\sum_{k=1}^{N\mathrm{M}_i^N} \mathbf{1}_{\sum_{l<j}P_{il}^N < U_{ik} \leq \sum_{l\leq j} P_{il}^N}.$$

Then there exists a Gaussian vector G *independent of* M^N *and* P^N *and a random variable* Z^N *with the same law as* Y^N *such that*

$$\sqrt{N}(Z^N - \mathrm{M}^N P^N) \xrightarrow{\text{a.s.}} G.$$

Moreover the covariance of the vector G *is the matrix* D:

$$\left\{\begin{array}{lcl} D_{jj} &=& \sum_i m_i p_{ij}(1-p_{ij}) \\ D_{jk} &=& -\sum_i m_i p_{ij} p_{ik} \qquad (j\neq k).\end{array}\right. \tag{5.17}$$

Proof. As (M^N,P^N) and $(U_{ik})_{1\leq i\leq S,k\geq 1}$ are independent, they can be viewed as functions on independent probability space Ω and Ω'. For all $(\omega,\omega')\in\Omega\times\Omega'$, we define $X_\omega^N(\omega') \stackrel{\text{def}}{=} \sqrt{N}(Y^N(\omega,\omega') - \mathrm{M}^N(\omega)P^N(\omega))$.

By assumption, for almost all $\omega \in \Omega$, $(\mathrm{M}^N(\omega), P^N(\omega))$ converges to (m, p). A direct computation shows that, when N grows, the characteristic function of X_ω^N converges to $\exp(-\frac{1}{2}\xi^T D\xi)$. Therefore for almost all ω, X_ω^N converges in law to G, a Gaussian random variable on Ω'.

Therefore for almost all ω, there exists a random variable $\overline{X}_\omega^N$ with the same law as X_ω^N that converges ω'-almost surely to $G(\omega')$ (see [59] for details on that representation). Let $Z^N(\omega, \omega') \stackrel{\text{def}}{=} \mathrm{M}^N(\omega)P^N(\omega) + \frac{1}{N}\overline{X}_\omega^N(\omega')$. By construction of $\overline{X}_\omega^N$, for almost all ω, $Z^N(\omega, .)$ has the same distribution as $Y^N(\omega)$ and $\sqrt{N}(Z^N - Y^N P^N) \xrightarrow{\omega,\omega'-a.s} G$. Thus there exists a function $\overline{Z}^N(\omega, .)$ that has the same distribution as $Y^N(\omega)$ for all ω and that converges (ω, ω')-almost surely to G. □

We are now ready for the proof of Theorem 5.8.

Proof. Let us assume that the Equation (5.6) holds for some $t \geq 0$.

As $\sqrt{N}((\mathrm{M}^N, \mathrm{C}^N)(t) - (m, c)(t))$ converges in law to G_t, there exists another probability space and random variables $\overline{\mathrm{M}}^N$ and $\overline{\mathrm{C}}^N$ with the same distribution as M^N and C^N such that $\sqrt{N}((\overline{\mathrm{M}}^N, \overline{\mathrm{C}}^N)(t) - (m, c)(t))$ converges almost surely to G_t [59]. In the rest of the proof, by abuse of notation, we will write M and C instead of $\overline{\mathrm{M}}$ and $\overline{\mathrm{C}}$ and then we assume that $\sqrt{N}((\mathrm{M}^N(t), \mathrm{C}^N(t)) - (m, c)(t)) \xrightarrow{\text{a.s.}} G_t$.

G_t being a Gaussian vector, there exists a vector of $S+d$ independent Gaussian variables $U = (u_1, \ldots, u_{S+d})^T$ and a matrix Δ of size $(S+d)\times(S+d)$ such that $G_t = \Delta U$.

Let us call $P_t^N \stackrel{\text{def}}{=} K(a_t, \mathrm{C}^N(t))$. According to lemma 5.15 there exists a Gaussian variable H_t independent of G_t and of covariance D such that we can replace $\mathrm{M}^N(t+1)$ (without changing $\mathrm{M}^N(t)$ and $\mathrm{C}^N(t)$) by a random variable $\overline{\mathrm{M}}^N(t+1)$ with the same law such that:

$$\sqrt{N}(\overline{\mathrm{M}}(t+1)^N - \mathrm{M}(t)^N P_t^N) \xrightarrow{\text{a.s.}} H_t. \tag{5.18}$$

In the following, by abuse of notation we will also write M instead of $\overline{\mathrm{M}}$. Therefore we have

$$\begin{aligned}\sqrt{N}(\mathrm{M}^N(t+1) - m(t)P_t) &= \sqrt{N}\Big(\mathrm{M}(t+1) - \mathrm{M}^N(t)P_t^N + m(t)(P_t^N - P_t) + \\ &\qquad (\mathrm{M}^N(t) - m(t))P_t + (\mathrm{M}^N(t) - m(t))(P_t^N - P_t)\Big) \\ &\xrightarrow{\text{a.s.}} H_t + m(t)\lim_{N\to\infty}\sqrt{N}(P_t^N - P_t) + \lim_{N\to\infty}\sqrt{N}(\mathrm{M}^N(t) - m(t))P_t.\end{aligned}$$

By assumption, $\lim \sqrt{N}(\mathrm{M}^N(t) - m(t))_i = (\Delta U)_i$. Moreover, the first order Taylor expansion with respect to all component of C gives a.s.

$$\begin{aligned}\lim_{N\to\infty} m(t)\sqrt{N}(P_t^N - P_t)_j &= \sum_{i=1}^{S} m_i(t)\sum_{k=1}^{d}\frac{\partial K_{ij}}{\partial c_{t_k}}(a_t, c(t))(XU)_{S+k} \\ &= \sum_{k=1}^{d} Q_{kj}(\Delta U)_{S+k}.\end{aligned}$$

Thus, the jth component of $\sqrt{N}(\mathrm{M}^N(t+1) - m(t)P_t)$ goes to

$$H_t + \sum_{k=1}^{d} Q_{kj}(\Delta U)_{S+k} + \sum_{i=1}^{S}(\Delta U)_i P_{ij} \tag{5.19}$$

Using similar ideas, we can prove that $\sqrt{N}(\mathrm{C}_k^N(t+1) - c_k(t+1))$ converges almost surely to $\sum_{i=0}^{S} \frac{\partial g_k}{\partial m_i}(\Delta U)_i + \sum_{\ell=0}^{d} \frac{\partial g_k}{\partial c_{t_\ell}}(\Delta U)_{S+\ell}$. Thus $\sqrt{N}((\mathrm{M}^N(t+1), \mathrm{C}^N(t+1)) - (m(t+1), c(t+1)))$ converges almost surely to a Gaussian vector.

Let us write the covariance matrix at time t and time $t+1$ as two bloc matrices:

$$\Gamma_t = \left[\begin{array}{c|c} \Delta & O \\ \hline O^T & \mathrm{C} \end{array}\right] \text{ and } \Gamma_{t+1} = \left[\begin{array}{c|c} \Delta' & O' \\ \hline O'^T & \mathrm{C}' \end{array}\right].$$

For $1 \leq j, j' \leq S$, $\Delta'_{j,j'}$ is the expectation of (5.19) taken in j times (5.19) taken in j'. Using the facts that
$\mathbb{E}\left((\Delta U)_{S+k}(\Delta U)_{S+k'}\right) = \mathrm{C}_{kk'}$, $\mathbb{E}\left((\Delta U)_{S+k}(\Delta U)_i\right) = O_{ik}$ and $\mathbb{E}\left((\Delta U)_i(\Delta U)_{i'}\right) = \Delta_{ii'}$, this leads to:

$$\begin{aligned}
\Delta'_{j,j'} &= \mathbb{E}\left(H_j H'_j\right) + \sum_{k,k'} Q_{kj} Q_{k'j'} \mathrm{C}_{kk'} + \sum_{k,i'} Q_{kj} O_{i'k} P_{i'j'} \\
&\quad + \sum_{i,k'} Q_{k'j'} O_{ik'} P_{ij} + \sum_{i,i'} P_{ij} \Delta_{ii'} P_{i'j} \\
&= D_{jj'} + (Q^T \mathrm{C} Q)_{jj'} + (Q^T O^T P)_{jj'} + (P^T O Q)_{jj'} + (P^T \Delta P)_{jj'}.
\end{aligned}$$

By similar computation, we can write similar equations for O' and C' that lead to Equation (5.7). □

5.8.4. Proof of Theorem 5.9 (third order results)

In order to prove Theorem 5.9, we start with a result on the sharpness of the approximation of the sum of Bernoulli random variable by a Gaussian distribution (Lemma 5.16).

Let B_i be independent Bernoulli random variables (*i.e* $P(B_i = 1) = 1 - P(B_i = 0) = p$) and let $Y^N = \frac{1}{N}\sum_{i=1}^{N} B_i$. We know that in a sense, Y^N is close to $Z^N = p + \frac{1}{\sqrt{N}} G$ with G a normal random variable of variance $\sigma^2 = \mathbb{E}\left((X_0 - p)^2\right)$. We want to compute an asymptotic development of the quantity:

$$d_N = \left|\mathbb{E}\left(f\left(Y^N\right)\right) - \mathbb{E}\left(f\left(Z^N\right)\right)\right|$$

where f is a Lipschitz function of Lipschitz constant L. The quantity d_N is called the Wasserstein distance between Y^N and Z^N.

Let $F_{N,p} : \mathbb{R} \to [0:1]$ and F_p be respectively the CDF (Cumulative Distribution Function) of $\sqrt{N}(Y^N - p)$ and of the standard normal distribution: $F_{N,p}(x) = P(\sqrt{N}(Y^N - p) \leq x)$ and $F_p(x) = P(G \leq x)$ where G is a normal variable of mean 0 and variance $\sigma^2 = p(1-p) = \mathbb{E}\left((X_0 - p)^2\right)$. Let U be a random variable uniformly distributed in $[0;1]$. d_N can be rewritten as:

$$d_N = \left|\mathbb{E}\left(f(x + \frac{\sigma}{\sqrt{N}} F_{N,p}^{-1}(U)) - f(x + \frac{\sigma}{\sqrt{N}} F_p^{-1}(U))\right)\right| \leq \frac{L\sigma}{\sqrt{N}} \mathbb{E}\left(|F_{N,p}^{-1}(U) - F_p^{-1}(U)|\right)$$

where $F_{N,p}^{-1}(U) \stackrel{\text{def}}{=} \min\{y : F_{N,p}(y) \geq U\}$.

Therefore, the problem becomes to get an estimation of $\mathbb{E}\left(|F_{N,p}^{-1}(U) - F_p^{-1}(U)|\right)$.

Lemma 5.16. *There exists a constant α_1 independent of N, L, p such that if U is a random variable uniformly distributed on $[0;1]$ and $F_{N,p}$ and F be the cumulative distribution functions defined previously, then for N big enough,*

$$\mathbb{E}\left(|F_{N,p}^{-1}(U) - F_p^{-1}(U)|\right) \leq \alpha_1 \sqrt{\frac{\log(N)}{N}}, \tag{5.20}$$

where $\alpha_1 < 356$.

Proof. For more simplicity, in this proof, we omit the index p when writing $F_{N,p}$ and F_p.

$$\mathbb{E}\left(|F_N^{-1}(U) - F^{-1}(U)|\right) = \int_0^1 |F_N^{-1}(u) - F^{-1}(u)| du \tag{5.21}$$

The Berry-Essen theorem (see for example part 2.4.d of [59]) shows that $\sup_{y\in\mathbb{R}} |F_N(\sigma y) - F(\sigma y)| \leq \frac{3\rho}{\sigma^3\sqrt{N}}$ where $\rho = \mathbb{E}\left(|X_0 - p|^3\right)$. As F and F_N are increasing, for all $u \in (\frac{3\rho}{\sigma^2\sqrt{N}}; 1 - \frac{3\rho}{\sigma^2\sqrt{N}})$, we have:

$$F^{-1}(u - \frac{3\rho}{\sigma^2\sqrt{N}}) \leq F_N^{-1}(u) \leq F^{-1}(u + \frac{3\rho}{\sigma^2\sqrt{N}}).$$

Using these remarks, the symmetry of the function F and the fact that $F^{-1}(u+\epsilon) - F^{-1}(u) \geq F^{-1}(u) - F^{-1}(u-\epsilon)$ for $u > 1/2$, (5.21) is less than

$$2\left(\int_{\frac{1}{2}}^{k_N} \left(F^{-1}(u + \frac{3\rho}{\sigma^2\sqrt{N}}) - F^{-1}(u)\right) du + \int_{k_N}^1 |F_N^{-1}(u) - F^{-1}(u)| du\right) \tag{5.22}$$

for all constant $k_N \in (\frac{1}{2}; 1 - \frac{3\rho}{\sigma^2\sqrt{N}})$.

The function F^{-1} is continuously differentiable, therefore the mean value theorem says that there exists $v(u) \in \left(u; u + \frac{3\rho}{\sigma^2\sqrt{N}}\right)$ such that

$$F^{-1}(u + \frac{3\rho}{\sigma^2\sqrt{N}}) - F^{-1}(u) = \frac{3\rho}{\sigma^2\sqrt{N}}(F^{-1})'(v(u)) \leq \frac{3\rho}{\sigma^2\sqrt{N}}(F^{-1})'(u + \frac{3\rho}{\sigma^2\sqrt{N}}).$$

Thus, the first part of inequality (5.22) is bounded by

$$\begin{aligned}
\int_{\frac{1}{2}}^{k_N} \left(F^{-1}(u + \frac{3\rho}{\sigma^2\sqrt{N}}) - F^{-1}(u)\right) du &\leq \frac{3\rho}{\sigma^2\sqrt{N}} \int_{\frac{1}{2}}^{k_N} (F^{-1})'(u + \frac{3\rho}{\sigma^2\sqrt{N}}) du \\
&= \frac{3\rho}{\sigma^2\sqrt{N}} \left(F^{-1}(k_N + \frac{3\rho}{\sigma^2\sqrt{N}}) - F^{-1}(\frac{1}{2})\right) \\
&\leq \frac{3\rho}{\sigma^2\sqrt{N}} F^{-1}(k_N + \frac{3\rho}{\sigma^2\sqrt{N}}).
\end{aligned} \tag{5.23}$$

Using an integration by substitution with $x = F^{-1}(u)$ (and $F'(x)dx = du$) and an integration by part, we get:

$$\begin{aligned}
\int_{k_N}^1 F^{-1}(u) du &= \int_{F^{-1}(k_N)}^{\infty} x F'(x) dx \\
&= \left[x\left(F^{-1}(x) - 1\right)\right]_{F^{-1}(k_N)}^{\infty} - \int_{F^{-1}(k_N)}^{\infty} (F(x) - 1)\, dx \\
&= (1 - k_N)\, F^{-1}(k_N) + \int_{F^{-1}(k_N)}^{\infty} (1 - F(x))\, dx
\end{aligned} \tag{5.24}$$

For $x \geq 1$, the tail of the distribution of a Gaussian variable satisfies:

$$\frac{1}{2x\sqrt{2\pi}}\exp(-\frac{x^2}{2}) \leq \frac{x}{(1+x^2)\sqrt{2\pi}}\exp(-\frac{x^2}{2}) \leq 1-F(x) \leq \frac{1}{x\sqrt{2\pi}}\exp(-\frac{x^2}{2}) \leq \exp(-\frac{x^2}{2}).$$

Moreover, for all $u \geq x \geq 1$ we have $u^2/2 - u \geq x^2/2 - x$ and $\int_x^\infty \exp(-u^2/2)du \leq \int_x^\infty \exp(-x^2/2 + x - u)du = \exp(-x^2/2)$. This leads to

$$\int_{F^{-1}(k_N)}^{\infty} 1 - F(x)dx \leq \exp(-\frac{F^{-1}(k_N)}{2}) \leq 2\sqrt{2\pi}F^{-1}(k_N)(1-k_N) \tag{5.25}$$

for k_N such that $F_N^{-1}(k_N) \geq 1$.

Similarly, $\int_{F^{-1}(k_N)}^{\infty} 1 - F_N(x)dx$ can be bounded by the same method using Hoeffding's inequality $F_N(x) \leq \exp(-2x^2)$ and we get:

$$\int_{k_N}^{1} F_N^{-1}(u)du \quad \leq \quad (1-k_N)F_N^{-1}(k_N) + \exp(-2F_N^{-1}(k_N)) \tag{5.26}$$

with

$$\begin{aligned}\exp\left(-2F_N^{-1}(k_N)\right) &\leq \exp\left(-2F^{-1}(k_N - \frac{3\rho}{\sigma^2\sqrt{N}})\right)\\ &\leq 2\sqrt{2\pi}F^{-1}(k_N - \frac{3\rho}{\sigma^2\sqrt{N}})(1 - k_N + \frac{3\rho}{\sigma^2\sqrt{N}})\end{aligned}$$

Combining (5.22), (5.23), (5.24), (5.25) and (5.26), (5.21) is less than:

$$\begin{aligned}(5.21) &\leq (5.22)\\ &\leq 2\left[(5.23) + (5.24) + (5.26)\right]\\ &\leq 2\frac{3\rho}{\sigma^2\sqrt{N}}F^{-1}\left(k_N + \frac{3\rho}{\sigma^2\sqrt{N}}\right) + 2(1-k_N)(F_N^{-1}(k_N) + (1+2\sqrt{2\pi})F^{-1}(k_N))\\ &\quad +4\sqrt{2\pi}F^{-1}(k_N - \frac{3\rho}{\sigma^2\sqrt{N}})(1 - k_N + \frac{3\rho}{\sigma^2\sqrt{N}})\end{aligned} \tag{5.27}$$

Let $k_N \stackrel{\text{def}}{=} 1 - 2\frac{3\rho}{\sigma^2\sqrt{N}}$. Since ρ is the third moment of a Bernoulli variable of mean p and σ^2 its variance, we have $\rho/\sigma^2 = p^2 + (1-p)^2 \in [.5; 1]$. Moreover, the functions F^{-1} and F_N^{-1} are increasing. Using these facts, Equation (5.27) becomes

$$\begin{aligned}(5.21) &\leq F^{-1}\left(1-\frac{3\rho}{\sigma^2\sqrt{N}}\right)\left(2\frac{3\rho}{\sigma^2\sqrt{N}}\right)\left(2+4+8\sqrt{2\pi}+12\sqrt{2\pi}\right)\\ &\quad +4\frac{3\rho}{\sigma^2\sqrt{N}}F_N^{-1}(1-2\frac{3\rho}{\sigma^2\sqrt{N}}),\\ &\leq \frac{350}{\sqrt{N}}F^{-1}(1-\frac{3}{2\sqrt{N}}) + \frac{12}{\sqrt{N}}F_N^{-1}(1-\frac{3}{\sqrt{N}}),\end{aligned} \tag{5.28}$$

where we used the fact that $2\sqrt{2\pi} < 6$ and $6\sqrt{2\pi} < 16$.

Hoeffding's inequality $1 - F_N(x) \leq \exp(-2x^2)$ shows that $F_N^{-1}(y) \leq \sqrt{-\log(1-y)/2}$. Applying this formula to $1 - 3/\sqrt{N}$ leads to:

$$F_N^{-1}(1-\frac{3}{\sqrt{N}}) \leq \sqrt{-\log(\frac{3}{\sqrt{N}})/2} = \sqrt{\frac{\log(N)}{4} + \frac{\log(1/3)}{2}} \leq \frac{\sqrt{\log(N)}}{2}.$$

Similar inequality for the tail of the normal distribution leads to $F^{-1}(y) \leq \sqrt{-2\log(1-y)}$ and $F^{-1}(1-3/(2\sqrt{N})) \leq \sqrt{\log(N)}$.

This shows that:

$$(5.21) \leq 356\frac{\sqrt{\log(N)}}{\sqrt{N}}.$$

This bound could be improved, in particular by a more precise analysis of (5.28), but fulfills for our needs.

□

In the case where we sum only a fraction δ_N of the N Bernoulli variables, the result still holds.

Lemma 5.17. *Let $0 \leq \delta_N \leq 1$ be a random variable and $b \in [0;1]$ such that $E\left|\delta_N - b\right| \leq \alpha'\sqrt{\log(N)}/N$ and U a random variable uniformly distributed on $[0;1]$ independent of δ_N. Then:*

$$\mathbb{E}\left(\left|\sqrt{\delta_N}F^{-1}_{N\delta_N,p}(U) - \sqrt{b}F^{-1}_p(U)\right|\right) \leq \alpha_2\sqrt{\frac{\log(N)}{N}}$$

where $\alpha_2 = \alpha_1 + \max(\alpha', \sqrt{\alpha'} + 2)$

Proof. Again, to ease the notations, we omit to write p in $F_{N,p}$ and F_p.

$$\begin{aligned}
\mathbb{E}\left(\left|\sqrt{\delta_N}F^{-1}_{N\delta_N}(U) - \sqrt{b}F^{-1}(U)\right|\right) &\leq \mathbb{E}\left(\left|\sqrt{\delta_N}F^{-1}_{N\delta_N}(U) - \sqrt{\delta_N}F^{-1}(U)\right|\right) \\
&\quad + \mathbb{E}\left(\left|\left(\sqrt{\delta_N} - \sqrt{b}\right)F^{-1}(U)\right|\right) \\
&\leq \alpha_1\sqrt{\delta_N}\sqrt{\frac{\log(N\delta_N)}{N\delta_N}} + \mathbb{E}\left(\left|\sqrt{\delta_N} - \sqrt{b}\right|\right)\mathbb{E}\left(G\right)
\end{aligned}$$

The first part of the inequality comes from the Lemma 5.16 and is less than $\alpha_1\frac{\log(N)}{N}$ since $\delta_N \leq 1$. The last part comes from the fact that U and δ_N are independent. Moreover, the variance of G is the variance of a Bernoulli variable, so $\mathbb{E}\left(|G|\right) \leq 1/4$.

To bound $\mathbb{E}\left(\left|\sqrt{\delta_N} - \sqrt{b}\right|\right)$, we distinguish two cases. If $\sqrt{b} \geq 1/\sqrt{N}$, we have:

$$\begin{aligned}
\mathbb{E}\left(\left|\sqrt{\delta_N} - \sqrt{b}\right|\right) &\leq \mathbb{E}\left(\frac{\left|\delta_N - \sqrt{b}\right|}{\sqrt{\delta_N} + \sqrt{b}}\right) \leq \mathbb{E}\left(\frac{\left|\delta_N - \sqrt{b}\right|}{\sqrt{b}}\right) \\
&\leq \sqrt{N}\mathbb{E}\left(\left|\delta_N - \sqrt{b}\right|\right) \leq C'\sqrt{\frac{\log(N)}{N}}.
\end{aligned}$$

If $\sqrt{b} \leq 1/\sqrt{N}$, we have:

$$\mathbb{E}\left(\left|\sqrt{\delta_N} - \sqrt{b}\right|\right) \leq \mathbb{E}\left(\sqrt{\delta_N}\right) + \sqrt{b} \leq \mathbb{E}\left(\sqrt{|\delta_N - b|}\right) + 2\sqrt{b} \leq \sqrt{C'\frac{\sqrt{\log(N)}}{N}} + \frac{2}{\sqrt{N}}.$$

This shows the inequality. □

The following lemma uses the previous results in the case of multidimensional Bernoulli variables. A multidimensional Bernoulli variable B of parameter $(p_1 \ldots p_S)$ is a unit vector $\mathbf{e}_j$ on $\mathbb{R}^S$ (its j-th component equals 1 and all others equal 0). The probability that $B = \mathbf{e}_j$ is p_j.

Lemma 5.18. *Let $m \in \mathscr{P}_N(\mathcal{S})$ and let $(B_k^i)_{1\le i\le S, k\ge 1}$ be independent multidimensional Bernoulli random vectors on $\{0;1\}^S$ such that $\mathcal{P}\left(B_k^i = \mathbf{e}_j\right) = p_{ij}$. Let f be a Lipschitz function on $\mathbb{R}^S$ with constant L. Let us define $Y^N \stackrel{\text{def}}{=} \frac{1}{N}\sum_{i=1}^S \sum_{k=1}^{m_i N} B_k^i$. Then there exists a constant α_3 such that:*

$$\left|\mathbb{E}\left(f(Y^N)\right) - \mathbb{E}\left(f(mP + \frac{1}{\sqrt{N}}G)\right)\right| \le \alpha_3 L \frac{\sqrt{\log(N)}}{N},$$

where mP is the vector $(mP)_j = \sum_{i=1}^S m_i p_{ij}$ and G is a random Gaussian vector of mean 0 and covariance matrix D defined by Equation (5.15) ($D_{jj} = \sum_i m_i p_{ij}(1-p_{ij})$ and $D_{jk} = -\sum_i m_i p_{ij} p_{ik}$ for $j \ne k$).

Proof. Let $(b_k^{ij})_{i,j\le S, k\ge 1}$ be a collection of independent Bernoulli random variables of parameters $p_{ij}/(1-\sum_{\ell<j} p_{i\ell})$ (the parameter of b_k^{i1} is p_{i1}). The Bernoulli vector B_k^i is equal in law to $\bar{B}_k^i$ (denoted $B_k^i \stackrel{\mathcal{L}}{=} \bar{B}_k^i$) where

$$\bar{B}_k^i = b_k^{i1}\mathbf{e}_1 + (1-b_k^{i1})\left(b_k^{i2}\mathbf{e}_2 + (1-b_k^{i2})\left(b_k^{i3}\mathbf{e}_3 + (1-b_k^{i3})(\ldots)\right)\right).$$

Indeed, $\bar{B}_k^i$ is a vector with only one component equal to 1 and

$$\mathcal{P}\left(\bar{B}_k^i = \mathbf{e}_j\right) = (1-p_{i1})(1-\frac{p_{i2}}{1-p_{i1}})(1-\frac{p_{i3}}{1-p_{i1}-p_{i2}})\ldots\frac{p_{ij}}{1-\sum_{\ell<j} p_{i\ell}} = p_{ij}.$$

Let $T_{ij}^N \stackrel{\text{def}}{=} N^{-1}\sum_{k=1}^{m_i N} \mathbf{1}_{\{B_k^i = \mathbf{e}_j\}}$ be the proportion of objects going from state i to state j. By definition of b_k^{ij}, T_{ij}^N can be written

$$\begin{aligned}
T_{ij}^N &\stackrel{\mathcal{L}}{=} \frac{1}{N}\sum_{k=1}^{m_i N}(1-b_k^{i1})(1-b_k^{i2})\ldots(1-b_k^{i,j-1})b_k^{ij} \\
&\stackrel{\mathcal{L}}{=} \frac{1}{N}\sum_{1\le k\le m_i N \text{ s.t. } (b_k^{i1}=0)\wedge\ldots\wedge(b_k^{i,j-1}=0)} b_k^{ij} \\
&\stackrel{\mathcal{L}}{=} \frac{1}{N}\sum_{k=1}^{m_i N - \sum_{\ell<j} T_{i\ell}} b_k^{i\ell} \\
&\stackrel{\mathcal{L}}{=} \frac{p_{ij}}{1-\sum_{\ell<j} p_{i\ell}}\left(m_i - \sum_{\ell<j} T_{i\ell}^N\right) + \sqrt{\frac{m_i - \sum_{\ell<j} T_{i\ell}^N}{N}} F^{-1}_{m_i N - N\sum_{\ell<j} T_{i\ell}^N, \frac{p_{ij}}{1-\sum_{\ell<j} p_{i\ell}}}(U_{ij}).
\end{aligned}$$

where the function F is defined by $F_{A,q}(x) \stackrel{\text{def}}{=} \mathcal{P}\left(\frac{1}{\sqrt{A}}\left(\sum_{i=1}^A c_i - Aq\right) \le x\right)$ where $q \in [0;1]$, $A \in \mathbb{N}$ and c_i are scalar *i.i.d.* Bernoulli variables – *i.e.* $\mathcal{P}(c_i = 1) = 1 - \mathcal{P}(c_i = 0) = p$. By construction, $(\sum_{i=1}^S T_{ij}^N)_{1\le j\le S}$ has the same law as Y^N.

The variable H is constructed similarly. Denoting $F_p(x) \stackrel{\text{def}}{=} \mathcal{P}(G \ge x)$ where G is a normal variable of mean 0 and variance $p(1-p)$, we define the variables H_{ij}^N by:

$$\begin{aligned}
H_{i1} &\stackrel{\text{def}}{=} F^{-1}_{p_{i1}}(U_{i1}) \\
H_{ij} &\stackrel{\text{def}}{=} -\frac{p_{ij}}{1-\sum_{\ell<j} p_{i\ell}}\sum_{\ell<j} H_{i\ell} + \sqrt{\sum_{\ell<j} p_{i\ell}}\, F^{-1}_{\frac{p_{ij}}{1-\sum_{\ell<j} p_{i\ell}}}(U_{ij})
\end{aligned}$$

It is direct to prove that H has the same law as G by showing that H is a Gaussian vector and by computing the covariance matrix of H.

Using this representation of Y^N and G by T^N and H, we have:

$$\begin{aligned}\left|\mathbb{E}\left(f(Y^N)\right)-\mathbb{E}\left(mP+\frac{1}{\sqrt{N}}G\right)\right| &\leq \mathbb{E}\left(\left\|f(T^N)-f(mP+\frac{1}{\sqrt{N}}H)\right\|\right)\\ &\leq L\mathbb{E}\left(\left\|T^N-mP-\frac{1}{\sqrt{N}}H\right\|\right)\\ &\leq L\sum_{i=1}^{S}\sum_{j=1}^{S}\mathbb{E}\left(\left|T^N_{ij}-m_ip_{ij}-H_{ij}\right|\right)\end{aligned}$$

The next step of the proof is to bound $\mathbb{E}\left|T^N_{ij}-m_ip_{ij}-H_{ij}\right|$ which is less than:

$$\mathbb{E}\left|\sqrt{\frac{m_i-\sum_{\ell<j}T^N_{i\ell}}{N}}F^{-1}_{m_iN-N\sum_{\ell<j}T^N_{i\ell},\frac{p_{ij}}{1-\sum_{\ell<j}p_{i\ell}}}(U_{ij})-\sqrt{\sum_{\ell<j}p_{i\ell}}F^{-1}_{\frac{p_{ij}}{1-\sum_{\ell<j}p_{i\ell}}}(U_{ij})\right|$$
$$+\frac{p_{ij}}{1-\sum_{\ell<j}p_{i\ell}}\mathbb{E}\left|\sum_{\ell<j}(m_ip_{i\ell}-T^N_{i\ell}+H_{i\ell})\right|$$

where we used the fact that $\frac{p_{ij}}{1-\sum_{\ell<j}p_{i\ell}}m_i-m_ip_{ij}=\frac{p_{ij}}{1-\sum_{\ell<j}p_{i\ell}}\sum_{\ell<j}m_ip_{i\ell}$.

By induction on j and using Lemma 5.17, this quantity is less than $\alpha^{(j)}\frac{\sqrt{N}}{N}$ where the constant $\alpha^{(j)}\stackrel{\text{def}}{=}\alpha^{(j-1)}+\alpha_1+\max(\alpha^{(j-1)},\sqrt{\alpha^{(j-1)}}+2)$. The constant α_3 is equal to $S^2\alpha^{(S)}$. □

We are now ready for the proof of Theorem 5.9.

Proof of Theorem 5.9. Let a be a sequence of actions. We define by a backward induction on t the function $W^N_{t\ldots T}(.)$ that will be the expected reward of the mean field Gaussian approximation between t and T:

$$\begin{aligned}W^N_{T\ldots T,a}\left(M^N(t),C^N(t)\right) &= 0\\ W^N_{t\ldots T,a}\left(M^N(t),C^N(t)\right) &= \mathrm{r}\left(M^N(t),C^N(t)\right)+\mathbb{E}\left(W^N_{t\ldots T,a}\left(\widetilde{M}^N_a(t+1),\widetilde{C}^N_a(t+1)\right)\right)\end{aligned}\tag{5.29}$$

where $(\widetilde{M}^N_a(t+1),\widetilde{C}^N_a(t+1))$ is the mean field Gaussian approximation starting at time t in from $(M^N(t),C^N(t))$ and after one time step during which the controller took action a_t. Similarly, we define $V^N_{t\ldots T,a}\left(M^N(t),C^N(t)\right)$ the expected reward between t and T for the original system. We want to prove by induction that there exist constants γ_t such that for any t:

$$\left|V^N_{t\ldots T}(M^N,C^N)-W^N_{t\ldots T}(M^N,C^N)\right|\leq\gamma_t\frac{\sqrt{\log(N)}}{N}.\tag{5.30}$$

The constant γ_t may depend on the parameters of the system (such as the Lipschitz constants of the functions g,K,r) but not on the value of (M^N,C^N).

Equation (5.30) is clearly true for $t=T$ by taking $\gamma_T=0$. Let us now assume that (5.30) holds for some $t+1\leq T$. By a backward induction on t, one can show that

$W^N_{t\ldots T,a}(.,.)$ is Lipschitz for some constant L_{W_t}. $\left|V^N_{t\ldots T}(M^N,C^N) - W^N_{t\ldots T}(M^N,C^N)\right|$ can be written:

$$\begin{aligned}
&\left|\mathbb{E}\left(V^N_{t+1\ldots T,a}\left(M^N_a(t+1),C^N_a(t+1)\right)\right) - \mathbb{E}\left(W^N_{t+1\ldots T,a}\left(\widetilde{M}^N_a(t+1),\widetilde{C}^N_a(t+1)\right)\right)\right| \\
&\quad\leq \left|\mathbb{E}\left(W^N_{t+1\ldots T,a}\left(M^N_a(t+1),C^N_a(t+1)\right) - W^N_{t+1\ldots T,a}\left(\widetilde{M}^N_a(t+1),\widetilde{C}^N_a(t+1)\right)\right)\right| \\
&\qquad + \left|\mathbb{E}\left(W^N_{t+1\ldots T,a}\left(M^N_a(t+1),C^N_a(t+1)\right) - V^N_{t+1\ldots T,a}\left(M^N_a(t+1),C^N_a(t+1)\right)\right)\right| \\
&\quad\leq \alpha_3 L_{W_t} L_g \frac{\sqrt{\log(N)}}{N} + \gamma_t \frac{\sqrt{\log(N)}}{N}.
\end{aligned} \tag{5.31}$$

The fist part of the second inequality comes from Lemma 5.18 applied to the function $m \mapsto W^N_{t+1\ldots T,a}\left(m, g(a, C^N(t), m)\right)$ that is Lipschitz of constant $L_{W_t}L_g$. The second part comes from the hypothesis of induction. This concludes the proof in the case of a fixed sequence of actions.

The proof for $V^N_* - W^N_*$ is very similar. The first step of the proof is to write a similar equation as (5.29) for W^N_* and V^N_* which can be computed by a backward induction:

$$\begin{aligned}
W^N_{T\ldots T,*}\left(M^N(t),C^N(t)\right) &= 0 \\
W^N_{t\ldots T,*}\left(M^N(t),C^N(t)\right) &= \mathrm{r}\left(M^N(t),C^N(t)\right) \\
&\quad + \sup_a \mathbb{E}\left(W^N_{t\ldots T,a}\left(\widetilde{M}^N_a(t+1),\widetilde{C}^N_a(t+1)\right)\right).
\end{aligned}$$

We then get equations similar to (5.31) but with a $\sup_a$ before both expectation. The sup can be removed using the fact that for any functions f and g: $|\sup_a f(a) - \sup_a g(a)| \leq \sup_a |f(a) - g(a)|$. □

Chapter 6.

From Discrete to Continuous Optimization

Abstract of this chapter – In this chapter, we study the convergence of Markov Decision Processes made of a large number of objects to optimization problems on ordinary differential equations (ODE).

We show that the optimal reward of such a Markov Decision Process, satisfying a Bellman equation, converges to the solution of a continuous Hamilton-Jacobi-Bellman (HJB) equation based on the mean field approximation of the Markov Decision Process. We give bounds on the difference of the rewards, and a constructive algorithm for deriving an approximating solution to the Markov Decision Process from a solution of the HJB equations.

We illustrate the method on three examples pertaining respectively to investment strategies, population dynamics control and scheduling in queues. They are used to illustrate and justify the construction of the controlled ODE and to show the gain obtained by solving a continuous HJB equation rather than a large discrete Bellman equation.

Résumé du chapitre – Ce chapitre étudie la convergence de processus de décision markoviens composés d'un grand nombre d'objets vers des problèmes d'optimisation sur des équations différentielles. Nous montrons que le gain optimal du processus de décision converge vers la solution d'une équation continue de type "Hamilton-Jacobi-Bellman". La preuve utilise à la fois des outils classiques des modèles champs moyens et différents nouveaux couplages entre les modèles discrets et continus qui permettent de donner des bornes explicites. La méthode est ensuite illustrée par trois exemples concernant des stratégies d'investissement, du contrôle de dynamiques de population et un problème d'allocation de ressources.

6.1. Introduction

The purpose of this chapter is to study Markov decision process composed of a large number N of interacting objects.

In Chapter 5, we considered a case where the probability than an object changes its state between two time steps (called *the intensity* and denoted $I(N)$) does not scale with N, *i.e.* $I(N) = O_{N\to\infty}(1)$. In that case, the optimization problem of the system of size N converges to a deterministic optimization problem in discrete time. Solving the deterministic system allows one to compute policies that are asymptotically optimal as the number of objects grows.

In this chapter, we focus on the case where the intensity vanishes as N grows, *i.e.* $\lim_{N\to\infty} I(N) = 0$, which is substantially different from the case studied in the previous chapter. In Chapter 2, we have seen that if $\lim_{N\to\infty} I(N) = 0$, then under mild conditions the system without control converges to a continuous time dynamical system and can be described by ordinary differential equations (ODE). In this chapter, we show that when N goes to infinity, the optimization problem converges to an optimization problem on an ordinary differential equation.

More precisely, we show that when the Markov decision process is such that its empirical density measure is Markovian, then its optimal reward converges to the optimal reward of its mean field approximation, given by the solution of an HJB equation. Furthermore, the optimal policy of the limit continuous system is also asymptotically optimal for the original discrete system. Our method relies on bounding techniques used in stochastic approximation and learning [24, 17]. We also introduce an original coupling method, where, to each sample path of the Markov decision process, we associate a random trajectory, obtained as a solution of the ODE, i.e. the mean field limit, controlled by random actions.

This convergence result has an algorithmic counterpart. Basically, when confronted with a large Markov Decision problem, one can first solve the HJB equation for the associated mean field limit and then build a decision policy for the initial system that is asymptotically optimal.

Few papers in the literature are concerned with the problem of mixing the limiting behavior of a large number of objects with optimization. In [52], the value function of the Markov decision process is approximated by a linearly parametrized class of functions, and a fluid approximation of the MDP is used. It is shown that a solution of the HJB equation is a value function for a modification of the original MDP problem. In [145, 57], the curse of dimensionality of dynamic programming is circumvented by approximating the value function by linear regression.

Our results have two main implications. The first one is to justify the construction of controlled ODEs as good approximations of large discrete controlled systems. This construction is often done without rigorous proofs. In Section 6.4.3 we illustrate this point on an example in the field of computer system infection by malware.

The second implication concerns the effective computation of an optimal control policy. In the discrete case, this is usually done by using dynamic programming for the finite horizon case or by computing a fixed point of the Bellman equation in the discounted case. Both approaches suffer from the curse of dimensionality that makes them impractical when the state space is too large. In our context, the size of the state space is exponential in N, making the problem even more acute. In practice, modern supercomputers only allow one to tackle optimal control problems where N is no larger than a few tens.

The mean field approach offers an alternative to brute force computations. By letting N go to infinity, the discrete problem is replaced by a limit Hamilton-Jacobi-Bellman equation that is deterministic and where the dimensionality of the original system has been hidden in the density measure. Solving the HJB equation numerically is sometimes rather easy, as in the two first examples of Section 6.4.3. It provides a deterministic optimal policy whose reward with a finite (but large) number of objects is remarkably close to the optimal reward.

In this chapter we focus on the finite horizon case, though the technique applies mutatis mutandis to infinite horizon with discount.

The rest of the chapter is structured as follows. Section 6.2 contains definitions, some notation, and hypotheses. Section 6.3 gives the theoretical results and the resulting algorithms. Section 6.4 illustrates the application of our method on a few examples. Section 6.5 contains proofs.

6.2. Notations and Definitions

6.2.1. System with N Objects

We consider a system composed by N *objects* similar to the one of Section 2.2.2. Each object has a state in the finite set $\mathcal{S} = \{1 \dots S\}$. Time is discrete and the state of the object n at step $k \in \mathbb{N}$ is denoted $X_n^N(k)$. The state of the system at time k is $X^N(k) \stackrel{\text{def}}{=} \left(X_1^N(k) \dots X_N^N(k)\right)$. For all $i \in \mathcal{S}$, we denote by $M^N(k)$ the empirical measure of the objects $\left(X_1^N(k) \dots X_N^N(k)\right)$ at time k:

$$M^N(k) \stackrel{\text{def}}{=} \frac{1}{N} \sum_{n=1}^{N} \delta_{X_n^N(k)},$$

where δ_x denotes the Dirac measure in x. $M^N(k)$ is a probability measure on $\mathcal{S}$ and $M_i^N(k)$ denotes the proportions of objects in state i at time k (also called the density): $M^N(k)[i] = \sum_{n=1}^{N} \mathbf{1}_{X_n^N(k)=i}$.

The system $\left(X^N(k)\right)_{k\in\mathbb{N}}$ is a Markov process once the sequence of actions taken by the controller is fixed. This means that there exists a kernel $\Gamma^N(i_1 \dots i_N, j_1 \dots j_N, a)$ such that if the controller takes the action $A^N(k)$ at time t and the system is in state $X^N(k)$, then:

$$\mathcal{P}\left(X^N(k+1) = j_1 \dots j_N | X^N(k) = i_1 \dots i_N, A^N(k) = a\right) = \Gamma^N\left(i_1 \dots i_N, j_1 \dots j_N, a\right).$$

The main assumption on the kernel Γ^N is that it is invariant by any permutation of the objects. This implies in particular that the objects are only distinguishable through their state. Moreover, this means that the process $M^N(k)$ is also Markovian once the sequence of actions is given. In the following, we will focus on the process of density of the system, $\left(M^N(k)\right)_{k\in\mathbb{N}}$, whose kernel is denoted by Γ^N.

6.2.2. Action, Reward and Policy

At every time k, a centralized controller chooses an action $A^N(k) \in \mathcal{A}$ where $\mathcal{A}$ is called the action set. $(\mathcal{A}, d)$ is a compact metric space for some distance d. As pointed out in the previous chapter, Section 5.2.1, and in Chapter 1, Section 1.3.1, we only focus on

deterministic Markovian policies, since they are dominant. For each k, π_k is a function from $\mathscr{P}(\mathcal{S})$ to $\mathcal{A}$. $M_\pi^N(k)$ denotes the density of the system at time k when the controller applies policy π.

If the system has density $M^N(k)$ at time k and if the controller chooses the action $A^N(k)$, she gets an *instantaneous reward* $r^N(M^N(k), A^N(k))$. The expected average reward over a finite-time horizon $[0; H^N]$ starting from m_0 when applying the policy π is defined by

$$V_\pi^N(m) \stackrel{\text{def}}{=} \mathbb{E}\left(\sum_{k=0}^{\lfloor H^N \rfloor} r^N\left(M_\pi^N(k), \pi(M_\pi^N(k))\right) \middle| M_\pi^N(0) = m \right). \tag{6.1}$$

The goal of the controller is to find a optimal policy that maximizes the expected reward. We denote by $V_*^N(m)$ the optimal reward when starting from m:

$$V_*^N(m) = \sup_\pi V_\pi^N(m).$$

6.2.3. Scaling Assumptions

If at some time k, the system has density $M^N(k) = m$ and the controller chooses action $A^N(k) = a$, the system goes into state $M^N(k+1)$ with probabilities given by the kernel $\Gamma^N(M^N(k), A^N(k))$. The expectation of the difference between $M^N(k+1)$ and $M^N(k)$ is called the *drift* and is denoted by $F^N(m, a)$:

$$F^N(m, a) \stackrel{\text{def}}{=} \mathbb{E}\left(M^N(k+1) - M^N(k) | M^N(k) = m, A^N(k) = a\right).$$

In order to study the limit with N, we assume that F^N goes to 0 at speed $I(N)$ when N goes to infinity and that $F^N/I(N)$ converges to a Lipschitz continuous function f. In a sense, $I(N)$ represents the order of magnitude of the number of objects that change their state within one unit of time.

The changes of $M^N(k)$ during a time step is of order $I(N)$. This suggests a rescaling of time by $I(N)$ to obtain an asymptotic result. We define the continuous time process $\left(\hat{M}^N(t)\right)_{t\in\mathbb{R}^+}$ as the affine interpolation of $M^N(k)$, rescaled by the intensity function, i.e. $\hat{M}^N$ is affine on the intervals $[kI(N), (k+1)I(N)]$, $k \in \mathbb{N}$ and

$$\hat{M}^N(kI(N)) = M^N(k).$$

Similarly, $\hat{M}_\pi^N$ denotes the affine interpolation of the density under policy π. Thus, $I(N)$ can also be interpreted as the duration of the time slot for the system with N objects.

We assume that the time horizon and the reward per time slot scale accordingly, i.e. we impose

$$\begin{aligned} H^N &= \left\lfloor \frac{T}{I(N)} \right\rfloor \\ r^N(m, a) &= I(N) r(m, a) \end{aligned}$$

for every $m \in \mathscr{P}(\mathcal{S})$ and $a \in \mathcal{A}$.

6.2.4. Limiting System (Mean Field Limit)

We will see in Section 6.3 that as N grows, the stochastic system $\hat{M}_\pi^N$ converges to a deterministic limit m_π, the mean field limit. For more clarity, all the stochastic variables (*i.e.*, when N is finite) are in uppercase while their limiting deterministic values are in lowercase.

An action function $\alpha : [0;T] \to \mathcal{A}$ is a piecewise Lipschitz continuous function that associates to each time t an action $\alpha(t)$. Note that action functions and policies are different in the sense that actions functions do not take into account the state to define the next action. For an action function α and an initial condition m_0, we consider the following ordinary differential equation for $m(t)$, $t \in \mathbb{R}^+$:

$$m(t) - m(0) = \int_0^t f(m(s), \alpha(s))ds. \tag{6.2}$$

Under the foregoing assumptions on f and α, this ODE satisfies the Cauchy-Lipschitz condition and therefore has a unique solution once the initial condition $m(0) = m_0$ is fixed. We call ϕ_t, $t \in \mathbb{R}^+$, the corresponding semi-flow, i.e.

$$m(t) = \phi_t(m_0, \alpha)$$

is the unique solution of Equation (6.2) with $m(0) = m_0$.

As for the system with N objects, we define $v_\alpha(m_0)$ as the reward of the limiting system over a finite horizon $[0;T]$ when applying the action function α and starting from $m(0) = m_0$:

$$v_\alpha(m_0) \stackrel{\text{def}}{=} \int_0^T r\left(\phi_s(m_0, \alpha), \alpha(s)\right) ds. \tag{6.3}$$

This equation looks similar to the stochastic case (6.1) although there are two main differences. The first one is that the system is deterministic. The second is that it is defined for action functions and not for policies. We also define the optimal reward of the deterministic limit $v_*(m_0)$:

$$v_*(m_0) = \sup_\alpha v_\alpha(m_0),$$

where the supremum is taken over all possible action functions from $[0;T] \to \mathcal{A}$.

6.2.5. Summary of Assumptions

In Section 6.3 we establish theorems for the convergence of the discrete stochastic optimization problem to a continuous deterministic one. These theorems are based on several technical assumptions, which are given next. Since $\mathcal{S}$ is finite, the set $\mathscr{P}(\mathcal{S})$ is the simplex in $\mathbb{R}^{\mathcal{S}}$ and for $m, m' \in \mathscr{P}(\mathcal{S})$ we define $\|m\|$ as the ℓ^2-norm of m and $\langle m, m' \rangle = \sum_{i=1}^S m_i m'_i$ as the usual inner product.

(A0) (State space and action set) The state space $\mathcal{S}$ is finite. The action set $\mathcal{A}$ endowed with a metric d is compact.

(A1) (Transition Kernel) Objects can be observed only through their state, *i.e.*, the transition kernel Γ^N, defined by Eq.(6.2.1), is invariant by permutations of $1 \dots N$.

There exist some non random functions $I_1(N)$ and $I_2(N)$ such that $\lim_{N\to\infty} I_1(N) = \lim_{N\to\infty} I_2(N) = 0$ and such that for all m and any policy π, the number of objects that perform a transition between time slot k and $k+1$ per time slot $\Delta_\pi^N(k)$ satisfies

$$\begin{aligned} \mathbb{E}\left(\Delta_\pi^N(k) \middle| M_\pi^N(k) = m\right) &\leq N I_1(N), \\ \mathbb{E}\left(\Delta_\pi^N(k)^2 \middle| M_\pi^N(k) = m\right) &\leq N^2 I(N) I_2(N), \end{aligned}$$

where $I(N)$ is the intensity function of the model, defined in the following assumption A2.

(A2) (Convergence of the Drift) There exist some non random functions $I(N)$ and $I_0(N)$ and a function $f(m,a)$ such that $\lim_{N\to\infty} I(N) = \lim_{N\to\infty} I_0(N) = 0$ and

$$\left\| \frac{1}{I(N)} F^N(m,a) - f(m,a) \right\| \leq I_0(N).$$

The function f is defined on $\mathscr{P}(\mathcal{S}) \times \mathcal{A}$ and there exists L_2 such that $|f(m,a)| \leq L_2$.

(A3) (Lipschitz Continuity) There exist constants L_1, K and K_r such that for all $m, m' \in \mathscr{P}(\mathcal{S})$, $a, a' \in \mathcal{A}$:

$$\begin{aligned} \left\| F^N(m,a) - F^N(m',a) \right\| &\leq L_1 \left\| m - m' \right\| I(N), \\ \left\| f(m,a) - f(m',a') \right\| &\leq K(\left\| m - m' \right\| + d(a,a')), \\ \left| r(m,a) - r(m',a) \right| &\leq K_r \left\| m - m' \right\|. \end{aligned}$$

Because of (A1), the simplex $\mathscr{P}(\mathcal{S})$ is bounded. Together with the compactness of $\mathcal{A}$ and the continuity of r (A3), this implies that the reward is bounded and we denote

$$\|r\|_\infty \stackrel{\text{def}}{=} \sup_{m \in \mathscr{P}(\mathcal{S}), a \in \mathcal{A}} |r(m,a)| < \infty.$$

To make things more concrete, here is a simple but useful case where all assumptions are true.

- There are constants c_1 and c_2 such that the expectation of the number of objects that perform a transition in one time slot is $\leq c_1$ and its standard deviation is $\leq c_2$,
- and $F^N(m,a)$ can be written under the form $\frac{1}{N}\varphi(m,a,1/N)$ where φ is a continuous function on $\Delta_S \times \mathcal{A} \times [0,\epsilon)$ for some neighborhood Δ_S of $\mathscr{P}(\mathcal{S})$ and some $\epsilon > 0$, continuously differentiable with respect to m.

In this case one can choose $I(N) = 1/N$, $I_0(N) = c_0/N$ (where c_0 is an upper bound to the norm of the differential $\frac{\partial \varphi}{\partial m}$), $I_1(N) = c_1/N$ and $I_2(N) = (c_1^2 + c_2^2)/N$.

6.3. Mean Field Convergence

In this section we establish the main result, Theorem 6.4, which states the convergence of the optimization problem for the system with N objects to the optimization problem of the mean field limit. To this end we introduce two auxiliary systems. The former is the process $\phi_t(m_0, A_\pi^N)$ defined below, which is a random continuous time system, coupled to the original system with N objects. The latter is M_α^N, also defined below, which is a discrete time system with N objects under a deterministic action function.

6.3.1. First Auxiliary System

Consider the system with N objects under policy π. The process M_π^N is defined on some probability space Ω. To each $\omega \in \Omega$ corresponds a trajectory $M_\pi^N(\omega)$. For each $\omega \in \Omega$, we define an action function $A_\pi^N(\omega)$. This random function is piecewise constant on each interval $[kI(N), (k+1)I(N))$ $(k \in \mathbb{N})$ and is such that $A_\pi^N(\omega)(kI(N)) \stackrel{\text{def}}{=} \pi_k(M^N(k))$ is the action taken by the controller of the system with N objects at time slot k, under policy π.

Recall that for any $m_0 \in \mathscr{P}(\mathcal{S})$ and any action function α, $\phi_t(m_0, \alpha)$ is the solution of the ODE (6.2). For every ω, $\phi_t(m_0, A_\pi^N(\omega))$ is the solution of the limiting system with action function $A_\pi^N(\omega)$, i.e.

$$\phi_t(m_0, A_\pi^N(\omega)) - m_0 = \int_0^t f(\phi_s(m_0, A_\pi^N(\omega)), A_\pi^N(\omega)(s))ds.$$

When ω is fixed, $\phi_t(m_0, A_\pi^N(\omega))$ is a continuous time deterministic process corresponding to one trajectory $M_\pi^N(\omega)$. When considering all possible realizations of M_π^N, $\phi_t(m_0, A_\pi^N)$ is a random, continuous time function "coupled" to M_π^N. Its randomness comes only from the action term A_π^N, in the ODE. In the following, we omit to write the dependence in ω. A_π^N and M_π^N will always designate the processes corresponding to the same ω.

The following result is the main technical result; it shows the convergence of the controlled system in probability, with explicit bounds. Notice that it does not require any regularity assumption on the policy π (recall that $\hat{M}_\pi^N$ is the linear interpolation of the discrete time system with N objects).

Theorem 6.1. *Under Assumption (A0,A1,A2,A3), for any $\epsilon > 0$, $N \geq 1$ and any policy π:*

$$\mathcal{P}\left(\sup_{0\leq t\leq T}\left\|\hat{M}_\pi^N(t) - \phi_t(m_0, A_\pi^N)\right\| > \left[\left\|M^N(0) - m_0\right\| + I_0(N)T + \epsilon\right] e^{L_1 T}\right) \leq \frac{J(N,T)}{\epsilon^2} \tag{6.4}$$

with

$$J(N,T) = 8T\left\{L_1^2\left[I_2(N)I(N)^2 + I_1(N)^2\left(T + I(N)\right)\right] + S^2\left[2I_2(N) + I(N)\left(I_0(N) + L_2\right)^2\right]\right\}. \tag{6.5}$$

Note that $I_0(N)$ and $J(N,T)$ for a fixed T go to 0 as $N \to \infty$. The proof is given in Appendix 6.5.1.

Let π is a policy and A_π^N the sequence of actions corresponding to a trajectory M_π^N as we just defined. Eq.(6.3) defines the reward for the deterministic limit when applying a sequence of actions. This defines a random variable $v_{A_\pi^N}(m_0)$ which corresponds to the reward of System ∞ when applying A_π^N. The random part comes from A_π^N. $\mathbb{E}\left(v_{A_\pi^N}(m_0)\right)$ designates the expectation of this reward over all possible A_π^N. A first consequence of Theorem 6.1 is the convergence of $V_\pi^N\left(M^N(0)\right)$ to $\mathbb{E}\left(v_{A_\pi^N}(m_0)\right)$ with an error that can be uniformly bounded.

Theorem 6.2 (Uniform convergence of the reward)**.** *Let A_π^N be the random action function associated with M_π^N, as defined earlier. Under Assumptions (A0,A1,A2,A3),*

$$\left|V_\pi^N\left(M^N(0)\right) - \mathbb{E}\left(v_{A_\pi^N}(m_0)\right)\right| \leq B\left(N, \left\|M^N(0) - m_0\right\|\right)$$

with

$$\begin{aligned} B(N,\delta) &\stackrel{\text{def}}{=} I(N)\left\|r\right\|_\infty + K_r\left(\delta + I_0(N)T\right)\frac{e^{L_1T}-1}{L_1} \\ &+ \frac{3}{2^{\frac{1}{3}}}\left[\frac{K_r}{L_1}\left(e^{L_1T} - 1 + \frac{I(N)}{2}\right)\right]^{\frac{2}{3}}\left\|r\right\|_\infty^{\frac{1}{3}} J(N,T)^{\frac{1}{3}}. \end{aligned} \tag{6.6}$$

The proof is given in appendix 6.5.2.

Note that $\lim_{N\to\infty,\delta\to 0} B(N,\delta) = 0$; in particular, if $\lim_{N\to\infty} M_\pi^N(0) = m_0$ almost surely [resp. in probability] then $\left|V_\pi^N\left(M^N(0)\right) - \mathbb{E}\left(v_{A_\pi^N}(m_0)\right)\right| \to 0$ almost surely [resp. in probability].

6.3.2. Second Auxiliary System

We now introduce the second auxiliary system. Let α be an action function that specifies the action to be taken at time t. Although α has been defined for the limiting system, it can also be used in the system with N objects. In that case, the action function α can be seen as a policy that does not depend on the state of the system. At step k, the controller applies action $\alpha(kI(N))$. By abuse of notation, we denote by M_α^N, the state of the system when applying the action function α (it will be clear from the notation whether the subscript is an action function or a policy). Similarly we define

$$V_\alpha^N(m_0) \stackrel{\text{def}}{=} \mathbb{E}\left(\sum_{k=0}^{H^N} r\left(M_\alpha^N(k), \alpha(kI(N))\right)\middle| M_\alpha^N(0) = m_0\right).$$

A second consequence of Theorem 6.1 is the convergence of M_α^N and of the reward:

Theorem 6.3. *Assume (A0,A1,A2,A3); α is a piecewise Lipschitz continuous action function on $[0;T]$, of constant K_α, and with at most p discontinuity points. Let $\hat{M}_\alpha^N(t)$ be the linear interpolation of the discrete time process M_α^N. Then for all $\epsilon > 0$:*

$$\mathcal{P}\left(\sup_{0\leq t\leq T}\left\|\hat{M}_\alpha^N(t) - \phi_t(m_0,\alpha)\right\| > \left[\left\|M^N(0) - m_0\right\| + I_0'(N,\alpha)T + \epsilon\right]e^{L_1T}\right) \leq \frac{J(N,T)}{\epsilon^2} \tag{6.7}$$

with

$$I_0'(N,\alpha) = I_0(N) + I(N)Ke^{(K-L_1)T}\left(\frac{K_\alpha}{2} + 2\left(1 + \min(1/I(N),p)\right)\left\|\alpha\right\|_\infty\right).$$

Further,

$$\left|V_\alpha^N\left(M^N(0)\right) - v_\alpha(m_0)\right| \leq B'\left(N, \left\|M^N(0) - m_0\right\|\right) \tag{6.8}$$

with $B'(N,\delta)$ as in Eq.(6.6) but with $I_0(N)$ replaced by $I_0'(N,\alpha)$.

The proof is given in appendix 6.5.3.

Note that $\lim_{N\to\infty,\delta\to 0} B'(N,\delta) = 0$; in particular, if $\lim_{N\to\infty} M_\pi^N(0) = m_0$ almost surely [resp. in probability] then $\lim_{N\to\infty} V_\alpha^N\left(M^N(0)\right) = v_\alpha(m_0)$ almost surely [resp. in probability].

6.3.3. Convergence of Optimization Problems

Theorem 6.4 (Optimal System Convergence)**.** *Let Under Assumptions (A0,A1,A2,A3) hold. If* $\lim_{N\to\infty} M^N(0) = m_0$ *almost surely [resp. in probability] then:*

$$\lim_{N\to\infty} V_*^N\left(M^N(0)\right) = v_*\left(m_0\right)$$

almost surely [resp. in probability].

Proof of Theorem 6.4. This theorem is a direct consequence of Theorem 6.3 and Theorem 6.2. We do the proof for almost sure convergence, the proof for convergence in probability is similar. To prove the theorem we prove

$$\limsup_{N\to\infty} V_*^N(M^N(0)) \leq v_*(m_0) \leq \liminf_{N\to\infty} V_*^N(M^N(0)). \tag{6.9}$$

- Let $\epsilon > 0$ and $\alpha(.)$ be an action function such that $v_\alpha(m_0) \geq v_*(m_0) - \epsilon$ (such an action is $\epsilon-$optimal). Theorem 6.3 shows that $\lim_{N\to\infty} V_\alpha^N(M^N(0)) = v_\alpha(m_0) \geq v_*(m_0) - \epsilon$ a.s. This shows that $\liminf_{N\to\infty} V_*^N(M^N(0)) \geq \lim_{N\to\infty} V_\alpha^N(M^N(0)) \geq v_*(m_0) - \epsilon$; this holds for every $\epsilon > 0$ thus $\liminf_{N\to\infty} V_*^N(M^N(0)) \geq v_*(m_0)$ a.s., which establishes the second inequality in Eq.(6.9), on a set of probability 1.

- Let $B(N,\delta)$ be as in Theorem 6.2, $\epsilon > 0$ and π^N be a policy such that $V_*^N(M^N(0)) \leq V_{\pi^N}^N(M^N(0)) + \epsilon$. By Theorem 6.2, $V_{\pi^N}^N(M^N(0)) \leq \mathbb{E}\left(v_{A_{\pi^N}^N}(m_0)\right) + B(N,\delta^N) \leq v_*(m_0) + B(N,\delta^N)$ where $\delta^N \stackrel{\text{def}}{=} \left\|M^N(0) - m_0\right\|$. Thus $V_*^N(M^N(0)) \leq v_*(m_0) + B(N,\delta^N) + \epsilon$. If further $\delta^N \to 0$ a.s. it follows that $\limsup_{N\to\infty} V_*^N(M^N(0)) \leq v_*(m_0) + \epsilon$ a.s. for every $\epsilon > 0$, thus $\limsup_{N\to\infty} V_*^N(M^N(0)) \leq v_*(m_0)$ a.s.

□

In particular, this theorem, along with Theorem 6.3 shows that an optimal policy for the limiting system is asymptotically optimal for the system with N objects as N goes to infinity. Since the reward function $r(m,a)$ is bounded and the time-horizon $[0;T]$ is finite, the set of reward when starting from the point m, $\{v_\alpha(m) : \alpha \text{ action function}\}$, is bounded. This set is not necessarily compact since the set of action function is not closed (a limit of Lipschitz continuous functions is not necessarily Lipschitz continuous). However, as it is bounded, for all $\epsilon > 0$, there exists an action function α^ϵ such that $v_*(m) = \sup_\alpha v_\alpha(m) \leq v_{\alpha^\epsilon} + \epsilon$. Theorem 6.4 shows that α^ϵ is optimal up to a term 2ϵ for N big enough.

In particular, this shows that as N grows, policies that do not take into account the state of the system (*i.e.*, action functions) are asymptotically as good as adaptive policies. In practice however, adaptive policies might perform better, especially for small values of N. However, it is in general impossible to prove convergence for the adaptive policy.

In fact, in many cases, policies π used for the control of stochastic systems are not continuous in t or m and often exhibit thresholds. In particular, when π is not continuous in m, M_π^N does not necessarily converge and obtaining asymptotics can be difficult. In some particular case, like for the best response dynamics studied in [79], limit theorems can be obtain but at the cost of a greater complexity. In the next Chapter 7, we show how to tackle this discontinuity by introducing the notion of differential inclusions.

6.4. Applications

6.4.1. Hamilton-Jacobi-Bellman equation and dynamic programming

Let us consider the finite time optimization problem for the stochastic system and its limit on a constructive point of view. Since the state space is finite, one can compute the optimal reward by using a dynamic programming algorithm. If $U^N(m,t)$ denotes the optimal reward for the stochastic system starting from m at time $t/I(N)$, then $U^N(m,t) = \sup_\pi \mathbb{E}\left(\sum_{k=t/I(N)}^{T/I(N)} r^N(M_\pi^N(k)) : M^N(t) = m\right)$. The optimal reward can be computed by a discrete dynamic programming algorithm (see Algorithm 1.2 in Chapter 1) by setting $U^N(m,T) = r^N(m)$ and

$$U^N(m,t) = \sup_{a\in\mathcal{A}} \mathbb{E}\left(r^N(m,a) + U^N(M^N(t+I(N)), t+I(N))\middle| \bar{M}^N(t) = m, A^N(t) = a\right). \tag{6.10}$$

Then, the optimal cost over horizon $[0; T/I(N)]$ is $V_*^N(m) = U(m,0)$.

Similarly, let us denote by $u(m,t)$ the optimal cost over horizon $[t;T]$ for the limiting system. If $v(m,t)$ satisfies the Hamilton-Jacobi-Bellman equation:

$$\dot{v}(m,t) + \max_a \{\nabla v(m,t).f(m,a) + r(m,a)\} = 0, \tag{6.11}$$

and if there exists an optimal control $\pi(m,t)$ attaining the max above and such that $\dot{m}(t) = f(m(t), \pi(m,t))$ has a unique solution, then v is the optimal cost for the limiting system: $v(m,t) = u(m,t)$ and the trajectory corresponding to $\pi(m,t)$ is optimal (Proposition 3.2.1 of [35]).

This might provide a way to compute the optimal reward as well as the optimal policy by solving the partial differential equation above. Unfortunately, the Hamilton-Jacobi-Bellman equation is often hard to solve, even numerically, as soon as the dimension of m grows. There exist many way to solve the deterministic optimization problem, depending on the structure of the problem. One of these, is the Pontryagin's maximum principle, used in [94] to solve the example presented below.

6.4.2. Numerical methods

If the continuous time optimization problem can be solved numerically, Theorem 6.4 above can be used to design an effective construction of an asymptotically optimal policy for the system with N objects over the horizon $[0, H^N]$ by using a numerical method described by the Procedure 6.1.

$m := M^N(0)$
Compute the limit of the drift f
Solve the deterministic optimization problem (6.11) on the interval $[0, H]$. This provides an optimal control function $\alpha(t)$
while $k < H^N$:
  $\pi(M^N(k), k) := \alpha(t/I(N))$.
Return π

Procedure 6.1: Static policy π for the system with N objects, over the finite horizon $[0; H^N]$.

Theorem 6.4 says that under policy π, the total reward V_π^N is asymptotically optimal:

$$\lim_{N\to\infty} V_\pi^N(M^N(0)) = \liminf_{N\to\infty} V_*^N(M^N(0)).$$

The policy π constructed by Procedure 6.1 is static in the sense that it does not depend on the state $M^N(k)$ but only on the initial state $M^N(0)$, and the deterministic estimation of $M^N(k)$ provided by the differential equation. One can construct a more adaptive policy by updating the starting point of the differential equation at each step. This new procedure, constructing an adaptive policy π' from 0 to the final horizon H^N is given in Procedure 6.2.

for $k \leq H^N$
  $\alpha_k(M, \cdot) :=$ solution of (6.11) over $[kI(N), H]$ starting in M
 $\pi'(M, k) := \alpha_k(\phi_{kI(N)}(M, \alpha_k))$
return π'

Procedure 6.2: Adaptive policy π' for the system with N objects, over the finite horizon $[0; H^N]$.

In practice, the total reward of the adaptive policy π' is larger than the reward of the static policy π because it uses on-line corrections at each step, before taking a new action. However Theorem 6.4 does not provide a proof of its asymptotic optimality.

6.4.3. Examples

In this section, we develop three examples. The first one can be seen as a simple illustration of optimal mean field. The Limiting ODE is quite simple and can be optimized in closed analytical form.

The second example considers a classical virus problem. While virus propagations concern discrete objects (individuals or devices), most work in the litterature study a continuous approximation of the problem under the form of an ODE. The justification of passing from a discrete to a continuous model is barely mentioned in most papers (they mainly focus on the study of the ODE). Here we present a discrete dynamical system based on a simple stochastic mobility model for the individuals whose behavior converges to a classcial continuous model.

Finally, the last example comes from routing optimization in a queueing network model of volunteer computing platforms. The goal of this last example is to show that a discrete optimal control problem suffering from the curse of dimensionality can be replaced by a continuous optimization problem where an HJB equation must be solved over a much smaller state space.

Utility Provider Pricing

This is a simplified discrete Merton's problem. This example shows a case where the optimization problem in the infinite system can be solved in closed form. This can be seen as an ideal case for the mean field approach: while the original system is difficult to solve even numerically when N is large, taking the limit when N goes to infinity makes it simple to solve, in an analytical form.

We consider a system made of a utility and N users; users can be either in state S (subscribed) or U (unsubscribed). The utility fixes their price $\alpha \in [0, 1]$. At every time

step, one randomly chosen customer revises her status: if she is in state U [resp. S], with probability $s(\alpha)$ [resp. $a(\alpha)$] she moves to the other state; $s(\alpha)$ is the probability of a new subscription, while $a(\alpha)$ is the probability of attrition. We assume $s()$ decreases with α and $a()$ increases. If the price is large the instant gain is large but the utility loses customers, which eventually reduces the gain.

Within our framework, this problem can be seen as a Markovian system made of N objects (users) and one controller (the provider). The intensity of the model is $I(N) = 1/N$. Moreover, if the immediate profit is divided by N (this does not alter the optimal pricing policy) and if $x(t)$ is the fraction of objects in state S at time t and $\alpha(t) \in [0; 1]$ is the action taken by the provider at time t, the mean field limit of the system is:

$$\frac{\partial x}{\partial t} = -x(t)a(\alpha(t)) + (1 - x(t))s(\alpha(t)) = s(\alpha(t)) - x(s(\alpha(t)) + a(\alpha(t))) \tag{6.12}$$

and the rescaled profit over a time horizon T is $\int_0^T x(t)\alpha(t)dt$. The Equation (6.12) can be constructed directly from the description of the model. The first term $-x(t)a(\alpha(t))$ corresponds to the mean number of users that leave the provider while $(1 - x(t))s(\alpha(t))$ corresponds to the mean number of user that join the provider.

Call $u_*(t, x)$ the optimal benefit over the interval $[t, T]$ if there is a proportion x of subscribers at time t. The Hamilton-Jaccobi-Bellman equation is

$$\begin{gathered} \frac{\partial}{\partial t} u_*(t, x) + H\left(x, \frac{\partial}{\partial x} u_*(t, x)\right) = 0 \\ \text{with } H(x, p) = \max_{\alpha \in [0,1]} [p(s(\alpha) - x(s(\alpha) + a(\alpha)) + \alpha x] \end{gathered} \tag{6.13}$$

H can be computed under reasonable assumptions on the rates of subscription and attrition $s()$ and $a()$, which can then be used to show that the optimal policy is threshold based. To continue the rest of this illustration, we consider the radically simplified case where α can take only the values 0 and 1, in which case the ODE becomes

$$\frac{\partial x}{\partial t} = -x(t)\alpha(t) + (1 - x(t))(1 - \alpha(t)) = 1 - x(t) - \alpha(t), \tag{6.14}$$

and $H(x, p) = \max(x(1 - p), (1 - x)p)$. The solution of the HJB equation can be given in closed form. The optimal policy is to chose action $\alpha = 1$ if $x > 1/2$ or $x > 1 - \exp(-(T - t))$, and 0 otherwise. Figure 6.1 shows the evolution of the proportion of subscribers $x(t)$ when the optimal policy is used. The coloured area corresponds to all the points (t, x) where the optimal policy is $\alpha = 1$ (fix a high price) while the white area is where the optimal policy is to choose $\alpha = 0$ (low price).

To show that this policy is indeed optimal, one has to compute the corresponding value of the benefit $u(t, x)$ and show that it satisfies the HJB equation. This can be done using a case analysis, by computing explicitly the value of $u(t, x)$ in the zones Z_1, Z_2, Z_3 and Z_4 displayed in Figure 6.1, and check that $u(t, x)$ satisfies (6.13) in each case.

Infection strategy of a virus worm

This second example has two purposes. The first one is to provide a rigourous justification of the use of a continuous optimization approach for this classical problem in population dynamics. The second one is to show that the continuous limit provides insights on

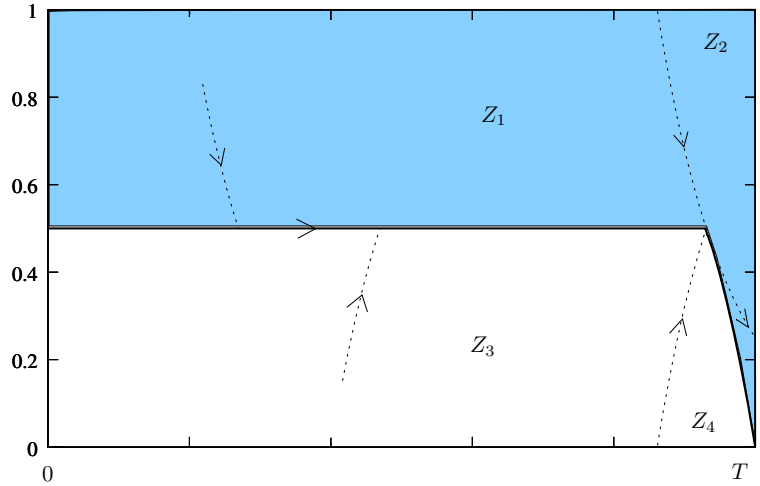


Figure 6.1.: Evolution of the proportion of subscribers (y-axis) under the optimal pricing policy.

the structure of the optimal behavior for the discrete system. Here, the optimal action function can be shown to be of bang-bang type for the limit problem using tools from continuous optimization such as the Pontryagin maximum principle. Theorem 6.4 shows that a bang-bang policy should also be asymptotically optimal in the discrete case.

This example is taken from [94] and considers the propagation of infection by a virus worm. Actually, similar epidemic models have been validated through experiments as well as simulations as a realistic representation of the spread of a virus in mobile wireless networks (see [54, 139]). A *susceptible* node is a mobile wireless device, not contaminated by the worm, but prone to infection. A node is *infective* if it is contaminated by the worm. An infective node spreads the worm to a susceptible whenever they meet, with probability β. The worm can also choose to kill an infective host, i.e., render it completely dysfunctional - such nodes are denoted *dead*. A functional node that is immune to the worm is referred to as *recovered*. The network operator can use security patches to immunize susceptibles (they become recovered) and can also heal infectives to the recovered state. Let the total number of nodes in the network be N. Let the proportion of susceptible, infective, recovered and dead nodes at time t be denoted by $S(t)$, $I(t)$, $R(t)$ and $D(t)$, respectively. Under a uniform mobility model, the probability that a susceptible node becomes infected is $\beta I/N$. The immunization of susceptibles (resp. infectives) happens at a fixed rate q (resp. b). This means that a susceptible (resp. infective) node gets immune with probability q/N (resp. b/N) at every time step.

At this point, the authors of [94] invoke the classical results of Kurtz [101] to show that the dynamics of this population process converges to the solution of the following differential equations.

$$\begin{aligned} \frac{\partial S}{\partial t} &= -\beta IS - qS \\ \frac{\partial I}{\partial t} &= \beta IS - bI - vI \\ \frac{\partial D}{\partial t} &= vI \\ \frac{\partial R}{\partial t} &= bI + qS. \end{aligned} \tag{6.15}$$

However, this does not show that the corresponding optimization problem converges.

This system actually satisfies assumptions (A_1, A_2, A_3), which allows us not only to obtain the mean field limit, but also to say more about the optimization problem. The

objective of the worm is to find $v(\cdot)$ such that the damage function $D(T)+\int_0^T f(I(t))dt$ is maximized under the constraint $0 \leq v \leq v_{\max}$ (where f is convex). In [94], this problem is shown to have a solution and the Pontryagin maximum principle is used to show that the optimal solution $v^*(\cdot)$ is of bang-bang type:

$$\exists t_1 \in [0 \dots T) \text{ s. t. } v^*(t) = 0 \text{ for } 0 < t < t_1 \text{ and } v^*(t) = v_{\max} \text{ for } t_1 < t < T.$$

Theorem 6.4 makes the formal link between the optimization of the model on an individual level and the previous resolution of the optimization problem on the differential equations, done in [94]. It allows us to formally claim that the policy v^* of the worm is indeed asymptotically optimal when the number of objects goes to infinity.

Brokering problem

Finally, let us consider a model of a volunteer computing system like BOINC http://boinc.berkeley.edu/. Volunteer computing means that people make their personal computer available for a computing system. When they do not use their computer, it is available for the computing system. However, as soon as they start using their computer, it becomes unavailable for the computing system. These systems become more and more popular and provide large computing power at a very low cost [98].

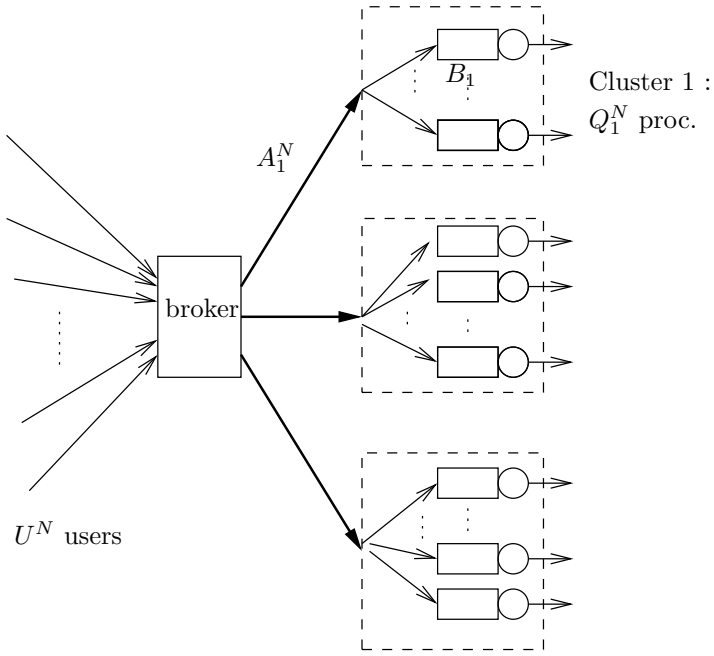


Figure 6.2.: The brokering problem in a desktop grid system, such as Boinc

The Markovian model with N objects is defined as follows. The N objects represent the users that can submit jobs to the system and the resources that can run the jobs. The resources are grouped into a small number of clusters and all resources in the same cluster share the same characteristics in terms of speed and availability. Users send jobs to a central broker whose role is to balance the load among the clusters.

The model is a discrete time model of a queuing system. Actually, a more natural continuous time Markov model could also be handled similarly, by using uniformization.

There are U^N users. Each user has a state $x \in \{\text{on}, \text{off}\}$. At each time step, an active user **s** ends one job with probability p_s^N and becomes **i**nactive with probability p_i/N. An inactive user sends no jobs to the system and gets **o**n with probability p_o/N.

There are C clusters in the system. Each cluster c contains Q_c^N computing resources. Each resource has a buffer of bounded size J_c. A resource can either be valid or broken. If it is valid and if it has one or more job in its queue, it completes one job with probability μ_c/N at this time slot. A resource gets **b**roken with probability p_b/N. In that case, it discards all the jobs of its buffer. A broken resource becomes **v**alid with probability p_v/N.

At each time step, the broker takes an action $a \in \mathscr{P}(\{1 \dots C\})$ and sends the jobs it received to the clusters according to the distribution a. A job sent to cluster c joins the queue of one resource, k according to a local rule (for example chosen uniformly among the Q_c^N resources composing the cluster). If the queue of resource k is full, the job is lost. The goal of the broker is to minimize the number of losses plus the total size of the queues over a finite horizon (and hence the response time of accepted jobs).

This model is represented in Figure 6.2.

The system has an intensity $I(N) \stackrel{\text{def}}{=} 1/N$. The number C of clusters is fixed and does not depend on N, as well as the sizes J_c of the buffers. However, both the number of users U^N, and the number of resources in the clusters Q_c^N, are linear in N. Finally, by construction, all the state changes occur with probabilities that scale with $1/N$.

The limiting system is described by the variable $m_o(t)$, that represents the fraction of users who are on, and the variables $q_{c,j}(t)$ and $b_c(t)$ that respectively represent the fraction of resources in cluster c having j jobs in their buffer and the fraction of resources in cluster c that are broken. For an action function $\alpha(\cdot)$, we denote by $\alpha_c(\cdot)$ the fraction of jobs sent to cluster c. Finally, let us denote by m the fraction of users (both active or inactive) and q_c the fraction of processors in cluster c. These fractions are constant (independent of time) and satisfy $m + q_1 + \cdots + q_C = 1$. The drift function can be computed directly from the description of the system and we get the following equations:

$$\frac{\partial m_o(t)}{\partial t} = -p_i m_o(t) + p_o(m - m_o(t)) \tag{6.16}$$

$$\frac{\partial q_{c,0}(t)}{\partial t} = p_a b_c(t) - \frac{\alpha_c(t) p_s m_o(t)}{q_c} q_{c,0}(t) + \mu_c q_{c,1} - p_b q_{c,0}(t) \tag{6.17}$$

$$\frac{\partial q_{c,j}(t)}{\partial t} = \frac{\alpha_c(t) p_s m_o(t)}{q_c}(q_{c,j-1}(t) - q_{c,i}(t)) + \mu_c(q_{c,j+1} - q_{c,j}) - p_b q_{c,j}(t) \tag{6.18}$$

$$\frac{\partial q_{c,J_c}(t)}{\partial t} = \frac{\alpha_c(t) p_s m_a(t)}{q_c} q_{c,J_c-1}(t) - \mu_c q_{c,J_c} - p_b q_{c,J_c}(t) \tag{6.19}$$

$$\frac{\partial b_c(t)}{\partial t} = -p_v b_c(t) + p_b \sum_{j=0}^{J_c} q_{c,j}(t), \tag{6.20}$$

where (6.17) and (6.19) hold for each cluster c and (6.18) holds for each cluster c and for all $j \leq J_c$. Each term of these equations is given by the probability for an event to appear multiplied by the drift induced by this event and rescaled by $I(N)$. For example, the term $-p_i m_o(t)$ of Equation 6.16 corresponds a user going from state "on" to state "off". If the proportion of user in state "on" is $m_o(0)$, This appears with probability $p_i m_o(t)$.

The cost associated to the action function α is:

$$\int_0^T \sum_{c=1}^{C} \sum_{j=1}^{J_c} j q_{c,j}(t) + \gamma \left(\sum_{c=1}^{C} \frac{\alpha_c(t) p_s m_o(t)}{q_c} (q_{c,J_c}(t) + b_c(t)) + \sum_{c=1}^{C} p_b \sum_{j=1}^{J_c} j q_{c,j}(t) \right) dt. \tag{6.21}$$

The first part of (6.21) represents the cost induced by the number of jobs in the system. The second part of (6.21) represents the cost induced by the losses. The parameter γ gives more or less weight on the cost induced by the losses.

The HJB problem becomes minimizing (6.21) while the variables $u_a, q_{k,i}, b_k$ satisfy Equations (6.16) to (6.20). This system is made of $(J+2)C$ ODEs. Solving the HJB equation numerically in this case can be challenging but remains more tractable than solving the original Bellman equation over J^N states. The curse of dimensionality is so acute for the discrete system that it cannot be solved numerically with more than 10 processors [34].

6.5. Appendix: proofs

6.5.1. Theorem 6.1

The proof is inspired by classical proof for stochastic approximation algorithm and is similar to the proof of Theorem 2.1, recalled in Chapter 1. The main idea of the proof is to write

$$\begin{aligned} \left\| M_\pi^N(k) - \phi_{kI(N)}(m_0, A_\pi^N) \right\| &\leq \left\| M_\pi^N(k) - M^N(0) - \sum_{j=0}^{k-1} f^N(j) \right\| \\ &\quad + \left\| M^N(0) + \sum_{j=0}^{k-1} f^N(j) - \phi_{kI(N)}(m_0, A_\pi^N) \right\| \end{aligned}$$

where $f^N(k) \stackrel{\text{def}}{=} F^N\left(M_\pi^N(k), \pi_k(M_\pi^N(k))\right]$ is the drift at time k if the empirical measure is $M_\pi^N(k)$. The first part is bounded with high probability using a Martingale argument (Lemma 6.6) and the second part is bounded using an integral formula.

Let is define $\bar{M}_\pi^N(t) \stackrel{\text{def}}{=} M_\pi^N\left(\left\lfloor \frac{t}{I(N)} \right\rfloor\right)$, i.e. $\bar{M}_\pi^N(kI(N)) = M_\pi^N(k)$ for $k \in \mathbb{N}$ and $\bar{M}_\pi^N$ is piecewise constant and right-continuous. Let $\Delta_\pi^N(k)$ be the number of objects that change state between time slots k and $k+1$. Thus,

$$\left\| M_\pi^N(k+1) - M_\pi^N(k) \right\| \leq N^{-1}\sqrt{2}\Delta_\pi^N(k) \tag{6.22}$$

and thus

$$\left\| \hat{M}_\pi^N(t) - \bar{M}_\pi^N(t) \right\| \leq N^{-1}\sqrt{2}\Delta_\pi^N(k) \tag{6.23}$$

as well, with $k = \left\lfloor \frac{t}{I(N)} \right\rfloor$. Define

$$Z_\pi^N(k) = M_\pi^N(k) - M^N(0) - \sum_{j=0}^{k-1} F^N\left(M_\pi^N(j), \pi_j(M_\pi^N(j))\right) \tag{6.24}$$

and let $\hat{Z}_\pi^N(t)$ be the continuous, piecewise linear interpolation such that $\hat{Z}_\pi^N(kI(N)) = Z_\pi^N(k)$ for $k \in \mathbb{N}$. Recall that $A_\pi^N(t) \stackrel{\text{def}}{=} \pi_{\lfloor t/I(N)\rfloor}(M^N(\lfloor t/I(N)\rfloor)) - A_\pi^N(t)$ is the action taken by the controller at time $t/I(N)$. It follows from these definitions that:

$$\begin{aligned}
\hat{M}_\pi^N(t) &= M_\pi^N(0) + \int_0^t \frac{1}{I(N)} F^N\left(\bar{M}_\pi^N(s), A_\pi^N(s)\right) ds + \hat{Z}_\pi^N(t) \\
&= M_\pi^N(0) + \int_0^t \frac{1}{I(N)} F^N\left(\hat{M}_\pi^N(s), A_\pi^N(s)\right) ds + \hat{Z}_\pi^N(t) \\
&\quad + \int_0^t \frac{1}{I(N)} \left[F^N\left(\bar{M}_\pi^N(s), A_\pi^N(s)\right) - F^N\left(\hat{M}_\pi^N(s), A_\pi^N(s)\right)\right] ds.
\end{aligned}$$

Using the definition of the semi-flow $\phi_t(m_0, A_\pi^N) = m_0 + \int_0^t f(\phi_s(m_0, A_\pi^N), A_\pi^N(s))ds$, the quantity $\hat{M}_\pi^N(t) - \phi_t(m_0, A_\pi^N)$ is equal to:

$$\begin{aligned}
&M_\pi^N(0) - m_0 + \hat{Z}_\pi^N(t) \\
&+ \int_0^t \frac{1}{I(N)} \left[F^N\left(\hat{M}_\pi^N(s), A_\pi^N(s)\right) - F^N\left(\phi_s(m_0, A_\pi^N), A_\pi^N(s)\right)\right] ds \\
&+ \int_0^t \left[\frac{1}{I(N)} F^N\left(\phi_s(m_0, A_\pi^N), A_\pi^N(s)\right) - f\left(\phi_s(m_0, A_\pi^N), A_\pi^N(s)\right)\right] ds \\
&+ \int_0^t \frac{1}{I(N)} \left[F^N\left(\bar{M}_\pi^N(s), A_\pi^N(s)\right) - F^N\left(\hat{M}_\pi^N(s), A_\pi^N(s)\right)\right] ds.
\end{aligned}$$

Applying Assumption (A2) to the third line, (A3) to the second and fourth lines, and Equation (6.23) to the fourth line leads to:

$$\begin{aligned}
\left\|\hat{M}_\pi^N(t) - \phi_t(m_0, A_\pi^N)\right\| &\leq \left\|M_\pi^N(0) - m_0\right\| + \left\|\hat{Z}_\pi^N(t)\right\| \\
&\quad + L_1 \int_0^t \left\|\hat{M}_\pi^N(s) - \phi_s(m_0, A_\pi^N)\right\| ds \\
&\quad + I_0(N)t + \frac{\sqrt{2}L_1 I(N)}{N} \sum_{k=0}^{\left\lfloor \frac{t}{I(N)} \right\rfloor} \Delta_\pi^N(k).
\end{aligned}$$

For all N, π, T, $b_1 > 0$ and $b_2 > 0$, define

$$\begin{aligned}
\Omega_1 &= \left\{\omega \in \Omega : \sup_{0 \leq k \leq \frac{T}{I(N)}} \sum_{j=0}^{k} \Delta_\pi^N(j) > b_1\right\}, \\
\Omega_2 &= \left\{\omega \in \Omega : \sup_{0 \leq k \leq \frac{T}{I(N)}} \left\|Z_\pi^N(k)\right\| > b_2\right\}.
\end{aligned} \tag{6.25}$$

Assumption (A1) implies conditions on the first and second order moment of $\Delta_\pi^N(k)$. Therefore by Lemma 6.5, this shows that for any $b_1 > 0$:

$$\mathcal{P}(\Omega_1) \leq \frac{TN^2}{b_1^2}\left[I_2(N) + \frac{I_1(N)^2}{I(N)^2}(T + I(N))\right]. \tag{6.26}$$

Moreover, we show in Lemma 6.6 that:

$$\mathcal{P}(\Omega_2) \leq 2S^2 \frac{T}{b_2^2}\left[2I_2(N) + I(N)\left[(I_0(N) + L_2)\right]^2\right]. \tag{6.27}$$

Now fix some $\epsilon > 0$ and let $b_1 = \frac{N\epsilon}{2\sqrt{2}L_1 I(N)}$, $b_2 = \epsilon/2$. For $\omega \in \Omega \setminus (\Omega_1 \cup \Omega_2)$ and for $0 \leq t \leq T$:

$$\begin{aligned}\left\|\hat{M}_\pi^N(t) - \phi_t(m_0, A_\pi^N)\right\| &\leq \left\|M_\pi^N(0) - m_0\right\| + \epsilon + I_0(N)T \\ &\quad + L_1 \int_0^t \left\|\hat{M}_\pi^N(s) - \phi_s(m_0, A_\pi^N)\right\| ds.\end{aligned}$$

By Grönwall's lemma:

$$\left\|\hat{M}_\pi^N(t) - \phi_t(m_0, A_\pi^N)\right\| \leq \left[\left\|M_\pi^N(0) - m_0\right\| + \epsilon + I_0(N)T\right] e^{L_1 t}. \tag{6.28}$$

and this is true for all $\omega \in \Omega \setminus (\Omega_1 \cup \Omega_2)$. We apply the union bound $\mathcal{P}(\Omega_1 \cup \Omega_2) \leq \mathcal{P}(\Omega_1) + \mathcal{P}(\Omega_2)$ which, with Eq.(6.26) and Eq.(6.27), concludes the proof.

The proof of Theorem 6.1 uses the following lemmas.

Lemma 6.5. *Let $(W_k)_{k\in\mathbb{N}}$ be a sequence of square integrable, nonnegative random variables, adapted to a filtration $(\mathcal{F}_k)_{k\in\mathbb{N}}$, such that $W_0 = 0$ a.s. and for all $k \in \mathbb{N}$:*

$$\begin{aligned}\mathbb{E}\left(W_{k+1} \middle| \mathcal{F}_k\right) &\leq \alpha, \\ \mathbb{E}\left(W_{k+1}^2 \middle| \mathcal{F}_k\right) &\leq \beta.\end{aligned}$$

Then for all $n \in \mathbb{N}$ and $b > 0$:

$$\mathcal{P}\left(\sup_{0\leq k\leq n} (W_0 + ... + W_k) > b\right) \leq \frac{n\beta + n(n+1)\alpha^2}{b^2}. \tag{6.29}$$

Proof. Let $m_n = \beta n + n(n+1)\alpha^2$ and $Y_n = \sum_{k=0}^n W_k$. It follows that $\mathbb{E}(Y_n) \leq \alpha n$ and

$$\mathbb{E}\left(Y_{n+1}^2 \middle| \mathcal{F}_n\right) \leq \beta + 2n\alpha^2 + Y_n^2$$

so that $Z_n = Y_n^2 - m_n$ is a supermartingale w.r. $\mathcal{F}_n$. Let T_n be the first time $k \leq n$ at which $Y_k > b$ if it exists, otherwise $T_n = n$, so that $Y_{T_n} > b$ if and only if $\sup_{0\leq k\leq n}(Y_k) > b$. By the optional stopping theorem [127, Thm 6.4.1]:

$$\mathbb{E}(Z_{T_n}) \leq \mathbb{E}(Z_0) = 0$$

thus $\mathbb{E}\left(Y_{T_n}^2\right) \leq \mathbb{E}(m_{T_n}) \leq m_n$. By Markov's inequality, $\mathcal{P}(Y_{T_n} > b) \leq m_n/b^2$. □

Lemma 6.6. *Define Z_π^N as in Eq.(6.24). For all $N \geq 2$, $b > 0$, $T > 0$ and all policy π:*

$$\mathcal{P}\left(\sup_{0\leq k\leq \left\lfloor \frac{T}{I(N)} \right\rfloor} \left\|Z_\pi^N(k)\right\| > b\right) \leq 2S^2 \frac{T}{b^2}\left[2I_2(N) + I(N)\left[(I_0(N) + L_2)\right]^2\right].$$

Proof. The proof is inspired by the methods in [17]. For fixed N and $h \in \mathbb{R}^S$, let

$$L_k = \langle h, Z_\pi^N(k)\rangle.$$

By the definition of Z^N, L_k is a martingale w.r. to the filtration $(\mathcal{F}_k)_{k\in\mathbb{N}}$ generated by M_π^N. Thus

$$\begin{aligned}\mathbb{E}\left((L_{k+1}-L_k)^2\middle|\mathcal{F}_k\right) &= \mathbb{E}\left(\langle h, M_\pi^N(k+1)-M_\pi^N(k)\rangle^2\middle|\mathcal{F}_k\right)\\ &\quad+\langle h, F^N\left(M_\pi^N(k),\pi_k(M_\pi^N(k))\right)\rangle^2.\end{aligned}$$

By Assumption (A2):

$$\left|\langle h, F^N\left(M_\pi^N(k),\pi(M_\pi^N(k))\right)\rangle\right| \le (I_0(N)+L_2)\, I(N)\, \|h\|\,.$$

Thus, using Eq.(6.22) and Assumption (A5):

$$\begin{aligned}\mathbb{E}\left((L_{k+1}-L_k)^2\middle|\mathcal{F}_k\right) &\le \|h\|^2\left[N^{-2}2\mathbb{E}\left(\Delta_\pi^N(k)^2\middle|\mathcal{F}_k\right)+[(I_0(N)+L_2)\, I(N)]^2\right]\\ &\le \|h\|^2\left[2I(N)I_2(N)+[(I_0(N)+L_2)\, I(N)]^2\right].\end{aligned}$$

We now apply Kolmogorov's inequality for martingales and obtain

$$\mathcal{P}\left(\sup_{0\le k\le n} L_k > b\right) \le \frac{n}{b^2}\|h\|^2\left[2I(N)I_2(N)+[(I_0(N)+L_2)\, I(N)]^2\right].$$

Let Ξ_h be the set of $\omega\in\Omega$ such that $\sup_{0\le k\le n}\langle h, Z_\pi^N(k)\rangle \le b$ and let Ξ be defined by $\Xi \stackrel{\text{def}}{=} \bigcap_{h=\pm\vec{e}_i, i=1\ldots S}\Xi_h$ where $\vec{e}_i$ is the ith vector of the canonical basis of $\mathbb{R}^S$. It follows that, for all $\omega\in\Xi$ and $0\le k\le n$ and $i=1\ldots S$: $\left|\langle Z_\pi^N(k),\vec{e}_i\rangle\right| \le b$. This means that for all $\omega\in\Xi$: $\left\|Z_\pi^N(k)\right\| \le \sqrt{S}b$. By the union bound applied to the complement of Ξ, we have

$$1-\mathcal{P}(\Xi) \le 2S\frac{n}{b^2}\left[I(N)I_2(N)+[(I_0(N)+L_2)\, I(N)]^2\right].$$

Thus we have shown that, for all $b>0$:

$$\mathcal{P}\left(\sup_{0\le k\le n}\left\|Z_\pi^N(k)\right\| > \sqrt{S}b\right) \le 2S\frac{nI(N)}{b^2}\left[I_2(N)+I(N)\left[(I_0(N)+L_2)\right]^2\right]$$

which, by changing b into $b/\sqrt{S}$, shows the result. □

6.5.2. Proof of Theorem 6.2

We use the same notation as in the proof of Theorem 6.1. By definition of V^N, v and the time horizons:

$$\begin{aligned}V_\pi^N(M^N(0))-\mathbb{E}\left(v_{A_\pi^N}(m_0)\right) &= \mathbb{E}\left(\int_0^{H^N I(N)} r(\bar{M}_\pi^N(s),A_\pi^N(s))-r(m_{A_\pi^N}(s),A_\pi^N(s))ds\right)\\ &\quad-\mathbb{E}\left(\int_{H^N I(N)}^{T} r(m_{A_\pi^N}(s),A_\pi^N(s))ds\right).\end{aligned}$$

The latter term is bounded by $I(N)\|r\|_\infty$. Let $\epsilon>0$ and $\Omega_0=\Omega_1\cup\Omega_2$ where Ω_1,Ω_2 are as in the proof of Theorem 6.1. Thus $\mathcal{P}(\Omega_0)\le\frac{J(N,T)}{\epsilon^2}$ and, using the Lipschitz continuity

of r in m (with constant K_r):

$$\left|V_\pi^N(M^N(0)) - \mathbb{E}\left(v_{A_\pi^N}(m_0)\right)\right| \leq I(N)\left\|r\right\|_\infty + \frac{2\left\|r\right\|_\infty J(N,T)}{\epsilon^2} + K_r\mathbb{E}\left(1_{\omega\notin\Omega_0}\int_0^T \left\|\bar{M}_\pi^N(s) - m_{A_\pi^N}(s)\right\| ds\right).$$

For $\omega \notin \Omega_0$ and $s \in [0,T]$: $\int_0^T \left\|\bar{M}_\pi^N(s) - \hat{M}_\pi^N(s)\right\| ds \leq \frac{\epsilon I(N)}{2L_1}$ and, by Eq.(6.28), $\int_0^T \left\|\hat{M}_\pi^N(s) - m_{A_\pi^N}(s)\right\| ds \leq \left(\left\|M^N(0) - m_0\right\| + I_0(N)T + \epsilon\right)\frac{e^{L_1T}-1}{L_1}$ thus

$$\left|V_\pi^N(M^N(0)) - \mathbb{E}\left(v_{A_\pi^N}(m_0)\right)\right| \leq B_\epsilon(N, \left\|M^N(0) - m_0\right\|) \tag{6.30}$$

where

$$B_\epsilon(N,\delta) \stackrel{\text{def}}{=} I(N)\left\|r\right\|_\infty + K_r\left(\delta + I_0(N)T + \epsilon\right)\frac{e^{L_1T}-1}{L_1} + \frac{K_rI(N)}{2L_1}\epsilon + \frac{2\left\|r\right\|_\infty J(N,T)}{\epsilon^2}$$

This holds for every $\epsilon > 0$, thus

$$\left|V_\pi^N(M^N(0)) - \mathbb{E}\left(v_{A_\pi^N}(m_0)\right)\right| \leq B(N, \left\|M^N(0) - m_0\right\|) \tag{6.31}$$

where $B(N,\delta) \stackrel{\text{def}}{=} \inf_{\epsilon>0} B_\epsilon(N,\delta)$. By direct calculus, one finds that $\inf_{\epsilon>0}\left(a\epsilon + b/\epsilon^2\right) = 3/2^{\frac{2}{3}}a^{\frac{2}{3}}b^{\frac{1}{3}}$ for $a > 0, b > 0$, which gives the required formula for $B(N,\delta)$.

6.5.3. Proof of Theorem 6.3

Let $\bar{\alpha}^N$ be the right-continuous function constant on the intervals $[kI(N);(k+1)I(N))$ such that $\bar{\alpha}^N(s) = \alpha(s)$. $\bar{\alpha}^N$ can be viewed as a policy independent of m. Therefore, by Theorem 6.1, on the set $\Omega \setminus (\Omega_1 \cup \Omega_2)$, for every $t \in [0;T]$:

$$\left\|\hat{M}_\alpha(t) - \phi_t(m_0,\alpha)\right\| \quad \leq \quad \left[\left\|M^N(0) - m_0\right\| + I_0(N)T + \epsilon\right]e^{L_1T} + u(t)$$

with $u(t) \stackrel{\text{def}}{=} \left|\phi_t(m_0,\bar{\alpha}^N) - \phi_t(m_0,\alpha)\right|$. We have

$$\begin{aligned} u(t) &\leq \int_0^t \left|f(\phi_s(m_0,\alpha),\alpha(s)) - f(\phi_s(m_0,\bar{\alpha}^N),\bar{\alpha}^N(s))\right| ds \\ &\leq \int_0^t K\left(\left\|\phi_s(m_0,\alpha) - \phi_s(m_0,\bar{\alpha}^N)\right\| + d(\alpha(s),\bar{\alpha}^N(s)\right) ds \\ &\leq K\int_0^t u(s)ds + Kd_1 \end{aligned}$$

where $d_1 \stackrel{\text{def}}{=} \int_0^T \left\|\alpha(t) - \bar{\alpha}^N(t)\right\| dt$. Therefore, using Grönwall's inequality, we have

$$u(t) \leq Kd_1e^{KT}.$$

By Lemma 6.7, this shows Eq.(6.7). The rest of the proof is as for Theorem 6.2.

Lemma 6.7. *If α is a piecewise Lipschitz continuous action function on $[0;T]$, of constant K_α, and with at most p discontinuity points, then*

$$\int_0^T d(\alpha(t), \bar{\alpha}^N(t))dt \leq TI(N)\left(\frac{K_\alpha}{2} + 2\left(1 + \min(1/I(N), p)\right)\|\alpha\|_\infty\right).$$

Proof of lemma 6.7. Let first assume that $T = kI(N)$. The left-hand side $d_1 = \int_0^T d(\alpha(t), \bar{\alpha}^N(t))dt$ can be decomposed on all intervals $[iI(N), (i+1)I(N))$:

$$\begin{aligned} d_1 &= \sum_{i=0}^{\lfloor T/I(N)\rfloor} \int_{iI(N)}^{(i+1)I(N)} \left\|\alpha(s) - \bar{\alpha}^N(s)\right\| ds \\ &\leq \sum_{i=0}^{\lfloor T/I(N)\rfloor} \int_{iI(N)}^{(i+1)I(N)} \|\alpha(s) - \alpha(iI(N))\| ds. \end{aligned}$$

If α has no discontinuity point on $[iI(N), (i+1)I(N))$, then

$$\int_{iI(N)}^{(i+1)I(N)} d(\alpha(s), \alpha(iI(N)))ds \leq \int_0^{I(N)} K_\alpha s ds \leq K_\alpha 2I(N)^2.$$

If α has one or more discontinuity points on $[iI(N), (i+1)I(N))$, then

$$\int_{iI(N)}^{(i+1)I(N)} d(\alpha(s)\alpha(iI(N)))ds \leq \int_{iI(N)}^{(i+1)I(N)} 2\|\alpha\|_\infty ds \leq 2\|\alpha\|_\infty I(N).$$

There are at most $\min(1/I(N), p)$ intervals $[iI(N), (i+1)I(N)]$ that have discontinuity points which shows that

$$d_1 \leq TI(N)(\frac{K_\alpha}{2} + \min(1/I(N), p)2\|\alpha\|_\infty).$$

If $T \neq kI(N)$, then $T = kI(N) + t$ with $0 < t < I(N)$. Therefore, there is an additional term of $\int_{kI(N)}^{kI(N)+t} d(\alpha(s), \bar{\alpha}^N(s))ds \leq 2\|\alpha\|_\infty I(N)$. □

Chapter 7.

Non-smooth Mean-Field Models

Abstract of this chapter – In this chapter, we study deterministic limits of Markov processes made of several interacting objects. While most classical results assume that the limiting dynamics has Lipschitz properties, we show that these conditions are not necessary to prove convergence to a deterministic system.

We show that under mild assumptions, the stochastic system converges to the set of solutions of a differential inclusion and we provide simple way to compute the limiting inclusion. When this differential inclusion satisfies a one-sided Lipschitz condition, there exists a unique solution of this differential inclusion and we show convergence in probability with explicit bounds.

This extends the applicability of mean field techniques to systems exhibiting threshold dynamics such as queuing systems with boundary conditions or controlled dynamics. This is illustrated by applying our results to several types of systems: fluid limits of priority queues, best response dynamics in games, push-pull queues with a large number of sources and a large number of servers and self-adapting computing systems.

Résumé du chapitre – Dans ce chapitre nous étudions le comportement limite de processus de Markov composés de multiples objets en interaction. Alors que la plupart des résultats classiques supposent que la dynamique limite a des propriétés de régularité de type Lipschitz, nous montrons que ces conditions ne sont pas nécessaires pour prouver la convergence vers un système déterministe. Nous montrons que sous des hypothèses assez faibles, le system stochastique converge vers la solution d'une inclusion différentielle et nous donnons un moyen simple d'obtenir cette inclusion différentielle limite. Quand elle satisfait une condition de semi-Lipschitz, il existe une solution unique à cette inclusion différentielle et nous montrons la convergence en probabilité en donnant des bornes explicites.

Cela permet d'étendre l'applicabilité des techniques de champs moyen à des systèmes avec des dynamiques à seuil, comme c'est le cas pour des systèmes de files d'attentes. Ceci est illustré par l'application de nos résultats à des files "push-pull" avec un grand nombre de sources de paquets et un grand nonbre de serveurs, qui constituent des modèles naturels de systèmes de calculs à volontaires.

7.1. Introduction

Let us consider a discrete stochastic dynamical system composed by a large number of objects. Under classical smoothness assumptions, there exist general results that show that the limiting system (when the number of objects goes to infinity) can be described by a system of deterministic ordinary differential equations

$$\dot{y}(t) = f(y(t)). \tag{7.1}$$

Such models are known as mean field limits and described in Chapter 2.

In most cases, the limiting drift function f in (7.1) is assumed to be Lipschitz. This strong condition hampers the applicability of these results in many practical cases, in particular, for systems exhibiting thresholds dynamics or with boundary conditions. This chapter studies the limiting behavior of such a system when the dynamics is non-smooth. Let us consider a simple queuing system with one buffer and many processors that can serve one packet per unit of time, on average. If y denotes the number of packets in the queue, then the average decrease of y is one packet per unit of time (under a proper rescaling of time) if the queue is non-empty (*i.e.* $y > 0$) and zero if the queue is empty. This leads to a deterministic limit behavior:

$$\dot{y}(t) = -1 \ \text{ if } y(t) > 0 \quad \text{and} \quad \dot{y}(t) = 0 \ \text{ if } y(t) = 0. \tag{7.2}$$

This dynamics is not continuous and therefore non Lipschitz which makes the classical approach inapplicable in that case.

Actually, most work using mean field limits for networks do not involve queues or when they do, the number of queues scale with the number of objects (as in the previous chapter), or convergence is obtained using *ad hoc* proofs (see for example [13]).

In the case of a non-continuous right-hand side, the differential equation (7.1) is not well-defined since there exist no function y that is differentiable and that satisfies (7.2). The proper way to define solutions of (7.2) is to use differential inclusions (DI) instead. Equation (7.1) is replaced by the following equation

$$\dot{y}(t) \in F(y(t)), \tag{7.3}$$

where F is a set-valued mapping: if $y \neq 0$ then $F(y) = \{-1\}$ and $F(0) = \{u : -1 \leq u \leq 0\}$. Of course a differential inclusion problem may (or may not) have multiple solutions.

In the following, we will provide generic convergence results of mean field type that show that under few conditions on the system, its behavior converges to the solutions of a differential inclusion (7.3) (Theorem 7.5). This result is generic and does not require any Lipschitz property on the set function F. In particular, it shows that when (7.3) has a unique solution, the behavior of the system converges to it. Moreover, we also show that when F satisfies a one-sided Lipschitz condition (7.9), we can bound the gap with the limiting dynamics with explicit bounds (Theorem 7.7).

The rest of the chapter is organized as follows. In Section 7.2, we briefly describe the general framework of our work and we give several examples. We will briefly recall some definitions and properties of differential inclusions in Section 7.2.1. Section 7.3 states the main theoretical results, Section 7.4 extends the results to time continuous Markov chains and Section 7.5 describes several application examples.

7.2. Description of the Model and Notations

We consider of system composed by N *objects* evolving in a finite state space $\mathcal{S} = \{1 \dots S\}$. Time is discrete and the state of object n at time step k is denoted $X_n^N(k)$ and $\mathcal{X}^N(k) \stackrel{\text{def}}{=} (X_1^N(k), \dots, X_N^N(k))$. The objects all evolve in a common environment, called the *context*. The context at time step k is denoted by $C^N(k) \in \mathbb{R}^d$. The state of the global system at time k is $(\mathcal{X}^N(k), C^N(k))$. We denote by $M^N(k)$ the empirical measure associated with the N objects:

$$M^N(k) \stackrel{\text{def}}{=} \frac{1}{N} \sum_{n=1}^{N} \delta_{X_n^N(k)}.$$

Since an object has S possible states, $M^N(k)$ can be represented by a vector with S components, its ith component being the proportion of objects in state i:

$$M_i^N(k) \stackrel{\text{def}}{=} \frac{1}{N} \sum_{n=1}^{N} \mathbf{1}_{X_n^N(k)=i}.$$

The system $(M^N(k), C^N(k))_k$ is assumed to be a Markov chain. In particular, this is true if the objects all have a Markovian dynamics, if the evolution of the context is deterministic and if the law of the whole system is invariant by any permutation of the N objects. The state space of this Markov chain is included in $\mathbb{R}^{S+d}$. To simplify the notations, we call $Y^N(k) \stackrel{\text{def}}{=} (M^N(k), C^N(k))$.

If at time k, the system is in state $Y^N(k) = y$, then the expected difference between $Y^N(k+1)$ and $Y^N(k)$ is called the *drift* and is denoted $f^N(y)$:

$$f^N(y) \stackrel{\text{def}}{=} \mathbb{E}\left(Y^N(k+1) - Y^N(k) | Y^N(k) = y\right). \tag{7.4}$$

This only defines f^N on the state space of Y^N, which is a subset of R^{d+S}. Outside the state space of Y^N, $f^N(y)$ is not defined at this point.

We assume that as N grows, the drift vanishes with speed $I(N)$. This means that there exists a function $I(N)$, called the *intensity* of the model, a set-valued function F and a continuation of the function f^N on all $\mathbb{R}^{d+S}$ such that

- The intensity vanishes: $\lim_{N\to\infty} I(N) = 0$.
- f^N converges uniformly to F in the following sense: there exists $J(N)$ with $\lim_{N\to\infty} J(N) = 0$ and a function f such that $f(y) \in F(y)$ for all $y \in \mathbb{R}^{d+S}$, satisfying
$$\sup_{y\in\mathbb{R}^{d+S}} \left\| \frac{f^N(y)}{I(N)} - f(y) \right\| \leq J(N). \tag{7.5}$$
where $\|.\|$ denotes the L^2 norm.

Actually, the following results (Theorems 7.5 and 7.7) remain valid under a weaker convergence condition of f^N to F. Instead of assuming that f^N has a limit f, one could simply assume that

$$\sup_{y\in\mathbb{R}^{d+S}} \inf_{u\in F(y)} \left\| \frac{f^N(y)}{I(N)} - u \right\| \leq J(N). \tag{7.6}$$

All the foregoing proofs would hold by replacing f by any h achieving the infimum in the above equation (7.6). However, in most practical cases (and in all the examples of this chapter), the rescaled drift $f^N/I(N)$ converges indeed to a (non-continuous) function f, so that we focus on that case in the following.

Finally, we assume that the second order of the drift is bounded. This means that there exists $b > 0$ such that:

$$\mathbb{E}\left(\left\|\frac{Y^N(k+1) - Y^N(k) - f^N(Y^N(k))}{I(N)}\right\|^2\right) \leq b. \tag{7.7}$$

All these assumptions are more or less necessary to use a mean field approach. The notable important feature of our model is that we do not assume any regularity property on the function f.

7.2.1. Differential inclusions

In Section 7.3, we will see that under mild conditions, the system described by the variables $(M^N(.), C^N(.))$ converges to the solutions of a deterministic differential inclusion. In this section, we recall the main concepts on differential inclusions. For a more complete description, the reader is referred to [1]. In all that follows, $\langle x, y\rangle$ denotes the classical inner-product on $\mathbb{R}^{d+S}$ and $\|x\| = \sqrt{\langle x, x\rangle}$ (L^2 norm) and $\|A\| = \sup_{x\in A} \|x\|$.

Definition 7.1. *Consider a differential inclusion problem:*

$$\dot{y}(t) \in F(y(t)), \quad y(0) = y_0, \tag{7.8}$$

where F is a set-valued function mapping each point $y \in \mathbb{R}^{d+S}$ to a set $F(y) \subset \mathbb{R}^{d+S}$. Let $I \subset \mathbb{R}$ be an interval with $0 \in I$. A function $y : I \to \mathbb{R}^{d+S}$ is a solution of the DI (7.8) with initial condition $y(0) = y_0$ if there exists a integrable function $\varphi : I \to \mathbb{R}^{d+S}$ such that:

(i) for all $t \in I$: $y(t) = y_0 + \int_0^t \varphi(s)ds$;

(ii) for almost every (a.e.) $t \in I$: $\varphi(t) \in F(y(t))$.

In particular, *(i)* is equivalent to say that y is absolutely continuous. *(i)* and *(ii)* imply that y is differentiable at almost every $t \in I$ with $\dot{y}(t) \in F(y(t))$.

Definition 7.2 (Upper Semi-Continuous (USC))**.** *The function F is said to be upper semi-continuous (USC) if for any $y \in \mathbb{R}^{d+S}$, $F(y)$ is a non-empty closed, convex and bounded set and if for any open set O containing $F(y)$, there exists a neighborhood V of y such that $F(V) \subset O$.*

Definition 7.3 (One-Sided Lipschitz (OSL))**.** *A set-valued function F is one-sided Lipschitz (OSL) with constant L if for all $y, \bar{y} \in \mathbb{R}^{d+S}$ and for all $u \in F(y)$ $\bar{u} \in F(\bar{y})$:*

$$\langle y - \bar{y}, u - \bar{u}\rangle \leq L \|y - \bar{y}\|^2 . \tag{7.9}$$

These two definitions give sufficient conditions for the existence (resp. uniqueness) of solutions for the differential inclusion (7.8). We recall the following results.

Proposition 7.4 (Theorems 2.2.1 and 2.2.2 of [100])**.**

- *If F is USC and if there exists c such that $\|F(x)\| \leq c(1 + \|x\|)$ then for all initial condition y_0, (7.8) has at least one solution on $[0; \infty)$ with $y(0) = y_0$.*
- *If F is OSL, then for all $T > 0$, there exists at most one solution of (7.8) on $[0; T]$.*

Of course (USC) and (OSL) combined insure that the DI has a unique solution.

7.2.2. A set valued function for the drift

In our framework, if $f^N(.)$ is the drift of the system defined in Equation (7.4), we define an adapted set-valued function F by:

$$F(y) = \bigcap_{\epsilon>0} \lim_{K\to\infty} \bigcup_{N\geq K} \overline{\text{conv}} \left\{ \frac{f^N(z)}{I(N)} : \|z - y\| \leq \epsilon \right\}, \tag{7.10}$$

where $\overline{\text{conv}}(A)$ is the closure of the convex hull of a set A. It should be clear that Equation (7.10) defines a set-valued function F such that for all y:

$$\lim_{N\to\infty} \inf_{u\in F(y)} \left\| f^N(y)/I(N) - u \right\| = 0.$$

Therefore, Equation (7.6) is satisfied as soon as this convergence holds uniformly in y. Furthermore, the function F defined in (7.10) is USC.

If we assume instead that the drift f^N converges uniformly to some function f, meaning that $\left\| f^N(y)/I(N) - f(y) \right\| \leq J(N)$, the set valued function F associated to the drift can be defined as:

$$F(y) = \bigcap_{\epsilon>0} \overline{\text{conv}} \left\{ f(z) : \|z - y\| \leq \epsilon \right\}. \tag{7.11}$$

In that case, if the function f is continuous in a point y, then F is a singleton: $F(y) = \{f(y)\}$ while if f is discontinuous in y, $F(y)$ is the convex closure of the limit set of f at point y.

7.3. Convergence results

This section contains the two main theoretical contributions of this chapter. The first one is Theorem 7.5 that shows that if F is USC then the stochastic system converges to the set of solutions of the differential inclusion. In particular, this implies that if the differential inclusion has only one solution, the stochastic system converges to this solution (Corollary 7.6).

This theorem does not give any bound on the speed of convergence. In fact, without further conditions, the convergence may be arbitrarily slow. However, when the differential inclusion is both USC and OSL, the the solution of the differential equation is unique and the speed of convergence can be lower-bounded (Theorem 7.7).

The rest of this section is organized as follows. We first state the two convergence results and then discuss their applicability. The proofs of the two theorems are postponed in the appendix. The proof of the first theorem is similar to classical proofs for the existence of solutions of differential inclusions while the second one is more similar to classical proofs of the convergence of stochastic approximation algorithms, like the proof of Theorem 2.1 in Chapter 2 or the proof of Theorem 6.1 in the previous chapter.

Let us now recall that at time step k, the system is in state $Y^N(k) \stackrel{\text{def}}{=} (M^N(k), C^N(k))$ and let us denote $\bar{Y}^N(t)$ the piecewise affine interpolation of $Y^N(k)$ when we scale time by a factor $I(N)$. Let us first define the continuous function $\tilde{Y}^N(t)$ as the component-wise linear interpolation of $\{Y^N(k)\}_{k\in\mathbb{N}}$: for all i,

$$\tilde{Y}_i^N(t) \stackrel{\text{def}}{=} (1-\alpha)Y_i^N(\lfloor t\rfloor) + \alpha Y_i^N(\lceil t\rceil),$$

with $\alpha = t - \lfloor t\rfloor$.

Then, by scaling time, $\bar{Y}^N(t) \stackrel{\text{def}}{=} \tilde{Y}^N(t/I(N))$.

Let us denote by $\mathscr{S}_T(y_0)$ the set of the solutions of the DI (7.8) starting from $y(0) = y_0$, we can show that $\bar{Y}(.)$ converges (in probability) to $\mathscr{S}_T(y_0)$. More precisely, the following theorem holds.

Theorem 7.5. *Let F be the limit of the drift defined by Equation (7.5). Assume that the drift satisfies (7.7) and that*

- *F is USC and there exists $c > 0$ s.t. $\|F(y)\| \le c(1 + \|y\|)$,*
- *$Y^N(0) \xrightarrow{\mathcal{P}} y_0$ (convergence in probability).*

Then for all $T > 0$:

$$\inf_{y\in\mathscr{S}_T(y_0)} \sup_{0\le t\le T} \left\|\bar{Y}^N(t) - y(t)\right\| \xrightarrow{\mathcal{P}} 0.$$

Proof. The proof is given in Section 7.6.1. □

This theorem shows that if N is large enough, the trajectory of the stochastic system $\bar{Y}^N$ on $T/I(N)$ steps is close to a solution of the differential inclusion (7.8) over $[0, T]$. In general a differential inclusion may have multiple solutions. Here, $\bar{Y}^N$ may converge to any solution of the DI, depending on its random innovations, making this result rather inefficient for performance evaluation. This result is of greater interest if the DI starting from y_0 has a unique solution: $\mathscr{S}_T(y_0) = \{y\}$. In that case, as a direct corollary of the preceding result, $\bar{Y}^N$ converges in probability to y on all intervals $[0; T]$.

Corollary 7.6. *Under the conditions of Theorem 7.5, if the DI (7.8) has a unique solution y, then for all T:*

$$\sup_{0\le t\le T} \left\|\bar{Y}^N(t) - y(t)\right\| \xrightarrow{\mathcal{P}} 0.$$

In some cases, like the example of push-pull queues described in Section 7.5, the limiting differential inclusion clearly has a unique solution which makes the preceding corollary directly applicable. The main drawback of the previous theorem is that it does not give any insight on the speed of convergence on the stochastic system towards its limit.

This limitation can be overcome when the function F satisfies the one-sided Lipschitz condition (7.9). Firstly, this ensures the uniqueness of the solution. Secondly, one can get precise bounds on the gap between the stochastic system and its limit in that case.

Theorem 7.7. *Under the conditions of Theorem 7.5 and if F is OSL with constant L, then the DI (7.8) has a unique solution y and there exist constants A_T, B_T, C_T depending only on T, L and c such that for all ϵ,*

$$\mathcal{P}\left(\sup_{0\le t\le T}\left\|Y^N(t)-y(t)\right\| \ge \left\|Y^N(0)-y(0)\right\| e^{LT} + \sqrt{I(N)}A_T + \sqrt{J(N)}B_T + \epsilon\right) \le \frac{I(N)}{\epsilon^2}C_T.$$

The constants A_T, B_T and C_T are given by

$$\begin{aligned} A_T &\stackrel{\text{def}}{=} e^{2LT}\sqrt{\frac{I(N)}{6}c^2(1+K_T)+\frac{K_T}{2L}(c(1+K_T)+J(N))}; \\ B_T &\stackrel{\text{def}}{=} \frac{\sqrt{K_T}e^{2LT}}{\sqrt{L}}; \\ C_T &\stackrel{\text{def}}{=} \frac{e^{2LT}}{2L}\left(4c^2(1+K_T)^2+b\right) \end{aligned}$$

with $K_T = \max\{\|y(0)\|+c, \|y_0^N\|+(c+J(N))T), \|Y_0^N\|+(c+J(N))T+\sqrt{bT}\}e^{cT}/c$ and c is defined in Theorem 7.5 and b in Equation (7.7).

Proof. The proof is given in Section 7.6.2. □

Note that if $F(.)$ is bounded by some K_F the terms $c(1+K_T)$ can be replaced by K_F. This is in particular true if Y^N is constrained to stay in a compact space of $\mathbb{R}^{d+S}$ or if the drift is bounded for all $y \in \mathbb{R}^{d+S}$.

These constants are of a similar order than bounds that can be obtained in the case where f is Lipschitz, such as the one of Theorem 6.1 of the previous chapter. However, the convergence speed with respect to N is in $O(\sqrt{I(N)})$ (compared with $O(I(N))$ is the Lipschitz case).

7.4. Extension to Non-smooth Density Dependent Population Processes

In this section, we show that our results can be adapted to the case of continuous time Markov chains and in particular for the famed model of density dependent population processes, described in Chapter 2, Section 2.2.1. The two theorems 7.5 and 7.7 can be transposed in this case.

Let $(D^N)_N$ be a density dependent population processes: D^N is a continuous Markov chain on $\frac{1}{N}\mathbb{Z}^d$ $(d \ge 1)$ and there exists a set $\mathcal{L} \subset \mathbb{Z}^d$ (with $0 \notin \mathcal{L}$), such that for each $\ell \in \mathcal{L}$ and $x \in N^{-1}\mathbb{Z}^d$, the rate of transition from x to $x+\ell/N$ is $N\beta_\ell(x) \ge 0$, where $\beta_\ell(.)$ does not depend on N. Let us assume that $\sum_{\ell\in\mathcal{L}}\sup_{x\in\mathbb{Z}^d}\|\beta_\ell(x)\| = \tau < \infty$ and let us define $f(x) = \sum_{\ell\in\mathcal{L}}\beta_\ell(x)\ell$ (in the following, we assume that this sum is well-defined).

If f is Lipschitz, then $D^N(.)$ converges to the solution of the equation $\dot{d}(t) = f(d(t))$ (see Theorem 2.1 in Chapter 2). Using uniformization of the Markov chain and the results from Section 7.3 we now show that this convergence still holds for general drifts, replacing f by its set-valued counterpart F, defined in (7.11).

Theorem 7.8. *If* $\sup_{x\in\mathbb{Z}^d}\sum_{\ell\in\mathcal{L}}\beta_\ell(x)=\tau<\infty$, *if* $\sum_{\ell\in\mathcal{L}}\|\ell\|^2\beta_l^2(y)<b<\infty$ *and if* F *is USC and satisfies* $\|F(x)\|\le c(1+\|x\|)$, *then for all* $T>0$:

$$\inf_{d\in\mathscr{S}_T(y_0)}\sup_{0\le t\le T}\left\|D^N(t)-d(t)\right\|\xrightarrow{\mathcal{P}}0,$$

where $\mathscr{S}_T(y_0)$ *is the solution set of the DI* (7.8) *starting in* y_0.

Moreover, if F *is OSL of constant* L, *then*

$$\mathcal{P}\left(\sup_{0\le t\le T}\left\|D^N(t)-d(t)\right\|\ge\left\|D^N(0)-d(0)\right\|e^{LT}+\frac{1}{\sqrt{N}}A_T+(1+c(1+K_T^y))\epsilon\right)\le\frac{C_T+\tau}{N\epsilon^2}$$

where A_T, C_T *and* K_T *are defined as in Theorem 7.7 with* $I(N)=N^{-1}$, *and* $d(\cdot)$ *is the unique solution of the differential inclusion* (7.8).

Proof. Since $\tau<\infty$, the rate of transition of $D^N(.)$ is bounded by $N\tau$. Using uniformization of continuous time Markov chain (see Theorem 1.2), there exists a Poisson process Λ^N of rate $N\tau$ and a discrete time Markov chain $Y^N(.)$ such that $D^N(t)=Y^N(\Lambda^N(t))$ and Y^N and Λ^N are independent. Moreover, for all x and $\ell\in\mathcal{L}$,

$$\begin{aligned}\mathcal{P}\left(Y^N(k+1)=x+\frac{\ell}{N}|Y^N(k)=x\right) &= \frac{1}{\tau}\beta_\ell(x),\\ \mathcal{P}\left(Y^N(k+1)=x|Y^N(k)=x\right) &= 1-\frac{1}{\tau}\sum_{\ell\in\mathcal{L}}\beta_\ell(x).\end{aligned}$$

For all $y\in\mathbb{R}^d$, the drift of $Y^N(.)$ is $\mathbb{E}\left(Y^N(k+1)-Y^N(k)|Y^N(k)=y\right)=(N\tau)^{-1}f(y)$. Moreover, $\mathbb{E}\left(\left\|Y^N(k+1)-Y^N(k)-f(y)\right\|^2|Y^N(k)=y\right)\le(N\tau)^{-2}\sum_{\ell\in\mathcal{L}}\|\ell\|^2\beta_l(y)<b$. Therefore, $Y^N(k)$ satisfies the conditions of Theorem 7.5. Moreover, F also satisfies the conditions of Theorem 7.5 which shows that $\inf_{y\in\mathscr{S}_T(y_0)}\sup_{t\le T}\left\|Y^N(tN)-y(t)\right\|=0$. When F satisfies the OSL condition of Theorem 7.7, we further get the result with explicit bounds.

As Λ^N is a Poisson process of rate $N\tau$, $\left|\Lambda^N(t)-tN\tau\right|^2$ is a sub-martingale and by Doob's inequality, $\mathcal{P}\left(\sup_{t\le T}\left|\Lambda^N(t)-tN\tau\right|\ge N\tau\epsilon\right)\le\mathbb{E}\left(\left|\Lambda^N(T)-TN\tau\right|^2\right)/(N\tau\epsilon)^2=(TN\tau)/(N\tau\epsilon)^2=T/(N\tau\epsilon^2)$. If y is a solution of the DI (7.8) on $[0;T]$, for all $t,s\in[0,T]$, $\|y(t)-y(s)\|\le c(1+K_T^y)\,|t-s|$ where K_T^y is defined in Lemma 7.11. This shows that if y is a solution of the differential inclusion, with probability greater than $T/(N\tau\epsilon^2)$, we have:

$$\begin{aligned}\left\|D^N(t)-y(t)\right\| &= \left\|Y^N(\Lambda^N(t))-y(t)\right\|\\ &\le \left\|Y^N(\Lambda^N(t))-y\left(\frac{\Lambda^N(t)}{N\tau}\right)\right\|+\left\|y\left(\frac{\Lambda^N(t)}{N\tau}\right)-y(t)\right\|\\ &\le \left\|Y^N(\Lambda^N(t))-y\left(\frac{\Lambda^N(t)}{N\tau}\right)\right\|+c(1+K_T^y)\epsilon.\end{aligned}$$

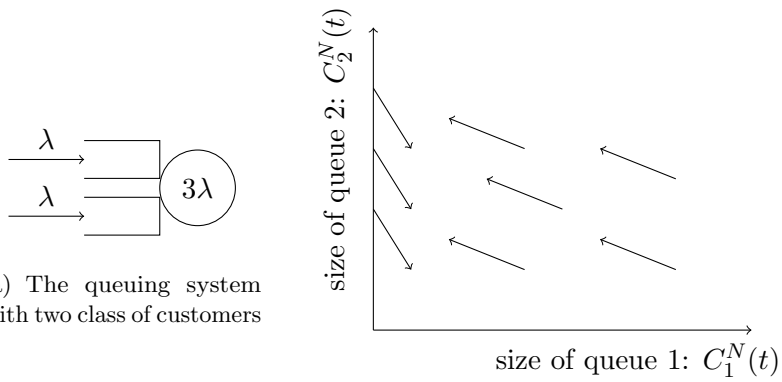
which concludes both parts of the theorem. □

7.5. Examples

7.5.1. Fluid limit of a system of parallel queues

Fluid limits are very popular for studying the dynamics of flows in a queuing network [56]. In this section, we show that our framework can be applied directly to prove the convergence of a queuing network to its corresponding fluid limit. Moreover, the differential inclusion approach allows us to compute the limiting dynamics as a direct application of Theorem 7.8.

We consider a simple model of a system composed by one server. There are two classes of customers: customers with high priority (HP) and customers with low priority (LP). High priority customers and low priority customer enter the system with rate λ (independently). If there are one or more HP customers in the queue, the server serves HP customers at rate 3λ. Only when there is no HP customer in the HP queue, can the server serve LP customers (with rate 3λ). Let $C_1(t)$ and $C_2(t)$ are the numbers of high priority and low priority customers at time t. This model is depicted in Figure 7.1(a) where each queue corresponds to a class of customers. The fluid limit of this system corresponds to the limiting behavior of the system when the initial state is scaled by a factor going to infinity.



(a) The queuing system with two class of customers

(b) Drift of the system

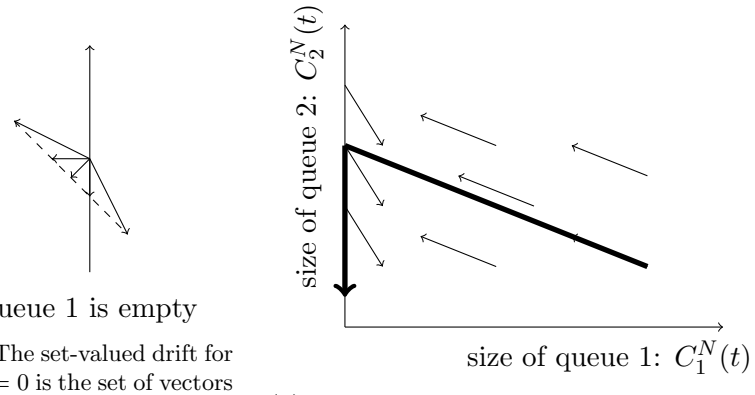
Figure 7.1.: System and the corresponding drift

This model fits in our density dependent population process framework by considering a system with 0 object and only a shared resource $C^N = (C_1^N, C_2^N)$. At time 0, let us define an arbitrary initial state with, say, $4N$ high priority customers and N low priority customers. Thus $C^N(0) \stackrel{\text{def}}{=} (4, 1)$. It should be clear that the expected variation of $C^N(t)$ is $f^N(c_1, c_2) = \frac{1}{N}(-2\lambda, \lambda)$ if $C_1^N(t) > 0$ and $f^N(c_1, c_2) = \frac{1}{N}(\lambda, -2\lambda)$ if $C^N(t) = 0$. Therefore, the intensity of the system is $I(N) = \frac{1}{N}$ and the limit of the drift is $f(c_1, c_2) = Nf^N(c_1, c_2)$ and is depicted in Figure 7.1(b).

As shown on Figure 7.1(b), the drift is constant for all $C_1^N > 0$ but is discontinuous for $C_1^N = 0$. Because of this discontinuity, there is no absolutely continuous function c such that $\dot{c}(t) = f(c)$ almost everywhere: the axis $c_1 = 0$ both attracts the trajectories from $c_1 > 0$ and repulses the trajectories starting from $c_1 = 0$.

However, the intuition tells us that when $C_1^N(t) = 0$, $C_1^N(t)$ should remain close to 0 and that the server should serve in average λ HP customers and 2λ LP customers. To

show that this intuition is correct, let us compute the set-valued function F corresponding to f (defined as in Equation (7.10)). For $c_1 > 0$, $F(c)$ is single-valued and $F(c) = \{f(c)\}$. For $c_1 = 0$, $F(c)$ is the convex hull of the vectors $\{(p, -2p), (-2p, p)\}$ and corresponds to the dashed line of Figure 7.2(a).



(a) The set-valued drift for $C_1 = 0$ is the set of vectors displayed by the dashed line.

(b) Unique solution of the differential inclusion.

Figure 7.2.: Convex hull of the drift at $C_1 = 0$ and unique solution of the fluid limit.

It should be clear that the model satisfies all the hypothesis of Theorem 7.5 and that the differential inclusion has a unique solution, depicted in Figure 7.2(b). Although this result can be shown directly, our framework provide an easy way to construct the fluid limit and prove the convergence of the original process.

7.5.2. Best Response in Rock-Paper-Scissors

The second example is taken from game theory and shows that the best response dynamics typically leads to a non-smooth limit system. In this case the limiting system will be USC and will have a unique solution but it will not satisfy a one-sided Lipschitz condition.

We consider a set of N individuals playing the game of rock-paper-scissors. The state of one individual can either be rock, paper or scissors. At each time step, 2 players are chosen at random. The first player cannot change its choice while the second can decide to play one of the three choices. The goal of the second player is to beat player one (the rule of the game is rock beats scissors, scissors beat paper and paper beats rock).

The best response of player two is the following. If there is a majority of rock (resp. paper or scissors), the second player should play paper (resp. scissor or rock). The corresponding limiting dynamics is drawn on Figure 7.3. The drift is not Lipschitz on the frontiers and the limit dynamics is non-smooth. The differential inclusion is USC but not OSL. However, for all initial conditions, the solution of the differential inclusion is unique and has at most one cusp point before converging in finite time to its unique attractor $(1/3, 1/3, 1/3)$, at constant speed.

7.5.3. Volunteer Computing

Here, we consider a model of a volunteer computing system, such as BOINC `http://boinc.berkeley.edu/`. The system is made of a single buffer and N desktop machines,

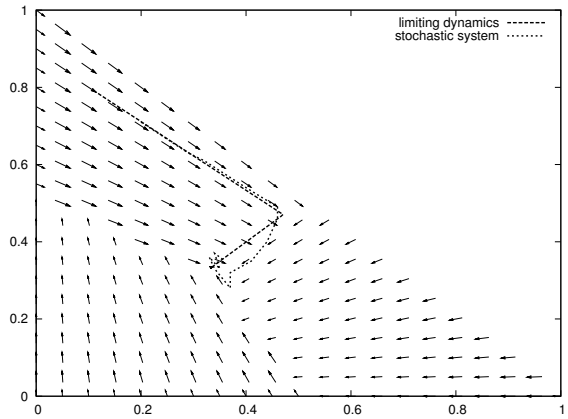


Figure 7.3.: (Vector field for the best response dynamics. The first (second) coordinate M_1 (M_2) is the proportion of Rocks (Paper).The proportion of Scissors is $M_3 = 1 - M_1 - M_2$. Two trajectories starting in $M(0) = (.8, .1, .1)$ are shown. One corresponding to a stochastic system with $N = 10$ players and one for the deterministic limit.

offered by their owners (volunteers), that serve the jobs of this buffer. However, as soon the owner of a processor wants to use it, she preempts it and the processor becomes unavailable for the computing system. As for the incoming jobs, they are assumed to arrive in the buffer according to a Poisson process at rate λ. These kinds of systems are often called push/pull models: The distributed applications *push* jobs to a central server that stores them in a buffer and whenever a processor becomes available, it *pulls* a job from the buffer and executes it.

Such systems fit our density dependent population process framework. The context $C(t)$ represents the size of the buffer while the N objects represent both the applications sending jobs and the hosts executing them. The state of a host is its availability and its idleness (whether it is executing a job or not). The non-smooth part of the dynamics comes from the buffer size. When $C(t) > 0$, if a host asks for a job, it gets it with probability one while when $C(t) = 0$, a host asking for a job will get nothing. In that case, one can show that this dynamics satisfies the conditions of Theorem 7.8 that can be used to study the limiting behavior of the system when the number of hosts and applications grows.

In the simplest case, the intensity of the system is $I(N) = 1/N$ and an application sends a job to the system at rate λ while jobs are completed at rate μ by each server. To represent the communication delays, every host gets jobs at rate γ. It becomes unavailable with rate p_u, and available with rate p_a if $C(t) > 0$ and 0 otherwise. If b, a, u denote respectively the proportion of busy, available and unavailable hosts, the limiting system is described by a DI:

$$\begin{aligned}
\dot{b}(t) &= -\mu b(t) + \gamma a(t)\mathbf{1}_{C(t)>0} \\
\dot{a}(t) &= \mu(t)b(t) + p_a u(t) - \gamma a(t)\mathbf{1}_{C(t)>0} \\
\dot{u}(t) &= -p_a u(t) + p_u a(t) \\
\dot{C}(t) &= -\gamma a(t)\mathbf{1}_{C(t)>0} + \lambda \mathbf{1}_{C(t)<C_{\max}}.
\end{aligned}$$

The formal DI is obtained by replacing $\gamma a(t)\mathbf{1}_{C(t)>0}$ by the singleton $\{\gamma a(t)\}$ if $C(t) > 0$ and the interval $[0; \gamma a(t)]$ when $C(t) = 0$.

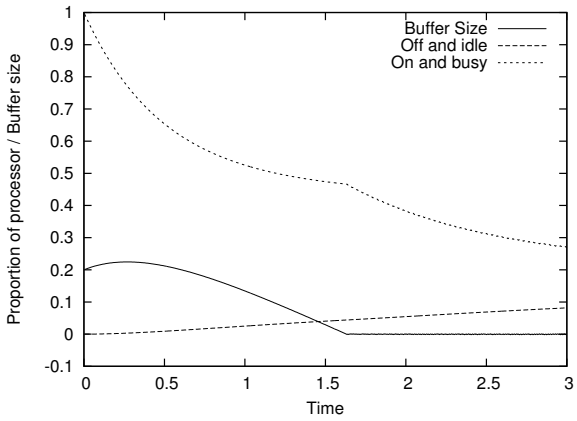


Figure 7.4.: Limit dynamics of a volunteer computing system. A non-differentiable point occurs when the buffer becomes empty.

At time $t = 0$, we consider that the size of the buffer is $C(0) = 0.2$ and that all processors are available and are serving a job. The behavior of the system is represented in Figure 7.4. One can see that there is a point of non-differentiability in the behavior of the system when the size of the buffer reaches 0.

7.5.4. Volunteer Computing: Day and Night Scenario

We now consider a similar model as the previous one except that the processors follow a day and night behavior. We consider that some of the processors are turned off at night. Therefore, the availability is larger during the day (between 7am and 5pm) than at night.

The limiting dynamics can be represented by a differential inclusion that depends (not continuously) on time:

$$\frac{\partial y(t)}{\partial t} \in F(y(t), t). \tag{7.12}$$

In the previous theorems, we assumed that the function F was time-invariant. There are three ways to tackle this problem. The first one is to adapt the proofs to the time dependent case. An other idea that can used here is the fact that for all finite interval of time, there is only a finite number of discontinuity points (7am, 5pm,...), and to apply the convergence results on a first sub-interval $[0; 7am]$. Using the fact that $Y^N(7) \to y(7)$, the convergence holds on $[7am, 5pm]$, and so forth. Yet another solution is to write $Z(t) \stackrel{\text{def}}{=} (Y(t), t)$. Then the differential inclusion (7.12) can be written:

$$\frac{\partial z(t)}{\partial t} \in (F(z(t)), 1)\,. \tag{7.13}$$

The fact that F is not continuous in t (and therefore in $z(t)$) is not a problem in our differential inclusion setting. It should be clear that in that case there is still a unique solution to the differential inclusion (7.13).

On Figure 7.5 we can observe two kinds of non-differentiable points. The first ones are the points representing the change from day time to night time. The other ones occur when the buffer becomes empty. The small oscillations of the buffer size around 0 and of the proportion of busy processors are just numerical integration artefacts (typical of

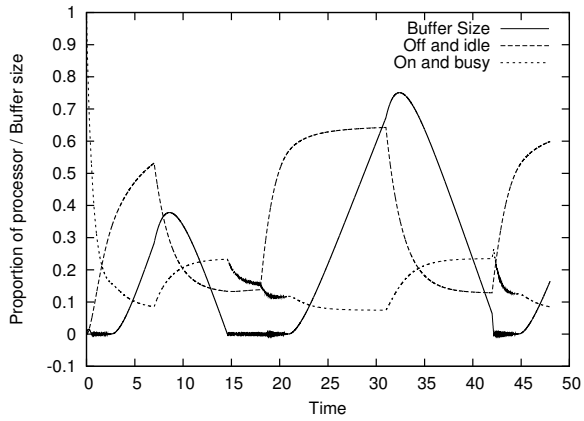


Figure 7.5.: Limit dynamics for volunteer computing under day and night availability.

numerical integration of differential inclusions). In both cases the exact solutions do not exhibit these jumps.

7.5.5. Join the Shortest Queue in Volunteer Computing

This example is similar to the one of Section 7.5.3 but with two identical time-homogeneous volunteer systems. Each time a job arrives, it is routed to the system with the smallest number of jobs. Here, the routing of jobs introduces a new cause of non-smoothness: there is a threshold in the dynamics of the system when both backlogs are equal.

Figure 7.6 shows the behavior of the limit differential inclusion. Once again, the limit behavior is unique once the initial condition is given.

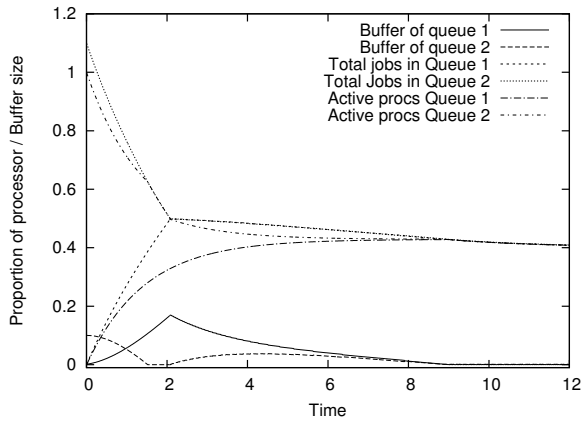


Figure 7.6.: Join the shortest queue for volunteer computing

As expected, new non-differential points appear when both buffers are equal.

7.5.6. OSL dynamics: energy-aware distributed computing system

The previous examples have dynamics for which the drift function did not satisfy an OSL condition. However, OSL conditions are commonly assumed in the non-smooth system litterature [48, 107]. In this section, we recall sufficient condition for a dynamical system

to be OSL and provide some examples of dynamics that are OSL. Recall that a set-valued function F is OSL with constant $L \geq 0$ if for all $y, \bar{y} \in \mathbb{R}^d$ and $u \in F(y), \bar{u} \in F(\bar{y})$: $\langle y - \bar{y}, u - \bar{u} \rangle \leq L \|y - \bar{y}\|^2$.

Let us first examine the example of Section 7.5.3. Let $y = (b, a, u, C)$ and $\bar{y} = (\bar{b}, a, u, 0)$, then $\langle y - \bar{y}, f(y) - f(\bar{(y)}) \rangle = -\mu \left|b - \bar{b}\right|^2 + \gamma a(b - \bar{b}) - \gamma a$. When $b - \bar{b}$ is small enough and positive, this expression is of order $a(b - \bar{b})$ which is greater than $L \|y - \bar{y}\|^2 = L(\left|b - \bar{b}\right|^2 + C^2)$. In fact, there are two types of non-smoothness in these equations. The first one is that the dynamics of C depends not continuously on C but in a OSL way (see below). The second type of discontinuity is that the dynamics of b depends non-smoothly on C. This latter discontinuity always leads to a term of order $(b - \bar{b})$ which is greater than $L \left\|b - \bar{b}\right\|^2$ whenever $b - \bar{b}$ is small enough. This is a general problem for the applicability of OSL: whenever the derivative of one coordinate depends non-smoothly on a other coordinate, the dynamics is never OSL. This is actually the case in all the examples of Sections 7.5.1 to 7.5.5.

Sufficient conditions to prove OSL

The next lemma links conditions on the drift function f to its set-valued counterpart.

Lemma 7.9. *Let $f : \mathbb{R}^d \to \mathbb{R}^d$ be a single valued function and F be the convex set-valued function associated to f, defined by*

$$F(y) = \bigcap_{\epsilon > 0} \overline{\text{conv}} \left\{ f(z) : \|z - y\| \leq \epsilon \right\}.$$

where $\overline{\text{conv}}$ denotes the convex hull of a set. Then:

(i) F is OSL with constant L iff *f is OSL with constant L.*

(ii) If f is Lipschitz of constant L, then F is OSL of constant L.

Sketch of proof. (i) – The proof of *(i)* is straightforward from the definition of OSL. The main idea is to use the fact that since R^d is a d dimensional space and by the definition of F, any $u \in F(y)$ can be written as the limit of a convex combination of d points $u_1 \dots u_d$ such that $u_i = f(y_i)$ and $\|y - y_i\| \leq \epsilon$. Using a similar combination for $\bar{u} \in F(\bar{y})$ and playing with the coefficient of the convex combination leads to *(i)*.

(ii) – The proof of *(ii)* comes from the Cauchy-Schwartz inequality. If f is Lipschitz with constant L, then for all $y, \bar{y} \in \mathbb{R}^d$: $\langle y - \bar{y}, f(y) - f(\bar{y}) \rangle \leq \|y - \bar{y}\| \|f(y) - f(\bar{y})\| \leq L \|y - \bar{y}\|^2$. This shows that f is OSL. By *(i)*, this implies that F is OSL. □

In order to show that a particular dynamic is OSL, we can also use the fact that the sum of two OSL functions is OSL. However, unlike in the case of Lipschitz functions, the product or the composition of two OSL functions is not necessarily OSL.

Lemma 7.10. *If F_1 and F_2 are two OSL with constant L_1 and L_2, then $F_1 + F_2$ is OSL with constant $L_1 + L_2$, where $F_1 + F_2$ is defined by:*

$$(F_1 + F_2)(y) = \{u_1 + u_2 : u_1 \in F_1(y), u_2 \in F_2(y)\}.$$

The basic example of a OSL function is the fluid limit of a single M/M/1 queue. Let us assume that jobs arrive in the system at rate $N\lambda > 0$ and are served by a server with rate

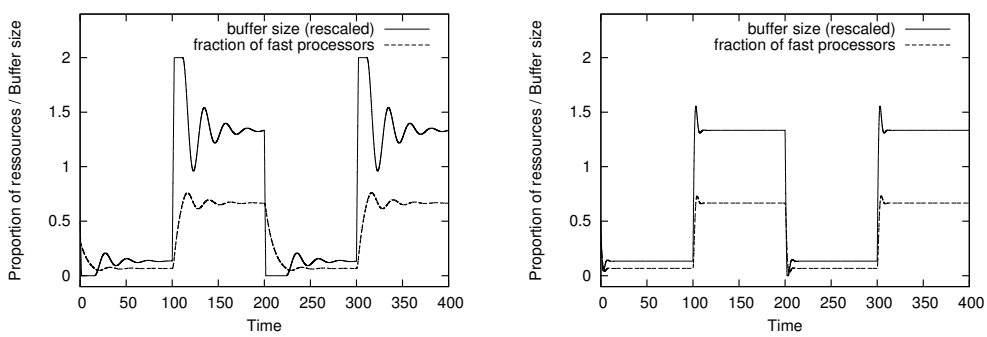
$N\mu > 0$. Let $Y^N(t)$ be the number of packet at time t in the queue rescaled by $\frac{1}{N}$ and let assume that $Y^N(0) = x$. The drift of the system is $f(y) = \lambda - \mu$ if $y > 0$ and $f(0) = \lambda$ (that can be extended to $f(y) = \lambda$ for $y < 0$). The set-valued drift corresponding to f is $F(y) = \{\lambda - \mu\}$ if $y > 0$, $F(y) = \{\lambda\}$ if $y < 0$ and $F(0) = [\lambda - \mu; \lambda]$. This function F satisfies a OSL condition with constant 0. Indeed, let $y, y' \in \mathbb{R}$. If y and y' are both positive or both negative, $f(y) - f(y') = 0$. If $y > 0$ and $y' \leq 0$,

$$\langle y - y', f(y) - f(y') \rangle = \langle y - y', -\mu \rangle \leq 0 \leq 0 \left\| y - y' \right\|^2 . \tag{7.14}$$

Therefore, the dynamics of a M/M/1 queue is OSL with constant 0 and one can apply Theorem 7.7 with $K_T = \lambda$, $I(N) = 1/N$, $J(N) = 0$ and any $L > 0$. Note that Theorem 7.7 requires $L > 0$ because of a term in $e^{2LT}/\sqrt{L}$ that comes from $I(N) \sum_{i=1}^{T/I(N)} (1 + 4LI(N))^i \leq \exp 4LT/(4L)$. When $L = 0$, this term can be replaced by T. The dependence in T of the constants A_T, B_T and C_T is e^{2LT} when $L > 0$ and T when $L = 0$.

Let us now consider a more sophisticated computing model made of a single buffer and N computing resources. Jobs arrive at rate $N\lambda$ in the buffer (where λ may vary) and $C^N(t)$ is the number of jobs in the buffer rescaled by N. The buffer size is upper bounded by $C_{\max}$. In order to optimize energy consumption, the computing resource can compute at rate μ_1 or $\mu_2 > \mu_1$. If there is a proportion $M_1^N(t) = m_1$ resources at speed μ_1 and $M_2^N(t) = m_2 = 1 - m_1$ resources at speed μ_2, the drift of $C^N(t)$ is $f_c(c, m_1, m_2) = \lambda - m_1\mu_1 - m_2\mu_2$ if $0 < c < C_{\max}$, $g(0, m_1, m_2) = \lambda$ and $f_c(C_{\max}, m_1, m_2) = -m_1\mu_1 - m_2\mu_2$. A computation similar to (7.14) shows that f_c is OSL.

The computing resources can not communicate and each resource updates it speed independently of the others as a function of the load $C^N(t)$ of the system. To do so, each resource measures the size of the queue $C^N(t)$ at rate γ. If the size of the buffer is c and the speed of the resource is μ_i, the speed is set to *fast* (*i.e.* μ_2) with probability $p_i(c)$ (and is set to *slow* with probability $1 - p_i(c)$). Therefore, the drift of $M_1^N(t)$ is $f_1(c, m_1, m_2) = \gamma(-m_1 p_1(c) + m_2(1 - p_2(c)))$ and the drift of $M_2^N(t)$ is $f_2(c, m_1, m_2) = -f_1(c, m_1, m_2)$. If $b \mapsto p_i(c)$ is Lipschitz, the functions f_1 and f_2 are both Lipschitz. This shows that the drift of the whole system (which is the sum of f_c, f_1 and f_2) is a OSL function.



(a) $\gamma = .1$ (b) $\gamma = 1$

Figure 7.7.: Differential inclusion dynamics of the energy-aware distributed systems for $\gamma = .1$ and $\gamma = 1$ and $C_{\max} = 2$. The y axis represents proportion of resources that are fast (*i.e.* $m_2(t)$) or the buffer size, depending on the curve.

In practical situation, measuring c and changing power of a computing resource is

costly in terms of time and energy. Thus, the choice of γ is a compromise between the reactivity of the system and the time and energy spent to adapt the power to the load. Figure 7.7 shows two examples of the evolution of the differential inclusion dynamics for two values of γ: $\gamma = .1$ and $\gamma = 1$. The parameters of the system are $\mu_1 = .5$, $\mu_2 = 2$ and $C_{\max} = 2$. On each figure, $p_i(c) = c/C_{\max}$. When $t \in [0; 100]$ or $t \in [200; 300]$, we set $\lambda = .6$ while for $t \in [100; 200]$ or $t \in [300; 400]$, $\lambda = 1.5$. Figure 7.7(a) shows that when γ is small, a brutal change of the arrival rate leads to a long period of oscillation of the system. When going from $\lambda = .6$ to 1.5, some jobs are lost at the beginning. Then, the system overreacts before stabilizing. When $\gamma = 1$ as in Figure 7.7(b), these oscillations have disappeared, except for a small burst at the beginning.

7.6. Proofs of Theorem 7.5 and Theorem 7.7

The idea of the proofs is to write the evolution of the system as a stochastic approximation algorithm. The state of the system at time $k+1$ can be written as

$$\begin{aligned} Y^N(k+1) &= Y^N(k) + I(N)f(Y^N(k)) + \left(f^N(Y^N(k)) - I(N)f(Y^N(k))\right) \\ &\quad + \left(Y^N(k+1) - Y^N(k) - f^N(Y^N(k))\right), \end{aligned} \tag{7.15}$$

where f is defined in (7.5). This equation can be seen as an Euler discretization of the DI (7.8) plus two error terms:

- $E^N(k) \stackrel{\text{def}}{=} f^N(Y^N(k))/I(N) - f(Y^N(k))$ which by Equation (7.5) is such that $\left\|E^N(k)\right\| \leq J(N)$;
- $U^N(k+1) \stackrel{\text{def}}{=} \left(Y^N(k+1) - Y^N(k) - f^N(Y^N(k))\right)/I(N)$ which by Equation (7.4) is such that $\mathbb{E}\left(U^N_{k+1} \middle| \mathcal{F}^N_k\right) = 0$ where $\mathcal{F}^N_k$ denotes the filtration associated with the process $(Y^N(k))_k$. Moreover, by Equation (7.7), $\mathbb{E}\left(\left\|U^N_{k+1}\right\|^2 \middle| \mathcal{F}^N_k\right) \leq b$.

Using this notation, and the definition of $F(y)$ by Equation (7.10), Equation 7.15 can be rewritten

$$Y^N(k+1) \in Y^N(k) + I(N)\left(F(Y^N(k)) + E^N(k) + U^N(k+1)\right). \tag{7.16}$$

Equation (7.16) is called a *stochastic approximation* with *constant step size* associated to the DI (7.8). The term *constant step size* comes from the fact that $I(N)$ does not vary with time. Both proofs of Theorem 7.5 and Theorem 7.7 are based on the convergence of such stochastic approximation (7.16) as N goes to infinity.

7.6.1. Proof of Theorem 7.5

This proof is inspired by classical proofs on differential inclusions, for example the proof of Theorem 2.2.1 of [100]. It is somewhat similar to the proof of Theorem 4.2 of [19] although there are two main differences. First, we focus on *constant step size* and are interested in the convergence over a finite time-horizon. Second, we do not need any assumption on the boundedness of the stochastic process since this is a consequence of the hypothesis in the finite time-horizon.

The idea of the proof is to show that for all sub-sequence of $\bar{Y}^N$, there exists a sub-sequence $\bar{Y}^{\sigma(N)}$ (of this sub-sequence) such that $d(\bar{Y}^{\sigma(N)}, \mathcal{D}_T(y_0)) \xrightarrow{\text{a.s.}} 0$. In all that

follows, let $\bar{Y}^{\sigma(N)}$ be a sub-sequence of $\bar{Y}^N$. In order to simplify the notations and because we will take several sub-sequences of sub-sequences, we omit the σ in the notation and we denote all sub-sequences $\bar{Y}^N$. In the first part of the proof, we consider the problem from a probabilistic point of view in order to make sure that the random part of the process goes almost surely to 0. Then we consider the problem from a trajectorial point of view using analytic arguments.

Developing the recurrence (7.16), the value of $Y^N(k+1)$ is equal to:

$$Y^N(k+1) = Y^N(0) + \sum_{i=0}^{k} I(N) f(Y^N(i)) + I(N) \sum_{i=0}^{k} E^N(i) + I(N) \sum_{i=0}^{k} U^N(i+1). \quad (7.17)$$

We define two functions $Z^N(t)$, and $V^N(t)$ to be piecewise affine functions such that for all k, $Z^N(kI(N)) = \sum_{i=0}^{k-1} I(N) f(Y^N(i))$ and $V^N(kI(N)) = \sum_{i=0}^{k-1} I(N) U^N(i+1)$. By Kolmogorov's inequality for martingales and since $\mathbb{E}\left(\left\|U^N(k+1)\right\|^2 |\mathcal{F}_k^N\right) \leq b$

$$\mathcal{P}\left(\sup_{0\leq t\leq KI(N)} \left\|V^N(t)\right\| \geq \epsilon\right) = \mathcal{P}\left(\sup_{0\leq k\leq K} \left\|I(N)\sum_{i=0}^{k-1} U^N(i+1)\right\| \geq \epsilon\right) \leq \frac{KI(N)^2}{\epsilon^2} b.$$

Therefore, for all T:

$$\mathcal{P}\left(\sup_{0\leq t\leq T} \left\|V^N(t)\right\| \geq \epsilon\right) \leq \mathcal{P}\left(\sup_{0\leq t\leq \left\lceil \frac{T}{I(N)} \right\rceil I(N)} \left\|V^N(t)\right\| \geq \epsilon\right) \leq \frac{\left\lceil \frac{T}{I(N)} \right\rceil I(N)^2}{\epsilon^2} b. \quad (7.18)$$

This shows that $\sup_{0\leq t\leq T} \left\|V^N(t)\right\|$ converges in probability to 0. Therefore, there exists a sub-sequence of V^N such that $\sup_{t\leq T} \left\|V^N(t)\right\|$ converges almost surely to 0.

We now reason from a trajectorial point of view. Let us now consider a trajectory $\omega \in \Omega$ of the system such that $\sup_{t\leq T} \left\|V^N(t)\right\|$ converges to 0. In particular, this implies that $\left\|V^N(t)\right\|$ is bounded for all N and t: $\sup_{N,0\leq t\leq T} \left\|V^N(t)\right\| \leq d < \infty$. Using (7.17) and since $\|F(y)\| \leq c(1+\|x\|)$, for all $k \leq T/I(N)$, $\left\|Y^N(k+1)\right\|$ can be bounded by:

$$\begin{aligned}
\left\|Y^N(k+1)\right\| &\leq \left\|Y^N(0)\right\| + \sum_{i=0} I(N) c(1+\left\|Y^N(i)\right\|) + \sup_{N,t} \left\|V^N(t)\right\| \\
&\leq \left\|Y^N(0)\right\| + ckI(N) + d + \sum_{i=0}^{k} I(N) \left\|Y^N(i)\right\| \\
&\leq \left(\left\|Y^N(0)\right\| + cT + d\right) \exp\left(cT\right)/c, \qquad (7.19)
\end{aligned}$$

where we used the discrete Grönwall's lemma and the fact that $kI(N) \leq T$.

Once we know that $\sup_{N,0\leq t\leq T} \left\|Y^N(t)\right\|$ is bounded, the rest of the proof can be adapted from classical results on the convergence of the Euler approximation for differential inclusions, see [100] for example. There exists $e > 0$ such that $\sup_{N,0\leq t\leq T} \left\|Y^N(t)\right\| \leq e$. Thus $\left\|f(Y^N(k))\right\| < c(1+e) < \infty$. This shows that the functions Z^N are Lipschitz with constant $c(1+e)$. Thus the sequence of functions $(Z^N)_N$ are equicontinuous and bounded. Therefore by the Arzéla-Ascoli theorem, for all sub-sequence of $(Z^N)_N$, there exists a sub-sequence that converges to some $z : [0;T] \to \mathbb{R}^d$. In the following, we will show that z is a solution of (7.3) which shows that $d(Z^N, \mathscr{S}_T(y_0)) \to 0$. As $\left\|Z^N - Y^N\right\| = \left\|V^N\right\| \to 0$, this implies that $d(Y^N, \mathscr{S}_T(y_0)) \to 0$. To prove this, we will construct a function φ such that:

(i) for all t: $z(t) = z(0) + \int_0^t \varphi(s)ds$;

(ii) for almost every t: $\varphi(t) \in F(z(t))$.

Let $\varphi^N(t)$ be a step function, constant on the intervals $[kI(N), (k+1)I(N))$ and such that for $t = kI(N)$, $\varphi^N(t) = f(Y^N(k))$. Therefore, the sequence φ^N is bounded in $L_2([0;T], \mathbb{R}^d)$. Thus, there exists a sub-sequence of φ^N converging weakly in L_2 to a function φ. Since L_2 is a reflexive space, if a sequence of functions φ^N converges to φ, this means that for all function v in L_2, there exists a sub-sequence of φ^N such that $\langle v, \varphi^N \rangle \to \langle v, \varphi \rangle$. Let $\xi \in \mathbb{R}^d$ and $t \in [0;T]$. Let the function v be defined by $v(s) \stackrel{\text{def}}{=} \xi$ for $s < t$ and $v(s) \stackrel{\text{def}}{=} 0$ for $t \geq s$. Since φ^N converges weakly to φ and $Z^N(t) \to z(t)$, we have:

$$\begin{aligned}
\langle Z^N(t), \xi \rangle &\to \langle z(t), \xi \rangle \\
\langle Z^N(t), \xi \rangle &= \langle Z^N(0), \xi \rangle + \langle \int_0^t \varphi^N(s)ds, \xi \rangle \\
&= \langle Z^N(0), \xi \rangle + \langle \varphi^N, v \rangle \\
&\to \langle z(0), \xi \rangle + \langle \varphi, v \rangle \\
&= \langle z(0) + \int_0^t \varphi(s)ds, \xi \rangle.
\end{aligned}$$

As this is true for all $\xi \in \mathbb{R}^d$, this shows that z is absolutely continuous: $z(t) = \int_0^t \varphi(s)ds$.

It remains to show that for a.e. t, $\phi(t) \in F(z(t))$. Let t^N denotes the greater multiple of $I(N)$ less than t ($t^N \stackrel{\text{def}}{=} \lfloor t/I(N) \rfloor I(N)$). Using that $f(Y^N(k)) \leq c(1+e)$ and that z^N converges uniformly to y, for all $\delta > 0$, there exists N_0 such that $N \geq N_0$ implies $\left\| z(t) - Y^N(t^N) \right\| \leq \delta$. Defining $F^\delta(y) \stackrel{\text{def}}{=} \cup_{z:\|z-y\| \leq \delta} F(z)$, this shows that $\phi^N(t) \in F^\delta(z(t))$. Since F is convex and z bounded, $\{\alpha \in L^2 : \alpha(t) \in F^\delta(z(t))\}$ is convex and closed. This shows that this set is weakly closed (see [128], Theorem 3.12). Therefore, for all t, $\phi(t) \in F^\delta(t)$. As this is true for all δ and F is USC, this shows that for a.e. t, $\phi(t) \in \cap_{\delta > 0} F^\delta(t) = F(z(t))$. Thus, z is a solution of the DI.

□

7.6.2. Proof of Theorem 7.7

This proof is based on two Lemmas. Lemma 7.12 shows the rate of convergence of an Euler scheme without error to the solution of a differential inclusion. Lemma 7.13) shows that the gap between the deterministic Euler scheme and the stochastic approximation can be bounded in probability with explicit bounds. The combination of the two directly implies Theorem 7.7.

We define by y_k^N the Euler scheme associated with the differential inclusion (7.3) with step $I(N)$ starting in $Y^N(0)$. That is:

$$\begin{aligned} y_0^N &= Y^N(0) \\ y_{k+1}^N &= y_k^N + I(N)\left(f(y_k^N) + E^N(k)\right) \end{aligned} \tag{7.20}$$

for some $f(y_k^N) \in F(y_k^N)$. Again, $y^N(t)$ denotes the piecewise interpolation of y_k^N where the time is scaled by a factor $I(N)$. More precisely, $y^N(t)$ is a piecewise affine function such that for $k \in \mathbb{N}$: $y^N(kI(N)) = y_k^N$. Finally, we denote by $y(t)$ a solution of the DI (7.3) (which is unique if the DI is OSL by Proposition 7.4).

We first start by a technical lemma that bounds the growth of the functions.

Lemma 7.11. *If F is USC and OSL with constant L and if there exists c such that for all x, $\|F(x)\| \leq c(1+\|x\|)$, then: if y_k^N is the piecewise interpolation of the Euler scheme (7.20) with $\left\|E^N(k)\right\| \leq J(N)$ and $y(.)$ the unique solution of the DI starting in y_0, then there exist constants $K_T^y, K_T^{y^N}$ and $K_T^{Y^N}$ such that $\sup_{0\leq t\leq T}\|y(t)\| \leq K_T^y$, $\sup_{0\leq t\leq T}\left\|y^N(t)\right\| \leq K_T^{y^N}$ a.s. and $\sup_{0\leq t\leq T}\sqrt{\mathbb{E}\left(\left\|f(\bar{Y}^N(t))\right\|^2\right)} \leq K_T^{Y^N}$ where*

$$\begin{aligned} K_T^y &\stackrel{\text{def}}{=} (\|y(0)\| + c)\frac{e^{cT}}{c} \\ K_T^{y^N} &\stackrel{\text{def}}{=} (\left\|y_0^N\right\| + (c + J(N))T)\frac{e^{cT}}{c} \\ K_T^{Y^N} &\stackrel{\text{def}}{=} (\left\|Y_0^N\right\| + (c + J(N))T + \sqrt{bT})\frac{e^{cT}}{c}. \end{aligned}$$

Proof. By the definition of y_k^N of Equation (7.20) and since $\|F(y)\| \leq c(1+\|y\|)$, we have

$$\begin{aligned} \left\|y_{k+1}^N\right\| &= \left\|y_0^N + I(N)\sum_{i=0}^{k}\left(f(y_i^N) + E^N(i)\right)\right\| \\ &\leq \left\|y_0^N\right\| + I(N)\sum_{i=0}^{k}\left(c(1+\left\|y_i^N\right\|) + J(N)\right) \\ &= \left\|y_0^N\right\| + (k+1)I(N)(c+J(N)) + I(N)c\sum_{i=0}^{k}\left\|y_i^N\right\|. \end{aligned}$$

Therefore, for all $k \leq T/I(N)$, by the discrete Grönwall's lemma, $\left\|y_k^N\right\| \leq K_T^{y^N}$ with $K_T^{y^N} \stackrel{\text{def}}{=} (\left\|y_0^N\right\| + (c+J(N))T)e^{cT}/c$. Similarly, $\|y(t)\| \leq \|y(0)\| + \int_0^t c(1+\|y(s)\|)ds \leq (\|y(0)\| + ct)e^{cT}/c$.

Similarly to the computation for y_k^N, and focusing on $Y^N(k)$ leads to:

$$\begin{aligned}\left\|Y^N(k+1)\right\| &\leq \left\|Y^N(0)\right\| + (k+1)I(N)(c+J(N)) \\ &\quad +I(N)\left\|\sum_{i=0}^{k} U^N(i+1)\right\| + I(N)c\sum_{i=0}^{k}\left\|y^N(i)\right\|\end{aligned}$$

which shows that $\left\|Y^N(k)\right\| \leq (\left\|Y_0^N\right\| + (c+J(N))T + I(N)\left\|\sum_{i=0}^{k} U^N(i+1)\right\|)e^{cT}/c$. Since $\mathbb{E}\left(U^N(k+1)\middle|\mathcal{F}_k^N\right) = 0$,

$$\mathbb{E}\left(I(N)\left\|\sum_{i=0}^{k} U^N(i+1)\right\|^2\right) \leq I(N)\sum_{i=0}^{k}\mathbb{E}\left(\left\|U^N(i+1)\right\|^2\middle|\mathcal{F}_k^N\right) \leq bkI(N).$$

Using the Jensen inequality on $x \mapsto (1+\sqrt{x})^2$, this implies that for all $k \leq T/I(N)$

$$\mathbb{E}\left(\left\|Y^N(k)\right\|^2\right) \leq (\left\|Y_0^N\right\| + (c+J(N))T + \sqrt{bT})^2\frac{e^{2cT}}{c^2}.$$

□

Lemma 7.12. *If F is USC and OSL with constant L and if there exists c such that for all y, $\|F(y)\| \leq c(1+\|y\|)$, then: if $y^N(.)$ is the piecewise interpolation of the Euler scheme (7.20) with $\left\|E^N(k)\right\| \leq J(N)$ and $y(.)$ the unique solution of the DI starting in y_0, then*

$$\sup_{0\leq t\leq T}\left\|y^N(t)-y(t)\right\| \leq e^{LT}\left\|y_0^N - y_0\right\| + \sqrt{I(N)}A_T + \sqrt{J(N)}B_T$$

where

$$\begin{aligned}A_T &\stackrel{\text{def}}{=} e^{2LT}\sqrt{\frac{I(N)}{6}c^2(1+K) + \frac{K}{2L}(c(1+K)+J(N))}; \\ B_T &\stackrel{\text{def}}{=} \frac{\sqrt{K}e^{2LT}}{\sqrt{L}},\end{aligned}$$

and $K_T \stackrel{\text{def}}{=} \max\{K_T^y, K_T^{y^N}\}$ with K_T^y and $K_T^{y^N}$ defined as in Lemma 7.11.

If y_0^N is bounded, then for T fixed, A_T is bounded for all N. This shows that the convergence of $y^N(t)$ to $y(t)$ is of order $O(\sqrt{I(N)})$. In fact, the rate of convergence of the Euler's method can be improved if the solution of the differential inclusion satisfies some properties. For example, if the solution has a finite number of point at which it is not differentiable the rate of convergence is $O(I(N))$, see for example [107], Theorem 4.1.

Proof. Let us fix N and let us call $t_i = iI(N)$. Let $r(t) \stackrel{\text{def}}{=} \left\|y^N(t)-y(t)\right\|^2$. Since $y^N(.)$ is piecewise affine and $y(.)$ is absolutely continuous, r is absolutely continuous and differentiable for almost every t. Therefore, for a.e. t such that $t_i \leq t < t_{i+1}$, we have

$$\begin{aligned}\dot{r}(t) &= 2\langle y^N(t)-y(t), f(y^N(t_i)) + E^N(i) - f(y(t))\rangle \\ &= 2\langle y^N(t_i)-y(t), f(y^N(t_i)) - f(y(t))\rangle + 2\langle y^N(t)-y^N(t_i), f(y^N(t_i)) - f(y(t))\rangle \\ &\quad +2\langle y^N(t)-y(t), E^N(i)\rangle \\ &= 2\langle y^N(t_i)-y(t), f(y^N(t_i)) - f(y(t))\rangle \qquad (7.21) \\ &\quad +2\langle (t-t_i)\left(f(y^N(t)) + E^N(t_i)\right), f(y^N(t_i)) - f(y(t))\rangle \qquad (7.22) \\ &\quad +2\langle y^N(t)-y(t), E^N(i)\rangle. \qquad (7.23)\end{aligned}$$

Equation (7.21) can be bounded by the OSL condition and is less than $2L\left\|y^N(t_i)-y(t)\right\|^2$ which is equal to $2L\left\|y^N(t_i)-y(t_i)+y(t_i)-y(t)\right\|^2 \leq 4L(r(t_i)+\|y(t_i)-y(t)\|^2)$. Moreover, since y is a solution of the DI (7.8), $\|y(t)-y(t_i)\| = \left\|\int_{t_i}^t f(y(s))ds\right\|^2 \leq ((t-t_i)c(1+K))^2$. By Cauchy–Schwarz inequality, the two terms of Equation (7.23) are bounded by $2(t-t_i)(c(1+K)+J(N))2K+4KJ(N)$.

Since $r(t)$ is absolutely continuous, $r(t_{i+1}) = r(t_i) + \int_{t_i}^{t_{i+1}} \dot{r}(t)dt$. Thus

$$\begin{aligned} r(t_{i+1}) \quad \leq \quad & r(t_i)\left(1+4LI(N)\right) + \frac{4L}{3}I(N)^3c^2(1+K)^2 \\ & +I(N)^2(c(1+K)+J(N))2K+4KJ(N)I(N). \end{aligned}$$

Using that $(1+4LI(N))^i \leq e^{4Lt_i}$ and that $\sum_{j=1}^i (1+4LI(N))^j = ((1+4LI(N))^{i+1}-1)/(4LI(N)) \leq e^{4Lt_i}/(4LI(N))$, we get by a direct induction that:

$$r(t_i) \leq e^{4Lt_i}r(0)+\frac{e^{4Lt_i}}{4L}\left(\frac{4LI(N)^2}{3}c^2(1+K)^2+2KI(N)(c(1+K)+J(N))+4KJ(N)\right). \tag{7.24}$$

This quantity is maximized for $t_i = T$. Therefore, we have

$$\sup_{0\leq t\leq T} r(t) \leq e^{4LT}r(0)+I(N)e^{4LT}\left(\frac{I(N)}{6}c^2(1+K)+\frac{K}{2L}(c(1+K)+J(N))\right)+J(N)\frac{Ke^{4LT}}{L}.$$

This gives the results using $\sqrt{a+b+c} \leq \sqrt{a}+\sqrt{b}+\sqrt{c}$. □

The following lemma states the convergence in probability of $Y^N(k)$ to the Euler scheme $y^N(k)$ with explicit probabilistic bounds. Together with Lemma 7.12, it shows that Y^N converges to the solution of the solution of (7.8) with explicit bounds (*i.e.* Theorem 7.7).

Lemma 7.13. *If F is USC and OSL with constant L and if there exists c such that for all y, $\|F(y)\| \leq c(1+\|y\|)$, then if $y^N(.)$ is the affine interpolation of the Euler scheme (7.20) and $Y^N(.)$ the affine interpolation of the stochastic approximation (7.16) with*

- $\left\|E^N(k)\right\| \leq J(N)$
- $\mathbb{E}\left(U_{k+1}^N \middle| \mathcal{F}_k^N\right) = 0$ *and* $\mathbb{E}\left(\left\|U_{k+1}^N\right\|^2 \middle| \mathcal{F}_k^N\right) \leq b$

then for all T and all $\epsilon > 0$,

$$\mathcal{P}\left(\sup_{0\leq t\leq T}\left|\bar{Y}^N(t)-y^N(t)\right| \geq \epsilon\right) \leq \frac{I(N)}{\epsilon^2}\frac{e^{2LT}}{2L}\left(2c^2\left((1+K_T^{y^N})^2+(1+K_T^{Y^N})^2\right)+b\right).$$

where $K_T^{y^N}$ and $K_T^{Y^N}$ are defined as in Lemma 7.11.

In particular, this shows that $\lim_{N\to\infty}\mathcal{P}\left(\sup_{0\leq t\leq T}\left|\bar{Y}^N(t)-y^N(t)\right| \geq \epsilon\right)$ goes to 0 with rate $I(N)$.

Proof. Using the equality $\|x+y+z\|^2 = \|x\|^2 + 2(\langle x,y\rangle + \langle x,z\rangle + \langle y,z\rangle) + \|y\|^2 + \|z\|^2$ and the definitions of $Y^N(k+1)$ and y^N_{k+1} (Eq. (7.16) and (7.20)), $\left\|Y^N(k+1)-y^N_{k+1}\right\|^2$ can be rewritten

$$\begin{aligned}
\left\|Y^N(k+1)-y^N_{k+1}\right\|^2 &= \left\|Y^N(k) - y^N_k + I(N)\left(f(Y^N(k)) - f(y^N_k) + U^N(k+1)\right)\right\|^2 \\
&= \left\|Y^N(k) - y^N_k\right\|^2 \\
&\quad +2I(N)\langle Y^N(k) - y^N_k, f(Y^N(k)) - f(y^N_k)\rangle \qquad (7.25)\\
&\quad +2I(N)\langle Y^N(k) - y^N_k + f(Y^N(k)) - f(y^N_k), U^N(k+1)\rangle \\
&\quad +I(N)^2\left\|f(Y^N(k)) - f(y^N_k)\right\|^2 + I(N)^2\left\|U^N(k+1)\right\|^2 \\
&\leq (1+2LI(N))\left\|Y^N(k) - y^N_k\right\|^2 \qquad (7.26)\\
&\quad +I(N)^2\left(\left\|f(Y^N(k)) - f(y^N_k)\right\|^2 + \left\|U^N(k+1)\right\|^2\right) \qquad (7.27)\\
&\quad +2I(N)\langle Y^N(k)-y^N_k + f(Y^N(k))-f(y^N_k), U^N(k+1)\rangle. \qquad (7.28)
\end{aligned}$$

where we use the OSL to go from (7.25) to (7.26).

Let us define a real valued sequence $W^N(k)$ by $W^N(0) = 0$ and for all $k \geq 0$:

$$\begin{aligned}
W^N(k+1) = (1+2LI(N))W^N(k) + I(N)^2\left(\left\|f(Y^N(k)) - f(y^N_k)\right\|^2 + \left\|U^N(k+1)\right\|^2\right) \\
+2I(N)\langle Y^N(k) - y^N_k + f(Y^N(k)) - f(y^N_k), U^N(k+1)\rangle.
\end{aligned}$$

Because of Equation (7.26)-(7.27)-(7.28), for all $k \geq 0$, $0 \leq \left\|Y^N(k+1)-y^N_{k+1}\right\|^2 \leq W^N(k+1)$ (almost surely). Moreover, since $\mathbb{E}\left(U^N(k+1)\middle|\mathcal{F}^N_k\right) = 0$, the expectation of (7.28) knowing $\mathcal{F}^N_k$ is also 0. This shows that:

$$\begin{aligned}
\mathbb{E}\left(W^N(k+1)\middle|\mathcal{F}^N_k\right) &= (1+2LI(N))W^N(k) + I(N)^2\Big(\left\|f(Y^N(k)) - f(y^N_k)\right\|^2 \\
&\qquad +\mathbb{E}\left(\left\|U^N(k+1)\right\|^2\middle|\mathcal{F}^N_k\right)\Big) \qquad (7.29)\\
&\geq W^N(k).
\end{aligned}$$

This shows that $W^N(k)$ is a positive sub-martingale. Therefore, by Doob's inequality, for $\epsilon > 0$, $\mathcal{P}\left(\sup_{0\leq i\leq k} W^N(i) \geq \epsilon^2\right) \leq \mathbb{E}\left(W^N(k)\right)/\epsilon^2$. Moreover, using Equation (7.29), the definition $W^N(0) = 0$, Lemma 7.11 and the hypothesis (7.7) on $\mathbb{E}\left(\left\|U(k+1)\right\|^2\middle|\mathcal{F}^N_k\right)$, $\mathbb{E}\left(W^N(k+1)\right)$ can be bounded by:

$$\begin{aligned}
\mathbb{E}\left(W^N(k+1)\right) =& \mathbb{E}\left(\mathbb{E}\left(W^N(k+1)\middle|\mathcal{F}^N_k\right)\right) \\
\leq& (1+2LI(N))\mathbb{E}\left(W^N(k)\right) + I(N)^2\left(\mathbb{E}\left(\left\|f(Y^N(k))-f(y^N(k))\right\|^2\right) + b\right) \\
\leq& (1+2LI(N))^{k+1}W^N(0) \\
&+ I(N)^2\sum_{i=0}^{k}(1+2LI(N))^i\left(2c^2\left((1+K^{y^N}_T)^2 + (1+K^{Y^N}_T)^2\right) + b\right) \\
=& I(N)\frac{(1+2LI(N))^{k+1}-1}{2L}\left(2c^2\left((1+K^{y^N}_T)^2 + (1+K^{Y^N}_T)^2\right) + b\right).
\end{aligned}$$

where $K^{y^N}_T$ and $K^{Y^N}_T$ are defined in Lemma 7.11. Note that we use Jensen's inequality on $x \mapsto (1+\sqrt{x})^2$ to show that $\mathbb{E}\left(\left\|f(Y^N(k))\right\|^2\right) \leq c^2(1+K^{Y^N}_T)^2$.

Taking the value for $k = T/I(N)$, using the preceding remarks and the fact that for $a > 0$, $(1+a)^i \leq e^{ai}$, we have

$$\begin{aligned}
\mathcal{P}\left(\sup_{0\leq t\leq T} \left\|\bar{Y}^N(t) - y^N(t)\right\| \geq \epsilon\right) &\leq \mathcal{P}\left(\sup_{0\leq i\leq T/I(N)} W^N(i) \geq \epsilon^2\right) \\
&\leq \frac{I(N)}{\epsilon^2}\frac{e^{2LT}}{2L}\left(2c^2\left((1+K_T^{y^N})^2 + (1+K_T^{Y^N})^2\right) + b\right).
\end{aligned}$$

□

Part III.

Other Contributions

Chapter 8.

Infinite Labeled Trees: from Rational to Sturmian Trees

Abstract of this chapter – This chapter studies infinite unordered d-ary trees with nodes labeled by $\{0, 1\}$. We introduce the notions of rational and Sturmian trees along with the definitions of (strongly) balanced trees and mechanical trees, and study the relations among them.

In particular, we show that (strongly) balanced trees exist and coincide with mechanical trees in the irrational case, providing an effective construction. Such trees also have a minimal factor complexity, hence are Sturmian. We also give several examples illustrating the inclusion relations between these classes of trees.

Résumé du chapitre – Dans ce chapitre, nous étudions certaines classes d'arbres non orientés étiquetés par $\{0, 1\}$. Nous introduisons les notions d'arbres rationnels et sturmiens ainsi que les définitions d'arbres (fortement) équilibrés et étudions les relations entre ces différentes classes.

En particulier, nous montrons l'existence d'arbres (fortement) balancés et que ceux ci correspondent aux arbres équilibrés dans le cas irrationnel. De plus ces arbres sont de complexité minimale et donc sturmiens. Nous présentons de nombreux exemples illustrant les relations entre les différentes notions présentées.

8.1. Introduction

Let us consider the following question: how to distribute ones and zeros over an infinite sequence $w = (w_n)_{n\in\mathbb{N}}$ such that the ones (and the zeros) are spread as evenly as possible. In a more formal way, the sequence w is *balanced* if the number of ones in a factor $w_i, \ldots, w_{i+\ell-1}$ of length ℓ, does not vary by more than 1, for all i and all ℓ. Such sequences exist and are called *Sturmian words* when they are not periodic.

Sturmian words are quite fascinating binary sequences: they have many different characterizations formulated in terms coming from as many mathematical frameworks, in which they always prove very useful. For example, Sturmian words have a geometric description as digitalized straight lines and as such have been used in computer visualization (see [97] for a review). They can also be defined with an arithmetic characterization using a repetitive rotation on a torus or continued fraction decompositions. From a combinatorial point of view, yet another characterization of Sturmian words is based on the balance between ones and zeros in all factors, as mentioned before. They are also used in symbolic dynamic system theory because they are aperiodic words with minimal factor complexity or because they have palindromic properties. Most of these equivalences have been known since the seminal work in [116]. More recently, Sturmian sequences have also been used for optimization purposes: they are extreme points of multimodular functions [85, 5, 76] and this has applications is scheduling theory [75].

Since then, there have been several constructions of generalized Sturmian words in the literature.

The first one concerns words over more than two letters. Billiard sequences in hypercubes extent the torus definition of Sturmian sequences while episturmian sequences [29] extend the palindromic characterization of Sturmian words, however, the other characterizations of Sturmian words are lost in both cases. Another extension is to two dimensions. A complete characterization of two-dimensional non-periodic sequences with minimal complexity is given in [50], here again the alternative characterizations are lost. Yet another extension of Sturmian words concerns discrete planes. Indeed, several characterizations of Sturmian lines can be extended to discrete planes. There exists interesting relations between multidimensional continued fraction decomposition of the normal direction of an hyperplane and the patterns of its discretization. These relations mimic what happens for Sturmian sequences, [63]. Finally, another generalization is to ordered trees [30], where Sturmian trees are defined as infinite binary automata such that the number of factors (subtrees) of size n is $n + 1$. The other characterizations of Sturmian words are lost once more.

The aim of this chapter is to do the same for unordered trees where things work better in the sense that several extensions coincide. We introduced in [67] a new type of infinite trees: unordered labeled trees, for which the left and right children of each node are not distinguishable and gave a brief presentation of their main properties. Here, we make an exhaustive study of such trees. We show that the balance property (even distribution of the labels over the vertices of the tree) coincides with a characterization of trees using integer parts of affine functions (called mechanicity). Furthermore these strongly balanced trees have a minimal factor complexity. Therefore, they can be seen as a natural extension of Sturmian sequence in more than one aspect. This brings some hope to use them as extreme points for adapted optimization problems.

Our purpose in the chapter is two-fold. The first part of the chapter is dedicated to the study of general unordered infinite trees with binary labels. In section 8.2, we provide

definitions of the main concepts as well as the basic properties of unordered trees with a special focus on the notion of density (the average number of ones) and rationality. Section 8.3 is dedicated to the study of the rational trees.

The second part of the chapter investigates balanced unordered trees and their properties. In particular, we show that strongly balanced trees (defined in section 8.4) are mechanical (so that they have a density and all labels can be constructed in almost constant time). Furthermore their factor complexity is minimal among all non-periodic trees. We also investigate the general shape of strongly balanced rational trees (section 8.5). We show that there essentially exists a unique strongly balanced tree with a given rational density. Also, once a strongly balanced tree is given, its density is easy to compute and we provide an efficient algorithm with polynomial complexity to test whether a rational tree is strongly balanced. Finally, Section 8.6 presents several examples and counter examples that illustrate the different notions presented in the chapter.

8.2. Infinite Trees

8.2.1. Ordered Infinite Trees or Tree-automata

Ordered infinite trees (also called tree-automata here) have been studied in [55, 30]. Ordered infinite trees are automata with an infinite number of states. An automata is a tree-automaton if it has one initial state and each state has a uniform in-degree equal to one (except for the initial state, whose in-degree is 0) and a uniform out-degree d with labels $a_1, \cdots, a_d$ on the arcs. Every node v is labeled by $\ell(v) = 1$ (resp. 0) if it is final (resp. non-final).

The language accepted by the tree-automaton $\mathcal{T}$ is a subset of $\mathcal{A}^*$ (where the alphabet $\mathcal{A} = \{a_1, \dots a_d\}$) and is denoted by $\mathcal{L}(\mathcal{T})$. Thus, a word w in the free monoid $\mathcal{A}^*$ corresponds to a node in $\mathcal{T}$, and a word w in $\mathcal{L}(\mathcal{T})$ corresponds to a node in $\mathcal{T}$ with label 1. Conversely, a unique tree-automaton can be associated to any subset L of $\mathcal{A}^*$, by labeling by one the nodes corresponding to the words in L.

Classically for automata, a family of equivalence relations can be defined over the nodes of tree $\mathcal{T}$: $v \sim_0 u$ if $\ell(v) = \ell(u)$, $v \sim_{n+1} u$ if $v \sim_n u$ and for all i, the ith child of u, ua_i and the ith child of v, va_i satisfy $ua_i \sim_n va_i$. By definition of $\sim_n$, $u \sim_n v$ if and only if the subtree rooted in u of height n is the same as the subtree rooted in v of height n.

$\mathcal{L}(\mathcal{T})$ is recognized by its minimal deterministic automaton (possibly infinite), say $A(\mathcal{T})$. Actually, $A(\mathcal{T})$ can be obtained from the tree $\mathcal{T}$ by merging all the states in the tree in the same equivalence classe of $\sim_n$ for all n.

An example is given in Figure 8.1 where the infinite tree-automaton and the minimal automaton recognizing all the prefixes of the Fibonacci[1] word over the alphabet $\{a, b\}$ is given together with the corresponding minimal automaton (which has an infinite number of states).

The number of distinct subtrees of height n in $\mathcal{T}$ is called the complexity $P(n)$, of $\mathcal{T}$. $P(n)$ is the number of equivalence classes of $\sim_n$. If $P(k) \leq k$ for at least one k, then it can be shown [30] that the complexity $P(n)$ is bounded by k. This implies that the minimal automaton $A(\mathcal{T})$ has less than k states. The tree is therefore rational, since it recognizes a rational language.

[1] the Fibonacci word is the limit of the sequence $f_{n+2} = f_n f_{n+1}$ with $f_0 = a$ and $f_1 = b$, see [109] for more details.

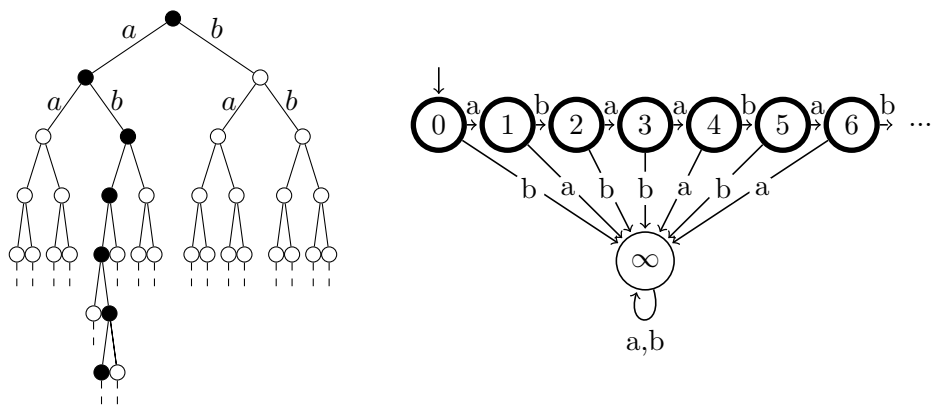


Figure 8.1.: The tree-automaton recognizing the Fibonacci word f and the corresponding minimal automaton. The states of this later are $0, 1 \ldots, \infty$. The final nodes are filled in black. There is a transition between nodes i and $i+1$ labeled by the ith letter of f and one between nodes i and ∞ labeled by the opposite of this ith letter.

If a tree-automaton $\mathcal{T}$ is such that $P(n) = n + 1$ for all n, then it has a minimal complexity among all non-rational trees. Such trees have been shown to exist and are called Sturmian in [30] by analogy with the factor complexity definition of Sturmian words (Figure 8.1 gives an example). In [30] several classes of Sturmian tree-automata are presented. However such trees are not balanced and no constructive definition (as the mechanical construction for words) is known.

8.2.2. Unordered Trees and Minimal Graph

In this chapter, we rather consider a different type of trees, namely infinite directed *graphs* with labels 0 or 1 on nodes and with uniform in-degree 1 and out-degree $d \geq 2$. Up to our knowledge, these types of trees have not yet been considered in the literature. The similarities as well as the discrepancies with ordered trees will be discussed all along the chapter.

In such trees, one node is special (with in-degree 0) and is called the root. Also, the children of a node are not ordered. Thus, the main difference with the previous type of trees is the fact that arcs are not labeled. Therefore such trees cannot be bijectively associated with languages.

We define the minimal multigraph (*i.e.* with multiple arcs) $G(T)$, associated with the tree T, mimicking the construction of the minimal automaton for ordered trees. To do that, we first introduce a family of equivalence relations $\equiv_n$ over the nodes of T:

- $v \equiv_0 u$ if u and v have the same label: $\ell(u) = \ell(v)$
- $v \equiv_{n+1} u$ if $v \equiv_n u$ and if there exists a bijection F between the children of v and the children of u such that for all child w of v, $w \equiv_n F(w)$.

Therefore, $v \equiv_n u$ if and only if the subtree with root v of height n is isomorphic to the subtree with root u of height n. By merging the nodes of T when they belong to the same equivalence classe $\equiv_n$ for all n, one gets the minimal multigraph $G(T)$ of the factors of T: all nodes merged in the same vertex of $G(T)$ are roots of the same subtrees, of every

height. In $G(T)$, the node corresponding to the root of T is distinguished. (graphically, this is done by adding an arrow pointing to the node).

An example of an unordered tree T is given in Figure 8.2. Actually, most figures in this chapter will represent binary trees (with out-degree $d = 2$), although all the discussion is carried throughout for arbitrary degrees. The nodes of the associated multigraph $G(T)$ are numbered arbitrarily and nodes with label 1 are displayed with a bold circle. The node corresponding to the root of the tree is pointed by an arrow.

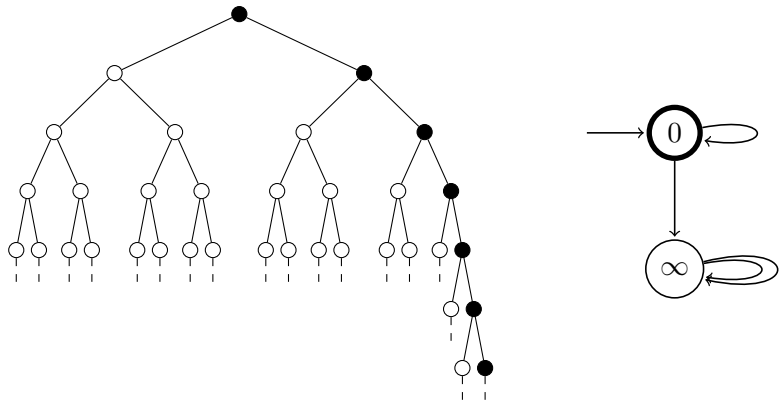


Figure 8.2.: A tree T and the associated minimal multigraph $G(T)$. The label of the black (white) nodes is 1 (0). The arcs are implicitly directed from top to bottom.

There exists a way to associate an ordered tree-automaton $\mathcal{T}$ to a tree T by choosing an order on the children of each node. This can be done by seeing $G(T)$ as an automaton by labeling arcs in $G(T)$ with letters $a_1, \ldots a_d$ in an arbitrary fashion. Conversely, a tree-automaton $\mathcal{T}$ can be converted into a graph T by removing the labels on the arcs. This graph is called the unordered version of $\mathcal{T}$. Figure 8.2 is the unordered version of the tree recognizing the Fibonacci word displayed in Figure 8.1. Note that while the minimal automaton is infinite, the minimal graph $G(T)$ is finite, with only two nodes; one corresponds to the subtree where all labels are 0 and the other one to the subtree with a branch with label 1 everywhere and all the other nodes with label 0 (Figure 8.2).

8.2.3. Irreducibility and periodicity

By analogy with Markov chains, we say that a tree T is *irreducible* if $G(T)$ is strongly connected.

A non-irreducible tree, $G(T)$ is made of *strongly connected components* , inter-connected by an acyclic graph. Also, an irreducible tree T is *periodic* with period p if the greatest common divisor of the lengths of all cycles in $G(T)$ is p. A tree with period 1 is also called *aperiodic*.

8.2.4. Factors, complexity and Sturmian trees

In this chapter, we will study properties of *factors* of infinite trees. For this purpose, we introduce two definitions:

- A *factor of height* n *(and base* 0*, by default)* is a subgraph of T which is a complete subtree of height n. The number of nodes in a factor of height n is denoted by $S(n) \stackrel{\text{def}}{=} \frac{d^n-1}{d-1}$.
- A *factor of height* n *and base* k (with root v), is a subgraph of T which is the subtree of height $k+n$ rooted in v minus the subtree of height k, rooted in v (see Figure 8.3 for an illustration). Such a subgraph is also called a factor of *shape* (n,k) in the following. The number of nodes of a factor of height n and base k is $S(n,k) \stackrel{\text{def}}{=} \frac{d^{n+k}-d^k}{d-1}$.

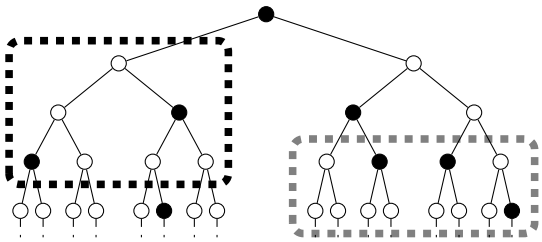


Figure 8.3.: Example of factors of a tree. On the left a factor of height 3 (and base 0) is surrounded in black. On the right is a factor of height 2 and base 2.

Similarly to what has been done for words or ordered trees, the factor *complexity* $\mathcal{P}_T(n)$ of a tree T is the number of distinct factors of height n and base 0.

The complexity of a tree $\mathcal{P}_T(n)$ can be bounded by the total number of ways to label trees of height n and degree d, say A_n. It should be clear that $A_1 = 2$ (a node can be labeled 0 or 1) and that $A_{n+1} = 2M(A_n, d)$ where $M(x,y)$ is the number of multisets with y elements taken from a set with x elements. Therefore using binomial coefficients,

$$A_{n+1} = 2\binom{A_n + d - 1}{A_n - 1}.$$

This is a polynomial recurrence equation of degree d. A change of variable, $u_n = \log A_n + \frac{1}{d-1}\log\frac{2}{d!}$ yields a new recurrence equation $u_{n+1} = (d+\varepsilon_n)u_n$ where $\epsilon_n = o(1)$. This implies that $A_n = \phi^{d^n + o(d^n)}$ for some ϕ with $1 < \phi < 2$.

As for lower bounds on the complexity of a tree, it will be shown in Section 8.3 that trees such that $\mathcal{P}_T(n) \leq n$ for at least one n are rational, *i.e.* have a bounded number of factors of any size (this implies that its minimal multigraph is finite). Therefore, trees T such that $G(T)$ is infinite and with a minimal complexity should satisfy $\mathcal{P}_T(n) = n+1$. These trees will be called *Sturmian trees* by analogy with words. This definition is close to the one of [30] for ordered trees. It is not difficult to exhibit such trees. For example, for any Sturmian word w, a d-ary tree such that all nodes on level i have label w_i is Sturmian.

Another more interesting example is the Dyck tree, represented on Figure 8.4. This tree is the unordered version of the tree-automata recognizing the Dyck language (language generated by the context-free grammar $S \to aSbS|\epsilon$), introduced in [30] and it is not hard to see that this tree is Sturmian. For that, consider the graph $G(T)$ associated with the Dyck tree T, also displayed in Figure 8.4. There are two factors of height 1 in T: those with a root labeled 1 (all associated with node 0 in $G(T)$) and those with a root labeled 0 (associated with nodes $\infty, 1, 2, \cdots$ in $G(T)$). This corresponds to the equivalence classes

for $\equiv_1$. All factors of height n with a root associated to nodes $\infty, n, n+1, n+2, \ldots$ have labels equal to 0: no path of length n in $G(T)$ reaches the only node with label 1, namely node 0. The factors of height n starting with a root i of $G(T)$ with $0 \leq i < n$ are distinct: their first node with label 1 is at level $i+1$. In other words, the equivalence classes for $\equiv_n$ are $\{\infty, n, n+1, \ldots\}, \{0\}, \{1\}, \ldots, \{n-1\}$. The number of distinct factors of height n is $n+1$.

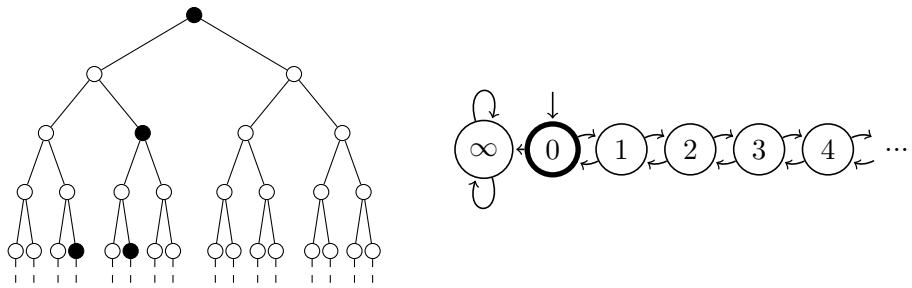


Figure 8.4.: The Dyck tree and its minimal graph.

8.2.5. Density

The density of a tree T is meant to capture the proportion of ones in the tree. For any node v and any height $n \geq 0$, the proportion of nodes with label 1 in the factor of height n with root v is denoted by $d_v(n)$. Let r be the root of the tree T. If the following limits exist, they define four notions of density:

- The *rooted density* of the tree is the limit of the density of the subtrees of the root r:
$$\lim_{n\to\infty} d_r(n).$$
- The *rooted average density* of the tree is the Cesaro limit of these densities:
$$\lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} d_r(i).$$
- The *density* of the tree is α if it has an identical rooted density for all nodes:
$$\forall v : \lim_{n\to\infty} d_v(n) = \alpha.$$
- The *average density* of the tree is α if it has an identical rooted average density for all nodes:
$$\forall v : \lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} d_v(i) = \alpha.$$

From the definitions, the following implications are direct: if a tree admits a density, then it admits an average density. In turn, a tree with an average density also has a rooted average density. Also, a tree with a density has a rooted density. See Figure 8.5 for some examples. These examples will be further developed in the following section on rational trees.

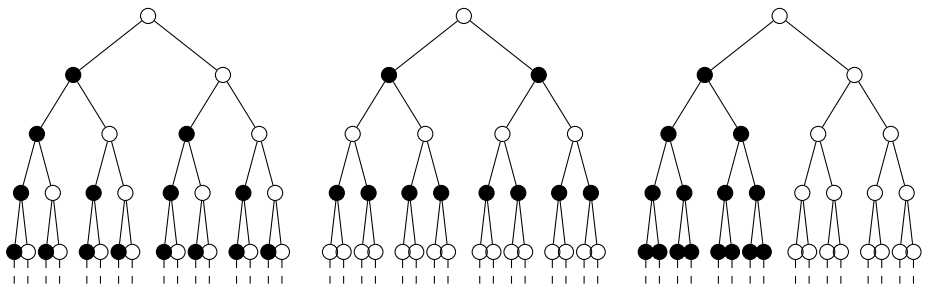


Figure 8.5.: The first tree has a density of 1/2, the second one an average density equal to 1/2 but no density. The last one has a rooted density 1/2 but no average density.

8.2.6. Ordered trees vs unordered trees

One of the main features of ordered labeled trees is the fact that there exists a bijection between ordered trees with finite degree and languages over finite alphabets, so that ordered trees benefit from the power of language theory formalism [55]. However, as shown in [30], the generalization of binary words to ordered trees with binary labels is surprisingly difficult. One of these difficulties comes from the combinatorial explosion due to the distinction of left and right children replacing a unique successor for words.

This is the basis for the introduction of unordered trees, where the unique successor is replaced more naturally by a pair (or more) of successors and indeed more properties of words can be generalized. Let us anticipate with the results shown in the following sections. First, the notion of density of a tree is very natural for unordered trees (definitions in Section 8.2.5) and leads to an algorithmic construction of balanced trees using a mechanical process based on the density and the *phase* of the root (Proposition 8.8). This can be viewed as a natural extension to trees of the mechanical construction of balanced words.

Also, the two main results of the chapter, namely the fact that the strongly balanced trees are the mechanical trees and have minimal complexity (Theorems 8.9 and 8.15) as well as the fact that rational strongly balanced trees are unique once the density is given (Theorem 8.16) are specific to unordered trees and generalize nicely the corresponding results for words.

8.3. Rational Trees

Definition 8.1. *A tree T is* rational *if the associated minimal multigraph $G(T)$ is finite.*

An example of a rational tree T is displayed in Figure 8.6 together with its multigraph $G(T)$. Note that this tree is not irreducible. It has one final strongly connected component of period 2 (corresponding to the alternating subtrees starting with ones and zeros, displayed as a left child) and a strongly connected component with period one (corresponding to the subtree with all its labels equal to one (displayed as a right child).

It is also possible to characterize rational trees using their complexity $\mathcal{P}(n)$, as shown in the following theorem.

Theorem 8.2. *The following statements are equivalent*

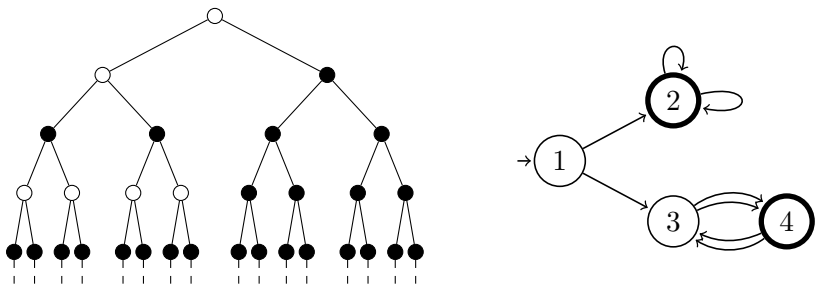


Figure 8.6.: A rational tree made of two distinct subtrees and its associated multigraph

1. *the tree T is rational;*

2. *there exists n such that $\mathcal{P}(n) \leq n$;*

3. *there exists n such that $\mathcal{P}(n) = \mathcal{P}(n+1)$;*

4. *There exists B such that for all n, $\mathcal{P}(n) \leq B$.*

Proof. The proof of this result is similar to the proof for words.
1 implies 2: If $G(T)$ is finite, then the number of factors of height n in T is smaller than the size of $G(T)$, therefore, there exists n such that $\mathcal{P}(n) \leq n$.
2 implies 3: Since $\mathcal{P}(1) = 2$ and $\mathcal{P}(n) \leq n$ and since $\mathcal{P}$ is non-decreasing with n, there exists $1 < k < n$ such that $\mathcal{P}(k) = \mathcal{P}(k+1)$.
3 implies 4: If $\mathcal{P}(n) = \mathcal{P}(n+1) = p$ then let us call by $A_1^n, \dots A_p^n$ all the distinct factors of height n in T. Since $P(n+1) = p$, each A_i^n is prolonged in a unique way into a tree of height $n+1$, called A_i^{n+1}. Now, each subtree A_i^{n+1} is composed of a root and d factors of height n, in the set $\{A_1^n, \dots A_p^n\}$. In turn, they are all prolonged into trees of height n in a unique way. Therefore, $\mathcal{P}(n+2) = p$. By a direct induction, $\mathcal{P}(k) = p$ for all $k \geq n$.
4 implies 1: If the number of factors of height n is smaller than B for all n, then this means that the number of equivalence classes for $\equiv_n$ is smaller than B for all n, this means that $G(T)$ has less than B nodes. □

8.3.1. Density of rational trees

Let T be a rational tree and let $G(T)$ be its minimal multigraph. The nodes of $G(T)$ are numbered $v_1 \cdots, v_K$, with v_1 corresponding to the root of T.

$G(T)$ can be seen as the transition kernel of a Markov chain by considering each arc of $G(T)$ as a transition with probability $1/d$. If $G(T)$ is irreducible then the Markov chain admits a unique stationary measure π on its nodes. The density of T and the stationary measure π are related by the following theorem.

Theorem 8.3. *Let T be an irreducible rational tree with a minimal multigraph $G(T)$ with K nodes. Let $\ell = (\ell_1, \dots \ell_K)$ be the labels of the nodes of $G(T)$ and let $\pi = (\pi_1, \dots, \pi_K)$ be the stationary measure of the Markov chain over the nodes of $G(T)$.*
If T is aperiodic, then T admits a density $\alpha = \pi\ell^t$ (where ℓ^t stands for the transpose of ℓ).
If T is periodic with period p then T admits an average density $\alpha = \pi\ell^t$.

Proof. Let V be the Markov chain corresponding to $G(T)$. Since $G(T)$ is irreducible, V admits a unique stationary measure , say $\pi = (\pi_1, \ldots, \pi_K)$. Let us call P the kernel of this Markov chain: $P_{i,j} = a/d$ if there are a arcs in $G(T)$ from v_i to v_j.

Now, let us consider all the paths of length n in T, starting from an arbitrary node v_i. By construction of $G(T)$, the number of paths that end up in the node v_i of $G(T)$ is given by the vector $d^n e_i P^n$, where e_i is the vector with all its coordinates equal to 0 except the ith coordinate, equal to 1.

The number of ones in a subtree of height n starting in v_i is $h_n(v_i) = e_i \sum_{k=0}^{n-1} d^k P^k \ell^t$.

Let us first consider the case where P is aperiodic. We denote by Π the matrix with all its lines equal to the stationary measure, π and by D_k the matrix $P^k - \Pi$. When P is aperiodic, then $\lim_{k\to\infty} ||D_k||_1 = 0$. Therefore, for all $k > n, ||D_k||_1 < \epsilon_n \to 0$.

Then the density of ones $d_{2n}(v_i) = \frac{d-1}{d^{2n}-1} h_{2n}(v_i)$ can be estimated by splitting the factors of height $2n$ into a factor of height n at the root and d^n factors of height n. One gets

$$\begin{aligned} d_{2n}(v_i) &= \frac{d-1}{d^{2n}-1} e_i \sum_{k=1}^{n} d^k P^k \ell^t + \frac{d-1}{d^{2n}-1} e_i \sum_{k=n+1}^{2n-1} d^k P^k \ell^t, \\ &= \frac{d-1}{d^{2n}-1} e_i (\sum_{k=1}^{n} d^k P^k + \sum_{k=n+1}^{2n-1} d^k D_k + \sum_{k=n+1}^{2n} d^k \Pi) \ell^t. \end{aligned}$$

When n goes to infinity, the first term goes to 0 because $e_i \sum_{k=1}^{n} d^k P^k \ell^t \leq d^{n+1}$. As for the second term $\frac{d-1}{d^{2n}-1} e_i \sum_{k=n+1}^{2n-1} d^k D_k \ell^t \leq \frac{1}{d^{2n}-1} d^{2n} \epsilon_n$. This goes to 0 when n goes to infinity.

As for the last term, $\frac{d-1}{d^{2n}-1} e_i \sum_{k=n+1}^{2n-1} d^k \Pi \ell^t = \frac{1}{d^{2n}-1}(d^{2n} - d^{n+2})(e_i \Pi)\ell^t$ goes to $\pi \ell^t$ when n goes to infinity.

The same holds by computing the density of factors of shape $2n+1$ by splitting them into the first $n+1$ levels and the last n levels. This shows that the rooted density of all the trees in T is the same, equal to $\pi \ell^t$.

Let us now consider the case when the tree is periodic with period p. In that case, the kernel of p steps of the Markov chain can be put under the form

$$P^p = \left[\begin{array}{c|c|c|c} P_1 & 0 & \cdots & 0 \\ \hline 0 & P_2 & \ddots & 0 \\ \hline \vdots & \ddots & \ddots & \vdots \\ \hline 0 & 0 & \ldots & P_m \end{array} \right].$$

The submatrices $P_1 \ldots P_m$ are the kernels of aperiodic chains defined on a partition $S_1 \ldots S_m$ of the nodes of $G(T)$. Let us denote by $\alpha_1 \ldots \alpha_m$ the densities of the factors of height np, starting in $S_1 \ldots S_m$, respectively (they exist because this has just been proved for aperiodic trees).

Starting from a node v the average density of a tree of size $n = pq_n + r_n$, $r_n < p$ is

$$\frac{1}{n} \sum_{k=0}^{n} d_k(v) = \frac{1}{pq_n + r_n} (\sum_{a=0}^{q_n} \sum_{b=0}^{p} d_{ap+b+r}(v) + \frac{1}{pq_n + r_n} \sum_{k=0}^{r_n} d_k(v)).$$

The first term goes to $(\alpha_1 + \ldots + \alpha_m)/m$ while the second term goes to zero, when n goes to infinity, independently of the root. Finally, $(\alpha_1 + \ldots + \alpha_m)/m = (\pi'_1 \ell^t_1 + \cdots + \pi'_m \ell^t_m)/m =$

$\pi\ell^t$ where $\pi'_1, \cdots, \pi'_m$ are the stationary probability for the kernels $P_1, \ldots, P_m$ and $\ell_1, \ldots \ell_m$ are the vectors of the labels in $S_1 \ldots S_m$. □

An example illustrating the computation of the density of an aperiodic irreducible rational tree is given in Figure 8.7. The stationary measure of the Markov chain is $\pi = (2/9, 3/9, 4/9)$. Therefore, the density is $\alpha = 2/9\ell_1 + 3/9\ell_2 + 4/9\ell_3 = 4/9$.

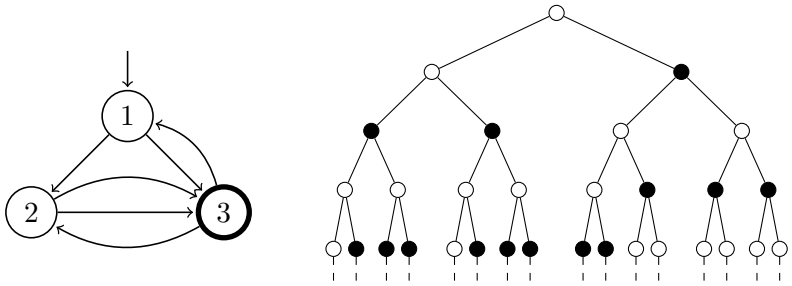


Figure 8.7.: An irreducible aperiodic rational tree and its minimal graph. The stationary probabilities over the associated Markov chain are $\pi = (2/9, 3/9, 4/9)$. The density of the tree is $\alpha = 4/9$.

As for the reducible case, it should be easy to see that a rational tree may have different (average) densities for some of its subtrees (this is the case for the rightmost tree in Figure 8.5). Therefore, a reducible tree does not have a density nor an average density in general.

Let us call $S_1 \cdots S_m$ the final strongly connected components of $G(T)$. Let $\alpha_1 \ldots \alpha_m$ be the average densities of the components $S_1 \cdots S_m$ respectively. Finally, let $r = (r_1 \ldots r_m)$ be the probability of reaching the components $S_1 \cdots S_m$ starting from the root v_1, in the Markov chain associated with $G(T)$. Then, the following theorem holds.

Theorem 8.4. *A rational tree has a rooted average density* $\alpha = (\alpha_1, \ldots \alpha_m)r^t$.

Proof. If P is reducible, P can be decomposed into

$$P = \left[\begin{array}{c|c|c|c} Q & K_1 & \cdots & K_m \\ \hline 0 & P_1 & \cdots & 0 \\ \hline \vdots & \cdots & \ddots & \vdots \\ \hline 0 & 0 & \cdots & P_m \end{array}\right] \text{ and } P^n = \left[\begin{array}{c|c|c|c} Q^n & K'_1 & \cdots & K'_m \\ \hline 0 & P_1^n & \cdots & 0 \\ \hline \vdots & \cdots & \ddots & \vdots \\ \hline 0 & 0 & \cdots & P_m^n \end{array}\right],$$

where $P_1 \ldots P_m$ are the transition matrices of the final strongly connected components.

Considering all the paths in $G(T)$ of length n, starting in the root, the number of paths ending in component S_ℓ is $N_\ell(n) = d^n \sum_{i \in S_\ell} P_{1i}^n$. Let us decompose all the paths ending in S_ℓ into two subpaths: one (of length k) before entering S_ℓ and one (of length $n-k$) inside S_ℓ, we get from the decomposition of P^n, $N_\ell(n) = d^n \sum_{k=0}^{n} (1, 0 \ldots 0) Q^k K_\ell u_\ell$, where u_ℓ is a vector whose coordinates are 1 in S_ℓ and 0 everywhere else.

The number of 1 in the rooted subtree of T of height $2n$ is the number of ones in all the paths of length n plus the number of ones in the subtrees of height n. When n is large, the number of ones in the paths can be neglected with respect to the number of ones in the end trees.

Finally, the number of ones in a tree of height $2n$ is the number of ones in each possible end-tree of height n times the number of such trees, namely $N_\ell(n)$. When n goes to

infinity, the density of ones goes to $\sum_{\ell=1..m} \alpha_\ell(1,0,\dots,0)(I-Q)^{-1}K_\ell u_\ell = (\alpha_1 \cdots \alpha_m)r^t$, with $r_\ell = (1,0,\dots,0)(I-Q)^{-1}K_\ell u_\ell$. □

An example of a reducible rational tree is given in Figure 8.6. The previous result can be used to compute its rooted average density. The graph $G(T)$ has two final components, one aperiodic component with density 1 and another one with period 2 with average density 1/2. Starting from the root, both components are reached with probability 1/2. Therefore, such a tree has an average rooted density $\alpha = 1/2.(1/2) + 1/2 = 3/4$.

Also, it is not difficult to show that if all final components have a density (rather than an average density), then the tree has a rooted density, given by the formula given by Theorem 8.4.

Finally, it is fairly straightforward to prove that since the transition matrix P of the Markov chain associated with $G(T)$ has all its elements of the form a/d, then the stationary probabilities π as well as the average rooted density α of a rational tree are rational numbers of the form c/b with $0 \le c \le b \le d^{K+1}$. This fact will be used in the algorithmic section 8.5 to make sure that the complexities of the algorithms do not depend on the size of the numbers.

8.4. Balanced and Mechanical Trees

In this section, we introduce the notions of strongly balanced trees and mechanical trees and explore the relations between them. In particular we will prove that in the irrational case they represent the same set of trees, giving us a constructive representation of this class of trees. These results are very similar to the ones on words, which are summarized below.

8.4.1. Sturmian, Balanced and Mechanical Words

One definition of a Sturmian word uses the complexity of a word. The complexity of an infinite word w is a function $\mathcal{P}_w : \mathbb{N} \to \mathbb{N}$ where $\mathcal{P}_w(n)$ is the number of distinct factors of length n of the word w. A word is periodic if there exists n such that $\mathcal{P}_w(n) \le n$. Sturmian words are aperiodic words with minimal complexity, *i.e* such that for any n:

$$\mathcal{P}_w(n) = n + 1. \tag{8.1}$$

If x is a factor of w, its height $h(x)$ is the number of letters equal to 1 in x. A balanced word is a word where the letters 1 are distributed as evenly as possible:

$$\forall x, y \text{ factors of } w, |x| = |y| \Rightarrow |h(x) - h(y)| \le 1. \tag{8.2}$$

A mechanical word can be constructed using integer parts of affine functions. Let $\alpha \in [0;1]$ and $\phi \in [0;1)$. The lower (resp. upper) mechanical word of *slope* α and *phase* ϕ, $w = w_1 w_2 \dots$ (resp. $w' = w'_1 w'_2 \dots$) is defined by:

$$\forall i \ge 1 \quad \begin{array}{l} w_i = \lfloor (i+1)\alpha + \phi \rfloor - \lfloor i\alpha + \phi \rfloor, \\ w'_i = \lceil (i+1)\alpha + \phi \rceil - \lceil i\alpha + \phi \rceil. \end{array} \tag{8.3}$$

These three definitions represent almost the same set of words. In the case of aperiodic words, they are equivalent: a word is Sturmian if and only if it is balanced and aperiodic if and only if it is mechanical of irrational slope. For periodic words, there are similar relations:

- A rational mechanical word is balanced.
- A periodic balanced word is ultimately mechanical.

A word is called ultimately mechanical if it can be written as xw where x is a finite word and w is a mechanical word. An example of a balanced word which is not mechanical (and just ultimately mechanical) is the infinite word only made of zeros except for one letter 1. For a more complete description of Sturmian words, we refer to [109].

8.4.2. Balanced and strongly balanced trees

Using the two definitions of factors of a tree, we define two notions of balance for trees: the first one and probably the most natural one, is what we call *balanced trees* and the other one is called *strongly balanced trees*.

Definition 8.5 (Balanced and strongly balanced trees)**.** *A tree is balanced if for all $n \geq 0$, the number of nodes with label 1 in any two factors of height n, differs by at most 1.*

A tree is strongly balanced if for all $n, k \geq 0$, the number of nodes with label 1 in any two factors of height n and base k, differs by at most 1.

As the name suggests, strong balance implies balance (by taking $k = 0$). Actually, this notion is strictly stronger (Section 8.6 displays an example of a balanced tree that is not strongly balanced). Although the balance property is weaker and seems more natural for a generalization from words, the following mostly focuses on strongly balanced trees that have almost the same properties as their counterparts on words.

Density of a balanced tree

Before beginning the full investigation of balanced trees, we start with a rather straightforward property: a balanced tree has a density.

Let us recall the definition of the density (section 8.2.5): for all node v and all height n, we call $h_v(n)$ the number of 1 in the factor of root v of height n and $d_v(n)$ the density of this factor, $d_v(n) \stackrel{\text{def}}{=} \frac{1}{S(n)} h_v(n)$. Using this notation, we can write the following result.

Proposition 8.6 (Density of balanced tree)**.** *A balanced tree has a density α.*

Moreover for all node v and for all height n:

$$|h_v(n) - \lfloor S(n)\alpha \rfloor| \leq 1. \tag{8.4}$$

Proof. Let m_n be the minimal number of 1 in all factors of height n. Since the tree is balanced, for all nodes v and $n \geq 1$:

$$m_n \leq h_v(n) \leq m_n + 1. \tag{8.5}$$

Now let us consider a factor of height $n + k$ and root v. It can be decomposed into a factor of height k of root v and d^k factors of height n at the leaves of the previous factor. The number of ones in these factors can be bounded by expressions depending on m_n and m_k:

$$m_k + d^k m_n \leq m_{n+k} \leq m_k + 1 + d^k(m_n + 1). \tag{8.6}$$

The density of a factor of height n is $\frac{m_n}{S(n)} \leq d_v(n) = \frac{h_v(n)}{S(n)} \leq \frac{m_n+1}{S(n)}$. Using these facts, we can bound $d_v(n+k) - d_v(n)$:

$$\frac{m_{n+k}}{S(n+k)} - \frac{m_n+1}{S(n)} \leq d_v(n+k) - d_v(n) \leq \frac{m_{n+k}+1}{S(n+k)} - \frac{m_n}{S(n)}.$$

Using (8.6), the left inequality can be lower bounded by

$$\begin{aligned}(d-1)\Big(\frac{d^k m_n + m_k}{d^{n+k}-1} - \frac{m_n+1}{d^n-1}\Big) &= (d-1)\Big(\frac{m_n + m_k/d^k}{d^n - 1/d^k} - \frac{m_n+1}{d^n-1}\Big) \\ &\geq (d-1)\Big(\frac{m_n}{d^n-1} - \frac{m_n+1}{d^n-1}\Big) \\ &\geq -\frac{1}{S(n)}.\end{aligned}$$

The same method can be used to prove that $d_v(n+k) - d_v(n) \leq \frac{1}{S(n)}$, which shows that for n big enough, $|d_v(n+k) - d_v(n)|$ is smaller than ϵ, regardless of k. Thus $d_v(n)$ is a Cauchy sequence and has a limit $\alpha = \lim_{n\to\infty} \frac{m_n}{S(n)}$. Because of Equation (8.5), this limit does not depend on v and the tree has a density.

Let us now prove that $|d_v(n) - \lfloor S(n)\alpha \rfloor| \leq 1$: dividing the Inequality (8.6) by $S(n,k)$ and taking the limit when k goes to ∞ leads to:

$$\frac{(d-1)m_n + \alpha}{d^n} \leq \alpha \leq \frac{(d-1)m_n + 1 + \alpha}{d^n}.$$

This shows that: $S(n)\alpha - 1 \leq m_n \leq S(n)\alpha$, which implies Equation (8.4). □

Similar ideas can be used to show that Equation (8.4) can be improved in the case of strongly balanced trees. In a strongly balanced tree, for all base and height $k, n \geq 0$, the number of ones $h(n,k)$ in a factor of height n and base k satisfies:

$$\big|h(n,k) - \lfloor S(n,k)\alpha \rfloor\big| \leq 1. \tag{8.7}$$

This is false in general for balanced trees.

8.4.3. Mechanical Trees

Building balanced tree is not that easy. According to formula (8.4), each factor of height n must have $\lfloor \alpha S(n) \rfloor$ or $\lfloor \alpha S(n) \rfloor + 1$ or $\lceil \alpha S(n) \rceil - 1$ nodes labeled one. This leads to the following construction, inspired by the construction of mechanical words.

Definition 8.7 (Mechanical tree)**.** *A tree is mechanical with density $\alpha \in [0;1]$ if for all nodes v, there exists a phase $\phi_v \in [0;1)$ that satisfies one of the two following properties:*

$$\forall n : h_v(n) = \Big\lfloor S(n)\alpha + \phi_v \Big\rfloor, \tag{8.8}$$

$$\text{or } \forall n : h_v(n) = \Big\lceil S(n)\alpha - \phi_v \Big\rceil. \tag{8.9}$$

In the first case, ϕ_v is an *inferior phase of v. In the second case, ϕ_v is* a *superior phase of v.*

This definition suggests that the phases of all nodes could be arbitrary. In fact, we will see that there exists a unique mechanical tree once the phase of the root is given. The second question raised by this definition is the existence and uniqueness of the phase: we call ϕ_v "a" phase of a node ϕ_v and not "the" phase of ϕ_v since there may exist several phases leading to the same tree. This is further discussed at the end of this section.

We begin by a characterization of mechanical trees.

Proposition 8.8 (Characterization of mechanical trees)**.** *Given $\alpha \in [0;1]$ and $\phi \in [0;1)$, there exists a unique mechanical tree of density α such that ϕ is an inferior (resp. superior) phase of the root.*

Moreover, if ϕ is an inferior (resp. superior) phase of a node then $\phi_0 \leq \cdots \leq \phi_{d-1}$ are inferior (resp. superior) phases of its d children, with

$$\phi_i = \frac{\phi + \alpha + i - \lfloor \alpha + \phi \rfloor}{d} \quad \left(\text{resp. } \phi_i = \frac{\phi - \alpha + i - \lceil \alpha - \phi \rceil}{d}\right). \tag{8.10}$$

Proof. The proof will be done in two steps. Firstly, we will see that if we define the phases as in (8.10) then the tree is mechanical. Secondly, we will see that this is the only way to do so.

Existence. Let $\alpha \in [0;1]$ and $\phi \in [0;1)$. We want to build a mechanical tree whose root has an inferior phase ϕ (the case of a superior phase if similar and is not detailed here). Let $\mathcal{A}$ be an infinite tree. To each node v, we associate a number ϕ_v defined by:

- $\phi_{\text{root}} = \phi$.
- If the phase of a node v is ϕ_v, its d children satisfy Equation (8.10).

Then we build a labeled tree by putting to each node v the label $\lfloor \alpha + \phi_v \rfloor$. Let us prove by induction on n that the following relation holds.

$$\text{For all v} : h_v(n) = \left\lfloor S(n)\alpha + \phi_v \right\rfloor. \tag{8.11}$$

By definition of the labels, (8.11) holds when $n = 1$. Let $n \geq 0$ and let us assume that (8.11) holds for n. Let v be a node with phase ϕ_v and let $\phi_0 \ldots \phi_{d-1}$ be the phases of its children. We assume that $\alpha + \phi_v < 1$, which means that the label of the node is 0 (a similar calculation can be done in the other case $(\alpha + \phi_v > 1)$).

Using the well-known formula $\sum_{i=0}^{d-1} \lfloor x + \frac{i}{d} \rfloor = \lfloor dx \rfloor$, we can compute $h_v(n+1)$:

$$\begin{aligned} h_v(n+1) &= \sum_{i=0}^{d-1} \lfloor S(n)\alpha + \phi_i \rfloor \\ &= \sum_{i=0}^{d-1} \lfloor \frac{d^n - 1}{d-1}\alpha + \frac{\alpha + \phi + i}{d} \rfloor \\ &= \lfloor d(\frac{d^n - 1}{d-1}\alpha + \frac{\alpha + \phi}{d}) \rfloor \\ &= \lfloor S(n+1)\alpha + \phi \rfloor. \end{aligned}$$

Therefore, (8.11) holds for all n which means that the tree is mechanical.

Uniqueness. Now, let $\mathcal{A}$ be a mechanical tree of density α. Let v be a node and $\phi_0, \dots, \phi_{d-1}$ be the phases of its children. Let i and j be two children and let $h_i(n)$ be the number of ones in the ith subtree (of phase ϕ_i). We want to prove that either $\big($for all n: $h_i(n) \le h_j(n)\big)$ or $\big($for all n: $h_i(n) \ge h_j(n)\big)$. If the two nodes are both inferior (resp. superior), this is clearly true: $h_i(n) \le h_j(n)$ if and only if $\phi_i \le \phi_j$ (resp. $\phi_i \ge \phi_j$). If i is inferior and j is superior, it is not difficult to show that $\phi_i < 1 - \phi_j$ implies $h_i(n) \le h_j(n)$ and $\phi_i \ge 1 - \phi_j$ implies $h_i(n) \ge h_j(n)$.

Therefore we can assume (up to an exchange of the order of the children) that for all n:

$$h_0(n) \le h_1(n) \le \cdots \le h_{d-1}(n).$$

Moreover as $h_{d-1}(n) - h_0(n) \le 1$, there exists k such that $h_0(n) = h_1(n) = \cdots = h_k(n) < h_{k+1}(n) = \cdots = h_{d-1}(n)$. As $\sum_{i=0}^{d-1} h_i(n)$ does not depend on $\phi_0, \dots, \phi_{d-1}$, then for each n there is only one k that works and therefore there is only one possibility for $h_i(n)$ for all n and all i. By induction of the depth of the children, this implies that for every node v' in the subtree of root v, $h_{v'}(n)$ is fixed and therefore the tree with root v is unique.

As we have seen in the beginning of the proof, the phases ϕ_i defined in (8.10) provide correct values for $h_i(\cdot)$. Therefore such a phase ϕ_i is a possible phase for the ith child. $\square$

This theorem shows that when the phase is fixed the tree is unique. The converse is false and one can find several phases that lead to the same tree (for example, when $\alpha = 0$ all phases define the tree with label 0 everywhere) but we will show next that the set of densities α for which the phases are not necessarily unique has Lebesgue measure zero.

If for all n, $S(n)\alpha + \phi \notin \mathbb{N}$, then $\lfloor S(n)\alpha + \phi \rfloor = \lceil S(n)\alpha + \phi - 1 \rceil$. In that case, if ϕ is an inferior phase of a node then $1 - \phi$ is a superior phase of the node. Therefore -except for particular cases- there exists at least two phases of a node: one inferior and one superior. Let us now look at the possible uniqueness of the inferior phase.

Let us denote $\mathrm{frac}(x) \in [0; 1)$ the fractional part of a real number x and let us consider the sequence $\{\mathrm{frac}(S(n)\alpha + \phi)\}_{n \in \mathbb{N}}$. If this sequence can be arbitrarily close to 0, this means that for all $\psi < \phi$, there exists k such that $\lfloor S(k)\alpha + \psi \rfloor < \lfloor S(k)\alpha + \phi \rfloor$ and ψ can not be a phase of the tree. Also, if this sequence can be arbitrarily close to 1, then one can show similarly that for all $\psi > \phi$, ψ is not a phase of the node. Conversely, if there exists $\delta > 0$ such that $\mathrm{frac}(S(n)\alpha + \phi) > \delta$ (resp. $< 1 - \delta$) for all n and if we set $\phi' = \phi - \varepsilon$ (resp. $\phi' = \phi + \varepsilon$), with $\varepsilon < \delta$, then $\lfloor S(n)\alpha + \phi \rfloor = \lfloor S(n)\alpha + \phi' \rfloor$ for all n.

Thus, a phase ϕ is unique if and only if 0 and 1 are accumulation points of the sequence $\{\mathrm{frac}(S(n)\alpha + \phi)\}_n$.

Let us call $x \stackrel{\text{def}}{=} \frac{1}{d-1}\alpha$ and $y \stackrel{\text{def}}{=} \phi - x$ and $x_1, \dots, x_k, \dots$ (resp. $y_1, y_2, \dots$) be the sequence of the digits of x (resp. y) in base d (also called the d-decomposition). We want to study the sequence $\mathrm{frac}(S(n)\alpha + \phi) = \mathrm{frac}(xd^n - y)$.

$$xd^n - y = \underbrace{\sum_{k=1}^{n} x_k d^{n-k}}_{\in \mathbb{N}} + \sum_{k=1}^{\infty} (x_{k+n} - y_k) d^{-k}.$$

Therefore, $\mathrm{frac}(xd^n - y)$ is arbitrarily close to 0 implies that for arbitrarily big k, there exists n such that

$$x_n \dots x_{n+k-2} = y_1 \dots y_{k-1}, \quad x_{n+k-1} > y_k, \text{ or } \mathrm{frac}(xd^n - y) = 0. \tag{8.12}$$

Also, $\mathrm{frac}(xd^n - y)$ is arbitrarily close to 1 implies that for arbitrarily big k, there exists n such that

$$x_n, \ldots, x_{n+k-2} = y_1, \ldots, y_{n-1}, \quad x_{n+k-1} < y_n,$$

or the d-development of y is finite (*i.e.* with only zeros after some point $\ell : y = y_1, \ldots, y_\ell, 1, 0, 0 \ldots$) and that for arbitrarily big k, there exists n such that

$$x_n, \ldots, x_{n+k-2} = y_1, \ldots, y_\ell, 0, 1, \ldots, 1. \tag{8.13}$$

Using this characterization, three cases can be distinguished.

- If $\frac{\alpha}{d-1}$ is a number such that all finite sequences over $0, \ldots, d-1$ appear in its d-decomposition, then every phase is unique. In particular, all *normal numbers*[1] in base d verify this property and it is known that almost every number in $[0, 1]$ is normal (see [46] or [59]).
- If $\alpha \in \mathbb{Q}$, then the sequence $\mathrm{frac}(S(k)\alpha + \phi))$ is periodic and there are no phase ϕ such that ϕ is unique.
- If α is neither rational nor has the property that all d-sequences appear in α, then some ϕ can be unique and some others may not. For example, for $d = 2$, if α is (in base 2) the number

  $$\alpha = 0.10110011100011110000111110000 0 \ldots,$$

  then if $\mathrm{frac}(\alpha - \phi) = 0$, ϕ is unique (because α satisfies both Equations (8.12) and (8.13)). However ϕ_1 and ϕ_2 such that $\mathrm{frac}(\alpha - \phi_1) = 0.10100$ and $\mathrm{frac}(\alpha - \phi_2) = 0.1010$ are equivalent (generate the same tree).

  Other examples of the same type are the *rewind trees*, drawn on figure 8.16. The sequence of digits in base 2 of the density of such a tree is a Sturmian word. Half of the nodes of the tree are associated with node 0 in the minimal graph and therefore could have the same phase whereas the phases computed using Equation (8.10) are not all the same. Therefore, phases are not unique here.

Phases of a tree

Let us call Φ_v the set of numbers that can be phases of a node v and Φ the set of the possible phases of a tree. Φ is the union of all possible phases of its nodes: $\Phi = \cup_v \Phi_v$. The set Φ may be countable or uncountable. Countable for example when $\alpha/(d-1)$ is normal since there are at most as many phases as nodes. Uncountable for example for the tree with all label 0, for which for each node, all phases in $[0; 1)$ work.

In all cases, the set of possible phases is dense is $[0; 1)$. Indeed, at least all phases defined by the relation (8.10) are in Φ. If ϕ is the phase of the root, then all nodes at level k have a phase which is the fractional part of:

$$\frac{\frac{\frac{\phi+\alpha+i_k}{d}+\alpha+i_{k-1}}{d} + \cdots + \alpha + i_1}{d} = \alpha(\frac{1}{d^k} \cdots \frac{1}{d}) + \frac{\phi}{d^k} + \frac{i_k}{d^k} + \cdots + \frac{i_1}{d^1}, \tag{8.14}$$

with $0 \leq i_j < d$ for all j. Conversely all of these numbers are the phases of some node at level k.

[1] A number is normal in base d if all sequences of length k appear uniformly in its d-decomposition

As k goes to infinity and using a proper choice of $i_1, \ldots, i_k$ the fractional part of this number can be as close as possible to any number in $[0; 1]$. Thus the set of phases of the tree is dense in $[0; 1]$.

If the density is $\frac{p(d-1)}{d^{n+k}-d^k}$ (with $n+k$ minimal) one can show that the set of all possible phases for a given node is $[\frac{d^m-1}{d-1}\alpha; \min(\frac{d^{m+1}-1}{d-1}\alpha, 1))$ for some $m \in 0, \ldots, n+k-1$. As Φ is dense in $[0; 1)$, it contains all these intervals. Therefore, $\Phi = [0; 1)$ and the tree has exactly $n+k$ different factors of height greater than $n+k$. Hence its minimal graph has exactly $n+k$ nodes.

8.4.4. Equivalence between strongly balanced and mechanical trees

As seen in section 8.4.1, there are strong relations between balanced and mechanical words. This part shows the same results between strongly balanced and mechanical trees. This result is formally stated in the following theorem.

A tree is *ultimately mechanical* if all nodes are mechanical (*i.e.* satisfies Equation (8.8) or (8.9)), except finitely many.

Theorem 8.9. *The following statements are true.*

(i) *A mechanical tree is strongly balanced.*

(ii) *An irrational strongly balanced tree is mechanical.*

(iii) *A rational strongly balanced tree is ultimately mechanical.*

This theorem is the analog of the theorem linking balanced and mechanical words. The word 0^k10^∞ is balanced but not mechanical, only ultimately mechanical. Its counterpart for trees would be a tree with all labels equal to 0 except for one node which has label 1. The label 1 can be put as deep as desired, which shows that we can not bound the size of the "non-mechanical" beginning of the tree. A more complicated example is drawn in Figure 8.8.

Let us begin by the proof of the first part of the theorem:

Lemma 8.10. *A mechanical tree is strongly balanced.*

Proof. Let $n, k \in \mathbb{N}$. For all nodes v, $h_v(n, k)$ is the number of 1 in the factor of height n and base k rooted in v. We want to prove that for all pairs of nodes v and v': $|h_v(n, k) - h_{v'}(n, k)| \leq 1$.

By proposition 8.8, we can assume that all phases of the tree are inferior (the case where all phases are superior is similar). We call ϕ (resp. ϕ') a phase of the node v (resp. v').

$$h_v(n,k) - h_{v'}(n,k) = \lfloor \frac{d^{n+k}-1}{d-1}\alpha + \phi \rfloor - \lfloor \frac{d^k-1}{d-1}\alpha + \phi \rfloor - \lfloor \frac{d^{n+k}-1}{d-1}\alpha + \phi' \rfloor + \lfloor \frac{d^k-1}{d-1}\alpha + \phi' \rfloor.$$

Using the well-known inequality $x - x' - 1 < \lfloor x \rfloor - \lfloor x' \rfloor < x - x' + 1$, one can show that

$$-2 < h_v(n,k) - h_{v'}(n,k) < 2.$$

As $h_v(n, k)$ and $h_{v'}(n, k)$ are integers, we have $-1 \leq h_v(n, k) - h_{v'}(n,)k \leq 1$ which ends the proof of the lemma. □

We will see in the next section 8.4.5 that a strongly balanced tree is rational if and only if its density can be written as $\frac{p}{S(n,k)}$ $(p, k, n \in \mathbb{N})$, therefore we will do the proof of theorem 8.9 distinguishing strongly balanced tree with density of this form from the others.

Lemma 8.11. *If $\mathcal{A}$ is a strongly balanced tree of density α which can not be written as $\frac{p}{S(n,k)}$ $(p, k, n \in \mathbb{N})$ then $\mathcal{A}$ is mechanical.*

Proof. Let τ be a real number and v a node. At least one of the two following properties is true:

$$\forall n \geq 1 : h_v(n) \leq \lfloor S(n)\alpha + \tau \rfloor, \tag{8.15}$$

$$\forall n \geq 1 : h_v(n) \geq \lfloor S(n)\alpha + \tau \rfloor. \tag{8.16}$$

To prove this, assume that it is not true. Then there exists k, n such that $h_v(n) < \lfloor S(n)\alpha + \tau \rfloor$ and $h_v(k) > \lfloor S(k)\alpha + \tau \rfloor$. In that case the number of 1 in the factor of height n and base $n-k$ (or $k, k-n$ if $k > n$) is $h_v(n) - h_v(k) \leq \lfloor S(n)\alpha + \phi \rfloor - \lfloor S(k)\alpha + \phi \rfloor - 2 < \frac{d^n - d^k}{d-1}\alpha - 1$ which violates Formula (8.7).

Let us now define the number ϕ as the minimum τ that satisfies (8.15):

$$\phi = \inf_{\tau} \Big\{ \text{For all } n : h_v(n) \leq \lfloor S(n)\alpha + \tau \rfloor \Big\}.$$

For all $\tau > \phi$, the equation (8.15) is true, while for all $\tau' < \phi$, the equation (8.16) is true. This means that for all $\epsilon > 0$ and all n:

$$S(n)\alpha + \phi - \epsilon - 1 \leq \lfloor S(n)\alpha + \phi - \epsilon \rfloor \leq h_v(n) \leq \lfloor S(n)\alpha + \phi + \epsilon \rfloor \leq S(n)\alpha + \phi + \epsilon. \tag{8.17}$$

Taking the limit when ϵ tends to 0 shows that:

$$S(n)\alpha + \phi - 1 \leq h_v(n) \leq S(n)\alpha + \phi. \tag{8.18}$$

Therefore, unless $S(n)\alpha + \phi \in \mathbb{N}$, $h_v(n) = \lfloor S(n)\alpha + \phi \rfloor = \lceil S(n)\alpha + \phi - 1 \rceil$.

If there exists $n \in \mathbb{N}$ such that $S(n)\alpha + \phi \in \mathbb{N}$, then, as $\alpha \notin \{\frac{p}{S(n,k)}, p, k, q \in \mathbb{N}\}$, there are no other $k \in \mathbb{N}$ $(k \neq n)$ such that $S(k)\alpha + \phi \in \mathbb{N}$. If for this particular n $h_v(n) = S(n)\alpha + \phi = \lfloor S(n)\alpha + \phi \rfloor$, the node is inferior of phase ϕ. Otherwise, $h_v(n) = S(n)\alpha + \phi - 1 = \lceil S(n)\alpha + \phi - 1 \rceil$ and the node is superior of phase $1 - \phi$. □

Lemma 8.12. *Let $\mathcal{A}$ be a strongly balanced tree such that there exist n and k such that all factors of shape (n, k) have the same number of nodes with label 1. Then the tree is mechanical.*

Proof. Let us take n and k satisfying the property, such that $n + k$ is minimal and let p be the common number of ones in the factors of shape (n, k). Obviously, the tree as a density $\alpha = \frac{p(d-1)}{d^k(d^n-1)}$.

Let v be the root of the tree. The same proof as in the irrational case can be used to establish that there exists ϕ such that

$$S(n)\alpha + \phi - 1 \leq h_v(n) \leq S(n)\alpha + \phi,$$

and that the root is inferior of phase ϕ if there is no j such that $h_v(j) = \frac{d^j-1}{d-1}\alpha + \phi - 1$ – resp. superior of phase $1 - \phi$ if there is no i such that $h_v(i) = \frac{d^i-1}{d-1}\alpha + \phi$. Therefore the

tree is mechanical unless there exist i and j satisfying these equalities. Let us show that if there exist such i and j, there is a contradiction.

Let $i = \min_{i'}\{h_v(i') = \frac{d^{i'}-1}{d-1}\alpha + \phi\}$ and $j = \min_{j'}\{h_v(j') = \frac{d^{j'}-1}{d-1}\alpha + \phi - 1\}$. Either $i < j$ or $i > j$, let us assume that $j < i$, the other case is similar. The number of ones in the factor of height $i - j$ and base j is $p' = \frac{d^i - d^j}{d-1}\alpha + 1$. In that case we have $i \geq k + n$, otherwise this would violate the minimal property of $n + k$. If $j - i > n$ the factor of height $i - j$ and base j is composed of a factor of height $i - n$ and base j and d^{i-n-k} factors of height n and base k – that have exactly p nodes labeled one as assumed in the previous paragraph – and then the number of 1 in this subtree is:

$$h_v(i) - h_v(j) - d^{i-n-k}p + \phi + 1 = \alpha\frac{d^{i-n} - d^j}{d-1} + \phi + 1,$$

which violates the minimality of i.

Then if all factors of shape (k, n) have exactly p nodes labeled 1, the tree is mechanical. □

Lemma 8.13. *If $\mathcal{A}$ is a strongly balanced tree with a density $\alpha = \frac{p}{S(n,k)}$ then it has at most n factors of shape (n, k) with $p + 1$ ones or $p - 1$ ones.*

Proof. Using Equation (8.7), each factor of shape (n, k) has $p-1$, p or $p+1$ nodes labeled by 1. As the tree is strongly balanced, either there is no factor with $p - 1$ ones or no factor with $p + 1$ ones. Let us assume that there is no factor with $p - 1$ ones (the other case is similar). We claim that there are at most n factors of shape (n, k) with $p + 1$ nodes labeled by 1.

Indeed, let f be a factor of shape (n', k') with $n' = \ell n, i \in \mathbb{N}, k' \geq k$. This tree is composed of j blocks of shape (n, k) (where j depends on ℓ and k') and using Equation (8.7) again, the number of nodes with label 1 is either $jp - 1$, jp or $jp + 1$. Therefore at most one of the (n, k) blocks has $p + 1$ nodes labeled by 1.

If there were more than $n + 1$ blocks of shape (n, k) with $p + 1$ ones in the whole tree, starting respectively at line $l_1, \ldots$ and l_{n+1}, there would be two blocks with $l_i = l_j \bmod n$ and the block of height $l_j - l_i + n, l_i$ would have $jp + 2$ ones, which is not possible. Therefore there are at most n blocks of shape n, k with $p + 1$ nodes labeled by 1 in the whole tree. □

An example of a rational tree strongly balanced but not mechanical is presented in Figure 8.8.

Lemma 8.14. *A strongly balanced tree with density $\alpha = \frac{p}{S(n,k)}$, $p, n, k \in \mathbb{N}$, is ultimately mechanical. Furthermore, if the tree is irreducible, it is mechanical.*

Proof. Using Lemma 8.13, there are at most n factors of height n and base k with $p + 1$ nodes labeled 1, in the rest of the tree all factors of shape (n, k) have exactly p ones. Then the tree is ultimately mechanical by Lemma 8.12.

If the tree is irreducible, a factor appears either 0 or an infinite number of times. As there are at most n factors of shape (k, n) with $p + 1$ nodes labeled 1, there are no such factors and the tree is mechanical by Lemma 8.12.

Note that this lemma concludes the proof of Theorem 8.9. □

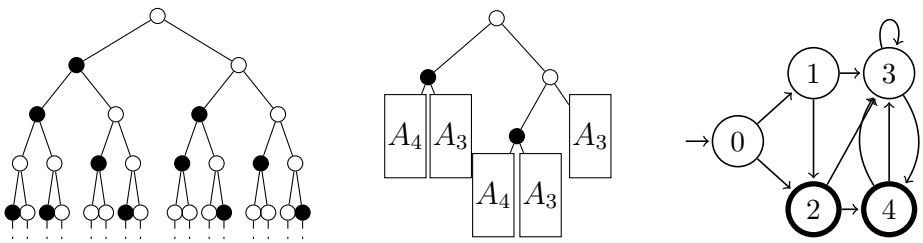


Figure 8.8.: Example of a rational tree that is strongly balanced but not mechanical. On the left is the tree itself. In the middle the mechanical suffixes of the tree are displayed and its minimal graph (reducible) is displayed on the right. There is one strongly connected component – the one corresponding to the nodes 3-4 – and two corresponding suffixes: A_3, starting with a 0, and A_4, starting with a 1.
One can verify on the picture that the beginning of this tree is strongly balanced and as it continues with density exactly 1/3, the whole tree is strongly balanced. However this tree is ultimately mechanical but not mechanical since in a mechanical tree of density 1/3, all factors of height 2 should have $\lfloor 1+\phi \rfloor = 1$ node labeled by one.

8.4.5. Link with Sturmian trees

In the case of words, Sturmian words are exactly the balanced (or mechanical) aperiodic words. The case of trees does not work as well since the Dyck Tree (Figure 8.4) and more generally all examples of Sturmian trees given in [30] are not balanced. However, the reverse implication holds as seen in the following theorem:

Theorem 8.15. *The following propositions are true.*

- *A strongly balanced tree of density different from* $\frac{p}{S(n,k)}$ *(for any* $p, n, k \in \mathbb{N}$*) is Sturmian.*
- *A strongly balanced tree of density* $\frac{p}{S(n,k)}$ *(*$p, n, k \in \mathbb{N}$*) is rational.*

This result has a simple implication: a strongly balanced tree is rational if and only if there exist $p, n, k \in \mathbb{N}$ such that its density is $\frac{p}{S(n,k)}$.

Proof. Let us consider the case of inferior mechanical trees (the superior case being similar).

Let $\mathcal{A}$ be a mechanical tree of density α, let v be a node and let $n \geq 0$. According to Proposition 8.8, the factor of root v of height n only depends on the phase ϕ_v of its root. In fact, one can show in the proof of Proposition 8.8 that this factor only depends on the values $\lfloor \frac{d^i-1}{d-1}\alpha + \phi_v \rfloor$ $(1 \leq i \leq n)$. If we write $f_i(\phi) \stackrel{\text{def}}{=} \lfloor \frac{d^i-1}{d-1}\alpha + \phi \rfloor$ $(i \geq 0, \phi \in [0:1])$, the number of factors of height n only depends on the values $f_1(\phi) \dots f_n(\phi)$.

As seen in (8.14), the set of phases is dense in $[0;1]$, therefore they are exactly as many trees as tuples $f_1(\phi), \dots, f_n(\phi)$ when $\phi \in [0;1)$ by right-continuity of f_i.

Each f_i is an increasing functions taking integer values and $h_i(1) - h_i(0) = 1$. Thus there are at most $n+1$ different tuples and then at most $n+1$ factors of height n and a mechanical tree is either rational or Sturmian.

Moreover if $\alpha \notin \left\{ \frac{p}{d^k S(n)} / p, n, k \in \mathbb{N} \right\}$, we can not have $i \neq j$ and $\frac{d^i-1}{d-1}\alpha+\phi$, $\frac{d^j-1}{d-1}\alpha+\phi \in \mathbb{N}$ and then there are exactly $n+1$ factors of height n.

If $\alpha = \frac{p}{S(n,k)}$, then the number of factors of height n is at most n. Therefore the tree is rational using Theorem 8.2 (see Section 8.4.3).

If the the tree is not mechanical, then Theorem 8.9 says that the tree has a density $\alpha = \frac{p}{S(n,k)}$ and is ultimately mechanical: there exists a depth $D \geq 1$ after which the tree is mechanical. Therefore, there are at most $S(D)+n$ factors of any height (n in the mechanical children because of the value of α plus $S(D)$ in the prefix subtree). In that case the tree is rational by Theorem 8.2. □

8.5. Algorithmic issues

8.5.1. Testing if a rational tree is strongly balanced

Given a finite description of a rational tree, let us consider the problem of checking whether this tree is strongly balanced. An algorithm that works in time $0(N^3)$ where N is the number of vertices of the minimal graph of the tree is presented.

The first focus is on the description of the special structure of the minimal graph of a rational strongly balanced tree. Then an algorithm for irreducible rational trees is described as well as a sketch of the algorithm for the general case.

Graphs of rational strongly balanced trees

The aim of this section is to study the general form of the minimal graphs of rational strongly balanced trees. In fact, we will see that they have a very particular form. The main results of this section are summarized in Theorem 8.16 and illustrated by Figures 8.9 and 8.10.

Theorem 8.16. *The following statements are true:*

(i) Two rational mechanical trees of the same density α have the same minimal graph G_α, up to the choice of the initial node of this graph. Moreover G_α is irreducible.

(ii) The minimal graph of a strongly balanced tree of density α has a unique strongly connected component that is final, G_α.

Proof. (i) Let us first consider a rational mechanical tree of density α. We know that there exist $p, k, n \geq 0$ such that $\alpha = \frac{p(d-1)}{d^k(d^n-1)}$. Using section 8.4.3, the minimal graph has exactly $n+k$ nodes, and for any node, the set of all possible phases of all its descendants is $[0;1)$. Therefore, the graph is strongly connected and unique. The only difference between two rational mechanical trees of the same density is to which node the root of the tree is associated. Figure 8.9 displays several examples. The (unique) minimal graph of the mechanical trees of density $1/3, 1/7, 4/15$ and $2/15$ are displayed.

(ii) If the tree is strongly balanced but not mechanical, it is ultimately mechanical (see proposition 8.14) which means that after a finite depth k, all suffixes are mechanical trees with the same density. All of these trees have the same graph, therefore the minimal graph has a unique final strongly connected component which is reached in at most k steps. Therefore, the minimal graph of a strongly balanced tree can be decomposed into a finite acyclic graph and one final strongly connected component, like in Figure 8.10. □

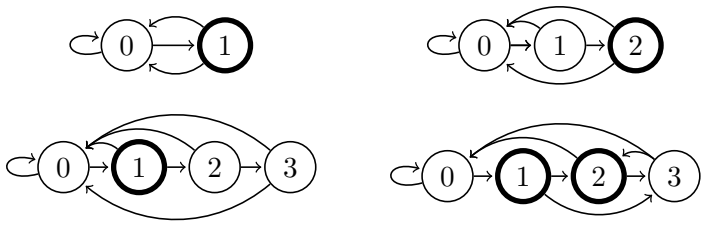


Figure 8.9.: All mechanical trees of the same density α have the same minimal graph G_α. These graphs represent G_α for $\alpha = 1/3$, $1/7$, $4/15$ and $6/15 = 2/5$. For all graphs with n nodes, there are exactly n different mechanical trees of this particular density, depending on which node is associated to the root. Note that the first three graphs have a very similar structure (Figure 8.16 displays more mechanical trees with this structure).

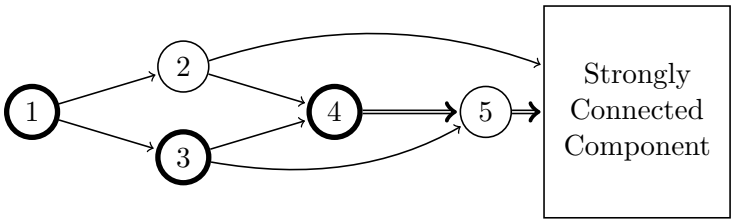


Figure 8.10.: General form of the graph of a reducible strongly balanced tree: an acyclic graph ending in a unique strongly connected component.

Irreducible trees

Testing if two graphs with a given fixed out-degree are isomorphic can be done in polynomial time [110]. Therefore using the result shown in the previous section 8.5.1, an algorithm to test if a graph represents a mechanical tree can be obtained by computing the density α of the graph and testing if the graph is isomorphic to the graph of all mechanical trees with density α. However this is not very efficient and here we propose an algorithm that tests the balance property directly.

Consider an irreducible rational tree $\mathcal{A}$ and let n_0 be the number of vertices of its minimal graph. Theorem 8.15 says that it is strongly balanced if and only if it is mechanical. In that case its density is $\frac{p}{S(n_0,k_0)}$ for some $p, k_0 \in \mathbb{N}$ and all subtrees of shape (k_0, n_0) have exactly p nodes with label 1. Such factors will be called *basic blocks* in the following.

Recall that the tree is strongly balanced if all factors of shape (n, k) have $\lfloor \alpha S(n, k) \rfloor$ or $\lfloor \alpha S(n, k) + 1 \rfloor$ nodes of label one. We want to show that testing it for all $n, k < n_0 + k_0$ is sufficient.

Let v be a node and $n, k \geq 0$ and let $h_v(F)$ be the number of labels 1 in the factor F of shape (n, k) with root v.

Starting from F, we construct a new factor F' by adding a new factor on top of F of shape $n_0, k - n_0$. This new factor can be partitioned into $d^{k-n_0-k_0}$ basic blocks. The total factor F' is of shape $(n + n_0, k - n_0)$ and its number of ones is $h_v(F') = h_v(F) + d^{k-n_0-k_0}p$ (see Figure 8.11).

The augmentation of the factor can be repeated until its shape n', k' is such that

$k' \leq k_0 + n_0$. Its number of ones is $h_v(F') = h_v(F) + H$ where H does not depend on v.

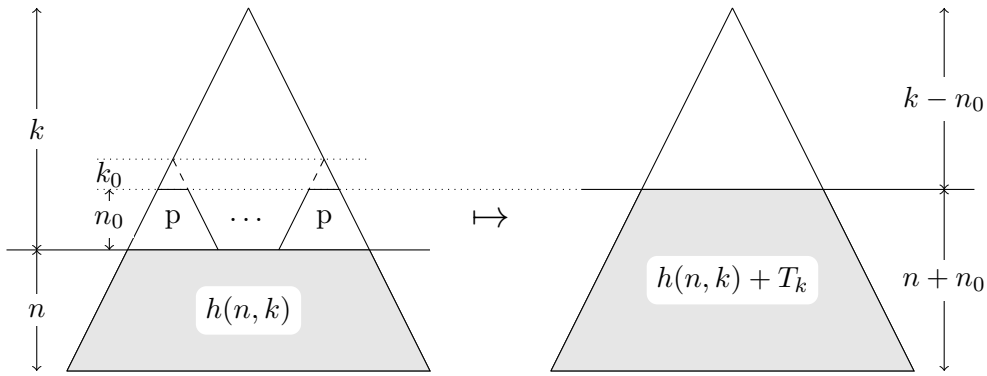


Figure 8.11.: The first transformation: if $k > n_0 + k_0$, we add a level of factors of shape n_0, k_0 that all contain exactly p ones. The shape of the factor becomes $(n + n_0, k - n_0)$. We repeat the transformation until the shape is (n', k') with $k' < n_0 + k_0$. In the figure, T_k stands for $pd^{k-n_0-k_0}p$.

The second phase consists in building a new factor F'' by removing a factor from F' of shape $n_0, k' + n' - n_0$. The removed part can be partitioned into $d^{n'-n_0-k_0}$ basic blocks. Therefore the number of ones in F'' is $h_v(F'') = h_v(F') - d^{n'-n_0-k_0}p$. This transformation is illustrated in Figure 8.12.

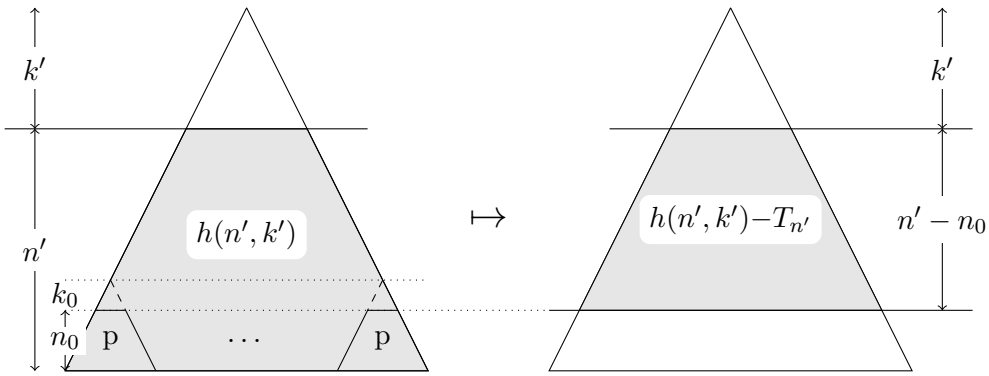


Figure 8.12.: The second transformation: if $n' > n_0 + k_0$, we can remove a level of factors of shape (n_0, k_0). The shape of the factor becomes $(n' - n_0, k')$. We repeat the transformation until the shape is (n', k') with $n' < n_0 + k_0$ (here, $T_{n'} = pd^{n'-n_0-k_0}$).

By repeating this transformation as long as $n'' > n_0 + k_0$, we get a final factor F'' whose shape is (n'', k'') with $n'' < n_0 + k_0$, $k'' < n_0 + k_0$ and whose number of ones is $h_v(F'') = h_v(F) + H - K$, where H and K do not depend on v but only on n and k.

Since $h_v(F) = h_v(F'') - H + K$, it is enough to compute the number of ones in all factors with shape (n'', k'') where $n'' < n_0 + k_0$, $k'' < n_0 + k_0$, to be able to obtain the number of ones in all factors on any shape.

Also, it is enough to test if all factors with shape (n'', k'') where $n'' < n_0 + k_0$, $k'' < n_0 + k_0$ satisfy the strong balance property for all factors on any shape to have the same property.

There are at most n factors of a given height and base. For $b < m$, let us call $h_{i,h,m}$ the number of 1 in the ith factor of height b and base $b+m$. Let us call $v(i) = (v_1(i), \ldots, v_d(i))$ the set of the d children of the tree rooted in i. $h_{i,b,m}$ can be computed using the formula:

$$h_{i,b,m} = \begin{cases} h_{i,1,0} & = & \ell(i) \\ h_{i,b,0} & = & \ell(i) + \sum_{j \in v(i)} h_{j,b-1,0} \\ h_{i,b,m} & = & \sum_{j \in v(i)} h_{j,b-1,m-1} \end{cases} \tag{8.19}$$

These considerations yield the Algorithm 8.1. The main steps of the algorithm are:

1. Compute the density α of the tree (cf Theorem 8.4).
2. If α can not be written as $p\frac{d-1}{d^N-d^k}$, the tree is not strongly balanced.
3. Check the strongly balanced property on the factors of shape $(n,k) < (N,N)$.

Require: Minimal graph G of a irreducible rational tree
Ensure: The tree corresponding to G is strongly balanced
  N:= number of vertices of G
  Compute the density α of the Markov Chain
  if for all k:$\frac{d^N-d^k}{d-1}\alpha \notin \mathbb{N}$ **then**
    return "not strongly balanced"
  end if
  for $1 \leq i, n, k \leq N$ **do**
    Compute $h_{i,n,k}$ according to (8.19)
    if $h_{i,n,k} \neq \lfloor \frac{d^n-d^k}{d-1}\alpha \rfloor$ and $h_{i,n,k} \neq \lfloor \frac{d^n-d^k}{d-1}\alpha \rfloor + 1$ **then**
      return "not strongly balanced"
    end if
  end for
  return "strongly balanced"

Algorithm 8.1: Testing if a irreducible rational tree is strongly balanced

Solving the Markov chain to get α takes at most $O(N^3)$ operations. Writing the density under the form $\frac{p}{d^N-d^k}$ is linear in N and computing all $h_{i,b,m}$ takes $O(N^3)$ operations using the formula (8.19). Therefore the algorithm runs in time $O(N^3)$.

General case

The general case is more complicated since there can be some factors of shape (n_0, k_0) with $p+1$ (or $p-1$) nodes labeled by 1. However the structure of the minimal graph of strongly balanced trees made in Section 8.5.1 can be useful.

- Indeed, the minimal graph must have only one strongly connected component and it must corresponds to a strongly balanced tree.
- If the density of the strongly connected component is $\frac{p}{2^{n_0}Ck_0}$, all factors of shape n_0, k_0 in the strongly component have exactly p nodes labeled by 1.

Therefore, using the same techniques of reduction of the size as in Figure 8.11, one can show that we just have to test the balanced property for factors of shape at most (N,N) where N is the number of vertices in the graph.

8.5.2. Counting

In this part, we address the problem of counting all possible factors of a mechanical tree. We will focus on trees of degree 2 and will compare this to the total number of possible factors of binary trees.

There are 2^n finite words on a binary alphabet of length n. Not all these words can be factors of a Sturmian words, for example 0011 can not since it is not balanced. In fact, the number of factors of length n of Sturmian words (see for example [31]) is:

$$1 + \sum_{i=1}^{n} (n - i + 1)\phi(i), \tag{8.20}$$

where ϕ is the Euler function ($\phi(i)$ is the number of integers less than i and coprime with i). Asymptotically, the number of factors is equivalent to n^3/π^2.

The number a_n of unordered complete binary trees of height n satisfies the equation:

$$a_{n+1} = a_n(a_n + 1) \tag{8.21}$$

According to [133], there is no simple solution of this equation but using the method described in [2], one can show that a_n is the nearest integer close to $\theta^{2^n} - 1/2$, where $\theta \approx 1.597910218\ldots$ is the exponential of the rapidly convergent series $\ln(3/2) + \sum_{n\geq 0} \ln(1 + (2a_n + 1)^{-2})$.

In section 8.4.5, we have seen that the number of factors of height n of a mechanical tree is the number of tuples $(f_1(\phi, \alpha), \ldots, f_n(\phi, \alpha))$ where $f_i(\phi, \alpha) = \lfloor (2^n - 1)\alpha + \phi \rfloor$. Let us call u_n this number.

To count the number of these tuples, consider the lines $\alpha \mapsto (2^n - 1)\alpha \bmod 1$, with $0 \leq \alpha \leq 1$(see Figure 8.13). The number of tuples is the number of different zones in this figure.

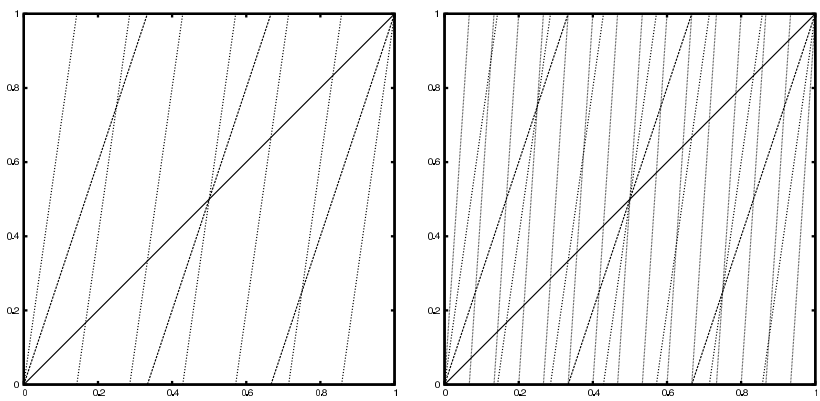


Figure 8.13.: On the left picture (resp. on the right one) the number of distinct factors of height 3 (resp. 4) are represented. The lines drawn are $\alpha \mapsto (2^n - 1)\alpha \bmod 1$ for $1 \leq n \leq 3$ (resp. $n \leq 4$). Each zone corresponds to a distinct factor of height 3 (resp. 4) of all mechanical trees. On the left picture, we can count that there are 20 factors of height 3. The difference between the left and the right picture is the addition of the lines $\alpha \mapsto (2^4 - 1)\alpha \bmod 1$. This leads to 60 factors of height 4.

An exact computation of u_n is cumbersome but good bounds can be computed easily. $u_{n+1}-u_n$ corresponds to the number of zones added by adding the lines $\alpha \mapsto (2^{n+1}-1)\alpha - i$. Each of these $2^{n+1}-1$ lines:

- add at least a new zone if it only crosses other lines at points $\phi = 0$ or $\phi = 1$. This is a very low estimate since it is only true for $i = 0$ or $i = 2^n - 2$, in the other cases it crosses at least the line $\alpha \mapsto \phi$.
- add at most $1+n$ zones if it crosses the n lines corresponding to $\alpha \mapsto (2^j - 1)\alpha - i_j$, $1 \leq j \leq n$ and if all these points are pairwise distinct.

Therefore we have an estimation for all $n \geq 2$:

$$2 + 2(2^{n+1} - 3) \leq u_{n+1} - u_n \leq (n+1)(2^{n+1} - 1). \tag{8.22}$$

This leads to the bounds for $n \geq 3$:

$$2^{n+2} \leq u_n \leq n2^{n+1}. \tag{8.23}$$

Improving these bounds seems difficult. To do so, one would have to count whether a "new" intersection has already been counted or if it is on the boundary $\phi = 0$. By simulation, it seems that the number of trees is closer to $n2^{n+1}$ than to 2^{n+1}.

8.6. Glossary

The aim of this part is to show the big picture and to provide several examples of trees that are either balanced, strongly balanced, reducible, irreducible, rational or Sturmian. In particular, we will give counter-examples that show that the inclusions between these classes are strict. The Figure 8.14 summarizes these results.

1. *Reducible Sturmian tree that is not balanced* – contrarily to the case of words where Sturmian words are balanced, there exist Sturmian trees that are not balanced. The Dyck tree (Figure 8.4), is one of them.
2. *Irreducible Sturmian trees that are not balanced* – An example of a Sturmian tree that is irreducible (but not balanced) is the *reflected random walk* tree represented in Figure 8.15. It is Sturmian since the equivalence classes of the relation $\equiv_n$ are $\{0\}, \ldots, \{n-1\}, \{n, n+1, \ldots\}$.
3. *Irreducible rational trees* – see Figure 8.7.
4. *Reducible rational trees* – see Figure 8.6.
5. *Irreducible strongly balanced rational tree* – see discussions in section 8.5.1 and Figure 8.9.
6. *Rational reducible strongly balanced trees that are not mechanical* – strongly balanced trees are not necessarily mechanical: if they are reducible, they are only ultimately mechanical, see Figure 8.8 for an example.

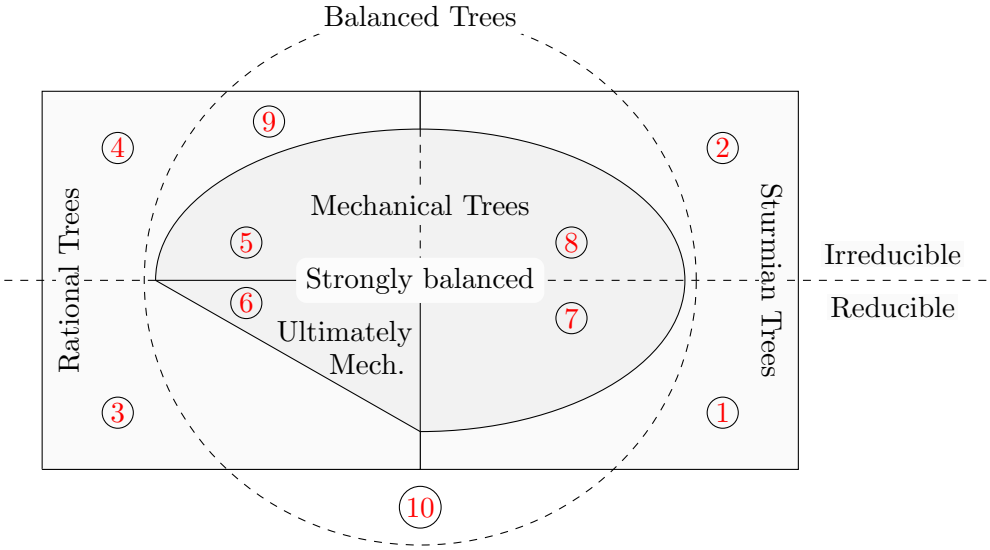


Figure 8.14.: Relations of inclusion linking the different classes of trees. Each number refers to an example detailed in section 8.6. For example, zone 6 is the set of trees that are rational, reducible, ultimately mechanical, strongly balanced, balanced and neither mechanical nor Sturmian.

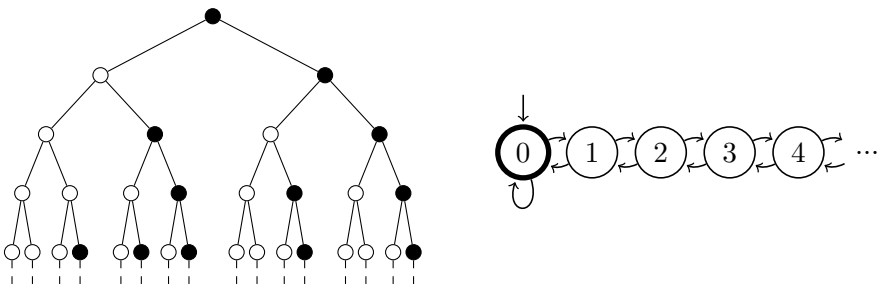


Figure 8.15.: The reflected random walk tree: each node of type n is followed by one of type $n-1$ and one of type $n+1$ (except for 0 that is followed by 0 and 1).

7. *Reducible mechanical trees* – let α be a normal number and consider the mechanical tree of density α and phase 0 at the root. As α is normal, there is a unique phase corresponding to each node of order k which is the fractional part of:

$$\alpha(\frac{1}{d^k}+\cdots+\frac{1}{d})+\frac{i_k}{d^k}+\cdots+\frac{i_1}{d}, \tag{8.24}$$

for a unique sequence $i_1,\ldots,i_k$ (see the end of section 8.4.3 for details about normal numbers and phases). If two phases corresponding to $i_1,\ldots,i_k$ and $i'_1,\ldots,i'_{k'}$ are equal, then we have

$$\mathrm{frac}\Big(\alpha(\frac{1}{d^k}+\cdots+\frac{1}{d^{k'+1}})+\sum_{j=1}^{k}\frac{i_j}{d^j}-\sum_{j=1}^{k'}\frac{i'_j}{d^j}\Big)=0.$$

As α is normal, it is irrational. Therefore $k=k'$ and $\mathrm{frac}(\sum_{j\le k}\frac{i_j}{d^j}-\sum_{j\le k'}\frac{i'_j}{d^j})=0$. By uniqueness of the decomposition of a number in base d, this implies that the

two sequences are equal. This shows that two different nodes in the tree have a different phase. Thus the minimal graph of this tree is exactly the tree itself which is in a sense the most reducible tree.

8. *Irreducible mechanical trees* – let w be a mechanical word and consider a graph with vertices $\{0, 1, \ldots, \}$, where a node $i \geq 0$ has label one if and only if $w_i = 1$. The node i has two outgoing arcs: one ending in $i + 1$, one ending in 0. We call this graph a *restart tree* since for a node n, we have the choice between restarting back in 0 or continuing in $n + 1$, an example is displayed in Figure 8.16.

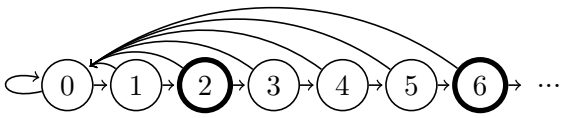


Figure 8.16.: Example of the restart tree corresponding to the word *aabaaab*...

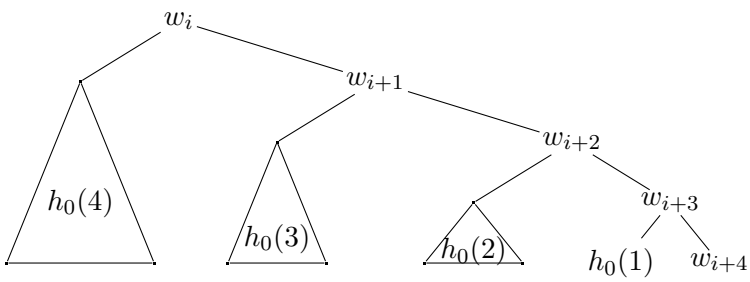


Figure 8.17.: Number of ones in a factor of the restart tree of height 5

As seen in Figure 8.17, the number of ones in a factor of height n that corresponds to the node i is

$$h_i(n) = w_i + \cdots + w_{i+n-1} + h_0(n-1) + \cdots + h_0(1), \tag{8.25}$$

and the number of ones in a factor of height n and base k is

$$h_i(n, k) = h_i(n) - h_i(k) = w_k + \cdots + w_{i+n-1} + h_0(n-1) + \cdots + h_0(k). \tag{8.26}$$

Therefore the tree is strongly balanced if and only if the word w is balanced. Since the tree is irreducible, in that case the tree is also mechanical. Moreover one can show that for any word w the tree has a density which is $\lim_{n\to\infty} \frac{h_0(n)}{2^n-1} = \frac{w_0}{2} + \frac{w_1}{4} + \frac{w_2}{8} + \cdots$.

Thus for any aperiodic balanced word, this provides an example of an irreducible irrational strongly balanced tree.

9. *Rational balanced tree that is not strongly balanced* – An example of a rational tree that is balanced but not strongly balanced is presented in Figure 8.18. One can show that all of its factors of height 3 have exactly 4 nodes with label one. Using this fact, one can show that the number of ones in a factor of height $3n + i$ $(0 \leq i \leq 3)$ rooted in a node j is:

Height	Node 1	Node 2	Node 3	Node 4
$3n$	$4\frac{8^n-1}{7}$	$4\frac{8^n-1}{7}$	$4\frac{8^n-1}{7}$	$4\frac{8^n-1}{7}$
$3n+1$	$1+2.4\frac{8^n-1}{7}$	$0+2.4\frac{8^n-1}{7}$	$0+2.4\frac{8^n-1}{7}$	$1+2.4\frac{8^n-1}{7}$
$3n+2$	$1+4.4\frac{8^n-1}{7}$	$1+4.4\frac{8^n-1}{7}$	$2+4.4\frac{8^n-1}{7}$	$2+4.4\frac{8^n-1}{7}$

This shows that the tree is balanced. It is not strongly balanced since there are factors of shape $(1,1)$ with 2 nodes labeled by one and others with 0 nodes labeled by one as seen in the bottom right part of figure 8.18. Also its minimal graph is not isomorphic to the unique minimal graph of a mechanical tree of density $4/7$ that has only 3 nodes (see the discussion about graphs of strongly balanced tree in Section 8.5.1).

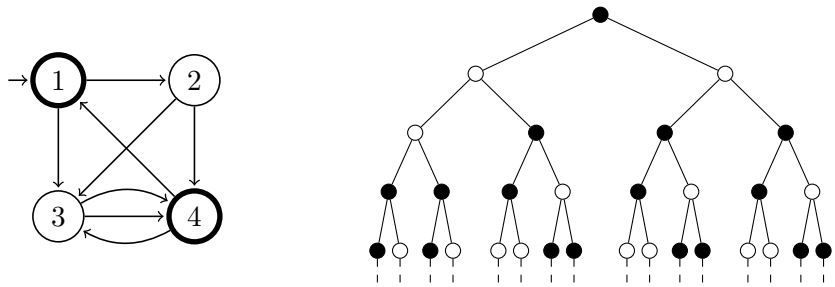
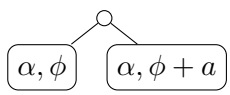


Figure 8.18.: A Rational Balanced Tree that is not strongly balanced

10. *Irrational balanced tree that is not strongly balanced* – Building an irrational tree not strongly balanced requires more work. We consider a tree which has a root r labeled by 0 and two children that are mechanical trees of density α and respective phases ϕ and $\phi+a$. We will see that under some conditions on α, ϕ and a, this will be an irrational tree that is balanced but not strongly balanced nor rational, nor Sturmian.



The two children of the root are balanced trees which means that the tree is balanced if and only if for all n:

$$\lfloor (2^{n+1}-1)\alpha \rfloor \leq h_r(n+1) \leq \lfloor (2^{n+1}-1)\alpha \rfloor + 1. \tag{8.27}$$

Let us call $k = \lfloor (2^n-1)\alpha + \phi \rfloor$ and $x = \mathrm{frac}((2^n-1)\alpha + \phi)$.

$$\begin{aligned} h_r(n+1) &= \lfloor (2^n-1)\alpha + \phi \rfloor + \lfloor (2^n-1)\alpha + \phi + a \rfloor \\ &= k + \lfloor k + x + a \rfloor. \end{aligned}$$

As $(2^{n+1}-1)\alpha = 2k + 2x + \alpha - 2\phi$, the equation 8.27 holds if for all $x \in [0;1)$, we have:

$$0 \leq k + \lfloor k + x + a \rfloor - \lfloor 2k + 2x + \alpha - 2\phi \rfloor \leq 1.$$

which holds if for all $x \in [0;1)$:

$$0 \leq \lfloor x + a \rfloor - \lfloor 2x + \alpha - 2\phi \rfloor \leq 1.$$

This equation is satisfied if and only if

$$(x+a<1 \text{ and } -1 \leq 2x-2\phi+\alpha<1) \text{ or } (x+a \geq 1 \text{ and } 0 \leq 2x-2\phi+\alpha<2).$$

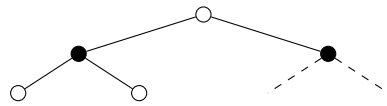
Looking at the extremal cases for $x+a<1$ and $x+a \geq 1$ which are $x=0,1-a,1$, one gets 4 relations:

$$\begin{aligned} 2(1-a)-2\phi+\alpha &< 1 \\ -1 &\leq -2\phi+\alpha \\ 2-2\phi+\alpha &< 2 \\ 0 &\leq 2(1-a)-2\phi+\alpha. \end{aligned}$$

Therefore the tree is balanced if and only if

$$\frac{\alpha}{2}<\phi \leq \frac{\alpha+1}{2}<\phi+a<1. \tag{8.28}$$

Moreover if $\alpha+\phi \geq 1$ and $3\alpha+\phi<2$, the tree is not strongly balanced since its beginning is



There are lots of triples α, ϕ, a satisfying conditions (8.28). For example a tree with $\alpha=\frac{1}{3}+\epsilon$, $\phi=0.6$ and $a=0.2$ where $\epsilon \in \mathbb{R} \setminus \mathbb{Q}$ with ϵ small enough (for example $\epsilon<0.01$ works since $\frac{\alpha}{2} \approx 0.21<0.6<\frac{\alpha+1}{2} \approx 0.71 \leq 0.8<1$ and $\alpha+\phi>1$, $3\alpha+\phi \approx 1.9<2$).

Chapter 9.

Distributed Delay-Power Control Algorithms for Bandwidth sharing in Wireless Networks

Abstract of this chapter – In this chapter, we formulate a Delay-Power Control (DPC) scheme for wireless networking, which efficiently balances delay against transmitter power on each wireless link. The DPC scheme is scalable, as each link autonomously updates its power based on the interference observed at its receiver; no cross-link communication is required. It is shown that DPC converges to a unique equilibrium power and several key properties are established, concerning the nature of channel bandwidth sharing achieved by the links. The DPC scheme is contrasted to the well-known Foschini-Miljanic (FM) formulation for transmitter power control in wireless networks and some key advantages are established. Based on the DPC and FM schemes, two protocols are developed, which leverage adaptive tuning of DPC parameters. One of them is inspired by TCP and exhibits analogous behavior. The chapter primarily focuses on the theoretical underpinnings of DPC and their practical implications for efficient protocol design.

Résumé du chapitre – Dans ce chapitre, nous présentons un algorithme de contrôle de puissance pour les réseaux sans fils. Ce mécanisme est un algorithme distribué dans lequel chaque émetteur essaie de trouver un équilibre entre la puissance consommée et le délais subis par les communications. Nous montrons que l'algorithme converge vers un point d'équilibre indépendant du point de départ étudions les propriétés de l'équilibre vers lequel il converge.

En se basant sur DPC et sur l'algorithm de Foschini-Miljanic, nous développons deux protocoles dont le but est d'adapter les paramètres de l'algorithme. Ce chapitre se concentre principalement sur les aspects théoriques de DPC et ses implications pratiques dans la construction d'un protocole de communication.

9.1. Introduction

Transmitter power control (and related rate control) is important in wireless networking for maintaining communication quality in the presence of interference, increasing network capacity, reducing power consumption, etc. It has received considerable attention in the past.

The prevalent formulation of the transmitter power control problem for streaming traffic is due to Foschini & Miljanic [64, 65]. It asserts that each communication link i in the shared wireless channel sets a target SINR $\hat{\gamma}_i$ (signal to interference plus noise ratio). It then aims to achieve $\hat{\gamma}_i$ in each time slot, by autonomously updating its transmitter power P_i^t, using the control algorithm

$$P_i^{t+1} = \frac{\hat{\gamma}_i}{\gamma_i^t} P_i^t, \tag{9.1}$$

in consecutive time slots, where γ_i^t is the link SINR observed in time slot t. If the target SINRs $\hat{\gamma}_i$ are feasible for all links, then the algorithm will provably converge to the set of SINR targets $\hat{\gamma}_i$. If not, all powers will diverge to infinity. The *asynchronous* version of the Foschini-Miljanic (FM) control (9.1) converges or diverges similarly [113].

The $\hat{\gamma}_i$ is measured at the receiver of link i and communicated to its transmitter (say, over a separate control channel). Hence, there is intra-link communication from the receiver to the transmitter. However, there is no cross-link communication and each link acts autonomously, a highly desirable property for scalability. In this sense, the FM algorithm (9.1) is distributed, except that there is need for a global guarantee that the links are feasible at their SINR targets $\hat{\gamma}_i$. Without it all links will go unstable (powers explode). Therefore, this is an essential global requirement, compromising somewhat the distributed nature of the algorithm.

Note that by targeting SINR $\hat{\gamma}_i$ in the FM formulation, link i essentially targets a *preset* effective transmission rate, hence, link communication delay. Therefore, this formulation is not geared towards managing the delay/power tradeoff, a fundamental one in wireless communication.

Another formulation [151] of the power control problem aims to maximize the minimum SINR of any link sharing the wireless channel, instead of managing the power/delay tradeoff. Game theoretic approaches to the problem are taken in [131, 112] and utility maximization ones in [135, 86], requiring various degrees of link coordination and not controlling the delay vs. power tradeoff explicitly. The latter, however, is controlled in a 'queueing dynamics sense' in the approach of [129, 91], where the power control instantaneously responds to the current packet backlog in the transmitter buffer and the interference observed at the link receiver. An optimal control is approximated, utilizing a model reduction where each link is considered operating in a random interference environment, generated by the others sharing the channel. This approach, however, uses instantaneous backlog information, which the FM one does not (neither the DPC one, as seen later).

The above approaches are very different from the one taken in this chapter. Specifically, the key idea here is that each wireless link manages selfishly its power vs. delay tradeoff, responding to the interference generated by other links sharing the channel, as follows. At each point in time, the link balances

1. the per-packet expected *transmission energy* spent overall until the packet (information block) under transmission is successfully received,

2. against the per-packet expected *transmission delay* until the packet is successfully received.

Additionally, if packets arrive continuously to the link transmitter buffer, the second concern can be extended to include the *queueing delay* of buffered packets. This idea is developed below and leads to a distributed Delay-Power Control (DPC), which is shown to reach a collective unique equilibrium when used by each link (even asynchronously). In contrast, the FM approach (9.1) does not exhibit 'awareness' of this balance. Both the FM and DPC approaches use the same minimal information (interference on link receiver) to control power.

The organization of the chapter is as follows. In Section 9.2 we introduce the wireless transmission model. In Section 9.3 we develop the rationale behind the DPC formulation and establish certain key DPC properties. In Section 9.4, we mathematically analyze DPC and prove that under general conditions it converges to a well-defined unique equilibrium[1] in a fully distributed and asynchronous way. Various useful mathematical properties of the DPC equilibrium are established, including that all feasible SINRs are attainable by DPC. The DPC behavior is demonstrated with selected numerical experiments. In Section 9.5, DPC is leveraged for designing protocols that automatically tune its parameters aiming to achieve efficient bandwidth sharing and avoid over-saturation (where high power is wasted for small SINR gains). We introduce two protocols, a Saturation-Averse DPC (which backs off from saturation when that settles in) and a Rate-Aggressive DPC (which aims for highest SINRs and link rates, up to the point of avoiding saturation). The latter is somewhat inspired by TCP and, not surprisingly, exhibits some analogous behavior. We report several interesting observations made in numerical experiments using the protocols. Finally, in Section 9.6 we discuss certain current and future lines of research.

Overall, the focus of this chapter is to introduce DPC and develop its theoretical underpinnings and core provable properties. Based on the latter we develop DPC-based protocols and investigate them via numerical experiments. It is not, however, within the scope (and limited space) of this chapter to study the mathematical properties of these protocols, which form highly non-linear complex systems; this is a topic of future research.

9.2. Transmission Model

Consider L communication links, indexed by $i \in \mathcal{L} = \{1, 2, ..., L\}$, sharing the same wireless medium and interfering with each other. Time is slotted, indexed by $t = \{1, 2, 3, ...\}$.

The power transmitted by the transmitter of link i in time slot t is P_i^t and is constant throughout the slot. The power received at the link's receiver from its transmitter is $G_{ii}P_i^t$, where G_{ii} is the constant power gain (attenuation) between the transmitter and the receiver of link i.

[1] We opt to take a direct system-theoretic point of view in this chapter, based on map iterations converging to a unique fixed point. However, as becomes evident later, the problem and results can equivalently be viewed within a game-theoretic framework, where iterations of best-responses of individual players eventually converge to a unique Nash equilibrium. The correspondence is immediate and we do not discuss it any further.

The thermal noise at the receiver of link i is constant N_i in all time slots. The power gain (attenuation) from the transmitter of link j to the receiver of link i is G_{ij} and is also constant in time. Hence, $G_{ij}P_j^t$ is the interference induced on the receiver of link i by the transmitter of link $j \in \mathcal{L} - i$ in time slot t. The total interference plus noise experienced by link i in time slot t is $N_i + \sum_{j \neq i} G_{ij}P_j^t$ and the signal to interference plus noise ratio (SINR) of link i at time t is

$$\gamma_i^t = \frac{P_i^t}{b_i^t} \tag{9.2}$$

where

$$b_i^t = \frac{N_i}{G_{ii}} + \sum_{j \neq i} \frac{G_{ij}}{G_{ii}} P_j^t \tag{9.3}$$

is total noise plus interference of link i, rescaled by the intra-link power gain G_{ii}. In the following, we liberally refer to b_i^t as *interference* (when no confusion arises).

In each time slot, the transmitter of each link transmits a packet to its receiver. If the packet is successfully received, it is removed from the transmitter buffer and the next one (if any) is transmitted in the next time slot. If the packet is not successfully received (due to corruption by interference/noise), it is repeatedly retransmitted until successfully received. It is assumed that transmission success/failure events are *statistically independent* with probabilities that depend on each link's SINR. Specifically, $S_i(\gamma_i) = S_i(P_i/b_i)$ is the probability that the packet transmitted in a time slot on link i will be successfully received, when the link's SINR is γ_i.

The success probability $S_i(\gamma_i)$ may be different for each link, depending on the modulation scheme used, the radio propagation environment, etc. We do not place any specific restrictions on $S_i(\gamma_i)$ at this point (see Def. 9.3 later), except that it is naturally assumed to be an *increasing* function[1] of the SINR γ_i of link $i \in \mathcal{L}$. Note[2] that $S_i(\gamma_i)$ is the *effective packet transmission rate* of link i under SINR γ_i.

At the end of each time slot, the receiver of each link needs to communicate to its transmitter 1) the interference it observed during the ending time slot, and 2) acknowledge whether the transmitted packet was received correctly or not. For simplicity, we assume that this reverse-communication occurs on a reliable network control channel, separate from the channel shared by the links. This control channel is a low-rate one, since the information transmitted per slot is minimal (interference value and acknowledgements). In any case, at minimum the receiver needs to acknowledge success/failure in packet reception (interference values could be piggy-backed on the Acks/Nacks). Note that this unavoidably introduces *intra-link* communication, but no cross-link communication is needed. Therefore, the concept of being autonomous or distributed here is at the level of individual links, as opposed to that of distinct transmitters/receivers.

9.3. Delay-Power Control (DPC)

We first develop the delay-power control *rationale* with reference to the head-of-line packet in the transmitter queue of each link. We then complete in section 9.3.1 this rationale, by incorporating packet arrivals and queueing delay.

[1] For example, typical SINR functions include $S_i(\gamma_i) = \gamma_i/(\xi + \gamma_i)$ for some positive constant ξ, and $S_i(\gamma_i) = 1 - e^{-\delta\gamma_i}$ for some positive constant δ, depending on system-specific particulars.

[2] We later extend S_i to include non-probability functions like $S_i(\gamma_i) = \ln(1 + \gamma_i)$, akin to information theoretic effective bit rates (see below).

Each link i controls its transmitter power *autonomously*, as follows. It first observes the interference $b_i = b_i^t$, which is communicated to the transmitter at the end of slot t. Then, link i reasons that

1. if all other links would not change their powers and
2. link i would transmit at constant power P_i in the future,

it will have to transmit the head-of-line packet for an average of $1/S_i(\frac{P_i}{b_i})$ time slots at power P_i, before it is successfully received. Recall that packet transmission success/failure events are statistically independent. Therefore, the link will incur

1. an average transmission *delay cost*: $1/S_i(\frac{P_i}{b_i})$
2. an[1] average *power cost*: $P_i/S_i(\frac{P_i}{b_i})$

until this packet is successfully received. Hence, the link chooses its power in the next slot $t+1$ to minimize the (weighted) sum of the per-packet[2] power and delay costs, as

$$P_i^{t+1} = \Phi_i(b_i^t) = \arg\min_p \left\{ \frac{\alpha_i}{S_i(\frac{p}{b_i^t})} + \frac{p}{S_i(\frac{p}{b_i^t})} \right\}. \tag{9.4}$$

The minimization is taken over $p \in [0, \infty)$. The weight factor α_i reflects how link i weighs delay vs. power. A delay sensitive (power conscious) link will chose a larger (smaller) α_i value.

Each link performs the power update (9.4) autonomously, without any cross-link communication, so the power control scheme is distributed (per link). Of course, after each power update the b_i^t changes and the scheme has to be applied again.

Under DPC (9.4) each link optimizes *selfishly* (over the link set) and *myopically* (over time) its delay-power tradeoff per transmitted packet, assuming that all other links will not change their powers. As the latter do, the link repeats the process. Another way to view DPC (9.4) is as directly responding to interference b_i^t via $P_i^{t+1} = \Phi_i(b_i^t)$.

Remark 9.1. *For example, when $S_i(x) = x/(x+1)$ we get $\Phi_i(b_i) = \sqrt{\alpha_i b_i}$ via direct calculations in (9.4), and the SINR of link i is $\sqrt{\alpha_i/b_i}$. In general, $\Phi_i(b_i)$ may not be obtainable in closed form and would have to be computed off-line and coded into a look-up table at node i.*

[1] Note that this is actually *energy cost*, but we liberally use the terms 'power' and 'energy' interchangeably in this chapter for simplicity, as time is slotted and the distinction is easily seen.

[2] A 'networking' (packetized) point of view is taken here. Each link can transmit a packet per slot, which is successfully received with probability $S_i(\frac{P_i}{b_i})$, which is assumed to have certain natural properties (discussed in Def. 9.3). The precise formula of $S_i(\frac{p}{b_i})$ does not matter and may actually not be computable in closed form. In principle, we may have to estimate it empirically via experiments for given link topology and technology. The key assumption, though, is that packet transmissions in various slots and links are *statistically independent*, which is standard in this line of research.

We can also 'twist' the above view into a 'digital communication' one. Indeed, the following results for DPC (9.4) under the 'networking' point of view (where $S_i(\cdot)$ is a probability) do hold true when we set $S_i(\frac{P_i}{b_i}) = \ln(1 + \frac{P_i}{b_i})$, which can be greater than 1 and is not a probability. How can then one (roughly) interpret the cost combination minimized in the DPC (9.4) algorithm?

Let us take a 'digital communication' point of view, seeing $\ln(1 + \frac{P_i}{b_i})$ as the effective bit rate. Then $1/\ln(1 + \frac{P_i}{b_i})$ an be (roughly) viewed as the (on average) delay between two successful bit transmissions, hence, the per bit delay. And $P_i/\ln(1 + \frac{P_i}{b_i})$ can be (roughly) viewed as the power during the average per-bit delay $1/\ln(1 + \frac{P_i}{b_i})$, hence, it is the average power per bit.

9.3.1. Incorporating Packet Arrivals and Queueing Delay

We now extend the control rationale to include packet arrivals and queueing delay, beyond the per packet transmission delay $1/S_i(\gamma_i)$ already taken into account.

Suppose that in each time slot a packet arrives at the transmitter queue of link i with probability λ_i. Packet arrival events are *statistically independent* of each other and all packet transmission success/failure events. If the interference were not to change on link i, then its transmitter queue could be modeled as a standard discrete-time queue (birth-death chain) with arrival rate λ_i and service rate $S_i(\gamma_i)$. The average queueing delay (buffering plus transmission time) incurred by a packet in the queue would simply be $1/(S_i(\gamma_i) - \lambda_i)$. Therefore, each user could be trying to selfishly and myopically optimize the weighted sum of the average per-packet queueing delay plus transmission power (energy), leading to the power control

$$P_i^{t+1} = \Phi_i(b_i^t) = \arg\min_p \left\{ \frac{\alpha_i}{S_i(\frac{p}{b_i^t}) - \lambda_i} + \frac{p}{S_i(\frac{p}{b_i^t})} \right\}, \tag{9.5}$$

where the minimization is over $p \in [0, \infty)$ and $\alpha_i > 0$ is again the weight of the packet expected queuing delay vs. power.

In order for the algorithm to be correctly defined, we need to have (by assumption) that $\lambda_i < \lim_{\gamma\to\infty} S(\gamma)$. Otherwise, the queue would not be stable even if no one else is transmitting. Then the issue arises (treated below) whether the arrival rates $\{\lambda_i, i \in \mathcal{L}\}$ are feasible (i.e. no transmitter queue explodes) when the system operates under DPC (9.5).

Note that DPC (9.4), which is oblivious to packet arrivals, is mathematically a *special case* of DPC (9.5) for $\lambda_i = 0, i \in \mathcal{L}$.

The DPC algorithm (9.5) or (9.4) can be relaxed to an *asynchronous version*, where each link power-updates at arbitrary time slots, observing the interference in the latest one.

Remark 9.2. *(*Coexistence of Heterogeneous Links*) Since DPC manages the delay vs. power tradeoff by choosing via α_i how much to weigh delay vs. power, it is very flexible in the following sense. In diverse wireless networking scenarios, power-conscious nodes transmitting elastic traffic (sensors, agents, etc.) could set their α_i low (leading to higher delay and lower power); but power-oblivious nodes transmitting delay-sensitive traffic (voice, media streaming, etc.) could set α_i high (squeezing the delay down at the expense of higher power). Since DPC allows each link to individually α_i (and is also distributed), it would be particularly appropriate even for highly heterogeneous mixtures of delay-tolerant and delay-sensitive links coexisting in a wireless channel.*

9.3.2. The Feasible Power Cone $\mathcal{J}(\lambda)$

DPC (9.5) is based on 'minimization' and so it never picks powers where either the delay or power costs are infinite.

For the delay cost $\frac{\alpha_i}{S_i(\frac{P_i}{b_i}) - \lambda_i}$ to be finite, the DPC must select a power P_i such that

$$\frac{P_i}{b_i} > \gamma_i^*(\lambda_i) \tag{9.6}$$

for each $i \in \mathcal{L}$, where γ_i^* is such that

$$S_i(\gamma_i^*) = \lambda_i. \tag{9.7}$$

Recall that $b_i(P) = \sum_{j \neq i} \frac{G_{ij}}{G_{ii}} P_j + \frac{N_i}{G_{ii}}$. Therefore, the set

$$\mathcal{J}(\lambda) = \{P < \infty : \frac{P_i}{b_i(P)} > \gamma_i^*(\lambda_i), \text{ for all } i \in \mathcal{L}\} \tag{9.8}$$

or (equivalently)

$$\mathcal{J}(\lambda) = \{P < \infty : G_{ii}P_i - \gamma_i^*(\lambda_i) \sum_{j \neq i} G_{ij}P_j > \gamma_i^*(\lambda_i)N_i, \quad \text{for all } i \in \mathcal{L}\} \tag{9.9}$$

is the set of feasible power vectors: any $P \in \mathcal{J}(\lambda)$ leads to finite costs on *all* links. Therefore, $\mathcal{J}(\lambda)$ is the power domain over which collective system optimization takes place. Note that $\mathcal{J}(\lambda)$ is a 'pure convex linear cone' (when nonempty), and is topologically open (does not include the boundaries).

The problem is that $\mathcal{J}(\lambda)$ may *not* exist (may be empty). Indeed, as the arrival rates λ_i increase, $\gamma_i^*(\lambda_i)$ also increases, from (9.7) and $S_i(\gamma)$ increasing in γ; therefore, $\mathcal{J}(\lambda)$ shrinks. Eventually, $\gamma_i^*(\lambda_i)$ tends to infinity, as λ_i increases, and the cone $\mathcal{J}(\lambda)$ 'evaporates.' Hence, there is only a bounded set of traffic loads $\lambda_i, i \in \mathcal{L}$ for which the cone $\mathcal{J}(\lambda)$ exits. We define it as

$$\Lambda = \{\lambda \in \mathbb{R}_0^L : \mathcal{J}(\lambda) \neq \emptyset\}. \tag{9.10}$$

This is the achievable throughput region of the system (from a queueing perspective) or the set of *feasible* arrival rates λ.

9.3.3. The Power Set $\mathcal{G}(\lambda)$

Define now the set $\mathcal{G}(\lambda)$ of all power vectors P for which $\frac{\alpha_i}{S_i(\frac{P_i}{b_i(P)}) - \lambda_i} + \frac{P_i}{S_i(\frac{P_i}{b_i(P)})}$ is higher than (or equal to) the minimum possible, that is, $P_i \geq \Phi_i(b_i)$ for each link $i \in \mathcal{L}$. Therefore,

$$\mathcal{G}(\lambda) = \{P < \infty : P_i \geq \Phi_i(b_i(P)), \text{ for all } i \in \mathcal{L}\}. \tag{9.11}$$

This is a topologically closed set (minima are attainable). Note that $\mathcal{G}(\lambda) \subseteq \mathcal{J}(\lambda)$ for each $\lambda \in \Lambda$. Indeed, $P \in \mathcal{G}(\lambda)$ implies that all $\Phi_i(b_i(P)) < \infty$ for all $i \in \mathcal{L}$, hence, the costs are finite and so $P \in \mathcal{J}(\lambda)$. We study the shape of $\mathcal{G}(\lambda)$ in Section 9.4.3, where we show that $\mathcal{J}(\lambda) \neq \emptyset$ implies $\mathcal{G}(\lambda) \neq \emptyset$.

9.4. Delay-Power Control (DPC) Analysis

We first consider the conditions under which DPC (9.5) and its special case (9.4) converge? We later examine the effects of varying the weights α_i.

9.4.1. Class $\mathcal{S}$ of functions $S_i(\gamma)$ for DPC Convergence

We next define a broad and natural class $\mathcal{S}$ of functions $S_i(\gamma)$ for which we can establish DPC convergence. Recall that $S_i(\gamma) \geq 0$ is the *effective transmission rate* of link i when its SINR is $\gamma \geq 0$.

Definition 9.3. *The function class $\mathcal{S}$ is comprised of the functions $S : \mathbb{R}^+ \to \mathbb{R}^+$ satisfying the following properties:*

1. $S_i(0) = 0$*; that is, no power results in no rate.*
2. $S_i(\gamma)$ *is* strictly increasing *($S_i'(\gamma) > 0$) and* strictly concave *($S_i''(\gamma) < 0$); that is, higher SINR results in higher rate, but exhibits diminishing gains as γ increases.*
3. $\lim_{\gamma\to\infty} \frac{S_i(\gamma)}{\gamma} = 0$*, that is, $S_i(\gamma)$ may increase*[1] *at most sublinearly at large γ.*
4. $S_i(\gamma)$ *is twice differentiable.*
5. *For all γ: $S_i'(\gamma)S_i(\gamma) - \gamma S_i'(\gamma)^2 + \gamma S_i(\gamma)S_i'(\gamma) < 0$.*

The above properties are quite natural for wireless systems. Only *Property 5* may not seem so at first glance, but Lemma 9.6 shows it is equivalent to $\Phi_i(b_i)$ in (9.5) *increasing* in b_i. That is, the link needs more power under higher interference to achieve its desired delay/power balance. This is consistent with intuition.

Remark 9.4. *Via standard calculations, one can easily show that typical $S_i(\gamma)$ functions, like $\gamma/(\xi_i + \gamma)$ for fixed $\xi_i \geq 0$ and $1 - e^{-\delta_i\gamma}$ for fixed $\delta_i \geq 0$ belong*[2] *to $\mathcal{S}$.*

9.4.2. Key Properties of DPC

We first show that $\Phi_i(b_i)$ in (9.5), and in special case (9.4) with $\lambda = 0$, is well-defined and identify some of its properties, assuming that $S_i \in \mathcal{S}$.

Proposition 9.5. *If $S_i \in \mathcal{S}$, there exists a* unique *minimizing value $\Phi_i(b_i) \in (0, \infty)$ of*

$$\min_p \left\{ \frac{\alpha_i}{S_i(\frac{p}{b_i}) - \lambda_i} + \frac{p}{S_i(\frac{p}{b_i})} \right\} \tag{9.12}$$

over $p \in [0, \infty)$, for each fixed delay/power weight $\alpha_i > 0$. Hence, there is a unique optimal power for each link i (given fixed $\alpha_i > 0$, $b_i > 0$, $\lambda_i \geq 0$) and $\Phi_i(b_i)$ is well-defined.

Proof. The proof can be found in Appendix 9.7.1. □

Lemma 9.6. *Suppose $S_i \in \mathcal{S}$ (satisfying Properties 1-5 of Def. 9.3). Then, $\Phi_i(b_i)$ is* strictly increasing *in b_i (given fixed $\alpha_i \geq 0$, $\lambda_i \geq 0$). In particular, Property 5:*

$$S_i(\gamma)S_i'(\gamma) - \gamma S_i'(\gamma)^2 + \gamma S_i(\gamma)S_i'(\gamma) < 0, \tag{9.13}$$

for $\gamma = \frac{P}{b} > 0$, is key to this result.

Proof. The proof can be found in Appendix 9.7.2. □

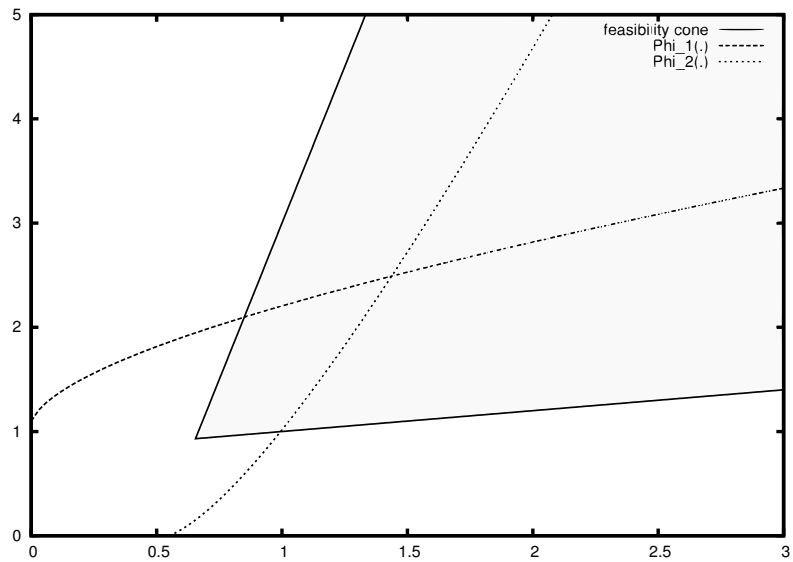


Figure 9.1.: Illustration of power cone $\mathcal{J}(\lambda)$ and power set $\mathcal{G}(\lambda)$ for a system of two links and a feasible arrival rate $\lambda = (\lambda_1, \lambda_2) \in \Lambda$ with $\lambda > 0$, operating under DPC (9.5). The horizontal axis tracks the power P_1 of link 1 and the vertical one the power P_2 of link 2. The region *above* the eventually lower curved line is where $P_2 \geq \Phi_2(b_2) = \Phi_2(\frac{N_2}{G_{22}} + \frac{G_{21}}{G_{22}} P_1)$.

9.4.3. Existence and Shape of Power Set $\mathcal{G}(\lambda)$

It is interesting to see how $\Phi_i(b_i)$ grows for large $b_i >> 1$.

Proposition 9.7. *If $S_i \in \mathcal{S}$, then*

$$\Phi_i(b_i) - \gamma_i^* b_i \approx C_i \sqrt{\alpha_i} \sqrt{b_i}, \text{ when } b_i >> 1, \tag{9.14}$$

for some constant $C_i > 0$ and γ_i^* such that $S(\gamma_i^*) = \lambda_i$.

Proof. The proof can be found in Appendix 9.7.3. □

This is depicted in Fig. 9.1 for the simple case of two links. The region *to the right* of the eventually upper curved line is where $P_1 \geq \Phi_1(b_1) = \Phi_1(\frac{N_1}{G_{11}} + \frac{G_{12}}{G_{11}} P_2)$. Each curved line is where equality is attained correspondingly. The interference $b_i(P) = \frac{N_i}{G_{ii}} + \sum_{j \neq i} \frac{G_{ij}}{G_{ii}} P_j$ is itself a function of power P. The set $\mathcal{G}(\lambda)$ is the region *above* the eventually lower curved line and *to the right* of the eventually upper curved line. The *intersection* of the two curved lines is a *fixed point* $P^* = (P_1^*, P_2^*) = (\Phi_1(b_1(P^*)), \Phi_2(b_2(P^*))) = \Phi(b(P^*))$. The shaded region is the cone $\mathcal{J}(\lambda)$ where $\frac{P_i}{b_i(P)} \geq \gamma_i^*(\lambda_i)$, so that no link queue explodes (recall $S_i(\gamma_i^*) = \lambda_i$). As per Proposition 9.7 the 'gap' $\Phi_i(b_i(P)) - \gamma_i b_i(P) \sim \sqrt{b_i(P)}$ when $b_i(p) >> 1$ (that is, for large P).

Remark 9.8. *($\mathcal{J}(\lambda) \neq \emptyset$ implies $\mathcal{G}(\lambda) \neq \emptyset$) Note that when the power cone $\mathcal{J}(\lambda)$ is* nonempty *(because $\lambda \in \Lambda$), Proposition 9.7 guarantees that the 'gap/distance' between*

[1] Of course, if $S_i(\gamma)$ is a probability it simply saturates to 1 for large γ. But, for example, $\ln(1+\gamma)$ also satisfies this condition.

[2] The function $\ln(1+\gamma)$ also belongs to $\mathcal{S}$.

the linear faces of the cone $\mathcal{J}(\lambda)$ and the corresponding (nonlinear) boundaries of the set $\mathcal{G}(\lambda)$ grows as $\sqrt{b_i}$ for large interference $b_i >> 1$ $(i \in \mathcal{L})$. Hence, the set $\mathcal{G}(\lambda)$ must also be nonempty*. An example is seen in Fig. 9.1.*

Remark 9.9. *(*DPC vs. FM*) Proposition 9.7 exposes a key difference between DPC and the FM algorithm (9.1). For $\lambda = 0$, DPC (9.4) will choose power*

$$P_i^{t+1} \sim \sqrt{b_i^t}, \tag{9.15}$$

for large interference $b_i^t >> 1$. On the contrary, FM (9.1) will choose $P_i^{t+1} = \frac{\hat{\gamma}_i}{\gamma_i^t} P_i^t = \hat{\gamma}_i b_i^t \sim b_i^t$, that is, linear in b_i^t.

9.4.4. DPC Convergence

We now address the key issue of collective converge of link powers under DPC (9.5) (and special case (9.4) with $\lambda = 0$) to a unique global equilibrium, despite the fact that each link optimizes its delay/power tradeoff selfishly (over the link set) and myopically (over time). We prove that the system,

$$P_i^{t+1} = \Phi_i(b_i^t), i \in \mathcal{L} \tag{9.16}$$

converges to a unique power equilibrium. Let us define the vector function

$$\Phi(b) = (\Phi_1(b_1), ..., \Phi_i(b_i), ..., \Phi_L(b_L)) \in \mathbb{R}_+^L \tag{9.17}$$

which yields the powers $\Phi_i(b_i)$ chosen by the links $i \in \mathcal{L}$ to apply in the next time slot, given the interference vector $b = (b_1, ..., b_i, ..., b_L) \in \mathbb{R}_+^L$ in the current time slot. The latter is induced by the vector $P = (P_1, ..., P_i, ..., P_L)$ of powers P_i used by the links $i \in \mathcal{L}$ in the current time slot via the vector function

$$b(P) = (b_1(P), ..., b_i(P), ..., b_L(P)) \in \mathbb{R}_+^L \tag{9.18}$$

where

$$b_i(P) = \frac{N_i}{G_{ii}} + \sum_{j \neq i} \frac{G_{ij}}{G_{ii}} P_j. \tag{9.19}$$

Recall that $b_i(P)$ is actually the noise plus interference rescaled by the intra-link power gain, but we refer to it concisely as 'interference' for simplicity.

The composition of the functions $\Phi(\cdot)$ and $b(\cdot)$ yields the power map

$$F(P) = \Phi(b(P)) \in \mathbb{R}_+^L \tag{9.20}$$

which is the power vector produced by DPC in the next time slot, when the power vector is P in the current one. Thus, the (synchronous) DPC algorithm can be succinctly expressed as

$$P^{t+1} = F(P^t) \tag{9.21}$$

for power vector $P^t = (P_1^t, ..., P_i^t, ..., P_L^t)$. To study the DPC convergence we focus on studying properties of the map $F(\cdot)$.

Remark 9.10. *(*Intrinsic Parameters*) Note that the receiver noise vector*

$$N = (N_1, ..., N_i, ..., N_L) \in \mathbb{R}_+^L \ . \tag{9.22}$$

is an intrinsic parameter of the map $b(\cdot)$ in (9.18), as is also the power gain matrix $\{G_{ij}, i, j \in \mathcal{L}\}$. Similarly, the delay/power weight vector

$$\alpha = (\alpha_1, ..., \alpha_i, ..., \alpha_L) \in \mathbb{R}_+^L \tag{9.23}$$

is an intrinsic parameter of the map $\Phi(\cdot)$ in (9.17), as is the set of functions $\{S_i(\cdot), i \in \mathcal{L}\}$. In general, we suppress intrinsic parameters in the notation below, unless explicitly involved in the argument under consideration.

We proceed to establish some key properties of the map $F(\cdot)$ that are essential to the convergence of the DPC algorithm.

Proposition 9.11 (Subhomogeneity). *The power map $F(\cdot)$ defined in (9.20) is*

1. *strictly increasing, i.e. $P < P' \Rightarrow F(P) < F(P')$, and*
2. *subhomogeneous, i.e. $F(\delta P) < \delta F(P)$,*

for all power vectors $P, P' \in \mathbb{R}_+^L$ and scalar $\delta > 1$. Inequalities between vectors are considered componentwise.

Proof. The proof is given in Appendix 9.7.4. □

For more on this class of functions, see e.g. [84].

We are now ready to consider the convergence of the synchronous and asynchronous versions of DPC.

Theorem 9.12 (DPC Convergence). *Suppose the link system operates under the* synchronous *or* asynchronous *Delay-Power Control (DPC) algorithm (9.5), or the special case (9.4) with $\lambda = 0$, with $S_i \in \mathcal{S}$. Let $\alpha > 0$ be the delay/power weight vector (9.23) and $N > 0$ the receiver noise vector (9.22). The system starts from arbitrary initial powers.*

Then, for each feasible arrival rate vector $\lambda \in \Lambda$ of (9.10), the power of each link $i \in \mathcal{L}$ converges to a unique *equilibrium*

$$\lim_{t \to \infty} P_i^t = P_i^{\alpha, N} \tag{9.24}$$

and the link eventually attains a unique equilibrium SINR

$$\lim_{t \to \infty} \gamma_i^t = \gamma_i^{\alpha, N} \tag{9.25}$$

correspondingly.

Proof. The proof is given in Appendix 9.7.5. □

Remark 9.13. In particular, Theorem 9.12 shows that the DPC (9.4), where we do not take arrival rates into account ($\lambda = 0$), always converges.

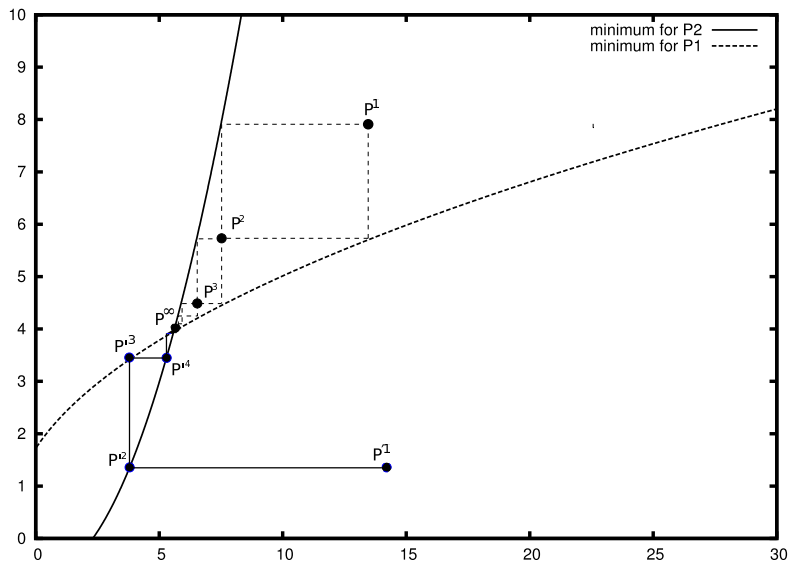


Figure 9.2.: DPC convergence to the unique equilibrium power P^∞ for an exemplary system of two links. $P^1, P^2, P^3, ...$ are the power steps of the synchronous DPC version and $P'^1, P'^2, P'^3, ...$ of the asynchronous one (superscripts denote time slots here). The axes correspond to individual powers of the two links. The curved lines are the loci where minimization in (9.5) occurs for each link correspondingly. See Fig. 9.1 for additional explanations.

Remark 9.14. *Viewing as a* generic *power control the iteration* $P_i^{t+1} = \Phi_i(b_i^t), i \in \mathcal{L}$ *with* $b_i^t = \frac{N_i}{G_{ii}} + \sum_{j \neq i} \frac{G_{ij}}{G_{ii}} P_j^t$*, one may ask under what conditions on* Φ_i *the iteration converges to a unique global equilibrium. From the proofs of Proposition 9.11 and Theorem 9.12 (and the example of Fig. 9.2), we can see that direct structural conditions on* $\Phi_i(b_i)$ *leading to the desired convergence are the following. 1) Strict* increasingness *of* $\Phi_i(b_i), i \in \mathcal{L}$*, reflecting the intuition the transmitter should respond with higher power to higher interference, as the channel is a 'congestable' medium. 2) And* $\frac{\Phi_i(b_i)}{b_i}$ *is strictly* decreasing *in* b_i*; that is,* $\Phi_i(b_i)$ *increases* sublinearly *in* $b_i, i \in \mathcal{L}$*. We do not elaborate further on this matter here.*

We have now established that (for any fixed feasible λ) the DPC algorithm converges to a unique equilibrium power

$$P^{\alpha,N} = (P_1^{\alpha,N}, ..., P_i^{\alpha,N}, ..., P_L^{\alpha,N}) \tag{9.26}$$

(for example, see Fig. 9.2) and unique equilibrium SINR

$$\gamma^{\alpha,N} = (\gamma_1^{\alpha,N}, ..., \gamma_i^{\alpha,N}, ..., \gamma_L^{\alpha,N}). \tag{9.27}$$

The power vector $P^{\alpha,N}$ induces the interference vector

$$b^{\alpha,N} = (b_1^{\alpha,N}, ..., b_i^{\alpha,N}, ..., b_L^{\alpha,N}) = b(P^{\alpha,N}) \tag{9.28}$$

Note that each link i chooses its power updates based on its own delay/power weight α_i and receiver noise N_i. However, in equilibrium, its power $P_i^{\alpha,N}$ is a function of the full vectors α and N. The other intrinsic parameters $\{G_{ij}, i, j \in \mathcal{L}\}$ and $\{S_i(\cdot), i \in \mathcal{L}\}$ and $\{\lambda_i, i \in \mathcal{L}\}$ are suppressed here for notational simplicity.

9.4.5. The Equilibrium Power $P^{\alpha,N}$

We proceed to establish some key properties of the equilibrium power vector, which provide important insights into the DPC dynamics and performance.

First, we can easily see that the powers $P_i^{\alpha,N}$ are Pareto optimal for the SINRs $\gamma_i^{\alpha,N}$ (in the sense that these SINRs could not be achieved with lower powers). Indeed, note that had $\gamma_i^{\alpha,N}$ been used as SINR targets in the FM algorithm (9.1), the latter would have converged to powers $P_i^{\alpha,N}$. But FM (9.1) is known to converge to Pareto optimal powers.

DPC Can Reach All Feasible γ

A critical issue is whether DPC can attain every *feasible* SINR vector γ; that is, any γ that can be induced by a positive (componentwise) power vector P, given the link system power gain matrix $\mathbf{G} = \{G_{ij}, i, j \in \mathcal{L}\}$ and receiver noise vector N. The space of feasible SINR vectors is

$$\Gamma = \left\{ \gamma : \gamma_i = \frac{G_{ii} P_i}{N_i + \sum_{j \neq i} G_{ij} P_j}, P_i > 0, i \in \mathcal{L} \right\}. \tag{9.29}$$

The region Γ is non-convex in general, as seen in Fig. 9.3 for example. We can rewrite the conditions in (9.29) as

$$P_i - \sum_{j \neq i} \frac{\gamma_i G_{ij}}{G_{ii}} P_j = \frac{\gamma_i N_i}{G_{ii}} \tag{9.30}$$

or in vector form

$$(\mathbf{I} - \mathbf{H}) P = U, \quad P > 0, \tag{9.31}$$

where $\mathbf{I}$ is the $L \times L$ identity matrix, $P = (P_i, i \in \mathcal{L})$ and $U = (\gamma_i N_i / G_{ii}, i \in \mathcal{L})$ are viewed as column vectors and $P > 0$ is interpreted componentwise, and

$$\mathbf{H} = \left\{ H_{ij} = \frac{\gamma_i G_{ij}}{G_{ii}}, i \neq j; \; H_{ii} = 0; \; i, j \in \mathcal{L} \right\} \tag{9.32}$$

is a matrix with zero diagonal terms and positive off-diagonal ones. From the Frobenius theory of non-negative matrices, it is well understood that (9.31) has a finite, positive, unique solution $P = (\mathbf{I} - \mathbf{H})^{-1} U$ iff the maximum modulus eigenvalue $\rho_{\mathbf{H}}$ (single and real) is less than 1. Note that $\mathbf{H}$ depends on γ (but not on N), so a vector γ is feasible iff $\rho_{\mathbf{H}} < 1$, in which case $P = (\mathbf{I} - \mathbf{H})^{-1} U$ induces this γ.

It is also well understood that if the SINR targets $\hat{\gamma}_i, i \in \mathcal{L}$ of the Foschini-Miljanic (FM) framework constitute a feasible SINR vector $\hat{\gamma}$, then the FM algorithm (9.1) will converge to a power vector $\hat{P}$ that induces exactly this $\hat{\gamma}$. Hence, the FM algorithm can 'scope out' the whole Γ.

The natural question then arises whether DPC can also 'scope out' Γ and attain any feasible γ. The answer to this important concern is positive. We develop it below, after we present a key result that we leverage for addressing the issue.

Proposition 9.15. *At equilibrium, (suppressing N for notational simplicity) the power vector P^α and SINR vector γ^α satisfy the set of equations:*

$$\alpha_i = \left(1 - \frac{\lambda_i}{S_i(\gamma_i^\alpha)}\right)^2 \left(b_i^\alpha \frac{S_i(\gamma_i^\alpha)}{S_i'(\gamma_i^\alpha)} - P_i^\alpha\right), \tag{9.33}$$

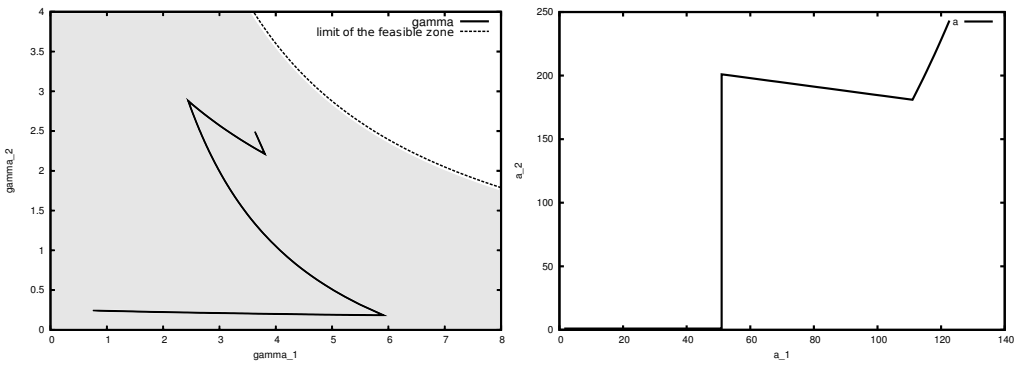


Figure 9.3.: The one-to-one mapping of $\alpha = (\alpha_1, \alpha_2)$ to $\gamma^\alpha = (\gamma_1^\alpha, \gamma_2^\alpha)$ at DPC (9.4) equilibrium for an exemplary system of two links. As α traces the trajectory shown in the bottom graph, γ^α traces the corresponding trajectory shown on the top graph, where also (part of) the boundary of the space of feasible γ is shown (upper-right curve).

or equivalently

$$P_i^\alpha \sigma_i(\gamma_i^\alpha) = \alpha_i \tag{9.34}$$

with

$$\sigma_i(x) = \left(1 - \frac{\lambda}{S_i(x)}\right)^2 \left(\frac{1}{x}\frac{S_i(x)}{S_i'(x)} - 1\right) \tag{9.35}$$

for each $i \in \mathcal{L}$. Moreover, each function $\sigma_i(x), x \geq \gamma_i^*$ is strictly increasing in x, starting from $\sigma_i(\gamma_i^*) = 0$.

Proof. The proof is given in Appendix 9.7.6. □

Note that from (9.34), if two distinct links i and j had the same rate functions $S_i(\cdot) = S_j(\cdot)$ and reached equal SINRs $\gamma_i^\alpha = \gamma_i^\alpha$ at equilibrium, then $\alpha_i/P_i^\alpha = \alpha_j/P_j^\alpha$.

Suppose now that we want DPC (9.5), or special case (9.4) with $\lambda = 0$, to reach in equilibrium a feasible SINR vector $\gamma = (\gamma_i, i \in \mathcal{L})$ with $\gamma > \gamma^*$. Since γ is feasible, we know from the discussion above that there exist a unique positive (componentwise) power vector $P = (P_i, i \in \mathcal{L})$ that induces this γ. We can thus plug the P_i and γ_i into (9.34) to produce a positive α_i for each link $i \in \mathcal{L}$. Running DPC with the so obtained delay/power weights would yield the equilibrium SINR vector γ. Therefore, by manipulating α we can reach any feasible γ. The mapping $\alpha \mapsto \gamma$ is one-to-one. This is demonstrated in Fig. 9.3.

$P^{\alpha,N}$ Variation and Scaling

Another issue is how the equilibrium power $P^{\alpha,N}$ and SINR $\gamma^{\alpha,N}$ of DPC (9.5) vary with respect to α, N and scale at the limit of those becoming very large or small.

Proposition 9.16 (Variation). *At equilibrium, the power vector $P^{\alpha,N}$ and SINR vector $\gamma^{\alpha,N}$ vary with α, N, as follows. For each link $i \in \mathcal{L}$*

1. *$P_i^{\alpha,N}$ is increasing in each $\alpha_j, j \in \mathcal{L}$.*
2. *$P_i^{\alpha,N}$ is increasing in each $N_j, j \in \mathcal{L}$.*

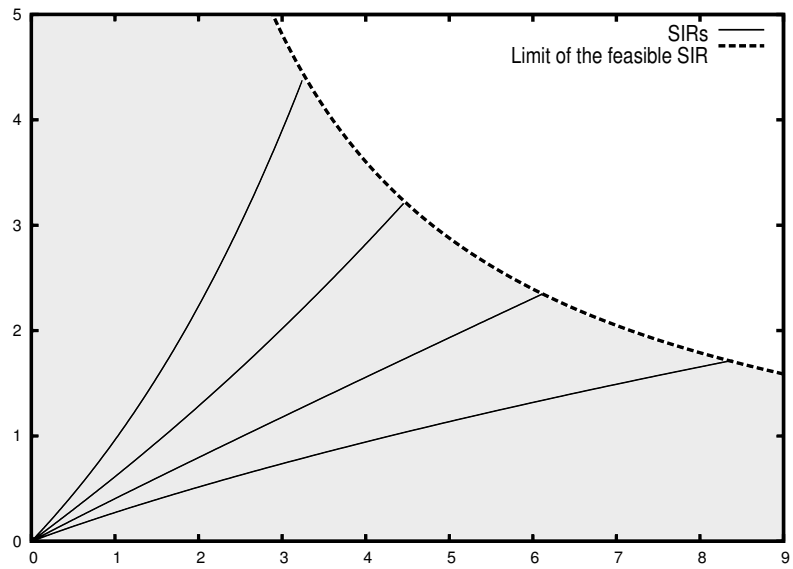


Figure 9.4.: Scaling of the DPC (9.4) equilibrium SINR $\gamma^{x\alpha}$ under delay/power weight $x\alpha$ in the feasible space Γ (whose part of its boundary is the upper-right curve) for an exemplary system of two links (same as in Fig. 9.3). The four left-to-right curves are the traces of $\gamma^{x\alpha}$ as x varies from 0 to ∞ for four fixed $\alpha = (\alpha_1, \alpha_2)$ with $\frac{\alpha_1}{\alpha_2}$ equal to .25, .5, 1, 2 correspondingly (higher to lower).

3. $\gamma_i^{\alpha,N}$ *is decreasing in each* $N_j, j \in \mathcal{L}$.

4. $\gamma_i^{\alpha,N}$ *is increasing in* α_i *and decreasing in* $\alpha_j, j \in \mathcal{L} - i$.

Proof. The proof is given in Appendix 9.7.7. □

Proposition 9.17 (Scaling). *At equilibrium, the power vector* $P^{\alpha,N}$ *and SINR vector* $\gamma^{\alpha,N}$ *scale in their parameters, as follows. Given any fixed* $\alpha, N \in \mathbb{R}_+^L$, *we have for each link* $i \in \mathcal{L}$,

$$\lim_{x\to\infty} \gamma_i^{x\alpha,N} = \lim_{y\to 0} \gamma_i^{\alpha,yN} = \gamma_i^{\alpha,*} < \infty, \tag{9.36}$$

for scalars $x, y \in \mathbb{R}$. *Also,*

$$\lim_{y\to 0} P_i^{\alpha,yN} = P_i^{\alpha,*} > 0. \tag{9.37}$$

Proof. The proof is given in Appendix 9.7.8. □

Perhaps it may seem surprising that as the receiver noise vector yN scales down to 0, DPC still operates with positive equilibrium link powers $P_i^{\alpha,*} > 0$. This simply means that the DPC equilibrium is not Pareto optimal when $N = 0$ (see Appendix 9.7.8 for DPC in this degenerate case).

9.4.6. DPC Dynamics - Joining and Leaving

An interesting issue is how DPC evolves and adapts to new links powering up and joining the channel or powering down and dropping out of it. We have explored this issue via numerical experiments in link systems (up to 6 links, randomly positioned). We have tracked the time evolution of γ_i^t and P_i^t under DPC (9.4) without arrival rates (with fixed

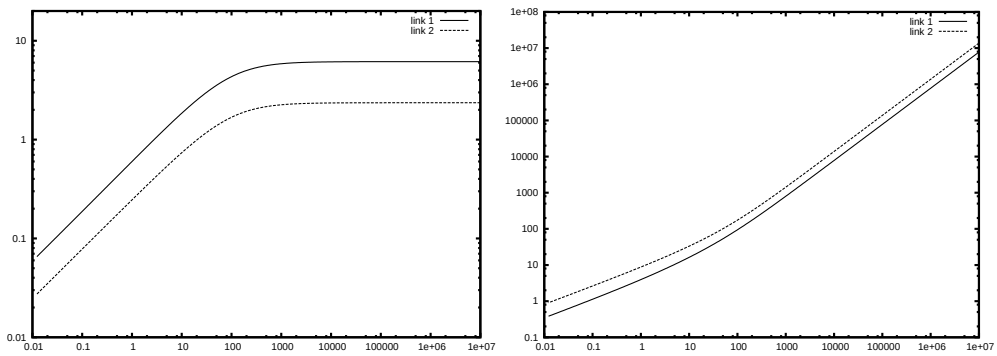


Figure 9.5.: Scaling of the DPC (9.4) equilibrium SINRs $\gamma_i^{x\alpha}$ (left) and powers $P_i^{x\alpha}$ (right), as scalar x varies from 10^{-4} to 10^7, for an exemplary system of two links (same as in Fig. 9.3, α is fixed, N is fixed and suppressed in the notation). Both axes are in log scale. Note the linear increase of $P_i^{x\alpha}$ and the saturation of $\gamma_i^{x\alpha}$ with large x, consistent with Proposition 9.17.

delay/power weight vectors α and the rest of intrinsic parameters) when links enter/exit the channel by powering up/down correspondingly. This is shown in Fig. 9.6 for a simple system of three links, which demonstrates adequately the system behavior and associated intuition (see fig. caption). Of course, it is assumed that the rate at which users join and leave is small in the time slot scale.

In Figure 9.6, it may look surprising that users with smaller SINRs have larger powers. Here is a simple explanation. Assume that $\alpha_i = a$ for all i and all functions S_i are the same. We deduce from Proposition 9.15 that $P_i\sigma(\gamma_i) = a$ for all i, with σ being 0 at the origin, increasing and continuous. This implies that, in this 'egalitarian' scenario, a node cannot have a small P_i and a small γ_i at the same time. In other words, at this equilibrium there is intrinsic propensity of users with higher delay (smaller SINR) to use higher power, trying to lower delay.

9.5. DPC-Based Protocol Design

The previous sections establish the theoretical underpinnings and key provable facts of the DPC algorithm. Leveraging these foundations, we now proceed to consider some design issues and introduce some DPC-based protocols for *efficient channel sharing* by the links. All protocols below are based on DPC (9.4), where $\lambda = 0$, as arrival rates are not taken into account and packets are assumed to always exist in the transmitter buffers.

The following discussion is based on insight gleaned from the theoretical analysis of DPC and tested via extensive numerical experiments (the most demonstrative of which we report below). We primarily take an empirical approach in this section and use numerical experiments for exploration and demonstration of the ideas.

9.5.1. Choosing α_i and Reassessing

In principle, each link $i \in \mathcal{L}$ chooses its delay/power weight α_i selfishly, expressing the required balance between delay and power that it strives to attain. This α_i could reflect inherent individual preference, belief, intrinsic utility, or simply liking (whether well

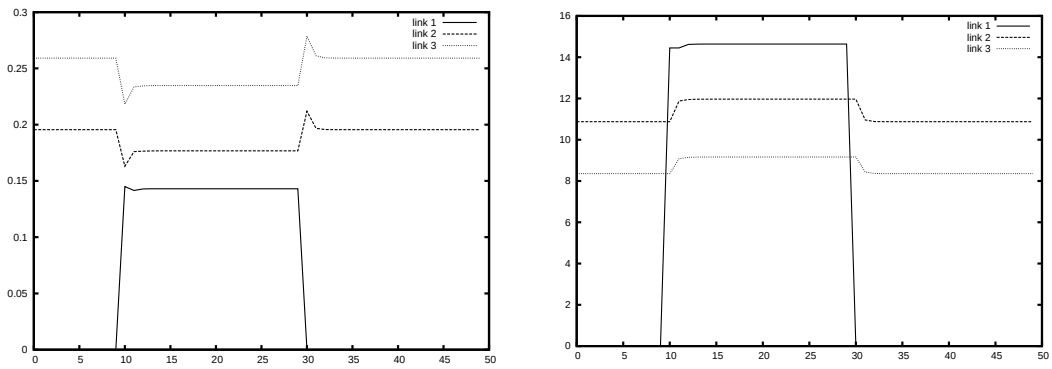


Figure 9.6.: Evolution of SINRs γ_i^t (left) and powers P_i^t (right) over time under DPC (9.4) for an exemplary system of three links. Links 2 and 3 coexist in equilibrium in the channel at time 0; link 1 powers up and joins the channel at time 10, causing the system to reach a new equilibrium, and powers down and leaves the channel at time 30, allowing the system to go back to its original equilibrium. The delay/power weight vector α is fixed. Note how fast the system adapts to entry/exit of links, reaching new equilibria for sharing the channel bandwidth.

chosen or not) in the interest of full link autonomy.

However, when the link attains its equilibrium operational point and observes its equilibrium delay vs. power balance, it may reassess its preferences, 'change its mind' and adapt, in an effort to reach a better equilibrium point. For example, it may change its α_i, or set some SINR target and invoke the FM algorithm (9.1) to attain it, or both.

We explore below two ways for each link to adjust its preferences and parameters and formulate them as DPC protocols, based on some of the DPC properties proven above.

9.5.2. The Over-Saturation Issue

Recall (from Propositions 9.16 and 9.17) that the equilibrium SINRs $\gamma_i^{x\alpha}$ increase and eventually saturate to finite $\gamma_i^{\alpha,*} < \infty$, as the scalar x increases to infinity (for any fixed delay/power weight vector α). At the same time, the equilibrium powers $P^{x\alpha}$ increase (and explode linearly eventually) and the $\gamma^{x\alpha}$ increases and approaches the boundary of the feasible SINR region Γ . We suppress N in the notation here, since it is fixed.

This behavior is demonstrated in Figs. 9.5 and 9.4. In particular, in Fig. 9.5 rapid saturation of the $\gamma^{x\alpha}$ is observed beyond a certain region around some value x_o, where a 'saturation knee' emerges and the curves become almost flat soon after it.

In general, if the links selections of α_i result in a 'large' vector α, the system could be in 'deep saturation' or *over-saturation*, that is, well beyond the saturation knee and deep in the flat region of γ_i^{α} (say, in Fig 9.5) and close to the frontier of the feasible SINR region Γ (say, in Fig. 9.4). The consequence of saturation (and especially over-saturation) is that each link is using high power with minuscule incremental return in SINR.

In view of the above, had each link i contracted its α_i somewhat to α_i', the system would have rapidly de-saturated, the $\gamma_i^{\alpha'}$ would be before (and close to) the saturation knee (see, for example, 9.5), and the $\gamma^{\alpha'}$ would have pulled away from frontier of Γ. At this 'near-saturation' regime additional power would result in substantial additional performance (SINR) on each link and would be justifiable.

We now propose two methods (formulated as protocols) for reaching near-saturation

but avoiding over-saturation.

9.5.3. Saturation-Averse DPC Protocol

We call the first protocol *Saturation-Averse (SA) DPC* (justified below). It simply augments DPC (9.4) with a subsequent phase of contracting a link system in over-saturation back to near- or sub-saturation, where power is well-spent for performance (SINR). Suppose each link $i \in \mathcal{L}$ initially chooses α_i. SA-DPC then implements two operational phases, the first of J steps, and the second of K steps, as follows.

1. *Phase 1:* During the first J steps the standard DPC algorithm is applied, which leads to equilibrium power P_i^α and SINR γ_i^α for each link. The number of steps J is taken to be long enough, so that DPC equilibrium is largely attained (DPC converges fast).

2. *Phase 2:* At the end of Phase 1, each link scales its (largely) attained equilibrium SINR γ_i^α down by a constant factor $0 < \mu < 1$ close to 1 (say, $\mu = 0.95$), akin to multiplicative decrease, and sets

$$\hat{\gamma}_i^\alpha = \mu \gamma_i^\alpha \tag{9.38}$$

   as its new SINR targets. It then applies the standard FM algorithm (9.1) for K steps. The number of steps K is taken to be long enough, so that the FM equilibrium is largely attained (FM converges fast). The FM algorithm is guaranteed to converge to $\hat{\gamma}_i^\alpha$ (at minimum power), since the $\hat{\gamma}_i^\alpha$ are automatically feasible, since derived from DPC and scaled down by μ.

SA-DPC is 'saturation averse' in the sense that, if γ^α of Phase 1 is too close to the frontier of the feasibility region Γ and the system is in over-saturation (so that nearly such γ_i^α could be reached with much smaller powers), then the contraction of the $\hat{\gamma}_i$ targets and the FM dynamics will drive the system a bit inside the feasibility region and desaturate it.

Of course, in case γ^α is already far enough from the feasibility frontier, Phase 1 will be almost without effect (provided μ is close enough to 1).

Fig. 9.7 illustrates the dynamics of SA-DPC under link joining/leaving, where the system periodically performs the two SA-DPC phases in order to cope with link entry/exit.

9.5.4. Rate-Aggressive DPC Protocol

We call the second protocol *Rate-Aggressive (RA) DPC*, because it strives to attain the highest possible SINRs γ_i (hence, rates $S_i(\gamma_i)$ also) without oversaturating and waisting power in the process. It achieves that by adapting the α_i and through those 'navigating' in the space of feasible SINRs Γ (recall that $\alpha \to \gamma$ is an one-to-one mapping), driving the system to a point close enough to the frontier of Γ to achieve high and fair SINRs but also far enough from it to avoid over-saturation and waste of power. Specifically, RA-DPC is designed to be:

(i) *generous* - it operates near saturation delivering high feasible SINRs (rates) close to the feasibility frontier of Γ;

(ii) *efficient* - does not oversaturate and waste power;

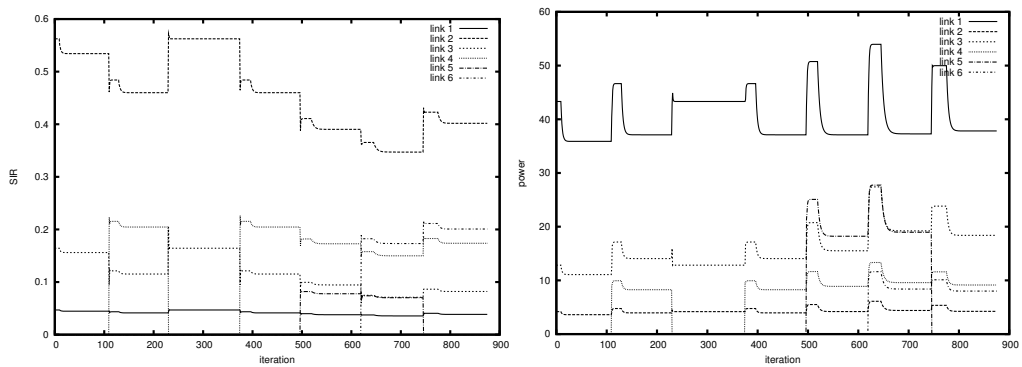


Figure 9.7.: Evolution trajectories of SINR (top) and power (bottom) under Saturation-Averse DPC for an exemplary system of 6 links, where link 4 joins the channel at time 110, leaves at time 225, and rejoins at time 375, while link 5 joins at time 500 and leaves at time 750. Note the fast readjustment of SINRs to accommodate links entering the channel, and recapture the 'void' when they leave.

(iii) *fair* - no link is starved;

(iv) *distributed* - no cross-link communication is needed.

Note that these properties are 'reminiscent of TCP' for wireline bandwidth sharing among communication sessions on a congestible link. We comment on this 'analogy' later below.

The RA-DPC (synchronous version) operates by applying repeatedly the following two phases in sequence.

1. *Phase 1 - Additive Increase of α:* At the beginning of Phase 1, each link $i \in \mathcal{L}$ increases its previous delay/power weight α_i to

$$\alpha_i' = \alpha + \Delta\alpha, \tag{9.39}$$

where the increment $\Delta\alpha > 0$ is a parameter of the RA-DPC protocol. Then the raw DPC (9.4) with α_i' is applied for J time slots, leading to equilibrium power $P_i^{\alpha'}$ and SINR $\gamma_i^{\alpha'}$. The number of steps J is chosen to be large enough so that DPC equilibrium is largely attained. This additive increase phase is designed with the desirable property (i) above in mind.

2. *Phase 2 - Multiplicative Decrease of γ:* Immediately after Phase 1, during Phase 2, each link $i \in \mathcal{L}$ performs the following operations:

   a) First, the link scales down its SINR $\gamma_i^{\alpha'}$ (attained in Phase 1) to

$$\hat{\gamma}_i = \mu\gamma_i^{\alpha'} \tag{9.40}$$

   by multiplying it by $\mu \in (0, 1)$ (multiplicative decrease). The μ is typically chosen to be close to 1 (say, 0.95).

   b) Then, the link runs for K time slots the FM algorithm (9.1), reaching equilibrium power $\hat{P}_i$. The number of steps K is chosen to be large enough so that FM equilibrium is largely attained.

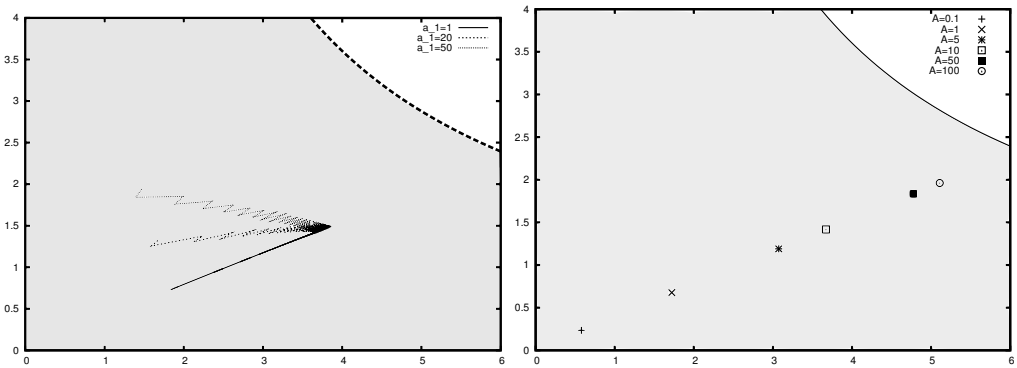


Figure 9.8.: Convergence of SINRs γ_i under *synchronous* RA-DPC for an exemplary system of two links. The multiplicative decrease parameter μ is 0.95. *Top Graph:* Evolution trajectories under RA-DPC for α=(1,1), (1,20), (1,50) - bottom to top curves correspondingly. The $\Delta\alpha$ is fixed. Note the convergence to the same equilibrium point. *Bottom Graph:* The RA-DPC equilibrium SINR points for $\Delta\alpha$=0.1, 1, 5, 10, 50, 100 - left to right points correspondingly. Note that higher (more aggressive) $\Delta\alpha$ pushes the equilibrium closer to the feasibility frontier.

c) Finally, at the end of Phase 2, the link computes

$$\hat{\alpha}_i = \hat{P}_i \left[\frac{1}{\hat{\gamma}_i} \frac{S_i\left(\hat{\gamma}_i\right)}{S'\left(\hat{\gamma}_i\right)} - 1 \right]. \tag{9.41}$$

Note that from Proposition 9.15, had each link chosen $\hat{\alpha}_i$ and applied DPC (9.4), it would have converged exactly to SINR $\hat{\gamma}_i$ in equilibrium. Based on that, the link resets its α_i to $\hat{\alpha}_i$, that is,

$$\alpha_i = \hat{\alpha}_i \tag{9.42}$$

and goes back to running Phase 1.

Note that the multiplicative decrease of SINRs by μ is designed with the desirable property (ii) above in mind. The μ is a protocol parameter that can be adjusted.

This completes the description of (synchronous) RA-DPC. It can be viewed as being 'rate-aggressive' because the additive increase Phase 1, during which it tries to progressively drive the SINR γ_i higher (hence, rate $S_i(\gamma_i)$ too) by increasing α_i and leveraging Proposition 9.16-4. On the other hand, the multiplicative decrease Phase 2 is 'saturation averse' and prevents the system from going into over-saturation.

Synchronous RA-DPC

We have gathered substantial experimental evidence (numerical) that the synchronous version of RA-DPC converges to a SINR equilibrium point in the feasibility region Γ, where each link has non-zero SINR and, hence, is not 'starved' - in line with desirable property (iii) above. In all cases analyzed (various values of $\Delta\alpha$, μ), we have consistently observed behavior similar to that depicted by Fig. 9.8 for a two link case. The equilibrium point depends on $\Delta\alpha$; the bigger $\Delta\alpha$, the closer the equilibrium SINRs to the feasibility frontier (see Fig. 9.8/bottom).

An interesting observation is that the synchronous RA-DPC appears to always converge to the *same* equilibrium point, irrespectively of the initial α (see Fig. 9.8/top). This point depends on the gain matrix $\{G_{ij}, i, j \in \mathcal{L}\}$, the receiver thermal noise vector N, and the protocol parameters $\Delta\alpha_i$ and μ. In contrast, the raw DPC (9.4) equilibrium point depends on $\{G_{ij}, i, j \in \mathcal{L}\}$, N and the delay/power weights α.

It is beyond the scope of this chapter to verify this observation by a complete mathematical proof; it is a matter of future research. However, here is a sketch of arguments that RA-DPC converges to a non-degenerate fixed point and is thus fair. (1) We first show that when α_i is multiplied by x, then γ_i will be multiplied by at most x. Thus an additive increase of α followed by a multiplicative decrease of γ cannot diverge to infinity. (2) The first argument can be repeated for all links, showing that the protocol must maintain the SINRs in a bounded region (no coordinate of this vector can tend to infinity). (3) The final step consists of deducing from the second argument that there ought to be a fixed point using again a contraction property (as we have done in the case of raw DPC).

We do not pursue further this matter here, but instead turn to the case of relaxing synchronization.

Asynchronous RA-DPC

We have experimentally investigated the asynchronous version of RA-DPC with respect to convergence and report below some interesting observations. First, recall that the raw DPC used in Phase 1 (of J time slots) and raw FM used in Phase 2 (of K slots) are both provably convergent under asynchronous updates within each phase (see Theorem 9.12 for DPC and [113] for FM). The issue then is whether lack of asynchrony of the phase boundaries across the links maintains convergence to equilibrium. This would go beyond RA-DPC having the desirable property (iv) of being distributed, to even being robust to lack of cross-link phase synchronization.

We have run numerical experiments of the asynchronous RA-DPC, where each link performs Phases 1 (J steps) and 2 (K steps) consecutively and repeatedly, but phases are 'shifted' across various links. Our general observation is that there is some good robustness to this lack of synchronization.

This is illustrated in Fig. 9.9 for a two link system with $J = K = 20$, where there is a phase shift/difference M across the two links, which varies from $M = 20$ (full synchronization) to $M = 0$ (full desynchronization; one link starts Phase 1, when the other starts Phase 2). The main observation is that for each phase-shift M (varying from 0 to 20) the asynchronous RA-DPC converges to a *limit cycle*. Although these limit cycles may be distinct for different M, they all appear in the vicinity of the synchronous RA-DPC equilibrium.

The numerical experiments allow us to conclude that even asynchronous RA-DPC exhibits good compliance with the desirable properties (i)-(iv) previously mentioned.

Of course, there is further work to be done to fully explain our findings and understand the nature of the fixed point observed in the simulation of synchronous RA-DPC.

Considering RA-DPC Variants

We have also explored numerically some RA-DPC variants on which we report below.

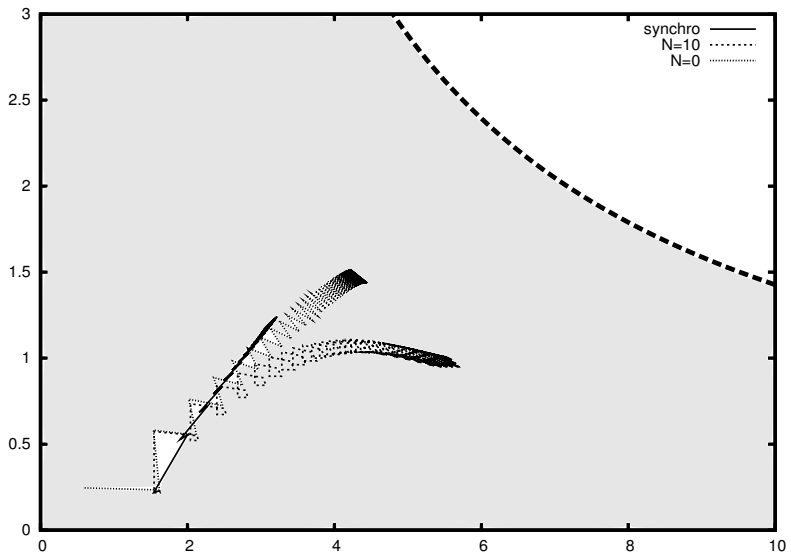


Figure 9.9.: Evolution of SINRs under *asynchronous* RA-DPC for an exemplary system of two links. The phase shift/difference between the two links is M=0(fully asynchronous; when one link starts Phase 1 the other starts Phase 2), 1 , 20 (synchronous). Note the convergence to limit cycles that are in the vicinity of the synchronous RA-DPC equilibrium.

First, we have considered the issue of whether in Phase 1 of RA-DPC additive increase of α_i could be substituted with a multiplicative increase (more aggressive). Based on numerical experiments, we report that multiplicative increase of α_i in Phase 1 does *not* work well. All link systems we experimented with were found to be unfair (even under full synchronization) in that at near-saturation, one user acquires all the bandwidth and starves the others.

Another issue we have considered is that of 'saturation' or 'stress' signals when congestion builds up. Recall that TCP on wireline session responds to saturation signals, like packet loss, to back off in a multiplicative manner and reduce congestion. In the DPC case a natural 'saturation indicator' could be the *efficiency* of a move (variation of α). Consider, for example, an increase of each α_i to α_i' moving the link system from equilibrium γ^α, P^α to $\gamma^{\alpha'}, P^{\alpha'}$. The efficiency of the move for link i is $E_i = (\gamma_i^{\alpha'} - \gamma_i^\alpha)/(P^{\alpha'} - P^\alpha)$. One expects that when the system is in saturation the efficiency E_i will be small, as even a large increase in power will result in a small increase in SINR performance. On the contrary, if the system is far enough from saturation the efficiency E_i will be large, as even a moderate increase in power will result in substantial increase in SINR. Therefore, each link could autonomously interpret low E_i as indicating saturation.

We need to explore the above idea more exhaustively, but initial results have not been encouraging. Even synchronous version of protocols implementing this idea that we have experimented with are very difficult to tune. It is not clear how to set a good rule for determining that the system is in saturation by observing the E_i; in the case of asynchronous implementations we have even observed substantial instabilities and lack of convergence to equilibrium.

Therefore, the version of DPC-based protocol that we recommend at this point is RA-DPC, which we have observed performing well under diverse operational scenarios. Other alternatives are under current investigation, but the results are too preliminary to

report on.

Note that RA-DPC is somewhat inspired by TCP dynamics. The analogy with TCP is twofold. First, RA-DPC aims at sharing bandwith for elastic traffic according to generic principles (i)-(iv) that are similar to those underpining the design of TCP. Secondly, its 'additive increase - multiplicative decrease' scheme is reminiscent of TCP Reno. Of course, there are also clear differences: TCP implements a reactive decrease whereas this is not the case for RA-DPC as proposed here. A deeper difference stems from the fact that the interaction between flows on wireless links is quite different from that in wireline networks.

9.6. Conclusions and Further Research

We have formulated an approach for incorporating delay considerations into transmitter power control, responding to interference in a wireless network. Contrary to the FM approach (9.1), the Delay-Power Control (9.5), (9.4) one balances the delay against power. It is distributed and converges to a unique global equilibrium, whose properties have been explored.

Based on the DPC properties, we have designed two protocols for efficient bandwidth sharing, and explored their behavior experimentally. Their theoretical investigation is the subject of current and future research on DPC.

Note that the minimization in (9.5) or (9.4) is over $[0, \infty)$. In practice, it would be over $[0, P_i^{max})$, where P_i^{max} is the finite power ceiling of link i. Actually, if the unique fixed point (equilibrium power vector) is in the power 'box' $[0, P_1^{max}) \times [0, P_2^{max}) \times \cdots [0, P_L^{max})$ there is no problem; DPC will still converge to it. But if the fixed point is not in the power 'box' the powers may 'stick' against and 'oscillate' on the 'box' boundaries. Whether the fixed point is in the 'box' or not is related to how congested the system is; the more congested, the further away from 0 the fixed point is and the further in danger of getting out of the 'box' (if not already out). Therefore, one needs to exercise admission control in order to prevent the unique equilibrium power to exit the power 'box.' This is another line of research we are currently pursuing.

9.7. Appendix

Here we provide the proofs of several facts stated above.

9.7.1. Proof of Proposition 9.5

For simplicity, we suppress the link index i below, since we operate on a single link in this proof. Also we often write S instead of $S(p/b)$. It suffices to show that the derivative of the cost with respect to p, that is,

$$\frac{\partial}{\partial p}\Big(\frac{\alpha}{S(p/b)-\lambda} + \frac{p}{S(p/b)}\Big) = -\frac{\alpha S'}{(S-\lambda)^2} + \frac{bS - pS'}{S^2} \tag{9.43}$$

becomes 0 at a unique point in $[0, \infty)$.

Let us define $g(p) = -\alpha S' \frac{S^2}{(S-\lambda)^2} + bS - pS'$ (eq. (9.43) times S^2). We prove below that $g(p)$ is negative for $p \approx 0$, positive for p large enough, and increasing for all p; those

properties imply that there exists a unique p such that $g(p) = 0$ and hence a unique minimum of (9.12).

Indeed, note first that for $p \approx 0$, we have $g(0) \approx -\alpha S' \frac{S^2}{(S-\lambda)^2} < 0$. Second, observe that for large $p >> 1$, we get $g(p) \approx bS(p/b) - pS'(p/b)$. Therefore, if we assume (arguing by contradiction) that $g(p) \leq 0$ for all large p, then $S(p/b) \leq \frac{p}{b} S'(p/b)$, which implies that $\frac{S'(\gamma)}{S(\gamma)} \geq \frac{1}{\gamma}$, for large γ. Integrating both sides, we get $\log S(\gamma) \geq C + \log \gamma$ implying $\log \frac{S(\gamma)}{\gamma} \geq C$ for some constant $C \in \mathbb{R}$ or equivalently $\frac{S(x)}{x} \geq \delta$ for some constant $\delta > 0$. The latter directly violates the sublinearity of $S \in \mathcal{S}$. Therefore, $\lim_{p\to\infty} g(p) > 0$.

Finally, note that for all $p \geq 0$ we have

$$g'(p) = -\alpha S'' \frac{S^2}{(S-\lambda)^2} + \alpha S' \frac{2\lambda S S'}{(S-\lambda)^3} - \frac{p}{b} S'' > 0, \tag{9.44}$$

using the increasingness and strict concavity of S. Therefore, $g(p)$ is strictly increasing for $p \geq 0$. This completes the proof.

9.7.2. Proof of Lemma 9.6

For simplicity, we suppress the link index i, since we operate on a single link in this proof. We simply write $\Phi(b)$ as the unique power that minimizes the function (9.12). The minimum is attained when the derivative (9.43) of the latter function is set to zero, hence, $\Phi(b)$ satisfies:

$$\frac{\alpha S'(\frac{\Phi(b)}{b})}{(S(\frac{\Phi(b)}{b}) - \lambda)^2} = \frac{bS(\frac{\Phi(b)}{b}) - PS'(\frac{\Phi(b)}{b})}{S^2(\frac{\Phi(b)}{b})} \tag{9.45}$$

We show that $\Phi(b)$ is increasing by establishing that $\Phi'(b) \geq 0$ for all $b \geq 0$.

Let us write $\gamma = \frac{\Phi(b)}{b}$, $S = S(\gamma)$, $S' = S'(\gamma)$, $S'' = S''(\gamma)$, $\Phi = \Phi(b)$, $\Phi' = \Phi'(b)$ and rewrite (9.45) as

$$\alpha = \Big(1 - \frac{\lambda}{S}\Big)^2 \Big(b\frac{S}{S'} - \phi\Big) \tag{9.46}$$

Using the fact that

$$\begin{aligned} \frac{\partial S(\frac{\Phi(b)}{b})}{\partial b} &= (\frac{\Phi'(b)}{b} - \frac{\Phi(b)}{b^2}) S'(\frac{\Phi(b)}{b}) \\ &= \frac{1}{b}(\Phi' - \gamma) S', \end{aligned}$$

and differentiating (9.46) w.r.t. b leads to

$$\begin{aligned} 0 = (1 - \frac{\lambda}{S})\Big(2\frac{\lambda}{b}(\Phi' - \gamma)\frac{S'}{S^2}(b\frac{S}{S'} - \Phi) + \\ (1 - \frac{\lambda}{S})(\frac{S}{S'} + \frac{b}{b}(\Phi' - \gamma)(1 - \frac{S''}{S'^2}) + \Phi')\Big). \end{aligned}$$

This can be written as:

$$\begin{aligned} \Phi'(b) = \Big(- \lambda S^2 \gamma S'' - \lambda S^2 S' - \lambda S \gamma S'^2 \\ + 2\lambda\gamma^2 S'^3 + S^3 \gamma S'' + S^3 S' - \gamma S^2 S'^2 \Big) \\ / \Big(2\gamma\lambda S'^3 + S^3 S'' - S^2 \lambda S'' - 2S\lambda S'^2\Big). \end{aligned} \tag{9.47}$$

If $\lambda = 0$, we get:

$$\Phi' = \frac{S'S - \gamma S'^2 + \gamma S S''}{S S''} \tag{9.48}$$

For all $\gamma > 0$, we have $S(\gamma) > 0$ and $S''(\gamma) < 0$ (by the strict concavity). Therefore, from (9.48), we see that $\Phi'(b) \geq 0$ for all λ implies:

$$S'(\gamma)S(\gamma) - \gamma S'(\gamma)^2 + \gamma S(\gamma)S''(\gamma) < 0\,. \tag{9.49}$$

Conversely, let us assume (9.49) and $\lambda \geq 0$. The denominator of Equation ((9.47)) can be rewritten as:

$$S^2 S''(S - \lambda) + 2\lambda S'^2(\gamma S' - S). \tag{9.50}$$

By the concavity assumption: $S'' < 0$ and $\gamma S' - S < 0$. Moreover, $S(\frac{\phi(b)}{b}) > \lambda$ by definition of DPC (9.5). Therefore the denominator of (9.47) is less than 0.

The numerator of (9.47) can be rewritten as:

$$S(S - \lambda)(\gamma S'' S + S'S - \gamma S'^2) + 2S'^2 \lambda \gamma(\gamma S' - S)\,. \tag{9.51}$$

By assumption (9.49) and the concavity of the function S, (9.51) is negative. This shows that (9.47) is positive and completes the proof.

9.7.3. Proof of Proposition 9.7

We use the notations of Appendix 9.7.2 for S and Φ.

Let us consider first the case of $\lambda > 0$. Give an interference b, the power response $\Phi(b)$ of the link satisfies the property:

$$\alpha = b\Big(1 - \frac{\lambda}{S}\Big)^2 \Big(\frac{S}{S'} - \frac{\Phi(b)}{b}\Big) \tag{9.52}$$

Using the concavity of S, $\frac{S}{S'} - \frac{\Phi(b)}{b}$ is maximum in γ^* and therefore is minimized by $\frac{\lambda}{S'(\gamma^*)} - \gamma^*$. When b goes to infinity and as the total product is equal to α, this implies that $1 - \frac{\lambda}{S}$ goes to 0. Thus $\frac{\Phi(b)}{b}$ is close to γ^*. Writing $\epsilon = \frac{\Phi(b)}{b} - \gamma^*$ and using a Taylor expansion w.r.t. ϵ, we have

$$\begin{aligned}(1 - \frac{\lambda}{S})^2 &= (1 - \frac{1}{1 + \frac{S'(\gamma^*)}{\lambda}\epsilon + O(\epsilon^2)})^2 \\ &= \frac{S'(\gamma^*)^2}{\lambda^2}\epsilon^2 + O(\epsilon^4).\end{aligned}$$

Using this, we get the following equation for ϵ:

$$\alpha \sim b\frac{S'(\gamma^*)^2}{\lambda^2}\epsilon^2\Big(\frac{\lambda}{S'(\gamma^*)} - \gamma^*\Big). \tag{9.53}$$

As $\Phi(b) = \gamma^* b + \epsilon b$, this leads to:

$$\Phi(b) - b\gamma^* \sim \sqrt{\alpha b}\frac{\lambda}{\sqrt{S'(\gamma^*)\lambda - \gamma^* S'(\gamma^*)^2}}. \tag{9.54}$$

This concludes the proof in the case $\lambda > 0$.

For $\lambda = 0$, a similar idea is used. When γ is small, $S(\gamma) = S'(0)\gamma + S''(0)\gamma^2 + O(\gamma^3)$ and $S'(\gamma) = S'(0) + 2S''(0)\gamma + O(\gamma^2)$. Solving the minimization Equation (9.45) when $\Phi(b)/b$ is small and $\lambda = 0$ leads to a second order polynomial:

$$S'(0)\gamma + S''(0)\gamma^2 = \frac{\alpha + P}{b}\big(S'(0) + 2S''(0)\gamma\big) + O(\gamma^3) \tag{9.55}$$

which leads to the equation:

$$-S''(0)\gamma^2 \sim \frac{\alpha}{b}S'(0), \tag{9.56}$$

that has the solution

$$\Phi(b) \approx_{b\to\infty} \sqrt{\alpha b}\sqrt{S'(0)/S''(0)}\,. \tag{9.57}$$

Therefore, when b tends to infinity, the SINR is small and the power $\Phi(b)$ behaves like $\sqrt{b}$. This completes the proof.

9.7.4. Proof of Proposition 9.11

For all links i, the function Φ_i is increasing. Moreover, the mapping $b : P \mapsto b(P)$ is increasing. Hence, the mapping $F = \Phi \circ b : P^t \mapsto P^{t+1}$ is increasing.

The subhomogeneity requires a little more work. For a particular link, let us define the function $h(p, b)$:

$$h(p, b) = -\alpha\frac{1}{b}\frac{S^2}{(S-\lambda)^2} + S - \frac{p}{b}S'. \tag{9.58}$$

As we have seen in the proof of proposition 9.5. $h(p, b)$ is negative if $p < \Phi(b)$, equal to 0 if $p = \Phi(b)$ and positive otherwise. Now let $p = \Phi(b)$ and $\delta > 1$. A direct compution shows that:

$$h(\delta p, \delta b) = -\alpha(\frac{1}{\delta b} - \frac{1}{b})S'(\frac{p}{b})\frac{S^2(\frac{p}{b})}{(S(\frac{p}{b}) - \lambda)^2} > 0, \tag{9.59}$$

which means that $\delta p > \Phi(\delta b)$.

Let now consider the power vector. When scaling P by δ, the thermal noise is not scaled, thus $b(\delta P) < \delta b(P)$. By monotonicity of the function Φ:

$$F(\delta x) = \Phi(b(\delta x)) < \delta F(x). \tag{9.60}$$

This completes the proof.

9.7.5. Proof of Theorem 9.12

The DPC algorithm can be viewed as a sequence of iterations of the function F of equation (9.20). This mapping belongs to the class of standard interference functions defined in [150], that is:

- For all i: $F_i(p) \geq 0$
- F is increasing (componentwise).
- F is subhomogeneous, that is, for any $\delta > 1$: $F(\delta P) > \delta F(P)$.

Therefore, using Theorems 1, 2 and 4 of Yates [150], if there exists a fixed point of $F(\cdot)$, it is unique and both asynchronous and synchronous version of DPC converges to this fixed point.

The existence of the fixed point requires more work and is rather more subtle than convergence above. Since $\lambda \in \Lambda$, we have $\mathcal{J}(\lambda) \neq \emptyset$. Then, Remark 9.8 implies that $\mathcal{G}(\lambda) \neq \emptyset$. Take a large initial power vector P^0 in $\mathcal{G}(\lambda)$, so that $P_i^0 > \Phi_i(b_i(P^0)$ for each $i \in \mathcal{L}$. This is doable because of Proposition 9.7; see also Figs. 9.2 and 9.1. In the first iteration $P^1 = F(P^0)$ the powers contract so that $P_i^1 = \Phi_i(b_i(P^0) < P_i^0$ for each $i \in \mathcal{L}$. Therefore, $P^0 > P^1$. Now apply the increasing function $F(\cdot)$ to get $F(P^0) = P^1 > F(P^1) = P^2$. Repeatedly applying this process we get $P^0 > P^1 > P^2 > P^3 > ... > P^t > P^{t+1} > ....$ This decreasing (and nonnegative) sequence of power vectors then converges to a fixed point P of $F(\cdot)$. Moreover, $b_i(P^t) \geq b_i(0) = \frac{N_i}{G_{ii}}, i \in \mathcal{L}$, hence, $P^{t+1} = \Phi(b(P^t)) \geq \Phi(b(0))$ for every t. Therefore, the fixed point $P \geq \Phi(b(0)) > 0$. This completes the proof.

9.7.6. Proofs of Proposition 9.15

These equations are just a rewriting of Equation (9.45). The monotonicity of $\sigma(x) = \left(1 - \frac{\lambda}{S(x)}\right)^2 \left(\frac{1}{x}\frac{S(x)}{S'(x)} - 1\right)$ comes from the fact that the first term of the product is clearly increasing in x while the derivative of the second term is:

$$\frac{xS'^2(x) - xS(x)S''(x) - S(x)S'(x)}{x^2 S'^2(x)} \tag{9.61}$$

which is positive, due to $S \in \mathcal{S}$ and Lemma 9.6. This completes the proof.

9.7.7. Proof of Proposition 9.16

The proofs of all the monotonicity properties of Proposition 9.16 are very similar so we will just prove the first one.

We denote by F^α the mapping that represents one iteration of the algorithm with the parameter α.

Using Equation (9.58), one can see easily that for all i, j, the mapping $\alpha_j \mapsto (F^{\alpha_j}(P^t, \alpha_j))_i$ (the power of link i after one iteration of the algorithm) is strictly increasing if $i = j$ and constant if $i \neq j$.

Moreover, as b_i is strictly increasing in every P_j for all $j \neq i$, after two iterations, if $\alpha_j < \alpha'_j$, then $(F^{\alpha_i})^2 < (F^{\alpha'_i})^2$, which shows that $\alpha_i \mapsto P^{\alpha,N}$ is increasing. Equation (9.58) cannot be satisfied by two different αs, showing that it is strictly increasing. This completes the proof.

9.7.8. Proof of Proposition 9.17

The equation (9.45) shows that if $P^{x\alpha,N}$ and $\gamma^{x\alpha,N}$ are the power and SINR at the equilibrium when parameters are $(x\alpha, N)$, then $\frac{1}{x}P^{x\alpha,N}$ and $\gamma^{x\alpha,N}$ are the power and SINR at the equilibrium when parameters are $(\alpha, \frac{1}{x}N)$. $\gamma^{x\alpha,N} = \gamma^{\alpha,\frac{1}{x}N} > \gamma^{\alpha,N}$ since γ is decreasing in N. Therefore $x \mapsto \gamma^{x\alpha,N}$ is increasing (and bounded since γ has to remain in the feasibility region), so they converge to some $\gamma^{\alpha,*}$ which corresponds in to the equilibrium with no thermal noise:

$$\lim_{x\to\infty} \gamma_i^{x\alpha,N} = \lim_{x\to\infty} \gamma_i^{\alpha,N/x} = \gamma_i^{\alpha,*} - \tag{9.62}$$

Let us call P^* (resp. γ^* and b^*) the limiting power (resp. SINR and noise) when N tends to 0. As everything is continuous in N, they satisfy the equation

$$b_i^* \frac{S(\gamma_i^*)}{S'(\gamma_i^*)} = \alpha_i \frac{S(\gamma_i^*)^2}{(S(\gamma_i^*) - \lambda)^2} + P_i^* \tag{9.63}$$

For all i, $\alpha_i > 0$ and $P_i^* \geq 0$. Hence for all i: $b_i \neq 0$. That means that at least one P_j^* is not zero which implies that none of the P_i^* is 0 since $\gamma_i^* \neq 0$.

Moreover, the DPC algorithm works in this very special case where all N_i are 0 and leads to the point γ_i^*. Of course in this particular case the algorithm is not Pareto optimal: one can reach the same γ^*'s with arbitrarily small powers by dividing all powers by the same constant. This completes the proof.

Conclusion and Bibliography

Conclusion

Throughout this document, we present different methods to control and optimize large scale systems. The main problem covered in this document is, given a stochastic model of the system, to analyze this model if the state space is too large.

A first approach, presented in Chapter 3, is to aggregate the state in a single variable, called the *potential*. The evolution of this potential depends on itself but also on another process (the number of steal requests in Chapter 3). To model the latter, we introduced an adversary that controlled this sequence of steal requests and tried to maximize the number of steal requests before running out of potential. This methodology is generic and is illustrated in Chapter 3 to study the total completion time of a large number of tasks scheduled by *work stealing*. Different scenarios are studied and simulations show that the theoretical bounds computed by our approach is sharp. We are currently investigating several extensions of these results that can also be generalized to the case of more general dependence graphs.

Although this adversary-potential approach is generic, when one considers a new problem, building an interesting function that will lead to interesting results is not easy. This is one of the reason to study mean field models, surveyed in Chapter 2. Classical *mean field models* focus on the description and the evaluation of a system. Starting from a description of the stochastic model as a system of interacting objects, it is rather easy to obtain deterministic equations that allow one to study various performance metrics. The methodology of this study is generic and is summarized in Figure 1.

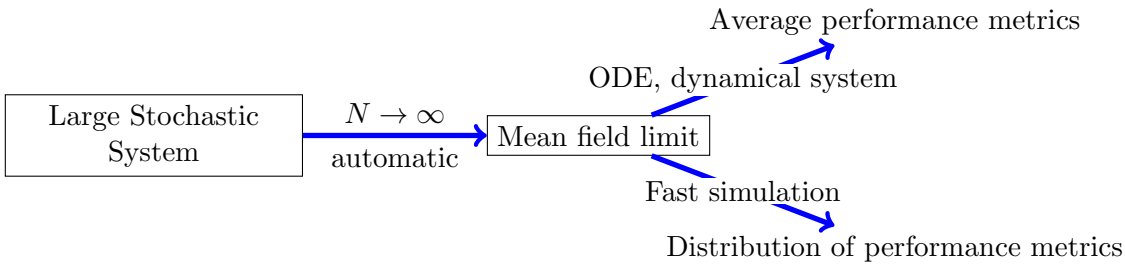


Figure 1.: The generic mean field method used to study the performance of a system, like in Chapter 4.

In particular, we applied this method in Chapter 4 to study the performance of *work stealing* in computational grids. We first build a stochastic model of the system where the N computing resources of the grid are represented by a system of N interacting objects. The computation of the deterministic limit is straightforward and can be done automatically using this representation, as in Section 4.3. Once the deterministic limit is built, it can be used to study the steady state, leading to *average performance metrics* or simulate the behavior of one object via *fast simulation*, which leads to the distribution of these performance metrics.

Mean field models have been widely used in the networking community to study the

behavior of a system. A large part of the work of this thesis has been to extend these results to the case of controlled dynamics. This is the focus of Part II where we study two kinds of problems. The first one is to compare the optimal control of a large stochastic system with the optimal control of its deterministic limit. The other problem is to study the limiting behavior of a system once a policy is fixed.

The problem of optimal control is to find a control policy that maximizes the expectation of an objective function. The approach used in Chapters 5 and 6 is summarized in Figure 2. Again, the first step of the analysis is to represent the system by a model of N interacting objects. Then, computing its mean field limit is automatic. This leads to a deterministic optimization problem. The main results of Chapters 5 and 6 of Part II are to show that solving this deterministic optimization problem leads to an asymptotically optimal solution for the original stochastic optimization problem. The problem of the limiting behavior of a controlled system is addressed in Chapter 7. We show that even when the dynamics of the policy present some discontinuity, the stochastic controlled system converges to its deterministic counterpart, described by a differential inclusion. Then, we can use the methodology of Figure 1 to perform a analysis of the system's behavior.

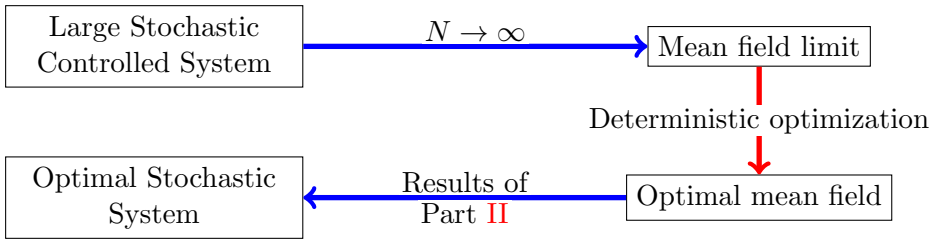


Figure 2.: The optimal mean field methodology of Part II. The blue arrows indicate an easy task, the red arrow indicates that this is not easy to solve.

The results of Part II are generic and have a wide range of applications, from brokering problems to controlled population dynamics. However, the main weaknesses of this method are the algorithmic issues coming from the deterministic optimization problem. This is due to the fact that even if the deterministic problem is often simpler than the original problem, it might still be hard to solve. In particular, there is no generic Hamilton-Jacobi-Bellman solver for an optimal control problem in d-dimensional space with $d \geq 3$. The principal interest of the optimal mean field is either to be solved for simple systems, like the one of Section 5.4 or to give intuitions on what should be the optimal policy like in Section 6.4.3. However, "*brute-force*" numerical methods seem to be hard to use.

The reason for these algorithmic issues is that we seek for an optimal control policy among all possible policies and that the set of such policies is huge. A natural extension of these results would be to restrict the choice of policy to a subset of all policies, and consider the optimization problem within this restricted choice. Using the mean field approach of Chapter 7, one can evaluate the cost of each policy on the mean field limit. Then if the set of policies considered is reasonably small, it should be straightforward to compute the best policy π_* for the deterministic limit. However, we have no results indicating that this policy should be optimal for the stochastic system, even asymptotically.

Another problem to look at would be to consider non-additive costs. For example, let us consider the problem of tasks arriving in the system and let $X(t)$ be the number of tasks at time t. By Little's formula, the average completion time is proportional to

$\sum_{t=1}^{T} X(t)$. Minimizing the average completion time is easily representable by a Markov decision processes. At each time step is associated a cost, $X(t)$, and the problem is to minimize the sum of these costs. However, if we consider a parallel application, tasks may arrive by bags of 100 or more and the real problem is not to minimize the average completion time of each task but to minimize the *total* completion time of the whole bag of tasks. This question is partially answered in Chapter 3 where we considered the total completion time of a single bag of tasks using work stealing. An interesting problem to look at would be to consider dynamics of such a system with an arrival process of bags of tasks. The problem is how to use the different resources in order to minimize the (average) total completion time of each bag of tasks.

A last but important question that should follow this work is to consider multiple objective criteria. In this work, we only focus on the problem of minimizing (or maximizing) a single real-valued utility function, like the *average response time* of tasks. Single-objective functions come naturally for someone who wants to exploit and dimension her computing resources or to minimize the delay suffered by users. However, today's problems become more and more multi-objective. One of the major issue in the near future will be energy. Someone trying to minimizing delay and energy is confronted to a dilemma: decreasing energy consumption implies increasing delays. This leads to consider cost functions of the form (D, E) (delay and energy) and raises several problems. In particular, unlike single objective functions, two solutions are not necessarily comparable. It is hard to decide if high delay and low energy consumption is better than the opposite.

A first step in this direction is done in Chapter 9, where we chose to replace the cost (D, E) by a function $E + \alpha D$ where α is a multiplicative factor that represents how each user values energy against delay. This leads every user to seek for a trade-off between the two. However, because of the difference of nature between the two costs (D, E), choosing this parameter α is very hard since we are adding two quantities that have not the same unit and are not comparable. From a theoretical point of view, a better answer is to compute the set of Pareto optimal solutions, that is solutions (D, E) such that there is no solutions (D', E') with $D' < D$ and $E' < E$. However, deciding which one is the best Pareto point is not easy.

For the moment, there is no commonly admitted solution on what should be the criteria to minimize. This is certainly something missing that should be developed in the future.

Bibliography

The number(s) at the end of each bibliographical reference indicates in which page(s) it appears.

[1] V. Acary and B. Brogliato. *Numerical methods for nonsmooth dynamical systems: applications in mechanics and electronics*. Springer Verlag, 2008. 140

[2] A.V. Aho and N.J.A. Sloane. Some doubly exponential sequences. *Fibonacci Quarterly*, 11(4):429–437, 1973. 188

[3] D. Aldous and J. Fill. Reversible Markov chains and random walks on graphs. *Book in preparation*, 2001. 16

[4] E. Altman, B. Gaujal, and A. Hordijk. Multimodularity, convexity, and optimization properties. *Mathematics of Operations Research*, 25(2):324–347, 2000. xxxii, xxxv

[5] E. Altman, B. Gaujal, and A. Hordijk. *Discrete-Event Control of Stochastic Networks: Multimodularity and Regularity*. Number 1829 in LNM. Springer-Verlag, 2003. 164

[6] J. Anselmi and B. Gaujal. Performance evaluation of a work stealing algorithm for streaming applications. In *13th Int. Conf. On Principles Of DIstributed Systems (OPODIS)*, Nîmes, 2009. 52

[7] J. Anselmi and B. Gaujal. The price of anarchy in parallel queues revisited. In *Proceedings of the ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, pages 353–354. ACM, 2010. xxxv

[8] N. S. Arora, R. D. Blumofe, and C. G. Plaxton. Thread scheduling for multiprogrammed multiprocessors. *Theory of Computing Systems*, 34(2):115–144, 2001. 34, 35, 42, 43, 48, 49

[9] Y. Azar, A. Z. Broder, A. R. Karlin, and E. Upfal. Balanced allocations. *SIAM Journal on Computing*, 29(1):180–200, 1999. 49

[10] F. Baccelli, N. Bambos, and N. Gast. Distributed delay-power control algorithms for bandwith sharing in wireless networks. *Submitted, under 2nd round of revision*, 2009. x, xxxii, xxxvi

[11] F. Baccelli and P. Bremaud. *Elements of queueing theory*. Springer-Verlag, 2003. 7, 16, 62

[12] F. Baccelli, A. Chaintreau, D. De Vleeschauwer, and D. McDonald. A mean-field analysis of short lived interacting TCP flows. In *Proceedings of the joint international conference on Measurement and modeling of computer systems*, pages 343–354. ACM, 2004. 19

[13] F. Baccelli, D. McDonald, and J. Reynier. A mean field model for multiple tcp connections through a buffer implementing red. *Performance Evaluation*, 49, 2002. 19, 138

[14] G.P. Basharin, A.N. Langville, and V.A. Naumov. The life and work of AA Markov. *Linear Algebra and its Applications*, 386:3–26, 2004. 16

[15] R.E. Bellman. *Dynamic Programming*. Princeton University Press, 1957. 16

[16] M. Benaïm. Recursive algorithms, urn processes and chaining number of chain recurrent sets. *Ergodic Theory and Dynamical Systems*, 18(01):53–87, 1998. 23

[17] M. Benaïm. Dynamics of stochastic approximation algorithms. *Séminaire de Probabilités XXXIII. Lecture Notes in Math*, 1709:1–68, 1999. 25, 116, 132

[18] M. Benaïm and M.W. Hirsch. Stochastic approximation algorithms with constant step size whose average is cooperative. *Annals of Applied Probability*, 9(1):216–241, 1999. 25

[19] M. Benaïm, J. Hofbauer, and S. Sorin. Stochastic approximation and differential inclusions. *SIAM J. Control and Optimization*, 44(1):328–348, 2005. 152

[20] M. Benaïm and J.-Y. Le Boudec. A class of mean field interaction models for computer and communication systems. *Performance Evaluation*, 65(11-12):823–838, 2008. 25, 26, 27, 80

[21] M. Benaïm and J. Weibull. Deterministic approximation of stochastic evolution in games: a generalization. Technical report, mimeo, 2003. 25

[22] M. A. Bender and M. O. Rabin. Online scheduling of parallel programs on heterogeneous systems with applications to cilk. *Theory of Computing Systems*, 35:2002, 2002. 48

[23] A. Benoit, L. Brenner, P. Fernandes, B. Plateau, and W.J. Stewart. The PEPS software tool. *Computer Performance*, pages 98–115, 2003. xxxi

[24] A. Benveniste, P. Priouret, and M. Métivier. Adaptive algorithms and stochastic approximations. *Springer-Verlag New York, Inc. New York, NY, USA*, page 365, 1990. 116

[25] P. Berenbrink, T. Friedetzky, and L. A. Goldberg. The natural work-stealing algorithm is stable. *SIAM Journal of Computing*, 32(5):1260–1279, 2003. 48

[26] P. Berenbrink, T. Friedetzky, L. A. Goldberg, P. W. Goldberg, Z. Hu, and R. Martin. Distributed selfish load balancing. *SIAM Journal on Computing*, 37(4), 2007. 48, 49

[27] P. Berenbrink, T. Friedetzky, Z. Hu, and R. Martin. On weighted balls-into-bins games. *Theoretical Computer Science*, 409(3):511 – 520, 2008. 49

[28] J. Bernard, J-L. Roch, and D. Traore. Processor-oblivious parallel stream computations. In *16th Euromicro Int. Conf. on Parallel, Distributed and network-based Processing*, Toulouse, 2008. 34

[29] J. Berstel. Sturmian and episturmian words (a survey of some recent result results). In G. Rahonis S. Bozapalidis, editor, *Conference on Algebraic Informatics*, Lecture Notes Comput. Sci. 4728, pages 23–47, 2007. 164

[30] J. Berstel, L. Boasson, O. Carton, and I. Fagnot. Sturmian trees. *Theory of Computing Systems*, pages 1–36, 2009. 164, 165, 166, 168, 170, 183

[31] J. Berstel and M. Pocchiola. Random generation of finite sturmian words. In *LIENS - 93 -8, DMI, ENS, LITP - Institute Blaise Pascal*, 1993. 188

[32] V. Berten and B. Gaujal. Brokering strategies in computational grids using stochastic prediction models. *Parallel Computing*, 2007. Special Issue on Large Scale Grids. 91

[33] V. Berten and B. Gaujal. Grid brokering for batch allocation using indexes. In *Euro-FGI NET-COOP*, Avignon, France, june 2007. LNCS. 91

[34] V. Berten and B. Gaujal. Grid brokering for batch allocation using indexes. *Network Control and Optimization*, 4465:215–225, 2007. 130

[35] D.P. Bertsekas. *Dynamic Programming and Optimal Control, vol. 1*. Athena Scientific, 1995. 16, 124

[36] D.P. Bertsekas. *Dynamic programming and optimal control, vol. II*. Athena Scientific, 2007. 16

[37] G. Bianchi et al. Performance analysis of the IEEE 802. 11 distributed coordination function. *IEEE Journal on selected areas in communications*, 18(3):535–547, 2000. xxxi, 19

[38] P. Billingsley. *Convergence of probability measures*. Wiley New York, 1968. 28

[39] D. Blackwell. Discounted dynamic programming. *The Annals of Mathematical Statistics*, pages 226–235, 1965. 16

[40] R. D. Blumofe and C. E. Leiserson. Scheduling multithreaded computations by work stealing. *Journal of the ACM*, 46(5):720–748, 1999. 48, 49

[41] A. Bobbio, M. Gribaudo, and M. Telek. Analysis of large scale interacting systems by mean field method. In *5th International Conference on Quantitative Evaluation of Systems(QEST)*, pages 215–224, St Malo, 2008. 27

[42] C. Bordenave and V. Anantharam. Optimal control of interacting particle systems. Technical Report 00397327, CNRS Open-Archive HAL, 2007. 80

[43] C. Bordenave, D. Mcdonald, and A. Proutière. Random multi-access algorithms: A mean field analysis. In *Proceedings of Allerton conference*. Citeseer, 2005. 19, 20, 31

[44] C. Bordenave, D. McDonald, and A. Proutiere. Performance of random medium access control, an asymptotic approach. In *Proceedings of the 2008 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, pages 1–12. ACM, 2008. xxxi, 31

[45] C. Bordenave, D. McDonald, and A. Proutière. A particle system in interaction with a rapidly varying environment: Mean field limits and applications. *Networks and Heterogeneous Media (NHM)*, 5(1):31–62, 2010. 27, 31, 80

[46] E. Borel. Les probabilités dénombrables et leurs applications arithmétiques. *Rend. Circ. Mat. Palermo*, 27:247–271, 1909. 179

[47] V. Borkar. *Stochastic Approximation: A Dynamical Systems Viewpoint*. Cambridge University Press, 2008. 100

[48] F. Bouchut and F. James. One-dimensional transport equations with discontinuous coefficients. *Nonlinear Analysis*, 32(7):891, 1998. 149

[49] F. Cappello, F. Desprez, M. Dayde, E. Jeannot, Y. Jégou, S. Lanteri, N. Melab, R. Namyst, P. Primet, O. Richard, et al. Grid'5000: a large scale, reconfigurable, controlable and monitorable Grid platform. *IEEE/ACM International Workshop on Grid Computing*, 2005. 53, 69

[50] J. Cassaigne. Double sequences with complexity mn+1. *J. Autom. Lang. Comb.*, 4(3):153–170, 1999. 164

[51] Q. Chen, H. Chang, R. Govindan, S. Jamin, S.J. Shenker, and W. Willinger. The origin of power-laws in internet topologies revisited. In *IEEE INFOCOM*, volume 2, pages 608–617. Citeseer, 2002. xxvii

[52] W. Chen, D. Huang, A. Kulkarni, J. Unnikrishnan, Q. Zhu, P. Mehta, S. Meyn, and A. Wierman. Approximate Dynamic Programming using Fluid and Diffusion Approximations with Applications to Power Management. In *Submitted to the 48th IEEE Conference on Decision and Control*, 2009. 116

[53] J. Cho, J.Y. Le Boudec, and Y. Jiang. On the Validity of the Decoupling Assumption for Analyzing the 802.11 MAC Protocol. Technical report, EPFL, 2010. xxxi

[54] R. Cole. Initial studies on worm propagation in manets for future army combat systems. Technical report, Pentagon Reports, 2004. 127

[55] B. Courcelle. Fundamental properties of infinite trees. *Theoretical Computer Science*, 25(2):95–169, March 1983. Fundamental study. 165, 170

[56] J.G. Dai. On positive Harris recurrence of multiclass queueing networks: A unified approach via fluid limit models. *The Annals of Applied Probability*, 5(1):49–77, 1995. 145

[57] DP De Farias and B. Van Roy. The linear programming approach to approximate dynamic programming. *Operations Research*, 51(6):850–865, 2003. 116

[58] L.E. Dubins and L.J. Savage. *How to gamble if you must: Inequalities for stochastic processes*. McGraw-Hill, 1965. 16

[59] R. Durrett. *Probability: theory and examples*. Wadsworth & Brooks/Cole, 1991. 89, 106, 108, 179

[60] A.K. Erlang. The theory of probabilities and telephone conversations. *Nyt Tidsskrift for Matematik B*, 20:33, 1909. ix, xxvii

[61] A.K. Erlang. Solution of some problems in the theory of probabilities of significance in automatic telephone exchanges. *The Post Office Electrical Engineers' Journal*, 10:189–197, 1918. ix, xxvii

[62] S.N. Ethier and T.G. Kurtz. *Markov processes: characterization and convergence*. Wiley New York, 1986. 18, 21, 22, 23, 28, 29, 30, 31, 54

[63] T. Fernique. *Pavages, Fractions continues et géométrie discrète*. PhD thesis, University of Montpellier, 2007. 164

[64] G.J. Foschini and Z. Miljanic. A simple distributed autonomous power control algorithm and its convergence. *IEEE Transactions on Vehicular Technology*, 42(4):641–646, 1993. xxiv, 196

[65] GJ Foschini and Z. Miljanic. Distributed autonomous wireless channel assignment algorithm withpower control. *IEEE Transactions on Vehicular Technology*, 44(3):420–429, 1995. 196

[66] M. Frigo, C. Leiserson, and K. Randall. The implementation of the cilk-5 multithreaded language. In *ACM SIGPLAN'98 Conf. on Programming Language Design and Implementation*, volume 33, pages 212–223, Montreal, 1998. xi, 34

[67] N. Gast and B. Gaujal. Balanced labeled trees: density, complexity and mechanicity. In *Words, 6th international conference on words*, Marseille, France, 2007. x, xxxii, xxxvi, 164

[68] N. Gast and B. Gaujal. Infinite labeled trees: From rational to Sturmian trees. *Theoretical Computer Science*, 2009. x, xxxii, xxxvi

[69] N. Gast and B. Gaujal. A mean field approach for optimization in particle systems and applications. *Fourth International Conference on Performance Evaluation Methodologies and Tools, ValueTools*, 2009. x, xxxii, xxxvi, 93

[70] N. Gast and B. Gaujal. A Mean Field Model of Work Stealing in Large-Scale Systems. *SIGMETRICS'10*, 2010. x, xxxii, xxxv, 18, 80

[71] N. Gast and B. Gaujal. A mean field approach for optimization in discrete time. *Accepted for publication in Discrete Event Dynamic Systems*, 2010. x, xxxii, xxxvi

[72] N. Gast and B. Gaujal. Mean field limit of non-smooth systems: a differential inclusion limit. *Submitted for publication*, 2010. x, xxxii, xxxvi

[73] N. Gast and B. Gaujal. Mean field limit of non-smooth systems and differential inclusions. *MAMA Workshop*, 2010. x, xxxii, xxxvi

[74] N. Gast, B. Gaujal, and J.Y. Le Boudec. Mean field for Markov Decision Processes: from Discrete to Continuous Optimization. *Submitted for publication*, 2010. x, xxxii, xxxvi

[75] B. Gaujal, A. Hordijk, and D. Van der Laan. On the optimal open-loop control policy for deterministic and exponential polling systems. *Probability in Engineering and Informational Sciences*, 21:157–187, 2007. 164

[76] B. Gaujal and E. Hyon. Optimal routing policy in two deterministic queues. *Calculateurs Parallèles*, 2001. 164

[77] B. Gaujal, E. Hyon, and A. Jean-Marie. Optimal Routing in two parallel Queues with exponential service times. *Discrete Event Dynamic Systems*, 16(1):71–107, 2006. xxi, xxxii, xxxv

[78] T. Gautier, X. Besseron, and L. Pigeon. KAAPI: A thread scheduling runtime system for data flow computations on cluster of multi-processors. In *Proceedings of PASCO'07*, pages 15–23, 2007. xi, 34, 43

[79] Z. Gorodeisky. Deterministic approximation of best-response dynamics for the Matching Pennies game. *Games and Economic Behavior*, 66(1):191–201, 2009. 123

[80] C. Graham. Chaoticity on path space for a queueing network with selection of the shortest queue among several. *Journal of Applied Probability*, 37(1):198–211, 2000. 18, 28, 31

[81] C. Graham. Chaoticity for multiclass systems and exchangeability within classes. *J. Appl. Prob*, 45:1196–1203, 2008. 21

[82] C. Graham and P. Robert. Interacting multi-class transmissions in large stochastic networks. *Annals of Applied Probability*, 2009. 21

[83] R. L. Graham. Bounds on multiprocessing timing anomalies. *SIAM Journal on Applied Mathematics*, 17:416–429, 1969. 34

[84] J. Gunawardena. *Idempotency*. Cambridge Univ Pr, 1998. 205

[85] B. Hajek. Extremal splittings of point processes. *Mathematics of Operation Research*, 10(4):543–556, 1985. 164

[86] P. Hande, S. Rangan, M. Chiang, and X. Wu. Distributed uplink power control for optimal SIR assignment in cellular data networks. *IEEE/ACM Transactions on Networking*, 16(6):1420–1433, 2008. 196

[87] D.S. Hochbaum. *Approximation algorithms for NP-hard problems*. PWS Publishing Co. Boston, MA, USA, 1996. 101

[88] W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, 1963. 103

[89] R. Hoffmann, M. Korch, and T. Rauber. Performance evaluation of task pools based on hardware synchronization. In *Proc. of Supercomputing*, 2004. 34

[90] O. Kallenberg. *Foundations of modern probability*. Springer Verlag, 2002. 58

[91] S. Kandukuri and N. Bambos. Multimodal dynamic multiple access (MDMA) in wireless packet networks. In *IEEE INFOCOM*, volume 1, pages 199–208. Citeseer, 2001. 196

[92] F.P. Kelly. *Reversibility and stochastic networks*. Wiley New York, 1979. 16

[93] W.S. Kendall. Perfect simulation for the area-interaction point process. *Probability towards*, 218:234, 2000. xxx

[94] M.H.R. Khouzani, Saswati Sarkar, and Eitan Altman. Maximum damage malware attack in mobile wireless networks. In *IEEE Infocom*, San Diego, 2010. 124, 127, 128

[95] L. Kleinrock. *Queueing Systems, Volume 1: Theory*. John Wiley & Sons, 1975. 7, 16

[96] L. Kleinrock. *Queueing Systems: Volume 2: Computer Applications*. John Wiley & Sons New York, 1976. 7, 16

[97] R. Klette and A. Rosenfeld. Digital straightness- a review. *Discrete Appl. Math.*, 139:197–230, 2004. 164

[98] D. Kondo, B. Javadi, P. Malecot, F. Cappello, and D. Anderson. Cost-benefit analysis of cloud computing versus desktop grids. In *18th International Heterogeneity in Computing Workshop*, Rome, 2009. 128

[99] A. Kukanov and M. Voss. The foundations for scalable multi-core software in intel threading building blocks. *Intel Technology Journal*, 11(4):309–322, 2007. xi, 34

[100] M. Kunze. Non-smooth dynamical systems. *Lecture notes in mathematics*, 2000. 140, 152, 153

[101] T. Kurtz. Solutions of ordinary differential equations as limits of pure jump markov processes. *Journal of Applied Probability*, pages 49–58, 1970. 127

[102] T. Kurtz. *Strong approximation theorems for density dependent Markov chains*. Stochastic Processes and their Applications. Elsevier, 1978. 80

[103] T.G. Kurtz. *Approximation of population processes*. Society for Industrial Mathematics, 1981. 21, 54

[104] H.J. Kushner and G. Yin. *Stochastic approximation and recursive algorithms and applications*. Springer Verlag, 2003. 25

[105] J.M. Lasry and P.L. Lions. Mean field games. *Japanese Journal of Mathematics*, 2(1):229–260, 2007. 21

[106] J.Y. Le Boudec, D. McDonald, and J. Mundinger. A generic mean field convergence result for systems of interacting objects. In *Quantitative Evaluation of Systems, 2007. QEST 2007. Fourth International Conference on the*, pages 3–18, 2007. 18, 19, 26, 27, 80, 81

[107] F. Lempio. Euler's method revisited. *Proceedings of the Steklov Institute of Mathematics*, 211(4):429–449, 1995. 149, 156

[108] D.A. Levin, Y. Peres, and E.L. Wilmer. *Markov chains and mixing times*. Amer Mathematical Society, 2009. 16

[109] M. Lothaire. *Algebraic combinatorics on words*. Cambridge University Press New York, 2002. 165, 175

[110] E. Luks. Isomorphism of graphs of bounded valence can be tested in polynomial time. *Journal of Computer and System Sciences*, 25:42–65, 1982. 185

[111] A.A. Markov. Rasprostranenie zakona bol'shih chisel na velichiny, zavisyaschie drug ot druga. *Izvestiya Fiziko-matematicheskogo obschestva pri Kazanskom universitete*, 2(15):135–156, 1906. 16

[112] F. Meshkati, HV Poor, SC Schwartz, and NB Mandayam. An energy-efficient approach to power control and receiver design in wireless data networks. *IEEE transactions on communications*, 53(11):1885–1894, 2005. 196

[113] D. Mitra. An asynchronous distributed algorithm for power control in cellular radio systems. *Wireless and Mobile Communications*, pages 177–186, 1994. 196, 215

[114] M. Mitzenmacher. Analyses of load stealing models based in differential equations. In *10th ACM Symposium on Parallel Algorithms and Architectures (SPAA)*, pages 212–221, 1998. 18, 48, 52, 66

[115] M. Mitzenmacher. The power of two choices in randomized load balancing. *IEEE Transactions on Parallel and Distributed Systems*, pages 1094–1104, 2001. 18

[116] M. Morse and G.A. Hedlund. Symbolic dynamics ii. sturmian trajectories. *Amer. J. Math.*, 62:1–42, 1940. 164

[117] A. Müller and D. Stoyan. *Comparison methods for stochastic models and risks*. Series in Probability and Statistics. Wiley, 2002. 68

[118] R. Nelson. *Probability, stochastic processes, and queueing theory: the mathematics of computer performance modelling*. Springer, 1995. xxx

[119] M. Norman. A central limit theorem for Markov processes that move by small steps. *The Annals of Probability*, 2(6):1065–1074, 1974. 24

[120] J. Palmer and I. Mitrani. Optimal and heuristic policies for dynamic server allocation. *Journal of Parallel and Distributed Computing*, 65(10):1204–1211, 2005. Special issue: Design and Performance of Networks for Super-, Cluster-, and Grid-Computing (Part I). 91

[121] Christos H. Papadimitriou and John N. Tsitsiklis. The complexity of optimal queueing network control. *Math. Oper. Res.*, 24:293–305, 1999. 81

[122] J.G. Propp and D.B. Wilson. Exact sampling with coupled Markov chains and applications to statistical mechanics. *Random structures and Algorithms*, 9(1-2):223–252, 1996. xxx

[123] M.L. Puterman. *Markov Decision Processes : Discrete Stochastic Dynamic Programming*. Wiley-Interscience, 2005. 10, 11, 12, 14, 15, 16, 38

[124] J.-N. Quintin and F. Wagner. Hierarchical work stealing. Technical Report 7077, INRIA, 2009. 53, 69

[125] H. Robbins and S. Monro. A stochastic approximation method. *The Annals of Mathematical Statistics*, 22(3):400–407, 1951. 25

[126] T. Rolski. Comparison theorems for queues with dependent interarrival times. In *Lecture Notes in Control and Information Sciences*, volume 60, pages 42–71. Springer-Verlag, 1983. 94

[127] S.M. Ross. *Stochastic Processes, 2nd ed.* John Wiley and Sons, Inc. USA, 1996. 132

[128] W. Rudin. *Functional Analysis.* McGraw-Hill Science, 1991. 154

[129] J.M. Rulnick and N. Bambos. Mobile power management for wireless communication networks. *Wireless Networks*, 3(1):3–14, 1997. 196

[130] S. Sano, N. Miyoshi, and R. Kataoka. m-balanced words: A generalization of balanced words. *Theoretical Computer Science*, 314(1-2):97–120, 2004. xxxv

[131] C.U. Saraydar, N.B. Mandayam, and D.J. Goodman. Efficient power control via pricing in wireless data networks. *IEEE transactions on Communications*, 50(2):291–303, 2002. 196

[132] J. T. Schwartz. *Nonlinear functional analysis.* Gordon and Breach Science Publishers, New York, 1969. 90

[133] N.J.A. Sloane et al. The On-Line Encyclopedia of Integer Sequences, 2009. 188

[134] M. Squillante and R. Nelson. Analysis of task migration in shared-memory multiprocessor scheduling. *SIGMETRICS Perf. Eval. Rev.*, 19(1):143–155, 1991. xxxi, 52

[135] S. Stanczak, M. Wiczanowski, and H. Boche. Resource Allocation in Wireless Networks-Theory and Algorithms. *Lecture Notes in Computer Science*, 4000, 2006. 196

[136] W.J. Stewart. *Introduction to the numerical solution of Markov chains.* Princeton University Press NJ, 1994. 5

[137] YM Suhov and ND Vvedenskaya. Dobrushin's mean-field approximation for a queue with dynamic routing. *Markov Processes and Related Fields*, 3(4):493–526, 1997. 18

[138] AS Sznitman. Topics in propagation of chaos. Ecole d'été de probabilités de Saint-Flour XIX-1989. *Lecture Notes in Math*, 1464:165–251, 1989. 28, 29, 30

[139] S. Tanachaiwiwat and A. Helmy. Vaccine: War of the worms in wired and wireless networks. In *IEEE INFOCOM*, 2006. 127

[140] M. Tchiboukdjian, N. Gast, D. Trystram, J.-L. Roch, and J. Bernard. A tighter analysis of work stealing. *ISAAC 2010*, 2010. x, xxxii, xxxv

[141] M. Tchiboukdjian, D. Trystram, J.-L. Roch, and J. Bernard. List Scheduling: The Price of Distribution. Research Report 7208, INRIA, 2010. 36

[142] H. Tembine, J.-Y. Le Boudec, R. El-Azouzi, and E. Altman. Mean Field Asymptotic of Markov Decision Evolutionary Games and Teams. *GameNets*, 2009. 21, 58

[143] P. Tinnakornsrisuphap and A.M. Makowski. Limit behavior of ECN/RED gateways under a large number of TCP flows. In *IEEE INFOCOM*, volume 2, pages 873–883. Citeseer, 2003. 19, 27

[144] D. Traoré, J.-L. Roch, N. Maillard, T. Gautier, and J. Bernard. Deque-free work-optimal parallel stl algorithms. In *Proceedings of Euro-Par'08*, pages 887–897, 2008. 35

[145] J.N. Tsitsiklis and B. Van Roy. An analysis of temporal-difference learning with function approximation. *IEEE Transactions on Automatic Control*, 42(5):674–690, 1997. 116

[146] R. Van Nieuwpoort, T. Kielmann, and H. Bal. Efficient load balancing for wide-area divide-and-conquer applications. *ACM SIGPLAN*, 36(7):34–43, 2001. 53, 69

[147] J.M. Vincent. Perfect simulation of monotone systems for rare event probability estimation. In *Proceedings of the 37th conference on Winter simulation*, page 537. Winter Simulation Conference, 2005. xxx, 5

[148] R. R. Weber and G. Weiss. On an index policy for restless bandits. *Journal of Applied Probability*, 27:637–648, 1990. 91

[149] P. Whittle. *A celebration of applied probability*, volume 25A, chapter Restless Bandits: activity allocation in a changing world, pages 287–298. J. Appl. Probab. Spec., 1988. 15, 91

[150] R.D. Yates. A framework for uplink power control in cellular radio systems. *IEEE Journal on Selected Areas in Communications*, 13(7):1341–1347, 1995. 220, 221

[151] J. Zander. Distributed cochannel interference control in cellular radiosystems. *IEEE Transactions on Vehicular Technology*, 41(3):305–311, 1992. 196

Abstracts

Abstract

The goal of this thesis is to provide methods for the control and the optimization of large-scale systems, starting from stochastic models that approximate its behavior. However, such models suffer from the curse of dimensionality: the number of states needed to represent a system explodes when the size of the system grows.

In a first part, we present two different methods to reduce the complexity of the model by aggregating the states of the different objects. These two methods are illustrated by analyzing the performance of a distributed load balancing strategy, namely work-stealing. We first show how the use of a potential function leads to a tight analysis of the total completion time of a bag of tasks. Then we show how a mean field approximation can be used to study the steady-state of a grid scheduled by work-stealing.

Then, we focus on the optimal control of large stochastic systems. We extend classical mean field methods to study the controlled behavior of large systems. We show that under mild assumptions, solving an optimal control problem for a system with a large number of objects can be reduced to the solving of a problem for a deterministic system as the number of objects grows large. In practice, this allows one to evaluate the performance of a policy and provide a way to asymptotically solve problems that used to be intractable.

The last part of the document studies two different problems. We first consider the question of how to distribute a infinite number of tasks on an infinite tree, as evenly as possible. Secondly we study a distributed power control algorithm in wireless network whose goal is to efficiently balance the trade-off between the delay and the power spent per transmitted packet.

Résumé

Dans cette thèse, nous étudions des méthodes permettant le contrôle et l'optimisation de systèmes à grande échelle à travers l'étude de modèles stochastiques. Les modèles classiques souffrent tous du même problème: le nombre d'états pour décrire le système explose lorsque la taille du système étudié grandit.

Dans une première partie, nous présentons deux méthodes différentes pour réduire la complexité du modèle en agrégeant les états des différents objets: l'introduction d'une fonction potentielle manipulée par un adversaire et les modèles champ moyen. Nous montrons comment ces méthodes peuvent être appliquées pour étudier le vol de travail.

Puis, nous nous intéressons au contrôle optimal de grands systèmes stochastiques. Nous étendons les résultats classiques champ moyen à des problèmes d'optimisation et de contrôle. Nous montrons que sous des hypothèses faibles, résoudre le problème d'optimisation initial est asymptotiquement équivalent à la résolution d'un problème d'optimisation déterministe. Cela permet en pratique d'évaluer les performances d'une politique de façon beaucoup plus rapide et de résoudre des problèmes jusqu'ici impossibles.

Dans la dernière partie, nous étudions deux problèmes différents. Dans un premier temps, nous considerons le problème de distribuer une infinité de tâches sur un arbre infini de manière la plus régulière possible. Puis nous étudions un mécanisme de contrôle de puissance dans des réseaux sans fils ayant pour but de trouver un compromis entre puissance consommée et délais.