



Une approche distribuée pour les problèmes de couverture dans les systèmes hautement dynamiques.

Mohamed Hamza Kaaouachi

► To cite this version:

Mohamed Hamza Kaaouachi. Une approche distribuée pour les problèmes de couverture dans les systèmes hautement dynamiques.. Réseaux et télécommunications [cs.NI]. Université Pierre et Marie Curie - Paris VI, 2016. Français. NNT : 2016PA066058 . tel-01289153v2

HAL Id: tel-01289153

<https://theses.hal.science/tel-01289153v2>

Submitted on 7 Jul 2017

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.



UNIVERSITE PIERRE ET MARIE CURIE - PARIS 6
SORBONNE UNIVERSITÉS

ÉCOLE DOCTORALE EDITE
ÉCOLE DOCTORALE INFORMATIQUE,
TÉLÉCOMMUNICATIONS ET ÉLECTRONIQUE

THÈSE

pour obtenir le grade de

Docteur en Sciences

Mention : SYSTÈMES INFORMATIQUES

présentée et soutenue par

Mohamed Hamza KAAOUACHI

**Une approche distribuée pour les problèmes de
couverture dans les systèmes hautement
dynamiques**

Thèse co-dirigée par **Franck PETIT** et **François JOUEN**
et co-encadrée par **Swan DUBOIS**

Préparée au **Laboratoire d'Informatique de Paris 6 (LIP6)**
Equipe-Projet INRIA REGAL

Thèse soutenue devant le jury composé de :

M. Eddy Caron	- MC, HDR, ENS Lyon	Rapporteur
M. Vincent Villain	- PR, UPJV	Rapporteur
Mme Lélia Blin	- MC, HDR, UPMC	Examineur
M. Marc Bui	- PR, EPHE	Examineur
M. Arnaud Casteigts	- MC, Univ. Bordeaux	Examineur
M. Swan Dubois	- MC, UPMC	Co-Encadrant
M. François Jouen	- PR, EPHE	Co-Directeur
M. Franck Petit	- PR, UPMC	Co-Directeur

Cette Thèse fut financée par le Labex SMART, lui même soutenu par des fonds de l'État français gérés par l'ANR dans le programme “Investissements d'Avenir” sous la référence ANR-11-LABX-65.

Résumé

Un système distribué est un système composé d'éléments de calcul autonomes dotés de capacité de communication. Il s'agit d'un modèle commun pour l'étude des réseaux. L'évolution rapide des réseaux sans fils et/ou mobiles aussi bien dans la vie quotidienne que dans la recherche amène progressivement à intégrer la dynamique (i.e. l'évolution dans le temps de la connectivité) dans les systèmes distribués. Concrètement, cela revient à ajouter l'hypothèse que les capacités de communication des éléments du système peuvent varier dans le temps.

De nombreux modèles considèrent ainsi la dynamique comme composante à part entière du système (et non pas comme une faute). De manière récente, une nouvelle approche, appelée graphe variant dans le temps, tente d'unifier tous ces modèles dans un formalisme commun qui permet de classer les systèmes en fonction de leurs propriétés de connexité temporelle. Dans cette thèse, nous nous intéressons à des systèmes distribués hautement dynamiques dans lesquels les hypothèses de connexité sont minimalistes. Plus précisément, nous concentrons nos efforts sur les systèmes connexes à travers le temps dans lesquels la seule garantie est que tout élément du système peut infiniment souvent envoyer un message à tout autre (sans garantie sur la pérennité de la route utilisée ni sur le délai de communication). Nous nous intéressons plus particulièrement aux problèmes de couverture (par exemple, ensemble dominant minimal, couplage maximal, ensemble indépendant maximal, ...) dans ces systèmes distribués hautement dynamiques.

Les contributions de cette thèse dans ce contexte sont les suivantes. Nous proposons tout d'abord une nouvelle définition pour les problèmes de couverture qui est plus adaptée aux systèmes distribués hautement dynamiques que les définitions existantes. Dans un deuxième temps, nous fournissons un outil générique qui permet de faciliter les preuves de résultats d'impossibilité dans les systèmes distribués dynamiques. Nous appliquons cet outil pour prouver plusieurs résultats d'impossibilité à propos de problèmes de couverture. Ensuite, nous proposons une nouvelle mesure de complexité en temps qui permet de comparer équitablement les performances de protocoles dans les systèmes distribués dynamiques. Enfin, nous donnons un algorithme de construction d'un ensemble dominant minimal dans les systèmes distribués hautement dynamiques.

Table des matières

1	Introduction	11
2	Contexte	15
2.1	Système distribué	15
2.1.1	Caractéristiques	16
2.1.2	Avantages	17
2.2	Graphes statiques	18
2.3	Modèles de graphes dynamiques	19
2.4	Problèmes distribués et systèmes dynamiques	24
3	Modèle	33
3.1	Graphe variant dans le temps	34
3.2	Familles de graphes variant dans le temps	40
3.2.1	Familles de GVT dotés de propriétés ponctuelles	43
3.2.2	Familles de GVT dotés de propriétés récurrentes	47
3.3	Modèle de communication et de calcul	54
4	Spécifications et systèmes dynamiques	61
4.1	Calculs d’empreintes	62
4.2	Ensemble dominant et couplage	63
4.3	Approches temporelles, évolutives et permanentes	64
4.3.1	Approche permanente	65
4.3.2	Approche évolutive	67
4.3.3	Approche temporelle	70
4.4	Approche ultime	72
4.5	Conclusion	77
5	Résultats d’impossibilité	79

5.1	Définitions	80
5.1.1	Sous-famille induite	80
5.1.2	Espace métrique et suite de Cauchy	82
5.1.3	Espace de graphes variant dans le temps	83
5.2	Outil de preuves d'impossibilité	85
5.3	Applications de l'outil	87
5.3.1	Empreinte ultime	87
5.3.2	Ensemble dominant minimal ultime	89
5.3.3	Mariage maximal ultime	94
5.3.3.0.1	Isthme.	94
5.4	Conclusion	97
6	Complexité	99
6.1	Mesure de complexité existantes	100
6.1.1	Complexité en messages	100
6.1.2	Complexité en temps	100
6.2	Nouvelle mesure de complexité	101
6.3	Exemple : construction de l'empreinte globale	104
6.3.1	Algorithme	104
6.3.2	Correction	105
6.3.3	Complexité	107
6.3.4	Optimalité	108
6.4	Conclusion	109
7	Construction de l'ensemble dominant ultime	111
7.1	L'algorithme EDMU	111
7.2	Conclusion	113
8	Conclusion	115

Remerciements

Pour commencer, je tiens à remercier **Franck Petit** et **François Jouen**, mes directeurs de thèse sans qui je n'aurais même pas envisagé cette aventure et sans lesquels rien de tout ceci n'aurait pu se réaliser. leurs conseils et leurs soutiens ont toujours profité à l'avancement de cette thèse et je leur en serai toujours infiniment reconnaissant.

Je tiens aussi à remercier chaleureusement **Swan Dubois**, mon encadrant durant cette thèse. Son accompagnement, ses conseils, son soutien et toutes les choses que j'ai pu apprendre grâce à lui ne peuvent se résumer en quelques lignes. Il est et restera toujours le meilleur mentor dont j'aurais pu rêver.

Je souhaiterais aussi remercier sincèrement **Vincent Villain** et **Eddy Caron**, les rapporteurs de cette thèse, pour l'attention et l'intérêt qu'ils ont porté à mes travaux, ainsi que pour leurs précieux commentaires. Je tiens également à remercier **Lélia Blin**, **Marc Bui** et **Arnaud Casteigts** pour avoir accepté de faire partie des examinateurs de ma thèse.

Je tiens particulièrement à remercier toutes les personnes qui ont travaillé avec moi durant ces trois dernières années et notamment à mon co-auteur **Nicolas Braud-Santoni**.

Je remercie aussi les membres du laboratoire **LIP6** et du laboratoire **ISIR**, et plus particulièrement **Pierre Sens** et **Maria Potop-Butucaru** avec qui les échanges ont toujours été très fructueux. Je remercie aussi **Marguerite Sos** et **Adela Kabbaklija** qui ont été au top administrativement parlant.

Bien entendu, je remercie le **Labex SMART** soutenu par des fonds de l'état français gérés par l'ANR dans le programme "Investissements d'Avenir" sous la référence **ANR-11-LABX-65** et tous ses membres pour le financement dont j'ai pu bénéficier et pour les discussions que nous avons pu avoir.

Enfin, je tiens à remercier ma famille et mes amis pour m'avoir supporté pendant ces trois ans. Je remercie ma chérie **Hanane**, pour son soutien et ses encouragements. Je remercie mon frère de m'avoir prêté de son temps sur les questions orthographiques. Je remercie mes parents, d'avoir fait de leur mieux pour que je puisse accomplir cette thèse dans les meilleures conditions possibles.

Chapitre 1

Introduction

Les communications sans fils se sont considérablement développées ces dernières années. En effet, la généralisation et la banalisation des connectivités sans fils et la simplicité d'intégration de capteurs sur différents appareils font que de plus en plus d'objets communicants font partie de notre quotidien. Désormais, humains, agents, équipements, robots, et applications interagissent de plus en plus au travers d'infrastructures hétérogènes comme les réseaux *ad-hoc* mobiles ou MANETs - de l'anglais mobile ad hoc networks -, les réseaux de capteurs et d'actionneurs (mobiles) - sensor and actuator networks -, les réseaux BANs - body area network -, les réseaux véhiculaires VANETs - vehicular ad hoc networks - et internet.

En général, les applications et les services déployés dans les réseaux s'appuient sur des caractéristiques fixes de ces réseaux, notamment leur topologie. Dans les réseaux contemporains, la topologie n'est généralement pas figée. Elle évolue plus ou moins rapidement dans le temps. Les participants (utilisateurs, appareils, liens, etc.) sont susceptibles de rejoindre, de quitter ou de se déplacer dans le réseau à des moments imprévisibles. La caractéristique commune de ces réseaux est leur *forte dynamique*, ce qui signifie que leur topologie ne cesse d'évoluer dans le temps, ce qui rend extrêmement difficile la conception d'algorithmes distribués susceptibles de fonctionner sur de telles infrastructures protéiformes.

Traditionnellement, les systèmes distribués sont modélisés par un graphe statique connexe non orienté où les sommets sont des processus (nœuds, processeurs, serveurs, etc.) et les arêtes correspondent aux liens de communication bidirectionnels. De toute évidence, un tel modèle ne convient pas aux réseaux hautement dynamiques. Par conséquent, il est nécessaire de concevoir et d'étudier des modèles théoriques capables de décrire plus fidèlement cette nouvelle caractéristique.

Dans cet environnement à caractère chaotique, des efforts constants ont été dédiés à la conception de modèles qui prennent en compte la dynamique du réseau depuis plusieurs décennies, par exemple [AKM⁺12, AE84, Fer04, SW09]. Certains travaux visent à unifier les approches citées précédemment. Par exemple, dans [BXFJ03], les auteurs introduisent le *graphe évolutif* -Evolving Graph. Ils y proposent de modéliser le temps de manière discrète par une suite de dates et la dynamique du système par une suite de graphes statiques, un pour chaque date. Plus récemment,

un autre formalisme de graphe appelé *graphe variant dans le temps* ($\mathcal{GVT}$) -Time-Varying Graph-, a été proposé dans [CFQS12]. Contrairement au graphe évolutif, le graphe variant dans le temps permet aux systèmes d'évoluer en temps continu. Toujours dans [CFQS12], les graphes variant dans le temps sont regroupés et classés en familles particulièrement selon deux critères principaux : la qualité de la connexité entre les sommets et la possibilité/impossibilité d'effectuer des tâches.

Dans cette thèse, nous nous intéressons à la faisabilité de problèmes distribués sur des environnements hautement dynamiques modélisés par des graphes variant dans le temps. Nous nous intéressons aux $\mathcal{GVT}$ dont le temps de vie est non borné et sous les plus faibles hypothèses de connexité dans la classification de $\mathcal{GVT}$ dans [CFQS12]. En termes de problèmes distribués, nous étudions principalement deux problèmes classiques. Nous étudions d'abord le problème du couplage qui désigne un ensemble d'arêtes (de couples de sommets voisins) qui n'ont pas de sommets en commun puis le problème de l'ensemble dominant qui désigne un ensemble de sommets tel que tout sommet n'appartenant pas à l'ensemble dominant possède un voisin dans l'ensemble dominant.

Le chapitre 2 est consacré à la description d'un système distribué dynamique, la présentation de certaines définitions traditionnelles et à un état de l'art des modèles dynamiques de la littérature.

Le chapitre 3 est consacré à la présentation formelle du modèle de communication et de calcul utilisé dans cette thèse et à la description détaillée de différentes notions connues des systèmes dynamiques.

Dans le chapitre 4, nous montrons que la définition traditionnelle des problèmes distribués peut être inadaptée dans un système distribué dynamique. Dans [CMM11], les auteurs présentent trois approches pour redéfinir les problèmes distribués. Ces approches sont les suivantes :

- l'approche *permanente* qui requiert le calcul d'une solution unique mais correcte à tout instant ;
- l'approche *évolutive* dont la spécification impose de calculer une nouvelle solution pour chaque évolution du graphe dynamique ; et
- l'approche *temporelle* qui consiste à calculer une solution unique sur la base du graphe sous-jacent d'un $\mathcal{GVT}$.

Ces trois approches peuvent s'avérer être inappropriées dans un environnement hautement dynamique. En effet, l'approche permanente peut ne pas exister puisqu'elle est dépendante de la connexité du graphe. L'approche évolutive n'est pas stable et est très dépendante de la fréquence d'évolution du graphe. L'approche temporelle, quant à elle, peut se montrer être inopérante ultimement.

Notre première contribution est de proposer une nouvelle approche de redéfinition des problèmes dans un système dynamique qui consiste à calculer une solution stable qui existe toujours et qui est utile ultimement. Cette approche repose sur l'existence du graphe sous-jacent ultime qui ne contient que les liens de communication qui sont infiniment souvent récurrent. Nous présenterons ensuite à travers l'exemple la redéfinition puis la spécification des problèmes de l'ensemble dominant minimal et du couplage maximal selon l'approche ultime.

Nous proposons dans le chapitre 5 un outil générique qui facilite les preuves formelles de résultats d'impossibilité dans les graphes variant dans le temps. Nous définissons d'abord une distance entre toute paire de $\mathcal{GVT}$ fondée sur la longueur du plus long préfixe en commun temporel. Cette distance permet d'étudier la convergence des suites de $\mathcal{GVT}$. Notre résultat principal consiste à montrer que, étant donné un algorithme $\mathcal{A}$ conçu pour un $\mathcal{GVT}$ et une suite de $\mathcal{GVT}$ qui converge vers un $\mathcal{GVT}$ G , alors la suite d'exécutions de $\mathcal{A}$ sur chaque $\mathcal{GVT}$ de la séquence converge également. En outre, cette dernière converge vers l'exécution de $\mathcal{A}$ sur G . Ce résultat permet de prouver des résultats d'impossibilité de la manière suivante : nous raisonnons par contradiction en supposons qu'il existe un algorithme $\mathcal{A}$ qui peut résoudre le problème étudié et supposons qu'il est possible de construire une suite de $\mathcal{GVT}$ partageant des préfixes communs qui ne cessent de croître de telle sorte que l'exécution de l'algorithme $\mathcal{A}$ n'arrive pas à converger sur ces préfixes. De ce fait, en appliquant notre résultat principal nous pouvons déduire que cette suite converge vers un graphe variant dans le temps G_ω de telle sorte que l'exécution de l'algorithme $\mathcal{A}$ sur G ne converge jamais. En d'autres termes, cet outil permet de construire formellement un contre-exemple en s'appuyant sur une limite de suite de graphes variant dans le temps. Nous montrons ensuite la pertinence de ce résultat en prouvant qu'aucun algorithme déterministe n'est capable de satisfaire les spécifications des problèmes de l'empreinte ultime, le couplage ultime et l'ensemble dominant ultime.

En algorithmique, une mesure de complexité permet de comparer les performances de différents algorithmes résolvant la même tâche. Dans les systèmes dynamiques, il existe principalement deux mesures de complexité pour calculer l'efficacité d'algorithmes distribués : la complexité en temps et la complexité en messages. Les mesures de complexité en temps existantes dans la littérature ne peuvent être utilisées que dans des familles particulières de $\mathcal{GVT}$ (synchrone et semi-synchrone) et ne conviennent pas aux systèmes asynchrones hautement dynamiques que nous étudions dans cette thèse. Notre contribution dans le chapitre 6 est de proposer une nouvelle mesure de complexité qui permet de mesurer la performance de nos algorithmes indépendamment des communications asynchrones et des changements topologiques. Nous proposons d'étendre la définition classique de la complexité de temps en rondes qui est largement adoptée dans le modèle asynchrone par passage de messages dans les systèmes statiques. Nous détaillons un exemple pour illustrer la pertinence de cette nouvelle mesure de complexité.

Notre dernière contribution, présentée dans le chapitre 7, concerne l'ensemble dominant minimal ultime. Nous prouvons une condition nécessaire et suffisante pour la résolution de ce problème par un algorithme déterministe distribué sur un $\mathcal{GVT}$ connexe à travers le temps. La complexité de cet algorithme dans un $\mathcal{GVT}$ est de l'ordre du diamètre de l'empreinte ultime (c'est-à-dire, le graphe statique ne comprenant que les arêtes infiniment souvent présentes du $\mathcal{GVT}$).

Les travaux présentés dans cette thèse ont fait l'objet de plusieurs publications [BSDKP15b, BSDKP15a, DKP15b, DKP15a].

Chapitre 2

Contexte

Sommaire

2.1	Système distribué	15
2.1.1	Caractéristiques	16
2.1.2	Avantages	17
2.2	Graphes statiques	18
2.3	Modèles de graphes dynamiques	19
2.4	Problèmes distribués et systèmes dynamiques	24

Dans ce chapitre, nous présentons dans un premier temps ce que nous entendons par un système distribué dynamique et ses caractéristiques (Voir la section 2.1).

Nous définissons le modèle de communication traditionnellement utilisé (à savoir un graphe statique) et nous définissons quelques notions de bases des graphes statiques dans la section 2.2.

Dans la section 2.3, Nous présentons différentes approches qui permettent de modéliser des systèmes distribués dynamiques au moyen de graphes.

Enfin, nous faisons un bref état de l’art sur les problèmes distribués dans des systèmes dynamiques dans la section 2.4.

2.1 Système distribué

Dans ce document nous considérons qu’un *système distribué* est un ensemble d’éléments, autonomes et décentralisés, dotés d’aptitude de communication. Ces éléments collaborent entre eux par le biais de calculs internes et d’échanges d’informations, afin de résoudre une tâche donnée. Dans ce qui suit, un élément sera désormais dénommé indifféremment par *sommet*, *entité*, *processus* ou *unité de calcul*. L’échange d’information se déroulent à travers des *liens de communication* aussi désignés par *canaux de communication*, *relations* ou *arêtes*. Un lien de communication entre une paire de sommets octroie uniquement la possibilité qu’une communication se produise entre ces deux sommets, que cela se concrétise ou non.

On dira que deux sommets sont *voisins* lorsqu'ils seront reliés au moins une fois par un lien de communication.

Nous considérons qu'un système distribué est *dynamique* à partir du moment où l'ensemble des sommets et/ou des relations entre les sommets varient dans le temps. Cela signifie que des sommets peuvent apparaître ou disparaître à tout instant ce qui peut correspondre par exemple à l'arrêt et l'allumage de périphériques par les utilisateurs. Il en va de même pour les liens de communication qui peuvent apparaître ou disparaître au cours du temps, chose qui peut se produire par exemple si les unités de calculs sont dotées de capacité de mouvement dans un environnement sans fils. En effet, dans un environnement sans fil, la mobilité d'un processus peut induire des disparitions de liens de communication dues à l'éloignement ou à l'apparition d'obstacles. Nous utiliserons communément les termes réseau, réseau dynamique et système dynamique pour désigner un système distribué dynamique.

Cette définition est suffisamment générale pour représenter toutes sortes de réseaux tel que les réseaux sociaux, les réseaux de capteurs, les réseaux véhiculaires, les réseaux satellites, les réseaux mobiles ad-hoc, les machines multi-cœurs, les réseaux routiers, les réseaux de transports, etc. Ainsi les sommets peuvent représenter tout aussi bien des micro-ordinateurs de différentes configurations, ou des processeurs de différentes puissances, que des processus plus ou moins gourmand dans un système d'exploitation. Les liens de communication peuvent être hétérogènes puisque dans un réseau entre différents éléments il peut exister des liens de communications de différentes natures. À titre d'exemple, on peut citer des réseaux routiers (autoroute, réseau national, etc.), des réseaux informatiques filaires (fibre optique, coaxial, etc.), ou des non filaires (Wi-fi, hertzien, Bluetooth, etc.).

Nous supposons toutefois dans notre modèle que les liens de communication sont bidirectionnels, qu'il n'y a pas de lien multiple entre toute paire de sommets et qu'il n'y a pas de *boucle* dans un sommet (c'est-à-dire, un lien de communication dont les deux extrémités sont le même sommet).

2.1.1 Caractéristiques

Un système distribué peut être asynchrone puisqu'il peut contenir un nombre important de composants hétérogènes. En effet, les processus ont une horloge locale dont la vitesse d'avancement peut varier d'un processus à un autre. De ce fait, les horloges des processus peuvent ne pas indiquer la même date. D'autre part, les liens de communication sont assujettis à des variations dans la vitesse de transfert d'informations. Par exemple, dans le cadre des réseaux routiers la circulation dans certaines routes peuvent varier selon la date et/ou l'heure. Il en va de même pour les réseaux informatiques tel que internet où la vitesse d'accès à certains sites web peut varier selon l'heure et/ou l'affluence.

2.1.2 Avantages

Contrairement à un système centralisé dans lequel les calculs sont concentrés pour résoudre un problème, un système distribué répartit les données et les tâches sur plusieurs unités de calcul qui collaborent afin de résoudre ce problème. Ce procédé comporte certains avantages.

Partage de ressources À l'origine cette caractéristique est très liée à la volonté de réduction des coûts. Le partage de ressources offre aussi la possibilité de concilier l'avantage de l'autonomie qu'octroie la multiplication des unités de calcul tout en conservant l'avantage essentiel des systèmes centraux, à savoir l'absence de redondance des informations et l'optimisation de l'utilisation des composants.

Échange d'information. La motivation principale derrière l'utilisation d'un système distribué consiste à permettre l'échange d'information à grande échelle. À titre d'exemple, grâce à internet des centaines de millions de personnes peuvent communiquer et partager une énorme quantité de données à travers des e-mails, de la voix sur IP, du chat, du Cloud, etc.).

Capacité accrue. Dans un système distribué, on a la possibilité d'accéder à une grande capacité (de calcul et de mémoire). En effet, dans un système distribué l'utilisateur dispose de l'ensemble des ressources réparties sur le système distribué. Néanmoins, ceci implique qu'il faut résoudre tous les problèmes de concurrence lors des demandes d'accès simultanés aux ressources.

Fiabilité. Une *faute* est une erreur/défaillance temporaire ou permanente d'un élément du système distribué. Lorsque les calculs sont concentrés sur un seul processeur et que ce dernier subit une faute, l'ensemble du service que propose la machine est interrompu. En répartissant la charge de calcul entre les différents processeurs, le système peut continuer à proposer ses services ou une partie de ses services dans certains cas même en présence de fautes temporaires ou permanentes de certains composants du système distribué.

Évolutivité. Puisque les systèmes distribués fonctionnent à travers des composants hétérogènes, ils sont intrinsèquement évolutifs. Contrairement au système centralisé où l'évolution est parfois synonyme à un changement du matériel à cause des contraintes liés à l'évolution de l'architecture matérielle, les systèmes distribués peuvent évoluer avec de simple ajout de nouveaux éléments aux systèmes. Par exemple, on peut faire évoluer certaines caractéristiques dans un réseau local en ajoutant simplement de nouvelles machines (de différentes architectures, de différentes puissances de calcul, et de différentes capacités) au réseau pour faire évoluer la capacité de mémoire et/ou de calcul global du système.

2.2 Graphes statiques

Traditionnellement, nous utilisons un graphe statique pour modéliser les capacités de communication d'un système distribué. Nous commençons donc par quelques définitions établies dans la théorie des graphes qui seront utiles pour la compréhension globale.

Définition 2.1 *Un **graphe non-orienté** g est un couple $g = (V, E)$ composée d'un ensemble de **sommets** V , d'un ensemble d'**arêtes** $E \subset V \times V$ (E est un ensemble de paire d'éléments de V). On dira que deux sommets p et q sont des **extrémités** d'une arête si $\{p, q\} \in E$.*

*Une **boucle** est une arête dont les deux extrémités sont le même sommet.*

*Les **arêtes multiples** sont des arêtes qui partagent les mêmes sommets comme extrémités.*

*Un **graphe simple non-orienté** est un graphe non-orienté ne comprenant ni boucle ni arêtes multiples.*

Dans cette thèse, le terme **graphe statique** désignera un graphe simple non orienté.

Définition 2.2 *On dira qu'une arête $e \in E$ d'un graphe $g = (V, E)$ est **consécutives** à une autre arête $e' \in E : e' \neq e$ si e et e' partagent une même extrémité.*

*Dans un graphe statique $g = (V, E)$, un **chemin** entre deux sommets p et q est une suite d'arêtes consécutives reliant le sommet p au sommet q .*

*La **longueur** d'un chemin est le nombre d'arêtes du chemin.*

Définition 2.3 *Étant donné un graphe statique $g = (V, E)$, une **distance** entre deux sommets p et q de V est la longueur de plus court chemin qui lie le sommet p au sommet q dans le graphe (V, E) .*

Définition 2.4 *Dans un graphe statique $g = (V, E)$, le **diamètre** est la plus longue distance entre deux sommets de V .*

Définition 2.5 *Dans un graphe statique $g = (V, E)$, un graphe est **biparti** si son ensemble de sommets peut être réparti en deux sous ensembles U et W tels que chaque arête de E a une extrémité dans U et l'autre dans W .*

Définition 2.6 *Un graphe statique $g = (V, E)$ est dit **complet** si et seulement si entre toutes paires de sommets p et q de V il existe une arête dans E dont les extrémités sont p et q .*

Définition 2.7 *Étant donné un graphe statique $g = (V, E)$, on dit que deux sommets p et q de V sont **connectés** si et seulement si il existe un chemin entre p et q dans g . Un graphe statique est **connexe** si quels que soient les sommets p et q de V , p et q sont connectés.*

Définition 2.8 Un **arbre** est un graphe acyclique connexe.

Une **forêt** est un graphe dont toutes les composantes connexes sont des arbres. Un **arbre couvrant** d'un graphe statique $g = (V, E)$ est un arbre qui contient tous les sommets de V .

Définition 2.9 Soit un graphe statique $G = (V, E)$. Le graphe $G' = (V', E')$ est un **sous graphe** de G si E' est inclus dans E et V' est inclus dans V .

Le graphe $G'' = (V, E'')$ est un **sous graphe couvrant** (ou **graphe partiel**) de G si E' est inclus dans E .

Tout comme les capacités de communication d'un système distribué statique sont naturellement représentées par un graphe statique dans bons nombres de travaux, le modèle de communication d'un système distribué dynamique peut naturellement être défini par un graphe dynamique. Dans la section suivante nous allons voir quelques modèles de communications de graphes dynamiques qui ont été utilisés pour représenter les capacités de communication de certains systèmes distribués dynamiques.

2.3 Modèles de graphes dynamiques

Les réseaux modernes s'orientent de plus en plus vers les communications sans fils qui octroient une certaine liberté de mouvements. En observant par exemple des réseaux sans fils, on peut constater que les déplacements d'appareils interconnectés entre eux par des technologies sans fils peuvent amener à la dynamique du système. En effet, les déplacements d'appareils dans et hors de la portée de communication les uns des autres font évoluer la topologie de ces réseaux dans le temps. Même, les réseaux statiques sans fil peuvent subir des évolutions topologiques et donc être dynamiques. Par exemple, dans un souci d'économie d'énergie sur un réseaux de capteurs, il est possible que ces derniers aient, au cours de leur vie, des cycles de repos pendant lesquels ils sont éteints. Ces cycles de repos sont source de dynamique dans ce type de réseaux.

Dans [HG97], on peut observer que les auteurs classifient la dynamique d'un graphe selon les entités qui peuvent changer au fil du temps comme suit :

- en graphe non orienté (ou orienté) à sommets dynamiques où l'ensemble des sommets varient dans le temps. En d'autres termes, au fil du temps il existe quelques sommets qui peuvent rejoindre le graphe ou des sommets qui peuvent le quitter voir les deux. Notons que lors de la disparition d'un sommet l'ensemble des arêtes (ou arcs) qui le lie à d'autres sommets disparaissent aussi.
- en graphe non orienté (ou orienté) à arêtes (ou arc) dynamiques tel que l'ensemble des arêtes (ou des arcs) peuvent varier au fil du temps, c'est-à-dire qu'il existe quelque arêtes (ou arcs) qui peuvent rejoindre le graphe ou le quitter.
- en graphe à sommets à poids dynamique lorsque les sommets sont pondérés et que leurs poids peut varier dans le temps.



FIGURE 2.1 – Une illustration d’un arbre dynamique par une suite de graphes qui représente l’évolution de l’arbre dynamique dans le temps. Le graphe évolue par l’ajout ou la suppression de nouveaux éléments tout en gardant la propriété d’être un arbre.

— en graphe à arête (ou arc) à poids dynamique lorsque les arêtes sont pondérées et que leurs poids peut varier dans le temps aussi.

La modélisation de cette dynamique et la conception d’algorithmes qui fonctionnent sous ces environnements ont reçu une attention considérable ces dernières années [AKL08, Ber96, CF04, CFMS10, Fer02, KKK00, ST81, ZT87]. Les différents concepts de graphes dynamiques que nous allons voir ci-après permettent de modéliser la dynamique d’un système, allant des réseaux ad-hoc sans fils en informatique, aux réseaux neuronaux en neurologie, en passant par les réseaux véhiculaires, les réseaux de transport, les réseaux satellites, les réseaux épidémiologiques et bien d’autres. Si ces réseaux sont considérés comme dynamiques dans la littérature c’est principalement dû au changement de leur structure au fil du temps.

Arbres dynamiques. Un modèle d’*arbre dynamique* [ST81] permet de représenter un système dynamique dont la topologie peut changer à condition de rester toujours un arbre. En d’autres termes, dans le modèle d’arbre dynamique il est possible de changer la racine de l’arbre ou de rajouter/supprimer un sommet ou un sous arbre dans l’arbre initial. Notez qu’une insertion ou une suppression d’arête tout en gardant la structure d’une forêt implique l’ajout d’un sommet ou d’un sous arbre dans la forêt initiale. La figure 2.1 illustre un exemple d’arbre dynamique.

Graphes aléatoires dynamiques Le modèle de *graphes aléatoires dynamiques* peut définir un système dont la dynamique est récurrente mais aléatoire. En effet, dans ce modèle un graphe de rang $i + 1$ est calculé à partir du graphe de rang i avec une certaine probabilité. Ce type de graphe est markovien et il est souvent synchrone. Dans [CF04], la dynamique d’un graphe aléatoire suit le raisonnement suivant : le graphe G_t de l’étape t est construit à partir du graphe de l’étape précédente G_{t-1} par l’ajout d’un nouveau sommet t et m arêtes aléatoires entre le sommet t et les sommets de G_{t-1} . Tandis que dans [AKL08], il est question d’un graphe aléatoire dynamique avec un nombre fixe de sommets où la dynamique du graphe réside dans l’ajout et la suppression d’arête à chaque étape. Des modèles tels que [BCF11, CCD⁺15, CMM⁺08] et bien d’autres reprennent l’hypothèse où la dynamique intervient uniquement au niveau des liens, plus précisément il est question d’un nombre fixe de sommets et d’une distribution aléatoire d’arêtes de telle sorte qu’à chaque étape, chaque arête, présente ou non, change de situation selon deux processus d’état markovien avec une probabilité p d’apparaître et une probabilité q de disparaître (c’est-à-dire, si une arête

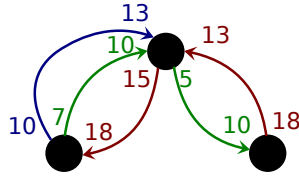


FIGURE 2.2 – Un exemple d'un réseau planifié. Les arêtes sont étiquetées par des dates de départ et d'arrivée des communications.

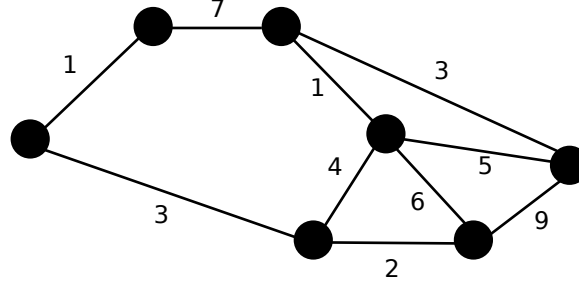


FIGURE 2.3 – Cet graphe illustre un réseau temporel à travers l'étiquetage des arêtes par les dates qui exprime les instants durant lesquelles elles peuvent être utilisées.

existe à une étape t il existe une probabilité q pour qu'elle disparaisse à l'étape $t + 1$ et si une arête n'existe pas à une étape t elle a une probabilité p d'apparaître à l'étape $t + 1$).

Réseaux planifiés Plusieurs problèmes dans les réseaux sont basés sur des relations fondamentales impliquant le temps comme les calendriers des vols aériens et des transports en commun. Dans ce contexte le modèle de *réseaux planifiés* -scheduled network- ou les réseaux d'arêtes planifiés -edge-scheduled networks- fut introduit dans [Ber96]. Ce modèle est représenté par un multigraphe orienté où à chaque arc sont associés deux dates, le temps de départ et le temps d'arrivée. La figure 2.2 illustre un exemple de réseaux planifiés.

Réseaux temporels Dans le même contexte que les réseaux planifiés [Ber96] et afin de mieux représenter des réseaux tel que la propagation d'une épidémie, les réseaux de transport planifiés etc., le modèle de *réseaux temporels* fut introduit dans [KKK00] pour modéliser à travers un graphe ces types particuliers de réseaux. Dans ce modèle on représente la dynamique d'un système par un graphe statique dont les arêtes sont étiquetées par des dates ponctuelles de telle sorte que la date sur chaque arête donne l'instant pendant lequel ses deux sommets extrémités peuvent communiquer.

La figure 2.3 illustre un exemple de réseaux temporels.

Réseaux espace-temps Les *réseaux espace-temps* -space-time networks- furent introduits par [ZT87] pour modéliser le flux des trafics (au cours du temps) de réseaux de transports routiers dotés d'une certaine capacité et composés de voies à sens unique et de voies à double sens. Dans ce modèle on considère un graphe orienté pondéré, où chaque arc a un poids, le poids de chaque arc représente le temps nécessaire pour le

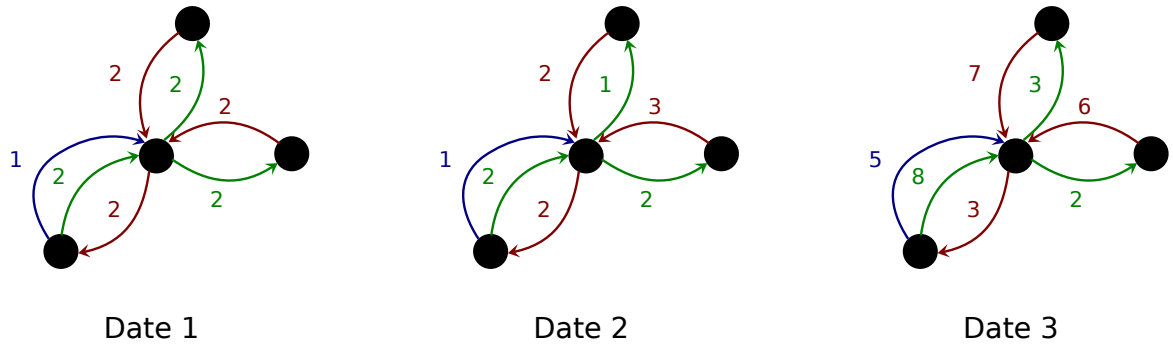


FIGURE 2.4 – Une illustration de l’évolution d’un réseau routier à travers un réseau espace-temps au fil du temps où seul le poids des arêtes est géré. Dans cet exemple, on remarque que les arcs vers le sommet central à la date 3 sont plus lourds ce qui peut exprimer un fort trafic dans ces arcs là à ce moment.

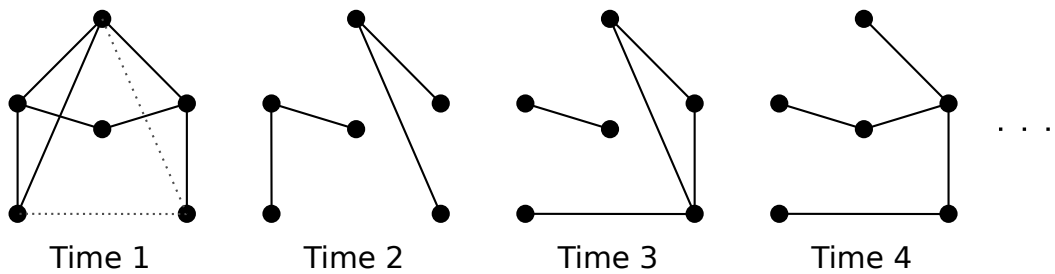


FIGURE 2.5 – Cette figure illustre la dynamique du modèle de graphe évolutif à travers une suite de graphes statiques. Chaque graphe correspond à la situation du système à l’instant Time “i”.

traverser, de telle sorte que ce poids peut changer au fil du temps. On y suppose que les arcs sont dotés de capacité à savoir qu’il existe une borne supérieure sur le nombre d’éléments qui peuvent y entrer et le traverser par unité de temps. Si cette capacité est excédée alors l’élément restant doit attendre au niveau du sommet ou suivre une autre route. La figure 2.4 illustre un exemple de réseau espace-temps.

Graphes évolutifs Le modèle connu sous le nom de *graphe évolutif* fut introduit pour la première fois dans [Fer02]. Son objectif fut d’exprimer, à travers un modèle combinatoire de référence, les réseaux dotés de caractéristiques qui évoluent dans le temps (comme l’évolution de l’ensemble des sommets et/ou de l’ensemble des arêtes). Pour sa simplicité, ce modèle de graphe est largement utilisé dans la littérature pour modéliser la dynamique des systèmes dynamiques pour n’en citer que quelques uns [BXFJ03, DRm14, IW13, RBK07]. Un graphe évolutif peut être vu comme une suite ordonnée de graphes statiques où chaque graphe de la suite exprime l’état du réseau à un instant donné. Il peut tout aussi bien être exprimé par un couple $g = (V, E)$ comme dans [KO11] où V correspond à l’ensemble des sommets du graphe et E à une fonction dynamique sur les arêtes qui procurent à chaque étape t l’ensemble des arêtes qui sont disponibles. La figure 2.5 illustre un exemple de graphe évolutif.

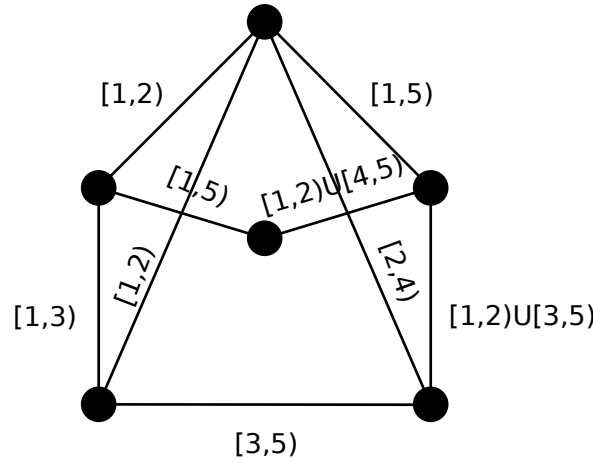


FIGURE 2.6 – Un exemple d’illustration de graphe variant dans le temps où l’on ignore les latences ζ et ϕ . Les intervalles de temps de chaque arête expriment les instants durant lesquels une arête est présente (cette information peut être obtenue par la fonction ρ).

Graphes variant dans le temps. L’étude des réseaux hautement dynamiques (où il est possible que le réseau reste non connexe à tout instant) se basait principalement sur des analyses ou des hypothèses probabilistes. Le peu d’approches déterministes qui existaient supposaient des réseaux mobiles bien structurés dont la dynamique suit fidèlement une période. Pour pallier à ce problème, les auteurs dans [CFMS10] ont introduit le modèle des graphes variant dans le temps où il est question de connexion à travers le temps. Cette nouvelle approche concernant l’étude théorique des réseaux dynamiques s’est très rapidement orientée vers l’extension du modèle initial vers un modèle plus générique. Depuis peu [CFQS12, CFMS15], le modèle de graphes variant dans le temps tente d’unifier les modèles, concepts et notions de plusieurs décennies dans un formalisme commun qui permet de classifier les systèmes dynamiques en fonction de leurs propriétés de connexité temporelle.

L’idée des graphes variant dans le temps est d’exprimer à travers des fonctions sur des graphes, les propriétés temporelles complexes des systèmes dynamiques. Le formalisme des graphes variant dans le temps peut être exprimé par le sextuplet $(V, E, \mathcal{T}, \rho, \zeta, \phi)$ où V exprime un ensemble de sommets muni chacun de latence qui varie dans le temps décrit par la fonction ϕ et qui interagissent entre eux durant un intervalle de temps $\mathcal{T}$, à travers des arêtes E dont les disponibilités sont représentées par une fonction ρ et dont les latences sont exprimées par la fonction ζ (Voir le chapitre 3 pour plus de détails concernant le formalisme des graphes variant dans le temps). La figure 2.6 représente une manière simplifiée d’illustrer un graphe variant dans le temps.

2.4 Problèmes distribués et systèmes dynamiques

Dans cette section nous allons aborder brièvement quelques problèmes et quelques résultats obtenus dans le cadre de l'étude de systèmes distribués dynamiques. La plupart des exemples, illustrations et discussions sur les redéfinitions de problèmes présentées dans cette section sont issus de [CF13].

La diffusion

Le problème de **diffusion** consiste en la propagation d'un message depuis un sommet initiateur (désigné par l'émetteur ou la source) vers tous les autres sommets du réseau. La figure 2.7 illustre ce problème. Parmi les techniques utilisées pour résoudre ce problème dans les graphes statiques connexes est d'inonder le réseau, c'est-à-dire que chaque sommet en possession de l'information continue de l'envoyer à ses voisins même si ces derniers sont préalablement informés. L'approche pour une

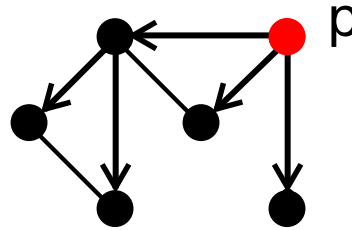


FIGURE 2.7 – Une illustration du problème de la diffusion dans un graphe statique. Le sommet p représente l'initiateur de la diffusion et il propage son message à travers des chemins dans le graphe.

diffusion dans les systèmes dynamiques est différente de l'approche traditionnelle car elle est impactée par les variations dans le temps que subissent les liens de communications. Parmi les conséquences de ces types de réseaux est l'émergence de nouveaux critères d'optimisation de la diffusion qui prennent en considération les trajets (des chemins qui nécessitent parfois des délais d'attente dans certains sommets). En effet, l'optimalité d'un trajet peut être définie selon des critères topologiques ou temporels. Dans [BXFJ03], les auteurs ont introduit trois critères pour désigner trois trajets aux propriétés différentes étudiés dans le cadre de réseau dont l'évolution est connue à l'avance par un algorithme centralisé. Les auteurs ont déterminé qu'un trajet peut être court, rapide ou terminant plus tôt dépendant du critère qui est considéré (Voir le chapitre 3 pour les définitions des trajets court, rapide ou terminant plus tôt). La figure 2.8 l'illustre des exemples de ses chemins dans un système dynamique.

Les auteurs dans [CFMS10] ont étudié la version distribuée du problème sur des graphes variant dans le temps dotés de certaines propriétés. Des propriétés telles que la récurrence (bornée ou non) des arêtes dans le temps. Les auteurs montrent par exemple que si le nombre de sommets n et ou la borne de récurrence des arêtes Δ est connu des sommets, il est alors possible de résoudre le problème de la diffusion terminant plus tôt avec une complexité en messages de $O(m)$ (m étant le nombre d'arêtes). Ils montrent aussi que s'il existe une borne sur la récurrences des arêtes et

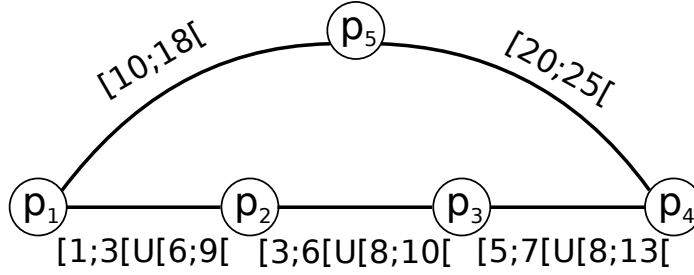


FIGURE 2.8 – Cette figure représente un graphe dynamique de telle sorte que la disponibilité de chaque arête est représentée par l'intervalle de temps qui l'étiquette. Nous supposons qu'une information peut traverser un nombre fini d'arêtes disponibles en tout instant. Dans cet exemple, il s'agit de chercher les meilleurs trajets du sommet p_1 vers le sommet p_4 , en supposant qu'il est possible de traverser jusqu'à trois arêtes par instant.

Si l'objectif est un trajet court en nombre de saut, alors passer par l'arête $\{p_1, p_5\}$ à l'instant 10 puis par $\{p_5, p_4\}$ est optimal.

Si l'idée est de mettre le moins temps possible pendant la traversé (trajet rapide) un idéal serait de passer par les arêtes $\{p_1, p_2\}$ puis $\{p_2, p_3\}$ et finalement $\{p_3, p_4\}$ à l'instant 8.

Si on cherche le trajet terminant plus tôt alors une solution serait de prendre l'arête $\{p_1, p_2\}$ à l'instant 1, puis les arêtes $\{p_2, p_3\}$ et $\{p_3, p_4\}$ à l'instant 5.

que n ou Δ est connu des sommets la complexité en temps est de $O(n\Delta)$.

Nous allons maintenant aborder le problème du routage qui a quelques points en commun avec la diffusion.

Le routage

Le problème du routage est très proche du problème de diffusion. La principale différence entre les deux est que l'information est destinée à un seul sommet initialement prédéfini avant le début de la propagation. Le problème du **routage** consiste à trouver le chemin qui permet de transmettre une information d'un sommet émetteur vers un sommet destinataire. Le routage est utilisé par de nombreux réseaux, tel que internet, les réseaux téléphoniques, les réseaux de transport, etc. Sa performance est importante, l'objectif est donc de trouver le meilleur chemin pour délivrer l'information au destinataire.

Dans le cadre des réseaux dynamiques le routage est aussi impacté par les propriétés temporelles de ces réseaux. De ce fait, les routages dans un cadre dynamique sont aussi formulés en terme de propriétés topologiques ou temporelles, on parle alors de route courte, rapide ou terminant plus tôt par référence aux trajets qu'elles empruntent. Plusieurs techniques furent développées dans les réseaux dynamiques afin de répondre au problème du routage. On peut trouver par exemple dans [JFP04], une approche de routage où il est supposé que la dynamique est connue au préalable. Dans [LDS03], c'est une approche probabiliste qui est utilisée. Tandis que dans [DFGV03, GV03, JLSW07], c'est une stratégie d'apprentissage en fonction de ce qui a été rencontré qui est préférée. Les lecteurs souhaitant explorer plus pro-

fondément le problème du routage sont invités à consulter [Zha06] qui propose une taxonomie des approches de routage.

Nous allons maintenant discuter du problème de l'exploration dans un cadre dynamique.

L'exploration

Le problème de l'exploration doit aussi être adapté au contexte des systèmes dynamiques. Nous présenterons un bref résumé de travaux récents qui ont traité ce problème sous différents types de réseaux dynamiques.

Avant de présenter le problème de l'exploration nous devons d'abord introduire le paradigme des agents mobiles qui sont majoritairement utilisés dans un contexte d'exploration de graphes.

Les agents mobiles. Dans un graphe, les **agents mobiles** sont des unités de calcul qui sont dotés de capacité de migrer (de se déplacer) et changer leur support d'exécution (le sommet sous-jacent où ils sont positionnés). Ainsi, les agents peuvent exécuter des calculs au niveau d'un sommet, se déplacer vers un autre et effectuer encore de nouveaux calculs, et continuer ainsi.

Plusieurs variantes existent selon les capacités dont sont dotées les agents (par exemple, transporter des données, modifier le sommet sous-jacent, communiquer avec d'autres agents), ou en fonction des conditions d'initialisation (en particulier si les agents sont d'abord dispersés sur plusieurs sommets ou regroupés sur un même sommet). Bien que sur plusieurs de ces variantes les modèles des agents mobiles sont équivalents aux modèles à passage de messages, comme on peut le retrouver dans [BFFS03, CGMO06, DFSY07], il représente néanmoins un paradigme de programmation qui est radicalement différent dans le fond et qui a pour conséquence de faciliter l'expression naturelle de certaines problématiques dans les systèmes distribués à l'image du problème de l'exploration.

L'exploration. Dans un graphe statique l'**exploration** est le problème qui consiste à ce qu'un ou plusieurs agents visitent l'ensemble des arêtes (ou l'ensemble des sommets) d'un réseau.

Le problème de construction d'une **cartographie** est une exploration dont l'objectif est de construire un réseau (graphe) sous-jacent par exploration du réseau (graphe). Il existe des variantes qui supposent que certains sommets peuvent être malveillants en essayant d'empêcher les agents de réussir l'exploration. Par exemple, l'entité malveillante la plus répandue dans ce contexte est connu sous le nom de *trou noir*. Un **trou noir** est un sommet détruisant tous les agents y pénètrent.

Les objectifs d'une exploration peuvent varier, par exemple l'exploration peut être perpétuelle, ou elle peut être avec arrêt (après avoir identifié que le problème est résolu). Les performances d'une exploration sont souvent associées au nombre d'arêtes traversées ou au nombre de déplacements effectués. Les lecteurs souhaitant

approfondir le sujet peuvent se référer à [Kel12].

Exploration dans un modèle aléatoire. Dans [CF04], les auteurs étudient l'exploration d'un graphe dynamique aléatoire croissant dans le cadre de l'évolution de la toile d'internet. Dans ce type de graphe, les auteurs montrent que l'exploration manquera de visiter une proportion de sommets qui tend vers une constante si à chaque étape le graphe évolue en raccordant un nouveau sommet au graphe. Dans les systèmes aléatoires où les sommets sont fixes et les arêtes dynamiques, les auteurs de [AKL08] s'intéressent à ce qu'ils désignent par *temps de couverture* qui représente le temps nécessaire pour résoudre le problème de l'exploration. Sous l'hypothèse que la dynamique des arêtes est gérée par un adversaire inconscient -oblivious adversary-, les auteurs prouvent que le temps de couverture par une marche aléatoire *simple* (où le prochain sommet voisin à visiter est choisi de manière totalement arbitraire) peut être exponentiel sous certaines stratégies de l'adversaire alors qu'elle devient polynomiale quel que soit la stratégie de l'adversaire avec une marche aléatoire *paresseuse* (c'est-à-dire que le prochain sommet voisin à visiter est choisi sous une probabilité $1/d_{max}$, où d est le degré maximum du graphe et où il y a une probabilité que l'agent reste dans le sommet actuel).

Exploration dans un modèle de bus. Dans les systèmes distribués dynamiques l'exploration fut étudiée dans un premier temps par [FMS09, FKMS10, FKMS12a, FKMS12b, FKMS12c] sous l'hypothèse que tous les événements topologiques se reproduisent après une certaine période, cette famille de graphes dynamiques est souvent désignée par **graphes variants périodiques**. Le type de graphe variant périodique étudié par les auteurs dans le cadre de l'exploration est défini par un système de transport en commun où les véhicules/transporteurs se déplacent entre des stations/sites de manière synchrone tout en suivant infiniment souvent leur route. Par conséquent, il existe trois types d'entités dans cette représentation : les véhicules, les stations, et les agents mobiles.

Dans [FMS09] les auteurs étudient la faisabilité et la complexité du problème de l'exploration dans les graphes variants périodiques. Pour explorer ce type de système les auteurs supposent que chaque agent mobile peut migrer d'un véhicule à un autre si et seulement si les deux véhicules sont situés à la même station, en d'autres termes les agents restent toujours sur des véhicules. Les agents mobiles situés dans la même station peuvent communiquer entre eux. Les auteurs montrent les résultats suivants :

- dans un système où les sommets n'ont pas d'identifiants, le problème de l'exploration admet une solution si et seulement si les agents mobiles connaissent la taille (ou une borne supérieure sur la taille) de la plus longue route ;
- dans un système où les sommets ont des identifiants uniques, le problème de l'exploration admet une solution si et seulement si les agents mobiles connaissent n le nombre des sommets, ou une borne supérieure sur la périodicité du système.

Ensuite est considéré la complexité de l'exploration dans les graphes variants

périodiques où fut exhibée une borne inférieure en nombre de migrations. En termes de complexité, les auteurs montrent que dans le cas général $\Omega(kp)$ déplacements sont nécessaires dans un système homogène où toutes les routes ont la même période, là où $\Omega(kp^2)$ déplacements sont nécessaires dans un système hétérogène lorsque les routes n'ont pas les mêmes périodes (avec p la longueur de la plus grande route et k le nombre de véhicule).

Dans [FMS09] les auteurs étudient aussi l'impact de la structure des routes sur la complexité du problème de l'exploration. Par exemple, lorsque les routes étudiées sont simples (c'est-à-dire, qu'elles ne contiennent ni boucle ni arêtes multiples), la complexité en nombre de migrations des agents mobiles passe à $\Omega(kp^2)$ pour un système homogène et à $\Omega(kp^4)$ dans un système hétérogène.

Exploration du métro. Des travaux récents [FKMS10, FKMS12a, FKMS12b, FKMS12c, Kel12] ont étendu les résultats de l'exploration obtenue au préalable sur les graphes variants périodiques en rajoutant l'hypothèse que les agents peuvent descendre d'un véhicule et attendre dans une station.

Le modèle de métro considère k agents qui débutent à des dates imprévisibles. Dans ce modèle un agent mobile peut passer d'un véhicule à une station (descendre du véhicule) ou d'une station vers un véhicule (monter à bord du véhicule) et ne peut plus migrer d'un véhicule à un autre directement. Durant son voyage à bord d'un véhicule un agent peut compter le nombre de stations (d'arrêts) que le véhicule traverse et décider de descendre du véhicule en conséquence. Ce modèle de métro est décrit en détail dans [Kel12].

Dans ce modèle on suppose que deux agents mobiles peuvent communiquer entre eux s'ils se trouvent dans la même station [FKMS10, FKMS12c] ou s'ils se trouvent dans le même véhicule [FKMS12b]. Tous les agents mobiles exécutent de manière asynchrone le même algorithme (à savoir que chaque agent prend un temps fini et non borné pour effectuer ses calculs). Plusieurs hypothèses sont considérées initialement :

- soit les agents mobiles débutent à une date variable en étant co-localisés/regroupés à la même station,
- soit les agents mobiles débutent à une date variable tout en étant éventuellement dispersés sur plusieurs stations.

Recherche du trou noir dans le cas général Avant [FMS09], la plupart des travaux de la littérature étudient le problème de la recherche d'un trou noir par des agents mobiles explorant le réseau sous l'hypothèse que des agents mobiles co-localisés/regroupés travaillent en équipe afin de trouver l'unique trou noir du réseau comme dans [DFPS02, DFPS06]. Les auteurs de [DFPS02, DFPS06], démontrent une borne inférieure de $\Omega(n^2)$ en nombre de déplacements lorsque les agents mobiles ignorent complètement le réseau sous-jacent. Ces résultats furent étendus de différentes manières. Dans [FMS09], les auteurs proposent d'une part une exploration utilisant des agents qui débutent en étant dispersés sur le réseau, d'autre part, ils autorisent à ce qu'il existe éventuellement plusieurs trous noirs et plusieurs arêtes

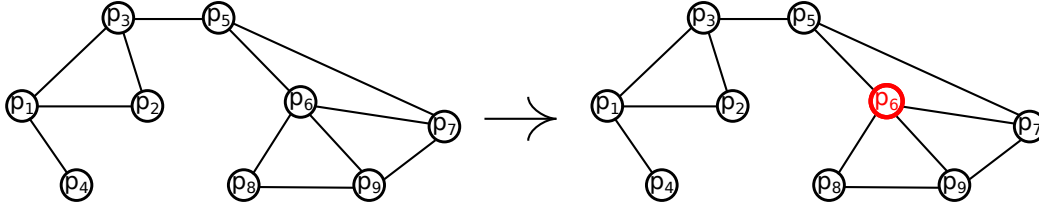


FIGURE 2.9 – Illustration d'une élection de leader dans un graphe statique. Le sommet en rouge représente un leader possible dans ce graphe

noir (c'est-à-dire, des arêtes qui détruisent tout agent qui essaye de les traverser), avec certaines limitations. La solution propose de résoudre le problème en $\Omega(mn)$ déplacements (où n est le nombre de sommets et m est le nombre d'arêtes).

Recherche de trou noir dans le modèle du métro La recherche de trous noirs fut aussi étudiée dans le modèle du métro sous l'hypothèse que les agents débutent en étant regroupés à la même station et communiquent entre eux dans les stations à travers des ardoises. Dans [FKMS10, FKMS12c] les auteurs prouvent certaines limitations et proposent une résolution du problème en $O(k \cdot n_C^2 \cdot l_R + n_C \cdot l_R^2)$ déplacement de véhicules.

Nous allons maintenant aborder le problème de l'élection de leader dans les systèmes dynamiques.

L'élection de leader

Dans un réseau statique l'**élection de leader** est le problème qui consiste à distinguer un sommet, nommé **leader**, parmi tous les autres, comme illustré par la figure 2.9. La désignation d'un leader octroie la possibilité de résoudre certains problèmes compliqués dans les systèmes distribués en affiliant au leader un rôle spécifique dont il a le monopole par rapport aux autres sommets. En effet, l'objectif d'une élection de leader est parfois de générer une certaine centralisation dans un système décentralisé, de manière à simplifier une coordination ultérieure des tâches. Certains problèmes distribués deviennent plus simple voir triviaux lorsqu'un unique leader est déterminé comme c'est le cas pour le *comptage* ou la *synchronisation* dans un réseau statique. Nous allons voir dans le chapitre 4 qu'il existe plusieurs approches pour redéfinir les problèmes dans un contexte dynamique. Cela s'applique aussi pour le problème de l'élection de leader.

Pour rappel dans un réseau dynamique il est possible que le réseau soit non connexe à tout instant. Par conséquent, il est possible qu'en tout instant le réseau soit composé de plusieurs composantes connexes qui peuvent varier dans le temps. Si le réseau est *connexe à travers le temps* (voir la définition 3.9 dans le chapitre 3), l'une des adaptations possibles du problème de l'élection de leader dans un contexte dynamique consiste à désigner un seul leader parmi tous les sommets du réseau comme l'illustre la figure 2.10(a). Un tel sommet peut être désigné par *leader temporel* puisque son rôle se déroule à travers le temps et est justifié par le fait que le réseau soit connexe

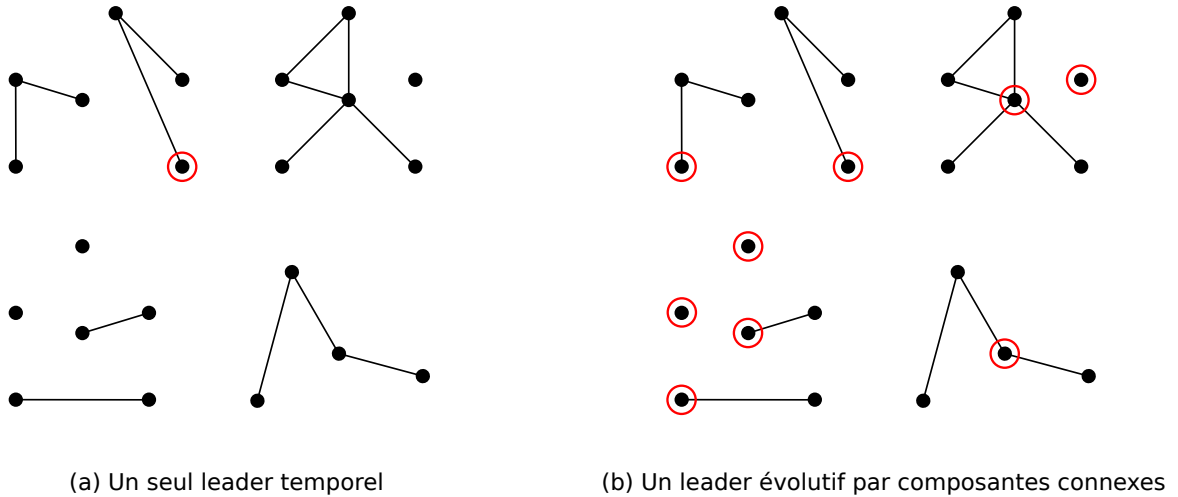


FIGURE 2.10 – Deux définitions pour l’adaptation de l’élection de leader dans un contexte dynamique

à travers le temps. Cependant, cette définition de leader temporel devient inadéquate si les composantes connexes durent un temps considérable de telle sorte que le temps d’interconnexion à travers le temps entre le leader et l’ensemble des sommets est long. En effet, la coordination des actions entre l’ensemble des sommets du réseau par l’unique leader prendrait alors un temps considérable qui dépendrait principalement des temps d’attente nécessaires permettant au leader de contacter tous les sommets du réseau. De ce fait, dans ce cas particulier il serait sans doute plus judicieux d’aborder le problème par rapport à l’état instantané du système à un instant donnée.

Une autre approche, consiste à définir l’élection de leader de telle sorte qu’il y ait un leader par composante connexe comme illustré dans la figure 2.9(b). Cette approche qu’on pourrait désigner par *élection de leader évolutif* est similaire à considérer que chaque composante connexe est un réseau à part sur lequel une élection de leader doit être réalisée. Cependant, puisque les sommets des composantes connexes sont assujettis à des dispersions et à fusionner avec d’autres sommets, le problème de l’élection de leader évolutif devient un problème de *maintenance*, à savoir que l’objectif serait de maintenir un seul leader par composante connexe en produisant un nouveau leader quand un groupe de sommets connexes se séparent en deux et en enlevant un leader lorsque deux composantes connexes se regroupent. Néanmoins, cette définition du problème peut aussi devenir inadaptée en fonction des hypothèses retenues dans les réseaux dynamiques. En effet, si par exemple la dynamique du graphe est très forte (changement radicaux) et que la fréquence de changement est très rapide, il est possible que la maintenance d’une solution pour l’élection de leader soit impossible, du fait des changements radicaux qui interviennent très vite en comparaison au temps nécessaire pour recalculer les leaders de chaque composante connexe.

L’arbre couvrant

Dans un graphe statique un **arbre couvrant** est un sous graphe couvrant connexe minimal. En d’autres termes, dans un graphe $G = (V, E)$ le couple $A = (V, E')$

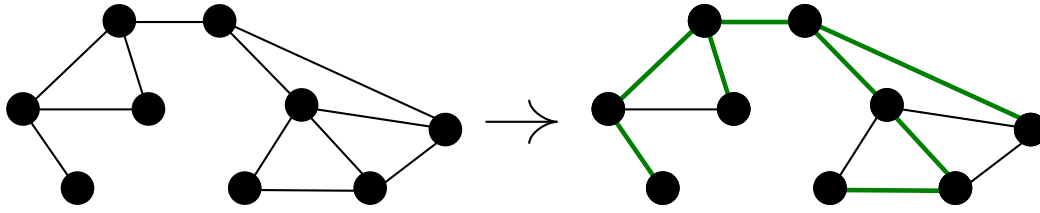
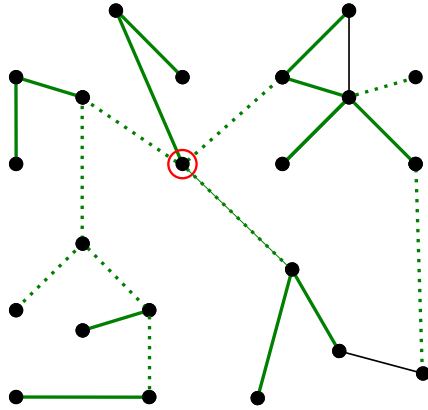
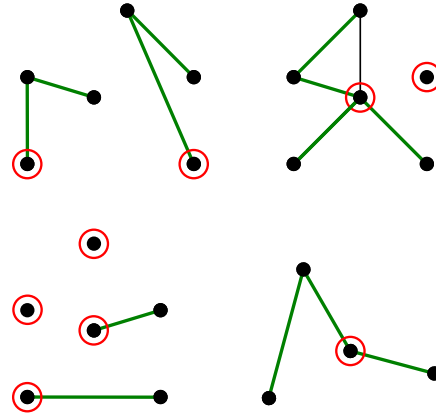


FIGURE 2.11 – Exemple d'un arbre couvrant dans les graphes statiques



(a) Un seul arbre couvrant temporel



(b) Maintenance d'une forêt d'arbre couvrant

FIGURE 2.12 – Une représentation de l'arbre couvrant temporel et l'arbre couvrant évolutif dans un réseau dynamique

est un arbre couvrant s'il n'existe pas de cycle dans A et si $E' \subseteq E$, $\forall p \in V, \exists e \in E'$ tel que p est une extrémité de e et l'ensemble des arêtes d'un arbre couvrant est un sous ensemble connexe des arêtes du graphe de telle sorte que chaque sommet de V est une extrémité d'une arête de E , comme illustré par la figure 2.11. Notez que dans un arbre couvrant il existe exactement un seul chemin entre toute paire de sommets.

Parfois on utilise l'élection de leader pour désigner la racine et ainsi construire un arbre couvrant progressivement à partir de cette racine. Le problème de l'arbre couvrant peut aussi être redéfini de différentes manières lorsqu'il s'agit de l'adapter à un système dynamique. Le problème de l'arbre couvrant peut être redéfini selon un critère temporel ou une propriété d'instantanéité.

L'approche temporelle consiste à définir le problème de l'arbre couvrant de telle sorte que la propriété de couverture se manifeste à travers le temps, une telle approche est basée sur la construction d'un seul arbre dont les arêtes logiques n'évoluent pas dans le temps. La figure 2.12(a) illustre un arbre couvrant temporel dont les arêtes en pointillés sont des liens logiques réapparaissent dans le temps.

L'approche d'instantanéité de l'arbre couvrant repose sur la construction d'un arbre couvrant par groupe de sommets connexes comme illustré par la figure 2.12(b).

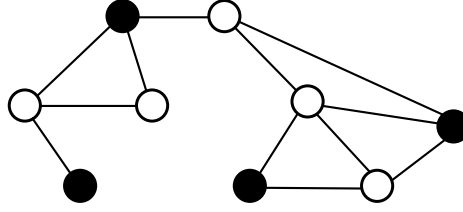


FIGURE 2.13 – Un exemple d'ensemble dominant dans un graphe statique. Comme on peut le constater tout sommet blanc a au moins un voisin noir.

L'ensemble dominant

Dans un graphe statique, un **ensemble dominant** désigne un sous ensemble de sommets du graphe (dit sommets dominants) de telle sorte que tout sommet non dominant du graphe ait un voisin parmi les sommets de l'ensemble dominant. La figure 2.13 illustre dans un graphe, un exemple d'un ensemble dominant représenté par les sommets noirs

Dans un contexte dynamique trois approches de redéfinition du problème existent et peuvent être retrouvées dans [CMM11]. Nous verrons en détails ces approches dans le chapitre 4.

Comptage. Le **comptage** désigne le problème qui consiste à calculer le nombre de sommets d'un graphe.

Dans [KLO10], les auteurs montrent, dans un système discret où l'évolution du graphe peut s'effectuer toute les x tours, que le problème de comptage ainsi qu'un problème particulier de dissémination de k -jetons (où k -jetons sont répartis sur le réseau et où tous les sommets doivent récupérer tous les k -jetons de telle sorte que k est égal au nombre de sommets du graphe) peuvent être résolus sous $\Theta(n^2)$ si le graphe peut changer à chaque étape et sous $\Theta(n + n^2 \log(n)/x)$ si le graphe est maintenu toutes les x étapes avant tout changement potentiel.

Chapitre 3

Modèle

Sommaire

3.1	Graphe variant dans le temps	34
3.2	Familles de graphes variant dans le temps	40
3.2.1	Familles de GVT dotés de propriétés ponctuelles	43
3.2.2	Familles de GVT dotés de propriétés récurrentes	47
3.3	Modèle de communication et de calcul	54

Ce chapitre détaillera de nombreuses définitions et notations que nous utiliserons tout au long de cette thèse.

De nombreux travaux traitent directement ou indirectement des systèmes distribués dynamiques. En effet, un nombre important de travaux prennent en compte les changements topologiques, pour ne citer que quelques uns, [AKM⁺12, AE84, CCF09, Fal03, Fer04, FGM07, SW09]. Suivant l’approche, le modèle utilisé peut être différent. Cependant, elles partagent toutes quelques caractéristiques et notions communes comme le changement topologique qui peut être induit par la disparition/la panne et l’apparition/rétablissement des liens de communications.

Des modèles de systèmes distribués dynamiques ont été développés lors de travaux récents (voir le chapitre 2, page 19 pour plus de détails). Il existe principalement deux modèles génériques qui peuvent représenter la dynamique d’un système sous toutes ses facettes. Dans [Fer04] on modélise la dynamique d’un système par une succession de graphes statiques appelés “graphe évolutif” (Evolving Graph). Quand le “graphe variant dans le temps” (Time-Varying Graph), présenté par les auteurs dans [CFMS10], est un modèle qui utilise un graphe unique associé à plusieurs fonctions qui décrivent les différents événements que les éléments du système subissent au cours du temps (comme l’apparition et la disparition de liens de communications ou de processus).

Des travaux récents [CCF09, CFQS12, CF13] proposent d’unifier la plupart des concepts, notions, formalismes et résultats de la littérature dans un seul formalisme des graphes variant dans le temps pour modéliser les différents systèmes distribués dynamiques.

Dans cette thèse, nous nous intéressons à l’étude de la faisabilité de certaines

questions dans le cadre de systèmes distribués hautement dynamiques (c'est-à-dire des systèmes dont les composantes subissent des changements fréquents et qui peuvent être non connexes à tout moment). Notre choix de modèle s'est porté sur les graphes variant dans le temps [CFQS12, CCF09], un modèle utilisé dans de nombreux travaux récents comme [CFMS10, SQF⁺11, CFMS15, DRm14].

3.1 Graphe variant dans le temps

Un graphe permet de décrire formellement les capacités de communications d'un système distribué. Le graphe variant dans le temps est un formalisme basé sur les graphes qui permet de formuler, à travers des fonctions, les propriétés temporelles complexes des systèmes distribués dynamiques tout en offrant une notation simple, intuitive et concise.

La motivation derrière le choix de modéliser les systèmes distribués par le formalisme du graphe variant dans le temps est due principalement à la richesse et la malléabilité de ce modèle qui permet de représenter un large éventail de systèmes dynamiques. Ce formalisme définit des graphes capables de modéliser les réseaux dynamiques sous tous leurs aspects. En effet, l'un des points forts des graphes variant dans le temps est leur capacité à remplacer les modèles dynamiques les plus utilisés dans la littérature, comme le montrent les auteurs dans [CFQS12].

Dans cette section nous allons présenter le formalisme des graphes variant dans le temps ainsi qu'une panoplie de notions très répandues dans la littérature autour des graphes dynamiques en général.

Le formalisme des graphes variant dans le temps [CFMS10] s'inspire des systèmes de ré-étiquetage de graphe [LMS99] (où l'étiquetage des arêtes peut changer au cours du temps à travers des règles bien définies) et du modèle de réseaux temporel [KKK00] (où il s'agit de modéliser la dynamique à travers un graphe statique dont les arêtes sont étiquetées par des dates ponctuelles). Ce formalisme est enrichi par de nouvelles fonctions dans [CFQS12] afin de modéliser toutes les particularités des réseaux dynamiques observés dans la littérature.

Un graphe variant dans le temps $\mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta, \phi)$ est un graphe avec un ensemble de sommets noté $V = (p_1, p_2, \dots, p_n)$ qui interagissent entre eux durant un intervalle de temps, fini ou non, $\mathcal{T} \subset \mathbb{T}$. L'interaction entre les sommets définit l'ensemble des arêtes ou relations bidirectionnelles $E \subseteq V \times V$, dont les disponibilités peuvent être irrégulières.

La fonction de présence des arêtes ρ décrit les apparitions et les disparitions des liens de communications dans un système distribué dynamique. En d'autres termes, la fonction ρ définit les disponibilités des arêtes au cours du temps.

Dans un système distribué, les communications se déroulent entre les différentes unités de calcul. Cependant, la communication entre deux unités de calcul peut être assujettie à des latences. De ce fait, la fonction de latences des arêtes ζ permet de représenter dans un graphe les variations de la latence de chaque arête au cours du temps. Notez que dans bon nombre de travaux cette fonction est soit ignorée,

soit constante et connue (par tous les processus).

Dans un réseau, le temps de traitement d'une action par des unités de calcul peut être aussi assujéti à des variations dans le temps. De ce fait, en plus de représenter le temps de traitement d'une action par chaque sommet, la fonction ϕ permet de modéliser aussi la variation de ce délai dans le temps.

Définition 3.1 (Graphe variant dans le temps ($\mathcal{GVT}$)) *Un **graphe variant dans le temps** [CFQS12] $\mathcal{G}$ est un tuple $(V, E, \mathcal{T}, \rho, \zeta, \phi)$ tel que :*

- V est un ensemble statique de sommets ;
- E un ensemble statique d'arêtes entre les sommets $E \subseteq V \times V$;
- $\mathcal{T}$, le **temps de vie** du graphe est inclus dans un domaine temporel $\mathbb{T}$. À titre d'exemple, $\mathbb{T}$ peut être un domaine temporel discret comme $\mathbb{N}$ ou un domaine temporel continu comme $\mathbb{R}^+$;
- $\rho : E \times \mathcal{T} \rightarrow \{0, 1\}$, appelée la **fonction de présence des arêtes**, indique pour une arête donnée si elle est présente ou non à un moment donné ;
- $\zeta : E \times \mathcal{T} \rightarrow \mathbb{T}$, appelée la **fonction de latence des arêtes**, indique le délai de communication à un instant donnée entre les deux extrémités d'une arête ;
- $\phi : V \times \mathcal{T} \rightarrow \mathbb{T}$, appelée **fonction de latence des sommets**, donne le temps nécessaire à une action interne d'un processus à un instant donnée.

Remarque 3.1 *Dans notre modèle nous considérons que l'ensemble des arêtes E ne comprend que les relations qui permettent à ses deux extrémités de communiquer **au moins une fois** durant le temps de vie du $\mathcal{GVT}$ (c'est-à-dire, le temps de présence de l'arête est suffisamment long par rapport à sa latence lors de son apparition). Formellement, nous supposons que dans un $\mathcal{GVT}$ $\mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta, \phi)$:*

$$\left\{ \begin{array}{l} \forall e \in E, \exists t \in \mathcal{T}, \rho(e, t) = 1 \\ \forall e \in E, \forall t \in \mathcal{T}, \exists \epsilon \in \mathcal{T}, (\rho(e, t) = 1 \wedge \forall t' \in]t - \epsilon, t[, \rho(e, t') = 0) \Rightarrow \\ \quad \forall t' \in [t, t + \zeta(e, t)], \rho(e, t') = 1 \end{array} \right.$$

On parle alors de **présence** d'une arête à un instant si et seulement si l'intervalle de temps entre l'apparition effective et la disparition effective de l'arête est supérieure au temps de latence de l'arête à cet instant. De la même manière, on ne parle d'**apparition** d'une arête e à un instant donnée que si l'arête e est présente tout de suite après cet instant et absente avant. On parle de **disparition** d'une arête e à un instant donné que si e était présente avant cet instant.

Nous considérons que l'apparition et la disparition d'une arête comme un **événement topologique**.

Étant donné un graphe variant dans le temps $\mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta, \phi)$, une suite de **dates caractéristiques d'une arête** $e \in E$, notée T_e , est la suite ordonnée des différentes dates d'apparitions et de disparitions de l'arête e durant le temps de vie de $\mathcal{G}$. La suite ordonnée de l'union de toutes les dates caractéristiques des arêtes de E est appelée la suite de **dates caractéristiques** de $\mathcal{G}$, que nous notons $T_{\mathcal{G}}$. La suite de

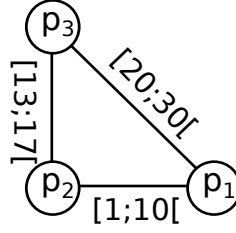


FIGURE 3.1 – Cette figure illustre un exemple de $\mathcal{GVT}$ dont l’empreinte globale est connexe alors que le $\mathcal{GVT}$ n’est jamais connexe durant son temps de vie, c’est-à-dire l’intervalle $[0,30[$.

dates caractéristiques d’un $\mathcal{GVT}$ $\mathcal{G}$ correspond aux différentes dates durant lesquelles des événements topologiques se produisent dans $\mathcal{G}$.

Les dates de ces événements topologiques peuvent-être perçues comme des dates de transition de $\mathcal{G}$. C’est-à-dire qu’à chaque date t_i de $T_{\mathcal{G}}$ le graphe change.

De ce fait, en utilisant ses dates caractéristiques, un graphe variant dans le temps $\mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta, \phi)$ peut être exprimé par la **suite d’évolutions** de $\mathcal{G}$ définie par une suite de graphes statiques $\mathcal{S}_{\mathcal{G}} = g_1, g_2, \dots$ où chaque $g_i = (V, E_i)$ correspond à l’état du graphe $\mathcal{G}$ à la date $t_i \in T_{\mathcal{G}}$. Notez que durant l’intervalle de temps $t \in [t_i, t_{i+1}[$ aucun événement topologique ne se produit (ni apparition ni disparition d’arête). De ce fait, un graphe variant dans le temps peut être exprimé aussi par $(V, E, S_{\mathcal{G}}, T_{\mathcal{G}}, \zeta, \phi)$.

Le **graphe sous-jacent** d’un graphe variant dans le temps, représente le graphe statique qui comprend tout sommet du $\mathcal{GVT}$ et toute arête qui est disponible au moins une fois durant le temps de vie du $\mathcal{GVT}$.

Définition 3.2 (Graphe sous-jacent) *Le **graphe sous-jacent** (ou **empreinte globale**) d’un graphe variant dans le temps $\mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta, \phi)$ noté $U_{\mathcal{G}}$, est le graphe (V, E) .*

Remarque 3.2 *Notez que même si l’empreinte globale $U_{\mathcal{G}}$ d’un graphe dynamique $\mathcal{G}$ est connexe, il est possible que $\mathcal{G}$ soit à tout instant non connexe. Comme illustré par la figure 3.1.*

Arête infiniment souvent présente. Une arête infiniment souvent présente dans un graphe variant dans le temps $\mathcal{G}$ est toute arête dont la présence est soit permanente soit récurrente infiniment souvent.

Définition 3.3 (Arête infiniment souvent présente) *Une arête **infiniment souvent présente** dans un graphe variant dans le temps $\mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta, \phi)$, est toute arête $e \in E$, pour tout $t \in \mathcal{T}$, il existe $t' \geq t$ tel que $\rho(e, t') = 1$.*

L’ensemble des arêtes infiniment souvent présentes de $\mathcal{G}$ est noté par $E^{\omega} = \{e \in E \mid \forall t \in \mathcal{T}, \exists t' > t, \rho(e, t') = 1\}$.

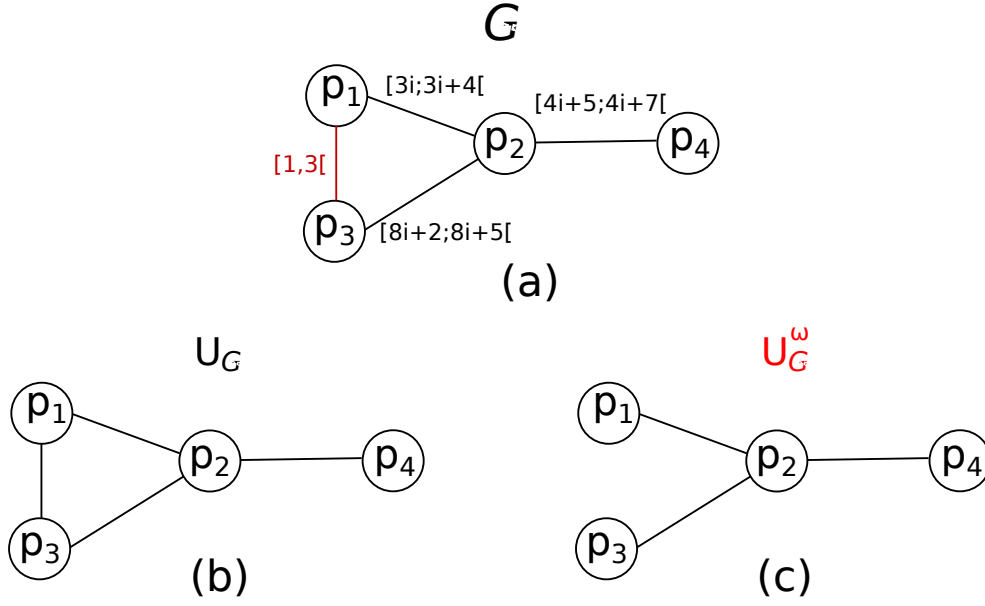


FIGURE 3.2 – La figure 3.2(a) illustre un graphe variant dans le temps $\mathcal{G}$ où l'arête $\{p_1, p_3\}$ est vouée à disparaître puisqu'elle est présente uniquement pendant l'intervalle de temps $[1, 3[$ tandis que les autres arêtes sont infiniment souvent présentes. La figure 3.2(b) représente l'empreinte globale $U_{\mathcal{G}}$ de $\mathcal{G}$. La figure 3.2(c) correspond à l'empreinte ultime $U_{\mathcal{G}}^{\omega}$ de $\mathcal{G}$. On constate notamment que la différence entre l'empreinte globale et l'empreinte ultime de $\mathcal{G}$ est la présence dans $U_{\mathcal{G}}$ et respectivement l'absence dans $U_{\mathcal{G}}^{\omega}$ de l'arête $\{p_1, p_3\}$.

Arête vouée à disparaître. Comme son nom l'indique, une arête vouée à disparaître dans un graphe variant dans le temps $\mathcal{G}$ est toute arête qui au bout d'un certain nombre d'apparitions et d'un certain temps cesse d'être présente dans $\mathcal{G}$.

Définition 3.4 (Arête vouée à disparaître) Une arête vouée à disparaître dans un graphe variant dans le temps $\mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta, \phi)$ est toute arête $e \in E \setminus E^{\omega}$. En d'autres termes, toute arête e est vouée à disparaître dans $\mathcal{G}$ si il existe $t \in \mathcal{T}$ tel que $\forall t' \geq t, \rho(e, t') = 0$.

Empreinte ultime.

Définition 3.5 Un **graphe sous-jacent ultime** (ou **empreinte ultime**) d'un graphe variant dans le temps $\mathcal{G}$ noté $U_{\mathcal{G}}^{\omega}$, est le graphe (V, E^{ω}) composé du même ensemble de sommets que $\mathcal{G}$ et où l'ensemble des arêtes E^{ω} correspond aux arêtes infiniment souvent présentes dans $\mathcal{G}$.

La figure 3.2 illustre la différence entre l'empreinte globale et l'empreinte ultime d'un graphe variant dans le temps qui admet une arête vouée à disparaître.

Comme le montre les auteurs dans [CFQS12] la notion classique de chemin dans les graphes statiques est parfois dénuée de sens dans les $\mathcal{GVT}$. Par conséquent,

un des concepts centraux dans les graphes dynamiques est la notion de **chemin temporel**, appelé aussi **trajet**. Un trajet entre un processus p et un processus q est une suite d'arêtes successives reliant le processus p au processus q telle que la disponibilité et la latence des arêtes permettent la transmission d'une information du processus p vers le processus q . Le concept d'un tel chemin a vu le jour dans la littérature sous différents noms. On s'y réfère par *trajet (journey)* [BXFJ03], *chemin temporel (temporal path)* [CHC⁺07, TMML10], *chemin conforme à un calendrier (schedule-conforming path)* [Ber96] et *chemin respectant un temps (time-respecting path)* [Hol05, KKK00].

L'existence d'un tel trajet d'un sommet p vers un sommet q dans un graphe variant dans le temps permet au sommet p de faire parvenir une information au sommet q si elle passe par ce trajet même si, à tout moment, il n'existe pas de route point à point instantanée (chemin traditionnel) entre le sommet p et le sommet q . Notez que ce concept implique que nous supposons qu'il existe un moyen de stocker une information dans un sommet afin qu'elle puisse être transmise ultérieurement. Nous désignerons l'hypothèse qui permet à un processus de garder une information et la transmettre plus tard, par la méthode de *stockage puis retransmission*.

Définition 3.6 Soit un graphe variant dans le temps $\mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta, \phi)$, un **trajet** (ou **chemin temporel**) du sommet p_1 de V vers le sommet p_k de V , noté $\mathbf{j}_{(p_1, p_k)}$ est une suite (de couple (arête, date)) $\{(\{p_1, p_2\}, t_1), (\{p_2, p_3\}, t_2), \dots, (\{p_k, p_{k+1}\}, t_k)\}$ où (les arêtes) $\{\{p_1, p_2\}, \{p_2, p_3\}, \dots, \{p_k, p_{k+1}\}\}$ sont un chemin dans le graphe (V, E) tel que $\forall i, 1 \leq i \leq k, \forall t \in [t_i, t_i + \zeta(\{p_i, p_{i+1}\}, t_i), \rho(\{p_i, p_{i+1}\}, t) = 1$ et $t_{i+1} \geq t_i + \zeta(\{p_i, p_{i+1}\}, t_i) + \phi(p_{i+1}, t_i + \zeta(\{p_i, p_{i+1}\}, t_i))$ pour tout $1 \leq i < k$.

La **date de départ** du trajet $\mathbf{j}_{(p_1, p_k)}$ notée $\mathbf{t}_{\text{depart}}(\mathbf{j})$, est t_1 . Elle correspond à la date associée à la première arête du trajet. Quand la **date de fin** du trajet $\mathbf{j}_{(p_1, p_k)}$ notée $\mathbf{t}_{\text{fin}}(\mathbf{j})$, est $t_k + \zeta(\{p_{k-1}, p_k\}, t_k)$ et elle correspond à la date définie par le cumul de la date associée à la dernière arête du trajet et la latence de cette arête.

On désignera l'**ensemble des trajets** possibles entre une paire de sommets p et q par $\mathcal{J}_{(p, q)}^*$. Tandis que par $\mathcal{J}_{(p, q)}^t$ on désignera tous les trajets possibles de p vers q à partir d'un instant supérieure ou égale à t .

Notez que dans les $\mathcal{GVT}$ hautement dynamiques parfois les seuls moyens qui permettent aux différents processus de communiquer entre eux sont des trajets.

La notation $p \rightsquigarrow q$ est parfois utilisée pour désigner un trajet entre le sommet p et le sommet q .

Remarque 3.3 Un trajet est non symétrique, c'est-à-dire que pour une paire de sommets p et q , s'il existe un trajet $q \rightsquigarrow p$, il n'existe pas forcément de trajet $p \rightsquigarrow q$, comme illustré par l'exemple dans la figure 3.3.

Trajet strict et non-strict. Les trajets stricts ont été définis principalement pour des $\mathcal{GVT}$ où le temps de vie du $\mathcal{GVT}$ est discret. Il s'agit d'un trajet qui autorise la traversée d'une seule arête par étape. Tandis qu'un trajet non-strict autorise la traversée d'un nombre quelconque d'arêtes consécutives par étape.

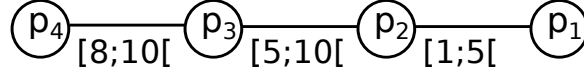


FIGURE 3.3 – Cette figure illustre un exemple de $\mathcal{GVT}$ sous l'hypothèse que les latences sont ignorées. Dans ce $\mathcal{GVT}$, il existe un trajet du sommet p_1 vers le sommet p_4 (par exemple, le trajet $\{(\{p_1, p_2\}, 1), (\{p_2, p_3\}, 5), (\{p_3, p_4\}, 8)\}$) alors que ce trajet est non symétrique : il n'existe aucun trajet du sommet p_4 vers le sommet p_1 dans ce $\mathcal{GVT}$.

Un **trajet strict** d'un sommet p vers un sommet q dans un $\mathcal{GVT}$ $\mathcal{G}$ est un trajet $j_{p,q}$ tel que, pour tout (e_i, t_i) et (e_{i+1}, t_{i+1}) de la suite $j_{p,q}$, le graphe à l'instant t_i est différent du graphe à l'instant t_j . En d'autres termes, au moins un événement topologique se produit entre t_i et t_{i+1} .

Notez que l'événement topologique ne concerne par forcément l'apparition de la prochaine arête du trajet.

Sous graphe temporel. Hormis la possibilité de considérer des sous graphes par restriction de liens et/ou de sommets dans un graphe dynamique comme dans les graphes statiques (voir la définition 2.9 pour le sous graphe), il est possible de restreindre le graphe à des intervalles de temps créant ainsi ce que l'on désigne par un sous graphe temporel.

Définition 3.7 Soit $\mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta, \phi)$ un graphe variant dans le temps, le graphe $\mathcal{G}' = (V, E, \mathcal{T}', \rho', \zeta', \phi')$ est un **sous graphe temporel** de $\mathcal{G}$ s'il satisfait ces différents points :

- $\mathcal{T}' \subset \mathcal{T}$;
- $E' = \{e \in E : \exists t \in \mathcal{T}' : \rho(e, t) = 1 \wedge t + \zeta(e, t) \in \mathcal{T}'\}$;
- $\rho' : E' \times \mathcal{T}' \rightarrow \{0, 1\}$ où $\forall t \in \mathcal{T}'$, $\rho'(e, t) = \rho(e, t)$;
- $\zeta' : E' \times \mathcal{T}' \rightarrow \mathbb{T}$ où $\forall t \in \mathcal{T}'$, $\zeta'(e, t) = \zeta(e, t)$;
- $\phi' : E' \times \mathcal{T}' \rightarrow \mathbb{T}$ où $\forall t \in \mathcal{T}'$, $\phi'(e, t) = \phi(e, t)$.

En pratique la notation $\mathcal{G}' = \mathcal{G}_{[t_i, t_j[}$ est utilisée pour désigner un sous graphe temporel de $\mathcal{G}$, où le temps de vie du graphe $\mathcal{T}'$ est restreint à $\mathcal{T} \cap [t_i, t_j[$. Ceci inclut aussi la notation $\mathcal{G}' = \mathcal{G}_{[t_i, +\infty[}$ qui définit un sous graphe temporel de $\mathcal{G}$ tel que le temps de vie du graphe est restreint de t_i à la fin de vie du graphe, ou la notation $\mathcal{G}' = \mathcal{G}_{[-\infty, t_j[}$ qui représente un sous graphe temporel de $\mathcal{G}$ tel que le temps de vie du graphe est borné du début de la vie du graphe à t_b .

Définition 3.8 Étant donné un graphe variant dans le temps $\mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta, \phi)$ et une date t donnée, un trajet $j_{(p,q)}$ débutant à $t_{\text{depart}} \geq t$ est un **trajet court** depuis l'instant t si et seulement si pour tout $j'_{(p,q)}$ débutant à $t'_{\text{depart}} \geq t$ la longueur en sauts (nombre d'arêtes du trajet) de $j'_{(p,q)}$ est supérieur ou égal à la longueur en sauts de $j_{(p,q)}$.

Définition 3.9 Étant donné un graphe variant dans le temps $\mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta, \phi)$, une **distance topologique** depuis un sommet $p \in V$ vers un sommet $q \in V$ à partir

d'une date t notée $d_{p,q}(t)$ est la longueur (en sauts) du plus court trajet de p vers q dans $\mathcal{G}$ débutant à une date $t_{\text{depart}} \geq t$.

Trajet terminant au plus tôt. Dans un graphe variant dans le temps, un **trajet terminant au plus tôt** entre les sommets p et q par rapport à une date donnée t est un trajet dont la date de fin t_{fin} est minimale parmi tous les trajets de $\mathcal{J}_{(p,q)}^t$.

Définition 3.10 Dans un graphe variant dans le temps $\mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta, \phi)$ et étant donnée une date $t \in \mathcal{T}$, un trajet $j_{p,q}$ entre p et q commençant à une date $t_{\text{depart}} \in \mathcal{T}$ est un **trajet terminant au plus tôt** si et seulement si $t_{\text{depart}} \geq t$ et $\forall j'_{p,q} \in \mathcal{J}_{p,q}^t$, $t'_{\text{fin}} \geq t_{\text{fin}}$.

Définition 3.11 Étant donné un graphe variant dans le temps $\mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta, \phi)$ et une date $t \in \mathcal{T}$, une **distance temporelle** depuis un sommet $p \in V$ vers un sommet $q \in V$ à partir d'une date t notée $\hat{d}_{p,q}(t)$ est la durée en temps entre t_{fin} du trajet terminant le plus tôt à partir de t .

Définition 3.12 Étant donné un graphe variant dans le temps $\mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta, \phi)$ et un instant donné t , un trajet $j_{p,q}$ entre les sommets p et q débutant à une date t_{depart} et terminant à une date t_{fin} est un **trajet rapide** si et seulement si $\forall j'_{p,q} \in \mathcal{J}_{p,q}^t$, $t_{\text{fin}} - t_{\text{depart}} \leq t'_{\text{fin}} - t'_{\text{depart}}$.

La figure 3.4 illustre la différence entre un trajet court, rapide et se terminant le plus tôt.

Trajet direct et indirect Un trajet direct est un trajet où les processus n'utilisent pas la méthode de stockage puis retransmission. Tandis que, dans un trajet indirect les processus l'utilisent. La figure 3.5 illustre la différence entre un trajet direct et un trajet indirect dans un graphe variant dans le temps.

Définition 3.13 Un trajet $j_{(p_1, p_k)} = \{(\{p_1, p_2\}, t_1), (\{p_2, p_3\}, t_2), \dots, (\{p_{k-1}, p_k\}, t_{k-1})\}$ est **direct** si $\forall i \in \mathbb{N}, i \leq k, t_{i+1} = t_i + \zeta(\{p_i, p_{i+1}\}, t_i) + \phi(p_{i+1}, t_i + \zeta(\{p_i, p_{i+1}\}, t_i))$.

La figure 3.6, synthétise les différents concepts vus précédemment.

3.2 Familles de graphes variant dans le temps

L'un des points intéressants introduits par le formalisme des graphes variant dans le temps dans [CFQS12] et dans [CCF09] est la hiérarchie de familles de graphes variant dans le temps.

Chaque famille regroupe des systèmes dynamiques satisfaisant certaines propriétés. Ces propriétés peuvent représenter les conditions nécessaires et parfois suffisantes que doivent satisfaire un système dynamique afin que la résolution de certains

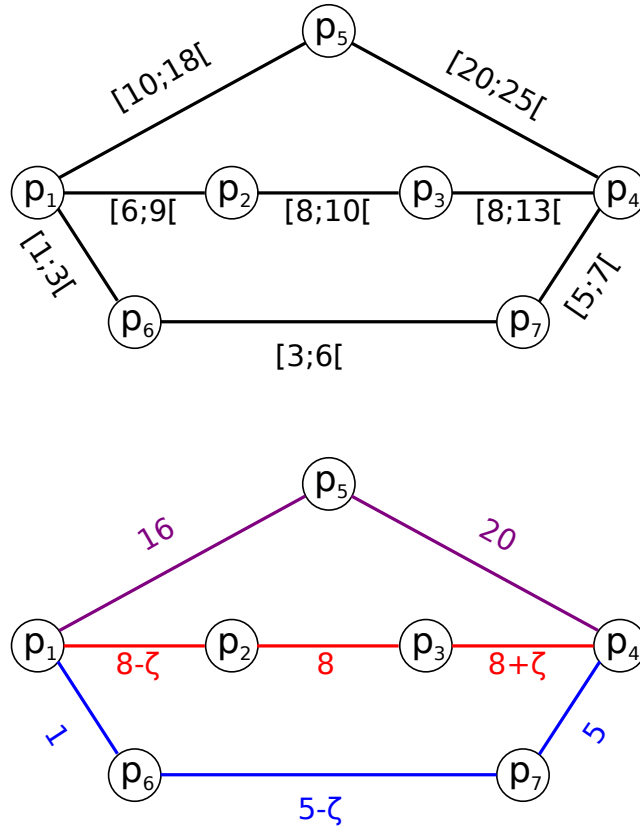


FIGURE 3.4 – Un exemple de trajet court, rapide et se terminant le plus tôt, pour $t = 1$, $0 \leq \zeta \leq 2$ et $\phi = 0$.

Trajet court : $\dot{j}_{(p_1, p_4)} = \{(\{p_1, p_5\}, 18 - \zeta), (\{p_5, p_4\}, 20)\}$

Trajet rapide : $\dot{j}_{(p_1, p_4)} = \{(\{p_1, p_2\}, 8 - \zeta), (\{p_2, p_3\}, 8), (\{p_3, p_4\}, 8 + \zeta)\}$

Trajet terminant au plus tôt : $\dot{j}_{(p_1, p_4)} = \{(\{p_1, p_6\}, 1), (\{p_6, p_7\}, 5 - \zeta), (p_7, p_4, 5)\}$

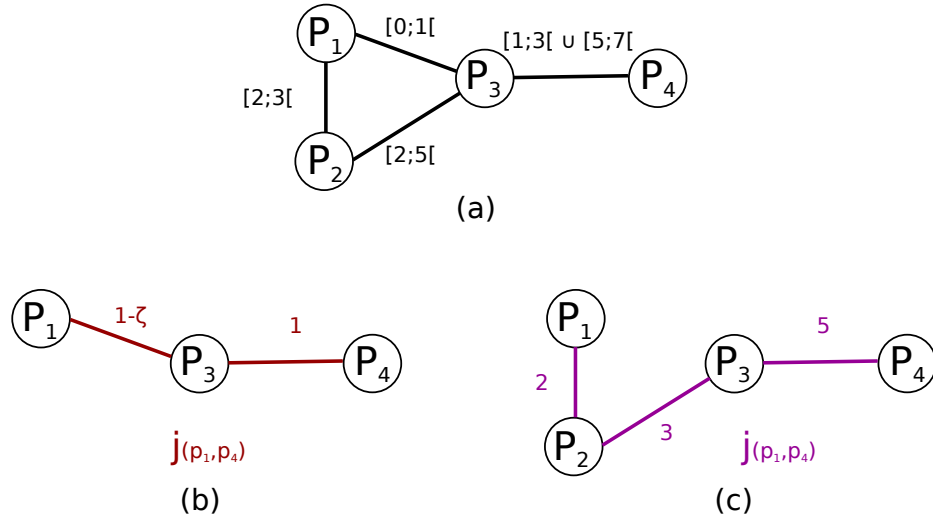


FIGURE 3.5 – La figure 3.5(a) illustre un $\mathcal{GVT}$ dont la latence ζ est constante et inférieure à 1 et la latence ϕ est égale à 0. Dans ce $\mathcal{GVT}$ nous étudions des trajets directs et indirects entre le processus p_1 et p_4 que nous illustrons dans les figures 3.5(b) et (c).

La figure 3.5(b) illustre l'exemple de trajet direct suivant : $\dot{j}_{(p_1,p_4)} = \{(\{p_1, p_3\}, 1 - \zeta), (\{p_3, p_4\}, 1)\}$.

La figure 3.5(c) illustre l'exemple trajet indirect suivant : $\dot{j}_{(p_1,p_4)} = \{(\{p_1, p_2\}, 2), (\{p_2, p_3\}, 3), (\{p_3, p_4\}, 5)\}$

Concept	Explication	Notation
Fonction de présence	d'une arête : Fonction qui indique si une arête $\{p, q\}$ est présente ou non à un instant t	$\rho(\{p, q\}, t)$
Fonction de latence	d'une arête : Fonction qui donne le délai de communication à un instant t entre p et q	$\zeta(\{p, q\}, t)$
Fonction de latence	d'un sommet : Fonction qui indique le temps que prend une action interne par un processus p à un instant t	$\phi(p, t)$
Trajet	Un chemin à travers le temps entre deux processus p et q	$j(p, q), p \rightsquigarrow q$
Trajet strict	Un chemin à travers le temps strict entre deux processus p et q	$j(p, q), p \overset{st}{\rightsquigarrow} q$
Temps de départ	d'un trajet :	$t_{depart}(j)$
Temps de fin	d'un trajet :	$t_{fin}(j)$
Tous les trajets	L'ensemble des trajets de p vers q	$\mathcal{J}_{(p,q)}^*$
	L'ensemble des trajets de p vers q à partir d'un instant t	$\mathcal{J}_{(p,q)}^t$

FIGURE 3.6 – Tableau de synthèse.

problèmes devienne possible, des problèmes tels que la diffusion de messages, la propagation d'informations avec accusé de réception, l'élection de leader, etc. Ces propriétés peuvent être aussi topologiques et/ou temporelles pour correspondre à des systèmes étudiés dans la littérature.

Notez que les familles de graphes variant dans le temps que nous allons voir ne sont pas définitives et peuvent-être étendues à souhait. En effet, dans [CFQS12, CCF09] les auteurs proposent à travers ces familles de modéliser les systèmes dynamiques les plus étudiés dans la littérature. Néanmoins, le spectre des systèmes qui peuvent-être modélisés par des graphes variant dans le temps est encore plus large (allant de l'ajout de nouvelles hypothèses aux familles existantes, à la création de nouvelles familles pour modéliser d'autres/nouveaux systèmes sous d'autres/nouvelles hypothèses.

Les figures 3.15 et 3.25 illustrent les relations d'inclusion entre les différentes familles de graphes dynamiques citées dans ce document.

3.2.1 Familles de GVT dotés de propriétés ponctuelles

Dans un premier temps nous allons présenter l'ensemble des familles auxquelles un système dynamique appartient si et seulement s'il satisfait certaines conditions au moins une fois durant sa vie. En effet, dans ces familles le caractère répétitif de certaines propriétés est exclu. Ce caractère non répétitif peut être utile dès lors que l'on suppose que temps de vie du système est fini ou lorsqu'il s'agit de considérer un nombre fini d'événements topologiques.

F1 : Un diffuseur. Cette famille comprend tout $\mathcal{GVT}$ dont un processus peut atteindre tous les autres à travers un trajet. Si un $\mathcal{GVT}$ n'appartient pas à cette famille alors il n'existe aucune chance qu'un processus puisse informer tous les autres. Par conséquent, cette propriété est nécessaire dans un $\mathcal{GVT}$ pour permettre à un processus de diffuser une information dans le $\mathcal{GVT}$ (la propager à l'ensemble des processus du $\mathcal{GVT}$).

Néanmoins, pour résoudre le problème de la diffusion dans un $\mathcal{GVT}$ cette propriété n'est pas suffisante. En effet, les propriétés définissant cette famille de $\mathcal{GVT}$ ne permettent pas de déterminer lequel de ces processus peut initialiser et satisfaire une diffusion.

Propriété 3.1 (F1) *Un $\mathcal{GVT}$ $\mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta, \phi)$ appartient à la famille F1 si $\exists p \in V : \forall q \in V, \exists j_{p,q}$.*

La figure 3.7 illustre un exemple de graphe variant dans le temps de la famille F1.

F2 : Tous diffuseurs/récepteurs. La particularité de cette famille de $\mathcal{GVT}$ est l'existence d'un trajet entre toute paire de processus du $\mathcal{GVT}$. Le succès d'une diffusion par n'importe quel processus d'un $\mathcal{GVT}$ dépend de l'appartenance ou non à cette

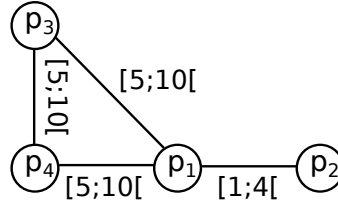


FIGURE 3.7 – Cette figure illustre un $\mathcal{GV}\mathcal{T}$ de la famille F1. Nous supposons que ζ est commune à toutes les arêtes, constante et égale à 1 ($\zeta = 1$) et $\phi = 0$ en tout instant et pour tout sommet. Ce $\mathcal{GV}\mathcal{T}$ satisfait la propriété de la famille F1 puisqu'il existe des trajets du sommet p_2 vers tous les autres.

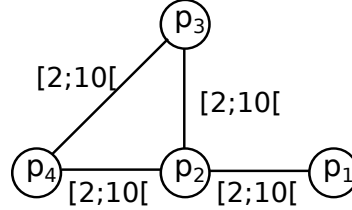


FIGURE 3.8 – Un exemple d'un $\mathcal{GV}\mathcal{T}$ de la famille F2. Dans ce $\mathcal{GV}\mathcal{T}$, nous supposons que $\zeta = 1$ et $\phi = 0$ partout et à tout instant. Sous ces conditions, on constate qu'entre tout couple de sommets il existe un trajet dans ce $\mathcal{GV}\mathcal{T}$.

famille. En effet, si un $\mathcal{GV}\mathcal{T}$ n'appartient pas à cette famille alors il existe au moins un processus qui échouera à propager une information s'il est désigné.

Cette propriété est nécessaire pour une élection de leader dans un $\mathcal{GV}\mathcal{T}$. En effet, un $\mathcal{GV}\mathcal{T}$ de cette famille permet à tout processus de prétendre à l'élection de leader et espérer être élu.

Propriété 3.2 (F2) *Un $\mathcal{GV}\mathcal{T}$ $\mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta, \phi)$ appartient à la famille F2 si $\forall (p, q) \in V^2, \exists j_{p,q}$.*

F3 : Un diffuseur strict. Les $\mathcal{GV}\mathcal{T}$ de cette famille admettent l'existence d'un processus qui partage un trajet strict vers tous les autres processus du $\mathcal{GV}\mathcal{T}$. La figure 3.9 illustre un exemple de cette famille.

Propriété 3.3 (F3) *Un $\mathcal{GV}\mathcal{T}$ $\mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta, \phi)$ appartient à la famille F3 si $\exists p \in V : \forall q \in V, \exists (p \xrightarrow{st} q)$.*

F4 : Tous diffuseurs/récepteurs strict. Cette famille correspond aux $\mathcal{GV}\mathcal{T}$ où il existe un trajet strict entre tous les processus du $\mathcal{GV}\mathcal{T}$. La figure 3.10 illustre un exemple de cette famille.

Propriété 3.4 (F4) *Un $\mathcal{GV}\mathcal{T}$ $\mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta, \phi)$ appartient à la famille F4 si $\forall (p, q) \in V^2, \exists (p \xrightarrow{st} q)$.*

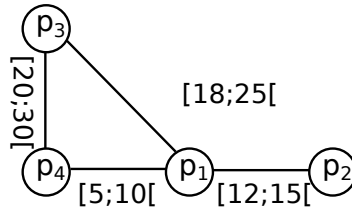


FIGURE 3.9 – Un exemple de graphe variant dans le temps où nous supposons que ζ est commune à toutes les arêtes, constante et égale à 1 et $\phi = 0$ en tout instant dans tout sommet du $\mathcal{GVT}$. Comme il existe des trajets stricts du sommet p_2 vers tous les autres, ce $\mathcal{GVT}$ appartient à la famille F3.

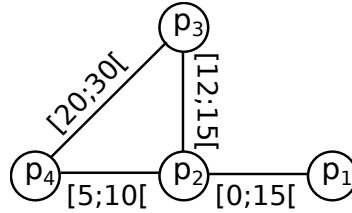


FIGURE 3.10 – Dans ce $\mathcal{GVT}$, nous supposons que $\zeta = 1$ et $\phi = 0$ en tout instant et partout. Notez que dans ce $\mathcal{GVT}$ il existe un trajet strict entre tout couple de sommets. Ce qui implique que ce $\mathcal{GVT}$ appartient à la famille F4.

F5 : Un diffuseur par lien direct Cette famille définit les $\mathcal{GVT}$ dans lesquels au moins un processus (à un certain moment de l'exécution) est lié par une arête (au court du temps) à tout autre processus. La figure 3.11 illustre un graphe de cette famille où le sommet p_2 partage une arête avec tout autre sommet du graphe au moins une fois à partir de l'instant 11.

Propriété 3.5 (F5) *Un $\mathcal{GVT}$ $\mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta, \phi)$ appartient à la famille F5 si $\exists p \in V : \forall q \in V \setminus \{p\}, \{p, q\} \in E$.*

F6 : Une clique temporelle Cette famille compte tout $\mathcal{GVT}$ où il existe au moins une fois une arête entre toute paire de ses processus du $\mathcal{GVT}$. La propriété de cette famille correspond aux conditions suffisantes pour satisfaire le problème du comptage qui consiste à calculer le nombre de processus d'un $\mathcal{GVT}$ [CCF09]. La figure 3.12 illustre un $\mathcal{GVT}$ de cette famille.

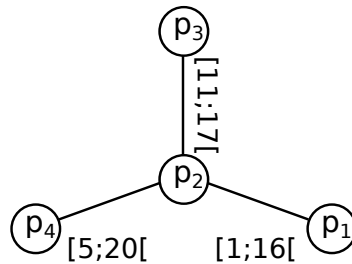


FIGURE 3.11 – Un graphe de la famille F5.

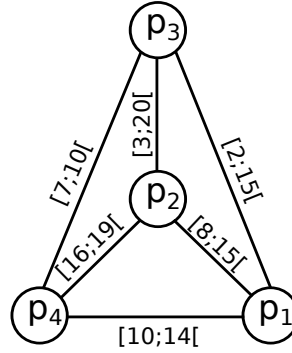


FIGURE 3.12 – Exemple de $\mathcal{GV}\mathcal{T}$ la famille F6. On peut remarquer que toute paire des sommets partageant au moins une fois une arête.

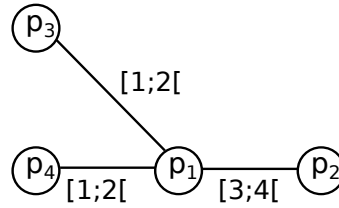


FIGURE 3.13 – Dans cet exemple illustré de $\mathcal{GV}\mathcal{T}$, nous supposons que $\zeta = 1$ et $\phi = 0$ partout et à tout instant. Tous les processus peuvent transmettre une information au processus p_2 à travers des trajets. Par conséquent, ce $\mathcal{GV}\mathcal{T}$ appartient à la famille F7.

Notez que bien que ce $\mathcal{GV}\mathcal{T}$ appartienne à la famille F7, il n'appartient pas à la famille F1 puisqu'il n'existe aucun processus qui puisse atteindre tous les autres à travers des trajets.

Propriété 3.6 (F6) *Un $\mathcal{GV}\mathcal{T}$ $\mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta, \phi)$ appartient à la famille F6 si $\forall (p, q) \in V^2, p \neq q \implies \{p, q\} \in E$.*

F7 : Un destinataire Cette famille comprend l'ensemble des $\mathcal{GV}\mathcal{T}$ qui admettent l'existence d'au moins un processus qui puisse être atteint, à travers des trajets, par tous les autres. On peut retrouver cette famille dans [CCF09] dans le cadre de l'étude d'algorithme de comptage distribué par les processus.

Les hypothèses sur cette famille correspondent aux conditions nécessaires pour résoudre le problème de comptage dans un $\mathcal{GV}\mathcal{T}$. La figure 3.13 illustre un exemple de cette famille.

Propriété 3.7 (F7) *Un $\mathcal{GV}\mathcal{T}$ $\mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta, \phi)$ appartient à la famille F7 si $\exists p \in V : \forall q \in V, \exists j_{(q,p)}$.*

F8 : Aller/retour Cette famille correspond aux $\mathcal{GV}\mathcal{T}$ caractérisés par l'hypothèse qu'entre toute paire de processus p et q , il existe un trajet de p vers q puis de q vers p . La présence d'une telle propriété dans un $\mathcal{GV}\mathcal{T}$ permet à tout processus de confirmer la bonne réception de ses communications.

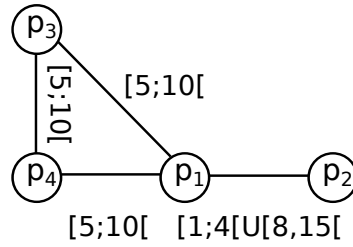


FIGURE 3.14 – Cette figure illustre un $\mathcal{GVT}$ de la famille F8, où nous supposons que $\zeta = 1$ et $\phi = 0$ partout et tout le temps. Entre tout couple de sommets de ce $\mathcal{GVT}$, il existe un trajet aller puis un trajet retour. Par exemple, entre le sommet p_2 et p_3 nous avons $j_{(p_2,p_3)} = \{(\{p_2, p_1\}, 1), (\{p_1, p_3\}, 5)\}$ pour l'aller et $j_{(p_3,p_2)} = \{(\{p_3, p_1\}, 8), (\{p_1, p_2\}, 9)\}$

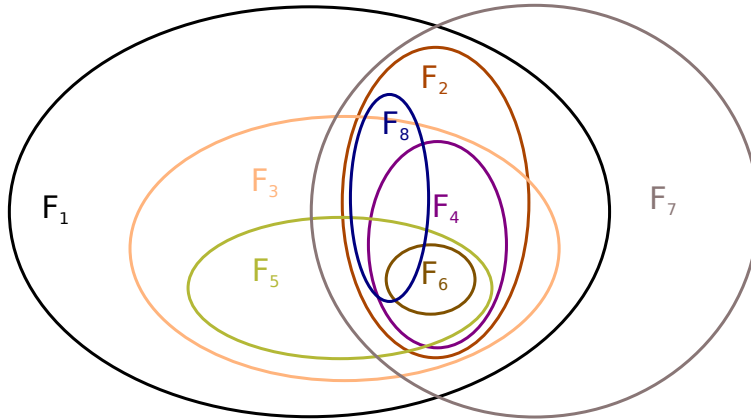


FIGURE 3.15 – Relation d'inclusion entre les différentes familles de $\mathcal{GVT}$ à propriétés non récurrentes.

La figure 3.14 illustre un exemple $\mathcal{GVT}$ de la famille F8.

Propriété 3.8 (F8) *Un $\mathcal{GVT}$ $\mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta, \phi)$ appartient à la famille F8 si $\forall (p, q) \in V^2, \exists j_{(p,q)}, \exists j'_{(q,p)} : t_{fin}(j_{(p,q)}) \leq t_{depart}(j'_{(q,p)})$.*

La figure 3.15 illustre la hiérarchie des familles de $\mathcal{GVT}$ à propriétés non récurrentes. On y constate que les familles F1 et F7 sont sous des hypothèses assez faibles et qu'elles comprennent une bonne partie des autres familles.

3.2.2 Familles de GVT dotés de propriétés récurrentes

Contrairement aux familles précédentes où certaines conditions devaient être satisfaites au moins une fois, nous allons voir ici des familles de $\mathcal{GVT}$ dont les propriétés les définissant sont récurrentes une infinité de fois. Dans ce cas précis, on suppose généralement que le temps de vie du graphe est non borné.

F9 : Connexe à travers le temps. Cette famille inclut l'ensemble des graphes dynamiques tel qu'il existe infiniment souvent dans le temps de vie du graphe un trajet entre toute paire de processus.

Comme on peut le constater la famille F9 est une variante de la famille F3. L'hypothèse de la présence d'un trajet entre toute paire de processus dans F3 devient récurrent dans F9. De ce fait, les $\mathcal{GVT}$ appartenant à cette famille supposent que la disponibilité et la latence des arêtes et des processus admet toujours l'existence d'un itinéraire qui permet à tout processus de communiquer avec un autre à travers des trajets. Notez que dans ce type de $\mathcal{GVT}$ il est possible qu'une arête apparaisse un nombre fini de fois durant la vie du graphe avant de disparaître à jamais.

Des travaux récents considèrent des variantes de cette famille, par exemple [GAS11] considère que la propriété de la famille F9 est vérifiée par au moins un ensemble de sommets stables (des processus qui ne quittent plus le système à partir d'une certaine date) et un ensemble de processus connus (des sommets identifiés par des processus stables) afin d'implémenter un détecteur de fautes dans un système dynamique.

La figure 3.16 illustre un exemple de graphe appartenant à cette famille.

Propriété 3.9 (Connexe à travers le temps $\mathcal{CTT}$) *Un $\mathcal{GVT} \mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta, \phi)$ appartient à la famille de $\mathcal{GVT}$ connexe à travers le temps si $\forall (p, q) \in V^2, \forall t \in \mathcal{T}, \exists j_{(p,q)} : t_{depart}(j_{(p,q)}) \geq t$.*

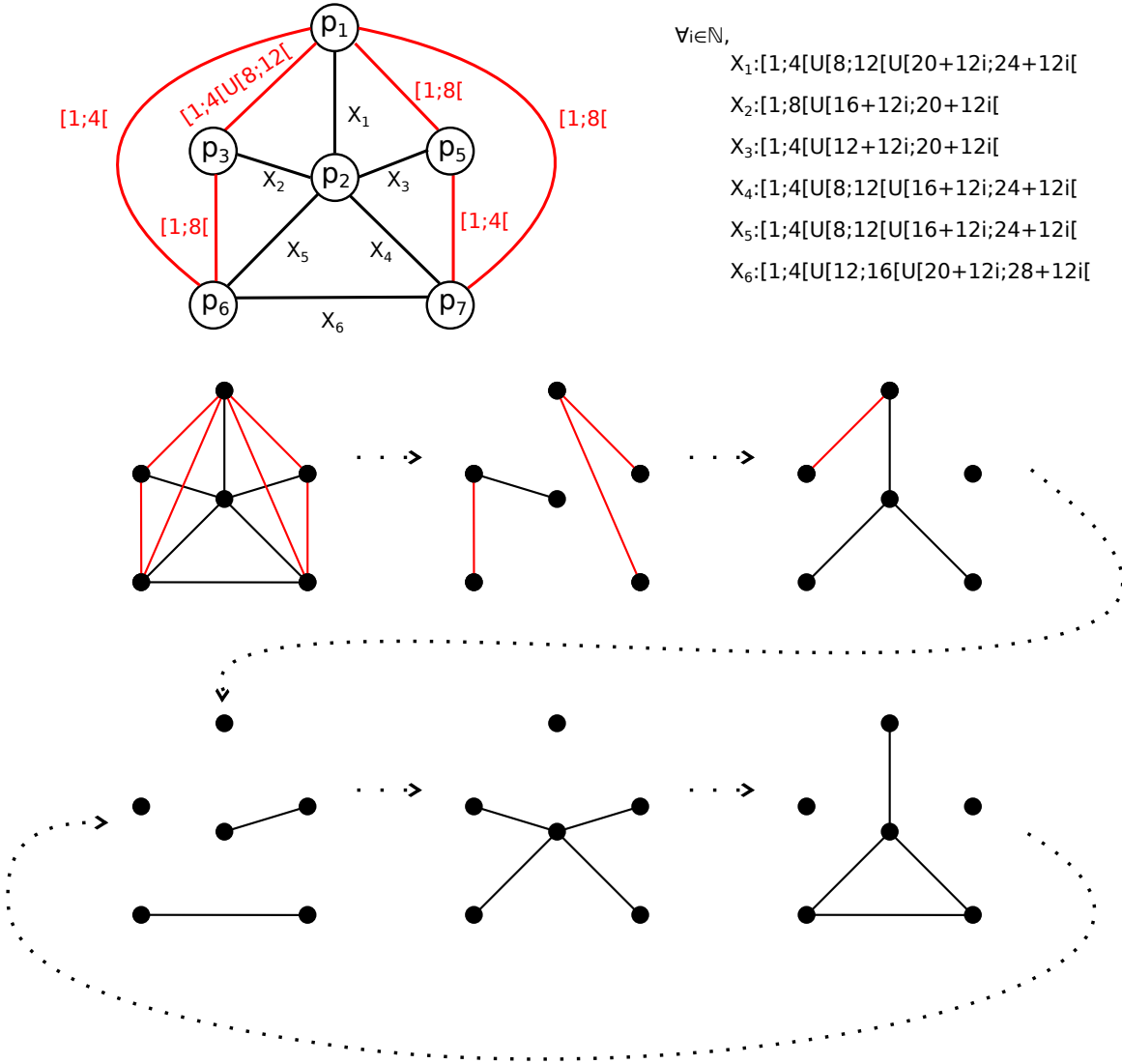
F10 : Arêtes récurrentes Cette famille rassemble l'ensemble des graphes dynamiques qui satisfont la propriété que l'empreinte globale est connexe et que toute arête $e \in E$ apparaît une infinité de fois pendant la vie du graphe. De par cette propriété, il existe un trajet entre toute paire de processus du $\mathcal{GVT}$. L'empreinte globale étant connexe, nous avons donc que $F10 \subseteq F9$. Cette famille ainsi que d'autres furent utilisées dans le cadre de l'étude de la faisabilité de diffusion courte, rapide et la diffusion se terminant le plus tôt. Dans [CFMS15], les auteurs prouvent la faisabilité d'une diffusion empruntant un trajet se terminant le plus tôt dans cette famille à condition que le nombre de processus $|V|$ soit connu par les sommets.

Une des particularités des graphes variant dans le temps appartenant à cette famille est que l'empreinte ultime est identique à l'empreinte globale.

La figure 3.17 illustre un graphe de cette famille.

Propriété 3.10 (Arêtes récurrentes) *Un $\mathcal{GVT} \mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta, \phi)$ appartient à la famille de $\mathcal{GVT}$ à arêtes récurrentes si $\forall e \in E, \forall t \in \mathcal{T}, \exists t' > t$ tel que $\rho(e, t') = 1$.*

F11 : Δ -Récurrent. Cette famille regroupe l'ensemble des graphes dynamiques satisfaisant la condition que l'empreinte globale est connexe et que toute arête $e \in E$ apparaît au moins une fois toutes les Δ unités de temps. Cette famille est donc une variante de la famille F10 puisqu'il existe un intervalle de temps (dite aussi borne) pendant laquelle toutes les arêtes du graphe apparaissent au moins une fois. De ce fait $F11 \subseteq F10$. Les réseaux familiaux où chaque membre interagit au moins une fois avec un membre de sa famille par jours/mois/années, peut correspondre à cette famille de $\mathcal{GVT}$. L'une des propriétés intéressantes que procure cette famille est la possibilité de

FIGURE 3.16 – Un exemple d'un $\mathcal{GVT}$ connecté à travers le temps.

Même si le $\mathcal{GVT}$ fini par devenir périodique, on peut constater que certaines arêtes n'apparaissent plus au bout d'un certain temps fini et non borné. Le graphe reste connexe à travers le temps y compris après la disparition de ces arêtes.

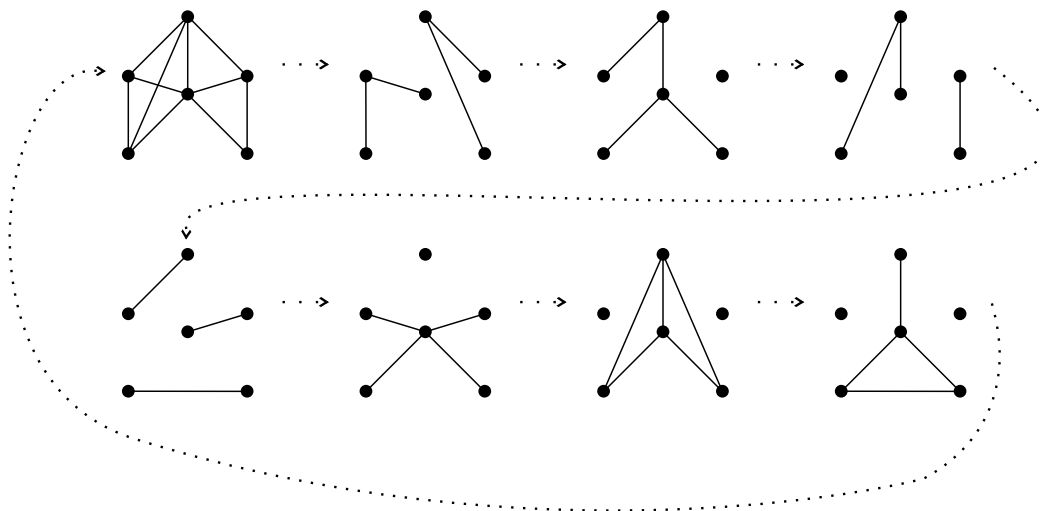
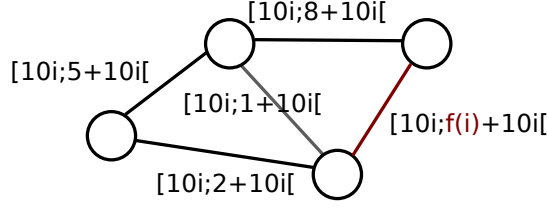


FIGURE 3.17 – Exemple d'un graphe de la famille F10. Les pointillés représentent une durée aléatoire finie mais non bornée où tout peut se produire (par exemple il se peut que tous les liens de communications disparaissent durant les phases représentées par des pointillés). Pour mieux représenter cette famille on peut considérer les graphes aléatoire en supposant un graphe initial (= à l'empreinte globale) au début et que tous les k instants les arêtes du graphe initial ont une probabilité a de disparaître tandis que les arêtes qui ont disparu ont une probabilité $1 - a$ de réapparaître. Le résultat nous assurera que toutes les arêtes de l'empreinte globale seront récurrentes satisfaisant par la même occasion la propriété de la famille à récurrence des arêtes. De plus, le caractère probabiliste des disponibilités des arêtes permet d'exclure définitivement cet exemple des classes bornées et périodiques.



$f(i)$: Retourne un nombre entre 1 et 5 en fonction de la valeur de i

FIGURE 3.18 – Exemple d'un $\mathcal{GVT}$ Δ -récurrent (F11). La fonction $f(i)$ désigne dans cet exemple une fonction qui retourne un nombre aléatoire qui correspond au temps de présence de l'arête tous les Δ instants. Par exemple pour $\Delta = 10$ on peut mettre $f(i) = \text{random}(i)\%5 + 1$ où $\%$ correspond au reste de la division.

calculer, avec détection de terminaison, l'ensemble des voisins lorsque la borne Δ est connue. La faisabilité d'une diffusion courte est possible dans les $\mathcal{GVT}$ de cette famille à condition que Δ soit connu par les processus [CFMS15].

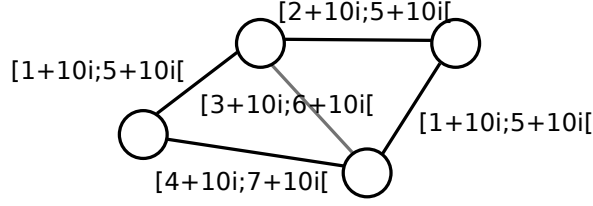
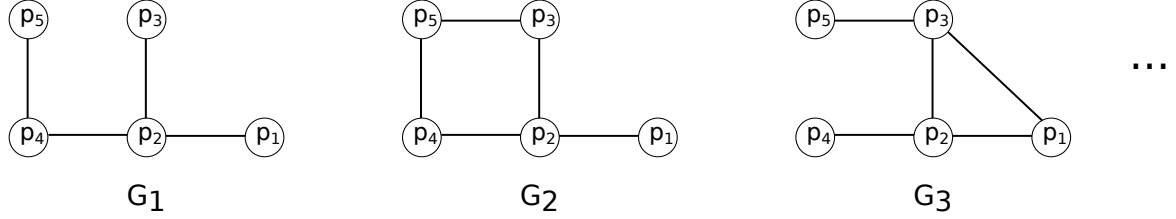
La figure 3.18 illustre un exemple d'un graphe de la famille Δ -Récurrent.

Propriété 3.11 (Δ -Récurrence) *Un $\mathcal{GVT}$ $\mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta, \phi)$ appartient à la famille de $\mathcal{GVT}$ Δ -Récurent si l'empreinte globale $U_{\mathcal{G}}$ est connexe et $\forall e \in E, \forall t \in \mathcal{T}, \exists \Delta \in \mathbb{T}, \exists t' \in [t, t + \Delta[: \rho(e, t') = 1$.*

F12 : Périodique Cette famille regroupe l'ensemble des graphes dynamiques périodiques dont l'empreinte globale est connexe. Par graphe dynamique périodique nous voulons dire qu'il y a une redondance dans la dynamique du graphe et que cette redondance est périodique. La caractéristique de cette famille correspond à bon nombre de propriétés réelles des réseaux, particulièrement toutes composantes mobiles dont la mobilité est périodique. Par exemple, cette famille convient parfaitement aux réseaux de transport (bus, métro, vol) et aux réseaux satellites. La diffusion rapide devient possible pour les graphes de cette famille à condition que la période p soit connu par les noeuds. Bon nombre de travaux supposent l'hypothèse de périodicité dans les réseaux dynamiques. Pour n'en citer que quelques uns : les travaux de [FMS09, IW11, SQF⁺11] dans le contexte d'exploration de réseaux et [KO09] en routage. Il est à noter que $F12 \subseteq F11$. La figure 3.19 représente un graphe appartenant à cette famille.

Propriété 3.12 (Périodique) *Un $\mathcal{GVT}$ $\mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta, \phi)$ appartient à la famille de $\mathcal{GVT}$ périodique si l'empreinte globale $U_{\mathcal{G}}$ est connexe et $\forall e \in E, \forall t \in \mathcal{T}, \exists p \in \mathbb{T}$ tel que $\forall k \in \mathbb{N}, \rho(e, t + kp) = \rho(e, t)$.*

Pour plus de lisibilité et de simplicité de lecture les familles qui suivront sont définies des $\mathcal{GVT}$ exprimés par leur suite d'évolutions 3.1 $(V, E, S_{\mathcal{G}}, T_{\mathcal{G}}, \zeta, \phi)$. De plus, pour ces familles on suppose que le temps de vie d'un $\mathcal{GVT}$ est discret.

FIGURE 3.19 – Exemple d'un $\mathcal{GVT}$ périodique (F12). Dans cet exemple la période est 10.FIGURE 3.20 – Exemple d'un $\mathcal{GVT}$ constamment connexe (F13).

F13 : Constamment connexe. Cette famille comprend tout graphe dynamique ayant la propriété de rester à tout instant connexe. La dynamique du réseau dans cette famille est libre de toute contrainte à condition que le graphe reste connexe. En d'autres termes, il est possible qu'une arête apparaisse un nombre fini de fois durant la vie du graphe avant de disparaître à jamais. Cette famille a été reprise par [KMO11a] dans le cadre de l'étude du problème du consensus. La figure 3.20 illustre un graphe variant dans le temps appartenant à cette famille, où le graphe reste à tout instant connexe. Des variantes de cette famille existent dans la littérature, à l'image de [DRm14] qui associe la propriété d'une connexion permanente à l'hypothèse que l'empreinte du graphe est un cactus dans le cadre de l'exploration du graphe par un agent.

Propriété 3.13 *Un $\mathcal{GVT}$ $\mathcal{G} = (V, E, S_{\mathcal{G}}, T_{\mathcal{G}}, \zeta, \phi)$ appartient à la famille de $\mathcal{GVT}$ constamment connexe si $\forall g_i \in S_{\mathcal{G}}, g_i$ est connexe.*

F14 : T-intervalle connexe. Cette famille constitue un cas particulier de la famille constamment connexe (F13). En effet, il existe un graphe partiel connexe g' pour chaque graphe g_i de $S_{\mathcal{G}}$ qui peut se maintenir durant une période T . Notez qu'un 1-intervalle connexe graphe est équivalent à un graphe de la famille F13. Cette famille fut introduite par [KLO10] dans le cadre de l'étude de problèmes tels que le comptage, la circulation de jeton et la résolution de fonction dont l'entrée est éparpillée sur l'ensemble des sommets. D'autres dérivés de cette famille ont vu le jour dans [IW13] dans le cadre de l'exploration de réseaux par des agents. Les auteurs ont regroupé la particularité d'un T-intervalle graphe avec la propriété de récurrence des arêtes d'un graphe sous l'hypothèse que l'empreinte globale est un anneau dans le cadre de l'exploration de réseaux par des agents.

La figure 3.21 illustre un graphe dynamique T-intervalle connexe.

Propriété 3.14 *Un $\mathcal{GVT}$ $\mathcal{G} = (V, E, S_{\mathcal{G}}, T_{\mathcal{G}}, \zeta, \phi)$ appartient à la famille de $\mathcal{GVT}$ T-intervalle connexe où $T \in \mathbb{N}$ si pour tout $i \in \mathbb{N}$, il existe un graphe partiel connexe g'*

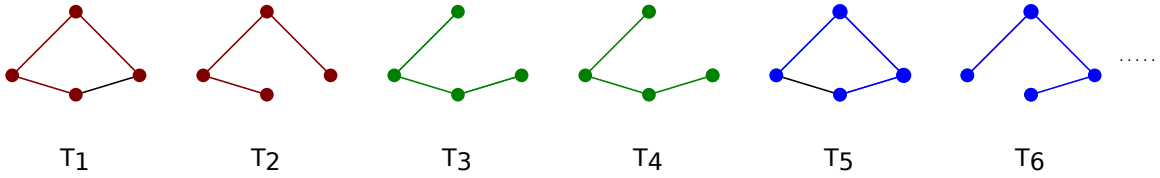


FIGURE 3.21 – Cette figure illustre un exemple de $\mathcal{GVT}$ T-intervalle connexe (F14) où $T=2$. On peut constater notamment que le même graphe partiel connexe est maintenant durant 2 unités de temps ce qui fait de ce graphe un graphe 2-intervalle connexe.

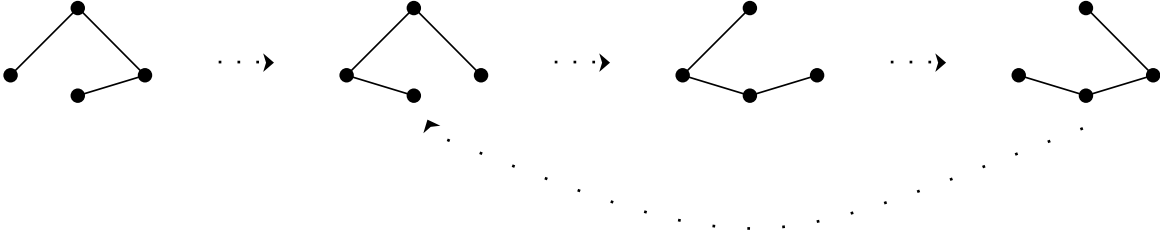


FIGURE 3.22 – Cette figure illustre un $\mathcal{GVT}$ connexe par moment. Dans cet exemple, les pointillés représentent une durée aléatoire finie mais non bornée où des événements topologiques peuvent se produire (par exemple il se peut que tous les liens de communications disparaissent). On peut notamment constater dans cette figure que pour tout instant il existe toujours une date future où le graphe est connexe.

de U_G tel que $\forall j \in [i, i + T - 1), g'$ est un graphe partiel de g_j .

F15 : connexe par moment Cette famille comprend l'ensemble des graphes dynamiques ayant la propriété qu'il existera toujours une date où l'empreinte globale G sera connexe. Cette famille peut représenter par exemple des réseaux dynamiques qui peuvent subir des pertes de connexions occasionnelles et passagères. On peut retrouver cette famille dans les travaux [RBK07] où il est désigné par *réseau dynamique finalement connexe*. La figure 3.22 illustre un $\mathcal{GVT}$ de cette famille.

Propriété 3.15 Un $\mathcal{GVT}$ $\mathcal{G} = (V, E, S_G, T_G, \zeta, \phi)$ appartient à la famille de $\mathcal{GVT}$ connexe par moment si $\forall i \in T_G, \exists j \in T_G : j \geq i, g_j$ est connexe.

F16 : Chemin par moment Cette famille est l'ensemble des graphes dynamiques, tel que partant d'un instant quelconque i , pour toute paire de sommets $(p, q) \in V^2$ il existe toujours une date dans la vie du graphe $j \geq i$ où un chemin existe entre p et q . La particularité de la famille F16 par rapport à la famille F15 est que ces chemins peuvent apparaître à différentes dates pour chaque paire de sommets différents.

Dans [RBK07], trois types de réseaux mobiles ad-hoc connexes sont considérés afin d'étudier la faisabilité de protocole de routage, ces réseaux correspondent aux familles F8, F15 et F16. Les auteurs démontrent que dans les familles F15 et F16, les

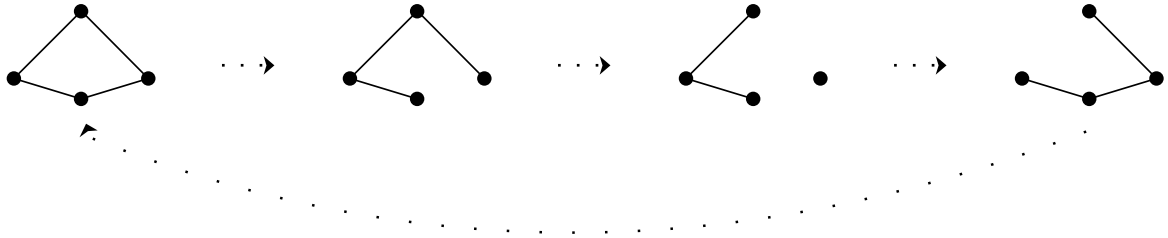


FIGURE 3.23 – Cette figure illustre un $\mathcal{GVT}$ doté de chemin par moment. Dans cet exemple, les pointillés représentent une durée aléatoire finie mais non bornée où des événements topologiques peuvent se produire (par exemple il se peut que tous les liens de communications disparaissent). On peut notamment constater que pour tout instant il existe toujours une date future où il existe un chemin entre tout couple de sommets.

protocoles de routage tolérant aux fautes transitoires dues aux changements topologiques finissent par fonctionner.

La figure 3.23 illustre un graphe de cette famille.

Propriété 3.16 *Un $\mathcal{GVT}$ $\mathcal{G} = (V, E, S_{\mathcal{G}}, T_{\mathcal{G}}, \zeta, \phi)$ appartient à la famille de $\mathcal{GVT}$ avec des chemins par moment si $\forall (p, q) \in V^2, \forall i \in T_{\mathcal{G}}, \exists j \in T_{\mathcal{G}}, j \geq i$, tel qu'il existe un chemin de p vers q dans g_j .*

F17 : Une clique temporelle récurrente Cette famille regroupe les graphes dynamiques de la famille F10 dont l'empreinte globale est une clique. Cette famille est une restriction de la famille des graphes avec récurrence des arêtes F9, où il est supposé que l'empreinte globale est un graphe complet.

La figure 3.24 illustre l'empreinte d'un graphe de la famille F17.

Propriété 3.17 *Un $\mathcal{GVT}$ $\mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta, \phi)$ appartient à la famille de $\mathcal{GVT}$ clique à récurrence des arêtes si $\forall e \in E, \forall t \in \mathcal{T}, \exists t' > t : \rho(e, t') = 1$.*

3.3 Modèle de communication et de calcul

Un système distribué peut être décrit sous différentes hypothèses de communication et de calcul. Ces modèles se distinguent par leur niveau d'atomicité. L'atomicité dépend de l'action qui est permise au cours d'une seule étape indivisible. Ci dessous quelques exemples classiques de modèles de communications (par ordre croissant d'atomicité) :

1. **Modèle à passage de messages [Tel00]** : Ce modèle est un modèle classique et très répandu de communication pour les systèmes distribués. Dans ce modèle, les sommets communiquent par le biais d'échange de messages à travers des canaux de communications. De telle sorte que durant une

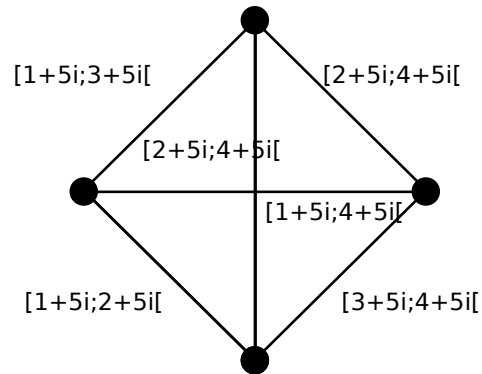


FIGURE 3.24 – Cette figure illustre un exemple de $\mathcal{GVT}$ appartenant à la famille de clique temporelle récurrente. Nous pouvons constater dans cet exemple que toutes les arêtes sont récurrentes infiniment souvent dans le temps en plus d'être périodiques. Ce qui fait de ce $\mathcal{GVT}$ un cas particulier de $\mathcal{GVT}$ de la famille de clique temporelle récurrente

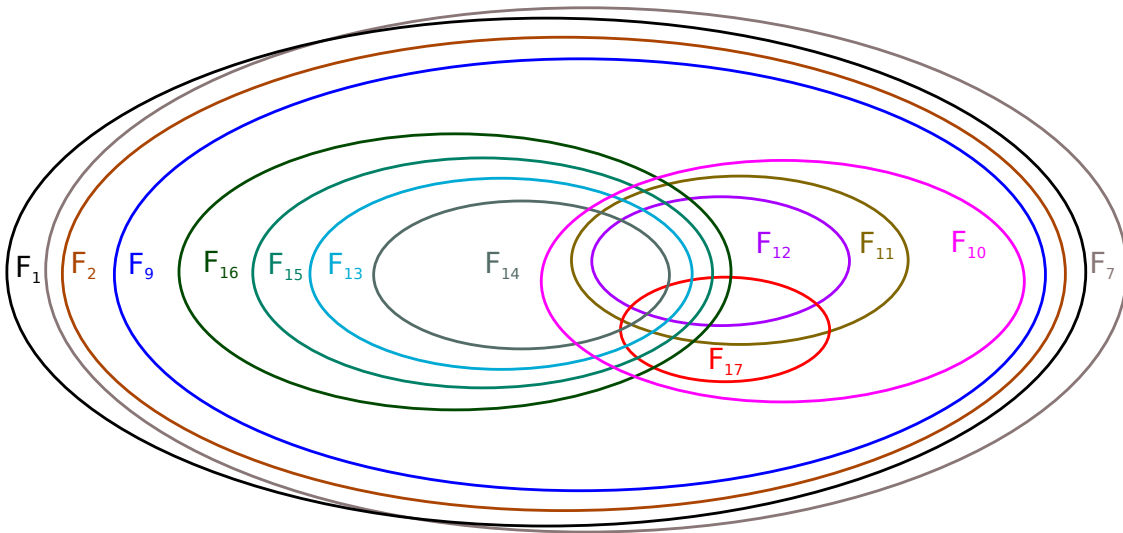


FIGURE 3.25 – Relation d'inclusion entre différentes familles de graphes variant dans le temps.

étape atomique, un processus peut envoyer un message à l'un de ses voisins ou recevoir un message de la part d'un de ses voisins mais pas les deux simultanément. Une illustration du modèle à passage de messages est dans la figure 3.26(a).

2. **Modèle à registre partagé** [Lam86] : Dans ce modèle, les différents processus souhaitant communiquer partagent une variable commune de telle sorte que seulement deux opérations sont autorisées par les processus sur cette variable. L'écriture qui permet de modifier la valeur de la variable par un processus et la lecture qui retourne la valeur de la variable au processus l'invoquant. Les processus souhaitant émettre une information essaient d'accéder en écriture au registre quand les processus récepteur de l'information y accède en lecture. En général, l'accès à la variable en écriture est exclusif à un certains nombre de processus à la fois (ça peut être un à la fois). Ce modèle de registre partagé est illustré dans la figure 3.26(c).
3. **Modèle à état** [Dij74] : Dans ce modèle, on suppose que chaque processus a une ressource mémoire qu'il partage avec l'ensemble de ses voisins. Cette ressource est accessible uniquement en lecture par les voisins. Dans ce modèle, on suppose qu'en une seule étape atomique un processus peut lire l'ensemble des états de ses voisins et mettre à jour son propre état. Ce modèle est illustré dans la figure 3.26(d).
4. **Modèle à registre** [DIM93] : Dans ce modèle, les sommets partagent leurs informations à travers des registres qui peuvent être lus par les voisins. Par conséquent, chaque processus a autant de registres que de voisins. En une étape atomique, un processus peut lire le registre d'un seul de ses voisins ou mettre à jour son propre registre. Cependant il ne peut faire les deux conjointement. De plus, chaque registre peut être lu par un seul voisin par étape. Ce modèle est illustré dans la figure 3.26(b).

Dans cette thèse nous utiliserons uniquement le modèle à passage de messages. Ce modèle est le plus réaliste des trois modèles étant donné que dans la réalité les échanges d'informations se font à travers des messages. Il existe cependant divers variantes de ce modèle (voir [Lyn96] par exemple) en fonction des hypothèses faites sur les propriétés des canaux de communications.

Les principales hypothèses courantes sont les suivantes :

1. les canaux de communications peuvent-être synchrones ou asynchrones, à savoir si le temps de transfert d'un message dans un canal de communication est borné ou non.
2. les canaux de communications sont fiables ou non. En d'autres termes chaque message envoyé est assuré d'être délivré ou à l'inverse des pertes peuvent se produire.
3. la capacité de chaque canal de communication est bornée ou non, (c'est-à-dire, le nombre de messages distincts que peut accueillir chaque canal de communication est borné ou non).
4. le canal de communication est FIFO (First In-First Out : le premier message entrant est le premier à sortir) ou non, c'est-à-dire que soit il ne peut

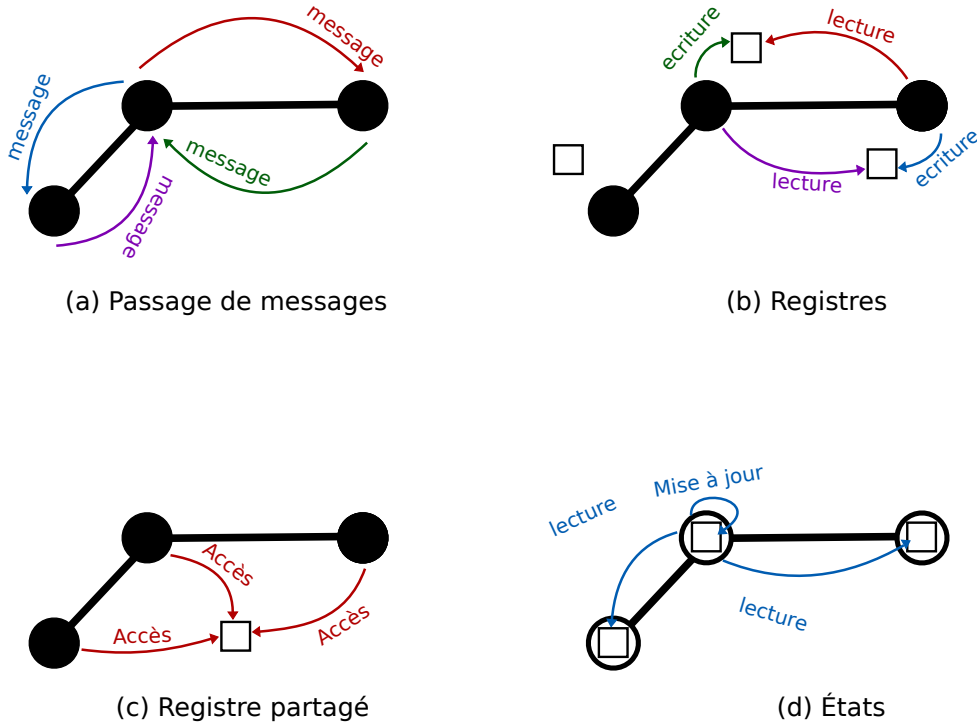


FIGURE 3.26 – Modèles de communication. Chaque couleur représente une étape atomique différente.

y avoir de dépassement entre les messages et un message sortant est toujours plus ancien que l'ensemble des messages qui résident encore dans le canal de communication, ou les dépassements sont permis et aucun ordre n'est défini.

Dans cette thèse nous supposons un modèle de passage de messages satisfaisant les propriétés suivantes :

1. les canaux de communications sont asynchrones (il n'existe aucune borne pour le temps de livraison des messages dans les canaux)
2. nous supposons que les liens de communications sont fiables. À savoir qu'un message ne peut être perdu par un canal de communication tant que ce dernier est présent.
3. nous considérons que la capacité des canaux de communications est bornée par un entier strictement positif x (au plus x messages peuvent être accueillis par un canal de communication)
4. finalement nous émettons l'hypothèse que les canaux de communications sont FIFOs. (premier entrant premier sortant), c'est-à-dire qu'il ne peut y avoir de dépassement entre les messages et un message sortant est toujours plus ancien que l'ensemble des messages qui résident encore dans le canal de communication. Cependant si un canal disparaît l'ensemble des messages qu'il contenait est perdu.

Nous utilisons les $\mathcal{GVT}$ pour décrire les capacités de communications d'un système distribué. Nous considérons un système distribué dynamique composé de n

processus désormais nommés indifféremment par sommets, nœuds ou unité de calcul. L'ensemble des unités de calcul du système distribué est représenté par l'ensemble V d'un $\mathcal{GVT}$ $\mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta, \phi)$. Nous supposons que chaque sommet de V a une mémoire locale, un algorithme déterministe séquentiel local et est doté de capacité de communication comme le veut un système distribué. Nous considérons l'hypothèse que chaque sommet a un identifiant unique. Par ailleurs, étant donnés deux sommets distincts p et q ayant pour identifiant respectivement id_p et id_q , on a soit $id_p > id_q$ soit $id_q > id_p$.

L'ensemble E quand à lui représente l'ensemble des liens de communications entre des paires de processus. Un lien de communication désormais sera désigné indifféremment par arête ou relation. Une paire de sommets p et q de V^2 sont considérés comme des voisins à partir du moment où $\{p, q\} \in E$. Nous utilisons la notation $\mathcal{N}_p$ pour désigner l'ensemble des voisins du sommet p , c'est-à-dire, l'ensemble des sommets $q | \{p, q\} \in E$. Nous supposons que l'intersection entre deux arêtes $e, e' \in E^2$ donne l'ensemble des sommets qu'elles partagent comme extrémité.

Nous émettons l'hypothèse que les processus communiquent en utilisant le modèle de passage de messages (voir 3.3 pour le modèle de passage de messages). La présence d'une relation entre deux processus p et q à un instant t implique que le processus p respectivement q peut envoyer un message à l'instant t au processus q respectivement au processus p . Nous émettons l'hypothèse que notre système distribué est asynchrone, à savoir qu'aucune paire de processus n'a accès à un quelconque périphérique partagé qui permettrait de synchroniser leur vitesse d'exécution. De plus, à aucun moment les processus n'ont accès à la valeur de la fonction de latence des arêtes ζ , les processus ne peuvent par conséquent pas connaître la latence des canaux de communications ni prédire une borne sur le retard d'un message. De même pour la fonction de latence des sommets ϕ , pour laquelle il n'existe pas de borne connue des sommets.

Notez que la capacité d'un sommet à envoyer un message à un autre sommet à un moment donné ne signifie pas que ce message sera délivré. En effet la dynamique du réseau peut induire à ce que le lien entre les deux sommets disparaisse avant la réception du message conduisant ainsi à sa perte.

Comme dans [CFMS15], nous faisons l'hypothèse que la présence et l'absence des liens de communications sont détectées instantanément par les deux sommets extrémités du lien. Nous supposons que notre système fournit à chaque processus une primitive de communication non bloquante nommée **Send_retry** [CFMS15] qui garantit la propriété suivante : quand un processus p invoque **Send_retry**(m, q) (où m est un message arbitraire et q un autre processus de V) à un moment donné t , la primitive délivre le message m au processus q en un temps fini à condition qu'il existe un temps $t' \geq t$ tel que l'arête $\{p, q\}$ est présente (voir la définition pour la présence d'une arête). En d'autres termes, la remise du message est assurée, après l'appel de la primitive **Send_retry**, s'il y a une disponibilité de l'arête qui dure suffisamment longtemps pour surmonter le délai de communication de l'arête à cet instant. Notez que cette primitive peut ne jamais délivrer un message. En effet, si l'arête utilisée pour l'envoi du message ne réapparaît plus jamais après l'invocation de la primitive, alors le message ne sera jamais délivré. L'avantage que nous procure cette primitive

est l'abstraction par rapport aux changements topologiques et aux communications asynchrones pour écrire des algorithmes de haut niveau.

Dans le cadre du modèle à passage de messages l'état du processus est défini par la valeur de ses variables locales. L'état des processus et des liens de communications du graphe $\mathcal{G}$ à un instant donné définissent une configuration. Formellement, étant donné un graphe variant dans le temps $\mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta, \phi)$ une configuration de $\mathcal{G}$ est un vecteur de $n+2$ éléments $(g_i, M_i, p_1, p_2, \dots, p_n)$ où g_i est un graphe statique qui appartient à la suite d'évolutions de $\mathcal{G}$ (Voir 3.1 page 36 pour la définition de la suite d'évolutions d'un $\mathcal{GVT}$), où M_i est l'ensemble de multi-ensembles des messages que transporte chaque lien de communication E_i et où p_1 jusqu'à p_n représentent les états des n processus de V . Nous supposons qu'un processus p retourne une valeur v dans une configuration γ si l'une de ses variables (appelée variable de sortie/retour) a la valeur v dans γ .

Exécution Une exécution d'un système distribué modélisé par un graphe variant dans le temps $\mathcal{G}$ est une suite de configurations $\mathcal{E} = \gamma_0, \dots, \gamma_k, \gamma_{k+1}, \dots$ telle que pour chaque $k \geq 0$ une étape d'exécution (γ_k, γ_{k+1}) , correspond à l'un des événements suivant : (i) $g_k \neq g_{k+1}$, ou bien (ii) au moins un processus reçoit un message d'un voisin ou envoie un message à un voisin ou exécute quelques instructions internes qui changent son état. L'algorithme exécuté sur $\mathcal{G}$ représente l'ensemble de toutes les actions internes autorisées par les sommets (en fonction de leur état courant ou des événements externes à l'image de la réception d'un message, ou l'expiration d'un délai d'attente) au cours d'une exécution dans $\mathcal{G}$.

Nous émettons l'hypothèse que pendant toutes les étapes (γ_k, γ_{k+1}) d'une exécution, si $g_k \neq g_{k+1}$ alors pour chaque arête e où $e \in E_k$ et $e \notin E_{k+1}$ (c'est-à-dire, e disparaît pendant l'étape (γ_k, γ_{k+1}) , plus aucun message transporté par e n'appartient à M_{k+1} . Cela signifie, la perte de tous les messages que transporte un lien de communication lors de sa disparition.

Nous supposons aussi que, pour toute arête e , si $e \in E_{k+1}$ et $e \notin E_k$ (c'est-à-dire que l'arête e apparaît lors de l'étape (γ_k, γ_{k+1})), alors e ne contient aucun message dans la configuration γ_{k+1} . En d'autres termes, tout lien de communication qui apparaît est vide.

Dans le cadre de nos travaux, nous étudions des graphes dynamiques qui ont un temps de vie non borné. Nous nous sommes plus particulièrement intéressés à la famille de graphes connectés à travers le temps que l'on notera $\mathcal{CTT}$. Rappelons que dans un tel $\mathcal{GVT}$, il existe toujours un trajet entre toute paire de processus de V ce qui donne à tout processus $p \in V$ la possibilité d'envoyer un message vers un processus $q \in V$ si la primitive `send_retry` est utilisée par chaque processus intermédiaire dans ce trajet. Par définition, un $\mathcal{GVT}$ qui appartient à $\mathcal{CTT}$ a un temps de vie infini. Nous supposons que tout processus ne connaît pas les trajets qui le lie avec les autres processus du graphe. De ce fait, la famille $\mathcal{CTT}$ nous permet de capturer un réseau hautement dynamique étant donnée qu'aucune autre hypothèse supplémentaire n'est définie (tels que des délais sur l'apparition des arêtes, ou sur des chemins utilisés par le passé) hormis l'existence de chemins temporels dans tout futur entre toute paire de

processus.

Chapitre 4

Spécifications et systèmes dynamiques

Sommaire

4.1	Calculs d’empreintes	62
4.2	Ensemble dominant et couplage	63
4.3	Approches temporelles, évolutives et permanentes . . .	64
4.3.1	Approche permanente	65
4.3.2	Approche évolutive	67
4.3.3	Approche temporelle	70
4.4	Approche ultime	72
4.5	Conclusion	77

Dans ce chapitre, nous étudions la spécification des problèmes distribués dans un système dynamique. En effet, l’évolution des modèles de systèmes distribués s’est dirigée progressivement vers des graphes qui prennent de plus en plus en compte la dynamique des éléments d’un système. Cela rend les spécifications traditionnelles (c’est-à-dire, celles définies dans un environnement statique) non adaptées voire incorrectes.

Les systèmes dynamiques peuvent apporter des problèmes qui n’existent pas dans les systèmes statiques. Par conséquent, il faut les définir. Les problèmes de calcul de l’empreinte globale et de l’empreinte ultime tombent dans cette catégorie (voir la section 4.1).

Ensuite, nous allons discuter des problèmes de couverture. Dans un graphe $g = (V, E)$, un problème de couverture est le problème du calcul d’un sous ensemble de V ou de E ayant certaines propriétés d’adjacence. Nous montrons que la définition traditionnelle de ces problèmes peut être inadaptée dans un contexte dynamique (Voir la section 4.2). Par conséquent, il faut redéfinir le problème pour l’adapter à la dynamique du système. Néanmoins, il existe plusieurs méthodes pour redéfinir les problèmes de couverture [CMM11]. Nous allons donc présenter les différents travaux de la littérature qui redéfinissent les problèmes de couverture pour les systèmes dynamiques (Voir la section 4.3). Nous présentons l’approche permanente dans laquelle la

spécification impose de calculer une solution unique mais correcte à tout instant. L'approche évolutive consiste à calculer une nouvelle solution à chaque étape d'évolution du $\mathcal{GV}\mathcal{T}$ dans le temps. Enfin, l'approche temporelle requiert le calcul d'une solution unique calculée sur l'empreinte globale du $\mathcal{GV}\mathcal{T}$. Nous présentons ces différentes approches sur les deux exemples suivant : le calcul de l'ensemble dominant et du couplage.

Après avoir présenté les défauts de ces différentes approches dans le modèle que nous étudions, nous introduisons dans la section 4.4 une nouvelle approche (dite ultime) que nous jugeons plus adaptée aux graphes hautement dynamiques de la famille $\mathcal{CTT}$. À titre d'exemple, nous spécifions les problèmes de couplage maximal et de l'ensemble dominant minimal selon cette approche.

Nous clôturerons par une discussion concernant les points faibles de cette nouvelle approche. Nous évoquons notamment l'impossibilité de satisfaire la spécification de certains problèmes sur la famille $\mathcal{CTT}$ de $\mathcal{GV}\mathcal{T}$ que nous aborderons dans le chapitre 5.

4.1 Calculs d'empreintes

Dans cette section, nous spécifions les problèmes du calcul de l'empreinte globale et ultime dans un système dynamique. Rappelons que l'empreinte globale d'un $\mathcal{GV}\mathcal{T}$ (voir la définition 3.2, page 36 pour l'empreinte globale) est le graphe statique comprenant l'ensemble des sommets du $\mathcal{GV}\mathcal{T}$ et l'ensemble des liens de communication qui sont présents au moins une fois durant du $\mathcal{GV}\mathcal{T}$. En particulier, les arêtes vouées à disparaître sont incluses dans l'ensemble des arêtes de cette empreinte globale. L'empreinte ultime d'un $\mathcal{GV}\mathcal{T}$ est le graphe statique composé de l'ensemble des sommets du $\mathcal{GV}\mathcal{T}$ et des arêtes qui sont infiniment souvent présentes dans le $\mathcal{GV}\mathcal{T}$.

Calcul de l'empreinte globale. De manière intuitive, le problème du calcul de l'empreinte globale consiste, pour chaque processus, à construire une copie locale de l'empreinte globale du $\mathcal{GV}\mathcal{T}$. Formellement, nous pouvons spécifier le problème de la manière suivante.

Spécification 4.1 (Problème du calcul de l'empreinte globale) *Un algorithme déterministe $\mathcal{A}$ satisfait la spécification du calcul de l'empreinte globale pour une famille de graphes variant dans le temps $\mathcal{C}$ si toute exécution $\mathcal{E} = \gamma_0, \gamma_1, \dots$ de $\mathcal{A}$ sur tout graphe variant dans le temps $\mathcal{G}$ de $\mathcal{C}$ a un suffixe $\mathcal{E}_i = \gamma_k, \gamma_{k+1}, \dots$ tel que chaque processus fournit en sortie l'empreinte globale de $\mathcal{G}$ pour toute configuration de $\mathcal{E}_i$.*

Calcul de l'empreinte ultime. De manière analogue, le problème du calcul de l'empreinte ultime revient, pour chaque processus, à construire une copie locale de l'empreinte ultime du $\mathcal{GV}\mathcal{T}$. Formellement, nous pouvons spécifier le problème de la manière suivante.

Spécification 4.2 (Problème du calcul de l’empreinte ultime) *Un algorithme déterministe $\mathcal{A}$ satisfait la spécification du calcul de l’empreinte ultime pour une famille de graphes variant dans le temps $\mathcal{C}$ si toute exécution $\mathcal{E} = \gamma_0, \gamma_1, \dots$ de $\mathcal{A}$ sur tout graphe variant dans le temps $\mathcal{G}$ de $\mathcal{C}$ a un suffixe $\mathcal{E}_i = \gamma_k, \gamma_{k+1}, \dots$ tel que chaque processus fournit en sortie l’empreinte ultime de $\mathcal{G}$ pour toute configuration de $\mathcal{E}_i$.*

4.2 Ensemble dominant et couplage

Avant d’aborder les différentes approches qui peuvent-être utilisées pour redéfinir les problèmes dans les systèmes dynamiques, nous rappelons la définition de deux problèmes très étudiés en théorie des graphes : à savoir l’ensemble dominant et le couplage.

Ces deux problèmes serviront d’exemple et seront reformulés au fur et à mesure selon les critères des approches de redéfinition que nous présentons dans la section 4.3 et 4.4.

Couplage. En théorie des graphes, un couplage est un ensemble d’arêtes (de couples de sommets voisins) qui n’ont pas de sommets en commun. Le couplage peut servir à résoudre bon nombre de problèmes. L’une des applications possibles consiste à répondre à un problème d’affectation de tâches aux ouvriers, où par exemple il faut affecter une tâche par ouvrier. On peut aussi se servir du couplage pour résoudre des problèmes complexes comme le voyageur de commerce de manière approximative [Chr76].

Définition 4.1 (Couplage) *Un **couplage** M dans un graphe $g = (V, E)$ est un ensemble d’arêtes deux à deux non adjacentes, c’est-à-dire, $M \subseteq E$ et $\forall (e_1, e_2) \in M^2, e_1 \neq e_2 \Rightarrow e_1 \cap e_2 = \emptyset$.*

*Un **couplage maximal** dans un graphe $g = (V, E)$ est un couplage M de g tel que, pour toute arête $e \in E$ qui n’est pas dans M , $M \cup \{e\}$ n’est pas un couplage de g .*

La figure 4.1 présente une illustration de ces définitions.

Ensemble dominant. Un ensemble dominant d’un graphe $g = (V, E)$ est un ensemble de sommets (appelé *dominants* ou *sommets dominants*) tel que tout sommet non dominant a un voisin dominant. Une des applications possibles de l’ensemble dominant dans le domaine des réseaux est la construction de cluster. En effet, l’une des méthodes possibles pour choisir les têtes de cluster d’un réseau consiste à construire un ensemble dominant et à prendre les sommets dominants comme têtes de cluster.

Définition 4.2 (Ensemble dominant) *Un **ensemble dominant** d’un graphe statique $g = (V, E)$ est un sous ensemble de sommets D de V de telle sorte que tout sommet n’appartenant pas à D est voisin d’au moins un sommet de D .*

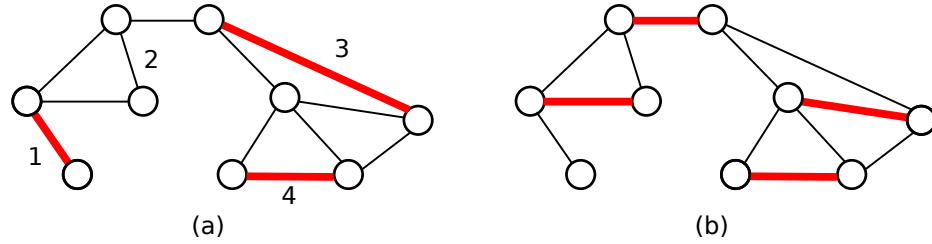


FIGURE 4.1 – Dans cet exemple les arêtes **en gras** représentent les arêtes du couplage. La figure 4.1(a) illustre un couplage $M_1 = \{1, 3, 4\}$ qui n'est pas maximal (puisque'il existe un couplage $M_2 = \{1, 2, 3, 4\}$ qui contient une arête supplémentaire tel que $M_1 \subset M_2$), tandis que la figure 4.1(b) illustre un couplage maximal.

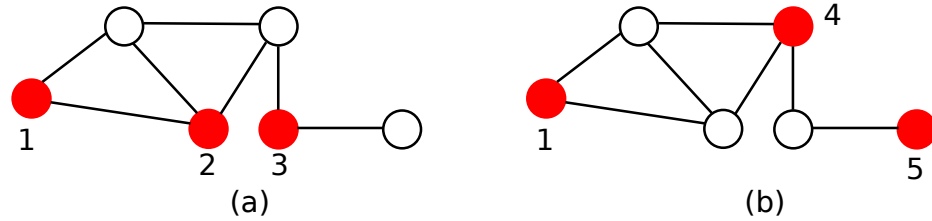


FIGURE 4.2 – Dans cet exemple, les sommets **pleins** représentent les sommets dominants tandis que les sommets **non pleins** représentent les sommets dominés. La figure 4.2(a) illustre un ensemble dominant $m_1 = \{1, 2, 3\}$ qui n'est pas minimal puisque l'ensemble $m_2 = \{2, 3\}$ est un ensemble dominant et $m_2 \subset m_1$. La figure 4.2(b) illustre un ensemble dominant minimal $m_3 = \{1, 4, 5\}$ puisqu'il n'existe pas de sous-ensemble strict de m_3 qui soit un ensemble dominant.

Un **ensemble dominant minimal** d'un graphe statique $g = (V, E)$ est un ensemble dominant D de g tel qu'il n'existe pas de sous ensemble strict de D qui soit un ensemble dominant de g .

La figure 4.2 correspond à une illustration d'un ensemble dominant minimal et non minimal.

4.3 Approches temporelles, évolutives et permanentes

Dans un contexte dynamique il devient nécessaire parfois, de redéfinir les problèmes traditionnels des graphes statiques. En effet, comme le prouvent les auteurs dans [CMM11], définie dans un cadre statique, la spécification d'un problème traditionnel des graphes peut s'avérer être inadaptée aux hypothèses de dynamique du système. La raison principale à cela est qu'elle ne prend pas en compte les changements topologiques.

Lorsqu'il s'agit d'envisager de redéfinir un problème pour l'adapter aux systèmes dynamiques il est tout à fait naturel de se poser certaines questions [CMM11].

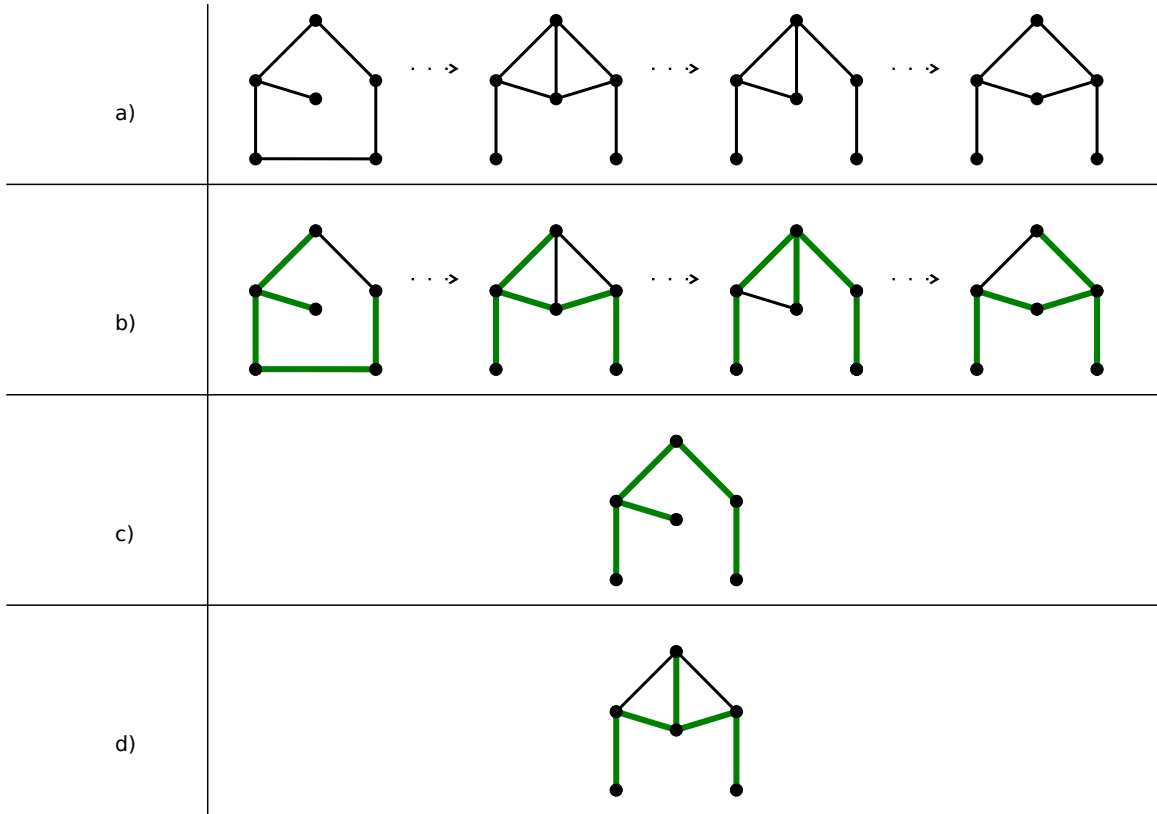


FIGURE 4.3 – Pour le graphe de la figure 4.3(a), la figure 4.3(b) illustre un exemple de graphe couvrant évolutif où pour chaque évolution du graphe dans le temps une solution est calculée. La figure 4.3(c) correspond à un graphe couvrant permanent où une seule solution doit satisfaire toute la dynamique du système. Enfin, la figure 4.3(d) illustre un graphe couvrant temporel où le temps est aplati de telle sorte que l'arbre proposé est défini sur les arêtes du système dans sa globalité

Doit-on définir le problème de telle sorte qu'une solution unique soit en tout instant vraie ? Ou doit-on résoudre le problème de manière classique pour chaque évolution du système ? Ou doit-on considérer l'évolution du système comme un tout afin de proposer une solution qui se vérifie dans le temps sous certaines conditions ? De toute évidence, cela dépend de l'importance des changements topologiques, de leurs fréquences et de ce que peut tolérer comme inexactitude l'application exécutée dans cet environnement. La figure 4.3 correspond à une illustration, dans un cadre dynamique, du problème de l'arbre couvrant défini sous différentes approches.

4.3.1 Approche permanente

L'idée de cette approche consiste à définir le problème de telle sorte que la solution recherchée est vraie dans un $\mathcal{GVT}$ à tout instant.

Dans ce qui suit, nous allons redéfinir le problème de l'ensemble dominant et le problème du couplage selon l'approche permanente.

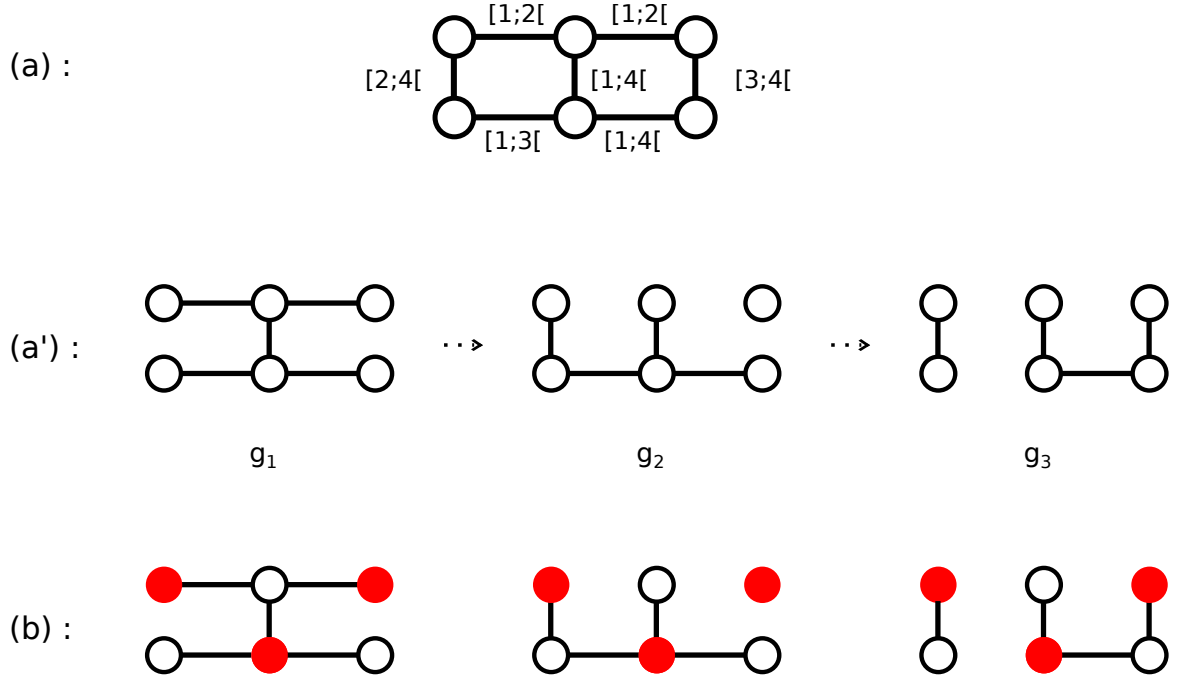


FIGURE 4.4 – Ensemble dominant permanent. Dans cet exemple, la figure 4.4(a) illustre un $\mathcal{GVT}$ $\mathcal{G}$ où ζ et ϕ sont égales à 0. La figure 4.4(a') correspond à l'illustration de $\mathcal{G}$ par sa suite d'évolutions. Les sommets **pleins** de la figure 4.4(b) correspondent aux sommets appartenant à l'ensemble dominant permanent.

Ensemble dominant permanent L'approche permanente appliquée au problème de l'ensemble dominant revient à chercher un ensemble dominant unique défini comme suite :

Définition 4.3 *Un ensemble dominant permanent d'un $\mathcal{GVT}$ $\mathcal{G} = (V, E, S_{\mathcal{G}}, \mathcal{T}_{\mathcal{G}}, \zeta, \phi)$ est un ensemble $D \subseteq V$ tel que D est un ensemble dominant de tout graphe de la suite $S_{\mathcal{G}}$.*

La figure 4.4 illustre un exemple d'un ensemble dominant permanent sur un $\mathcal{GVT}$.

Couplage permanent L'approche permanente appliquée au problème du couplage débouche sur la définition suivante :

Définition 4.4 *Un couplage permanent d'un $\mathcal{GVT}$ $\mathcal{G} = (V, E, S_{\mathcal{G}}, \mathcal{T}_{\mathcal{G}}, \zeta, \phi)$ est un ensemble d'arêtes $M \in E$ tel que M est un couplage de tout graphe de la suite $S_{\mathcal{G}}$.*

La figure 4.5 illustre un exemple d'un couplage permanent sur un $\mathcal{GVT}$.

Points faibles de l'approche permanente. La pertinence et même la possibilité de résoudre un problème donné redéfini par l'approche permanente dépend des composantes qui sont connexes à tout instant sur le $\mathcal{GVT}$. En effet, selon la dynamique du système les deux problèmes suivants peuvent apparaître :

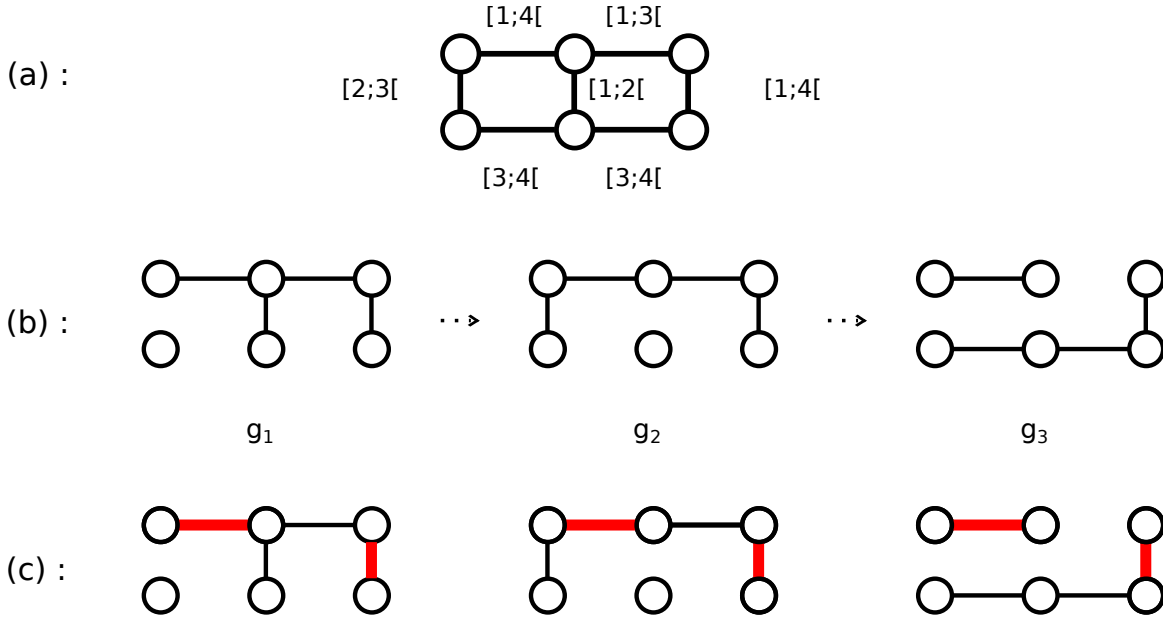


FIGURE 4.5 – Dans cet exemple, la figure 4.5(b) illustre la suite d'évolutions du $\mathcal{GVT}$ de la figure 4.5(a) sous l'hypothèse que les latences ζ et ϕ sont égales à 0. Les arêtes **en gras** dans la figure 4.5(c) représentent le couplage permanent.

- la solution obtenue est difficilement exploitable comme illustré par l'ensemble dominant permanent de la figure 4.6 ;
- ou l'inexistence de solution au problème, comme l'illustre la figure 4.7 pour le problème du couplage.

4.3.2 Approche évolutive

L'approche évolutive est la plus utilisée dans la littérature lorsqu'il s'agit de redéfinir la spécification de problèmes pour des systèmes dynamiques. L'idée derrière cette approche est assez simple. Elle consiste à considérer chaque évolution du $\mathcal{GVT}$ comme une suite de graphes statiques. Pour un problème donné, l'approche évolutive consistera à calculer une nouvelle solution pour chaque nouveau graphe statique de cette suite.

Il est important de noter que la solution sera probablement différente pour chaque graphe. Cette approche est donc une approche de maintenance étant donné que l'on recalcule (probablement) une nouvelle solution à chaque changement topologique.

Dans ce qui suit on considère la représentation des graphes variant dans le temps par leur suite d'évolutions $\mathcal{G} = (V, E, S_{\mathcal{G}}, \mathcal{T}_{\mathcal{G}}, \zeta, \phi)$ qui reflète l'évolution au cours du temps par une suite de graphes statiques $S_{\mathcal{G}}$ (voir 3.1 page 36 pour la suite d'évolutions d'un $\mathcal{GVT}$).

Nous illustrons dans la suite cette approche avec le problème de l'ensemble dominant et le problème du couplage.

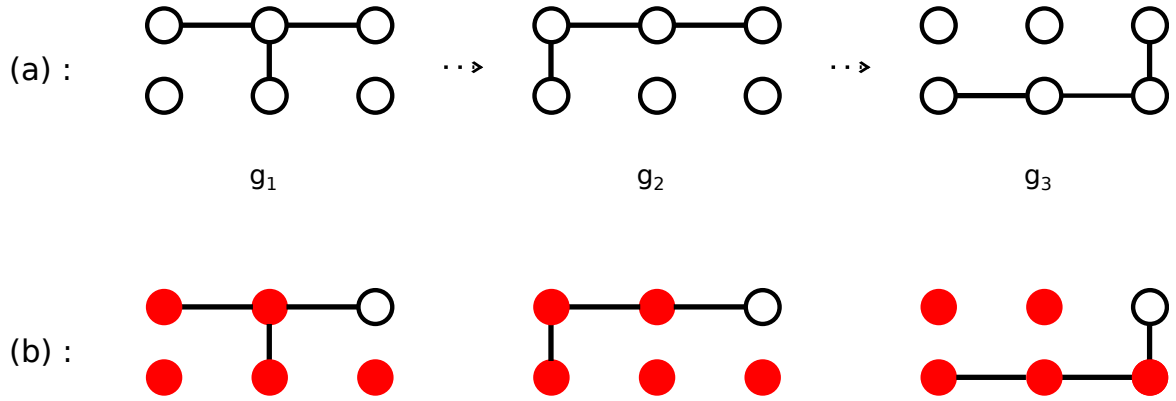


FIGURE 4.6 – L’unique solution pour résoudre le problème de l’ensemble dominant par l’approche permanente dans le $\mathcal{GV}\mathcal{T}$ de la figure 4.6(a) est de prendre presque tous les sommets du $\mathcal{GV}\mathcal{T}$ dans l’ensemble dominant. De manière générale, dans un $\mathcal{GV}\mathcal{T}$ de la famille $\mathcal{CTT}$ si toute arête disparaît au moins une fois durant la vie du $\mathcal{GV}\mathcal{T}$ alors l’ensemble dominant permanent est l’ensemble des sommets du $\mathcal{GV}\mathcal{T}$.

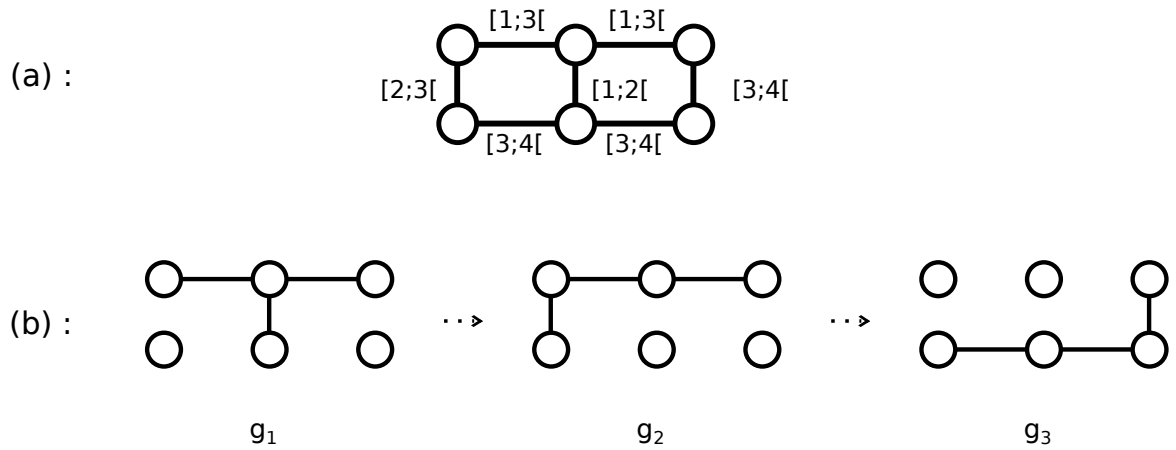


FIGURE 4.7 – Dans cet exemple, la figure 4.7(b) illustre la suite d’évolutions du $\mathcal{GV}\mathcal{T}$ de la figure 4.5(a) sous l’hypothèse que les latences ζ et ϕ sont égales à 0. Nous constatons que sur ce $\mathcal{GV}\mathcal{T}$ il n’existe pas de couplage permanent (aucune des arêtes du $\mathcal{GV}\mathcal{T}$ n’est présente à tout moment pour permettre un couplage permanent).

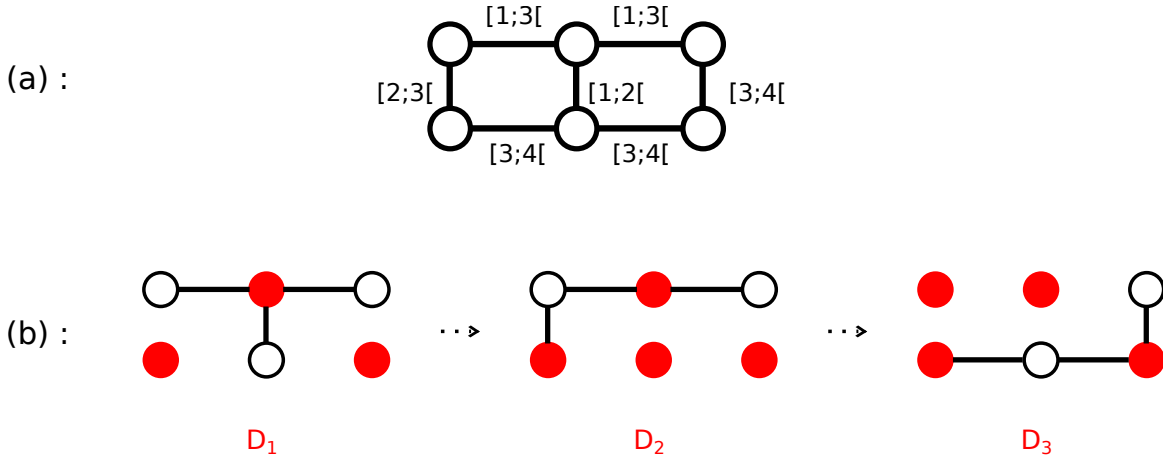


FIGURE 4.8 – La figure 4.4(a) représente un $\mathcal{GVT}$ $\mathcal{G}$ où ζ et ϕ sont égales à 0. La figure 4.4(b) correspond à l'illustration de $\mathcal{G}$ par sa suite d'évolutions. Les sommets **pleins** correspondent aux sommets appartenant à un ensemble dominant. Par conséquent, par les sommets **pleins**, la suite (D_i) de la figure 4.8(b) illustre un ensemble dominant évolutif du $\mathcal{GVT}$ $\mathcal{G}$.

Ensemble dominant évolutif. Les ensembles dominants évolutifs représentent l'approche la plus fréquente dans la littérature dès lors qu'il s'agit d'adapter la définition d'un problème à un système dynamique. L'ensemble dominant évolutif est une suite d'ensemble de sommets $\{M_i\}$ de telle sorte que chaque ensemble de sommets de la suite correspond à un ensemble dominant pour chaque graphe (statique) de la suite d'évolutions $S_{\mathcal{G}}$.

Définition 4.5 *Un ensemble dominant évolutif d'un $\mathcal{GVT}$ $\mathcal{G} = (V, E, S_{\mathcal{G}}, \mathcal{T}_{\mathcal{G}}, \zeta, \phi)$ est une suite $D_1, D_2, \dots, D_i$ où D_i est un ensemble dominant de g_i pour tout $i \in \mathbb{N}$.*

Un ensemble dominant minimal évolutif d'un $\mathcal{GVT}$ $\mathcal{G} = (V, E, S_{\mathcal{G}}, \mathcal{T}_{\mathcal{G}}, \zeta, \phi)$ est la suite $D_E = D_1, D_2, \dots, D_i$ où D_i est un ensemble dominant minimal de g_i pour tout $i \in \mathbb{N}$.

Il existe des définitions qui ajoutent des contraintes supplémentaires à l'approche évolutive. Par exemple, [CKZ04, CKZ05] introduit le h -ensemble dominant comme un ensemble dominant évolutif tel que chaque ensemble dominant de la suite est valable pendant au moins une durée h .

La figure 4.8 illustre un exemple d'un ensemble dominant évolutif dans un $\mathcal{GVT}$.

Couplage évolutif. Dans le même esprit, on peut redéfinir le couplage selon l'approche évolutive.

Définition 4.6 *Un couplage évolutif d'un $\mathcal{GVT}$ $\mathcal{G} = (V, E, S_{\mathcal{G}}, \mathcal{T}_{\mathcal{G}}, \zeta, \phi)$ est une suite $M_1, M_2, \dots, M_i, \dots$ où M_i est couplage de g_i pour tout $i \in \mathbb{N}$.*

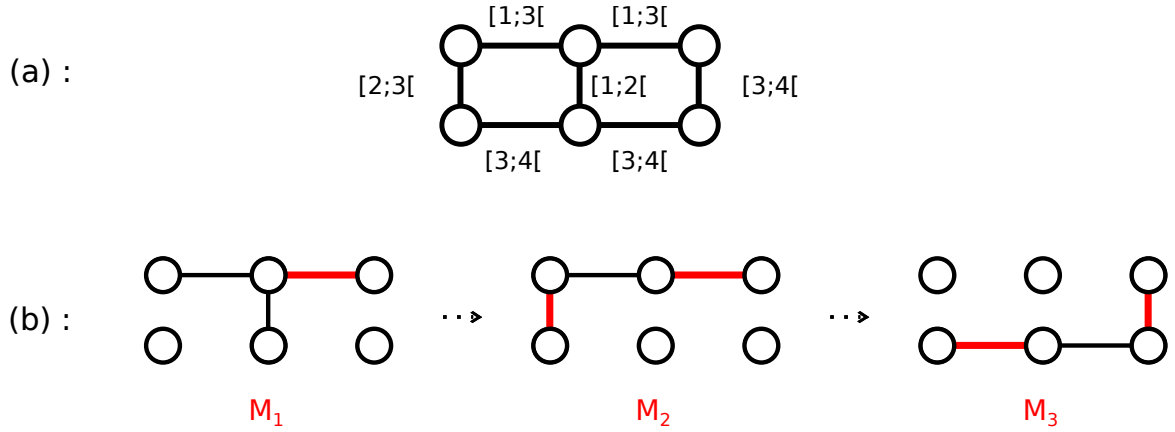


FIGURE 4.9 – Sur cet exemple, la figure 4.4(a) représente un $\mathcal{GVT}$ $\mathcal{G}$ où ζ et ϕ sont égales à 0. Les graphes de la figure 4.4(b) correspondent à la représentation de $\mathcal{G}$ par sa suite d'évolutions. L'ensemble d'arêtes **en gras** est un couplage de chaque g_i . La suite (M_i) est un couplage évolutif de $\mathcal{G}$.

Un **couplage évolutif maximal** d'un $\mathcal{GVT}$ $\mathcal{G} = (V, E, S_g, \mathcal{T}_g, \zeta, \phi)$ est la suite $M_E = M_1, M_2, \dots, M_i, \dots$ où M_i est un couplage maximal de g_i pour tout $i \in \mathbb{N}$.

La figure 4.9 illustre un exemple d'un couplage évolutif sur un $\mathcal{GVT}$.

Points faibles de l'approche évolutive. On peut reprocher à l'approche évolutive les deux points suivants. Premièrement, la qualité de la solution obtenue est très dépendante de la fréquence et de l'importance des changements topologiques du système. En effet, si les changements topologiques sont importants et suivent une cadence trop élevée, alors il est possible que l'algorithme qui doit construire ou maintenir la solution n'ait pas le temps nécessaire de le faire avant un nouveau changement topologique.

Deuxièmement, cette approche n'est adaptée que lorsque le $\mathcal{GVT}$ est souvent connexe. En effet, si on essaye d'appliquer cette approche sur un problème dans un contexte de $\mathcal{GVT}$ faiblement connexe (c'est-à-dire, un $\mathcal{GVT}$ avec très peu de composantes connexes à tout instant), la solution calculée peut être d'une utilité restreinte comme l'illustre l'exemple de l'ensemble dominant évolutif de la figure 4.10.

4.3.3 Approche temporelle

L'idée générale derrière cette approche est de considérer l'évolution du graphe dans son ensemble en considérant son empreinte globale comme support sur lequel les définitions doivent être adaptées. En d'autres termes, un algorithme suivant l'approche temporelle calcule une solution au problème sur l'empreinte globale du $\mathcal{GVT}$ sur lequel il s'exécute.

L'approche temporelle partage un point en commun avec l'approche permanente. En effet, dans ces deux approches il s'agit de trouver une solution stable qui

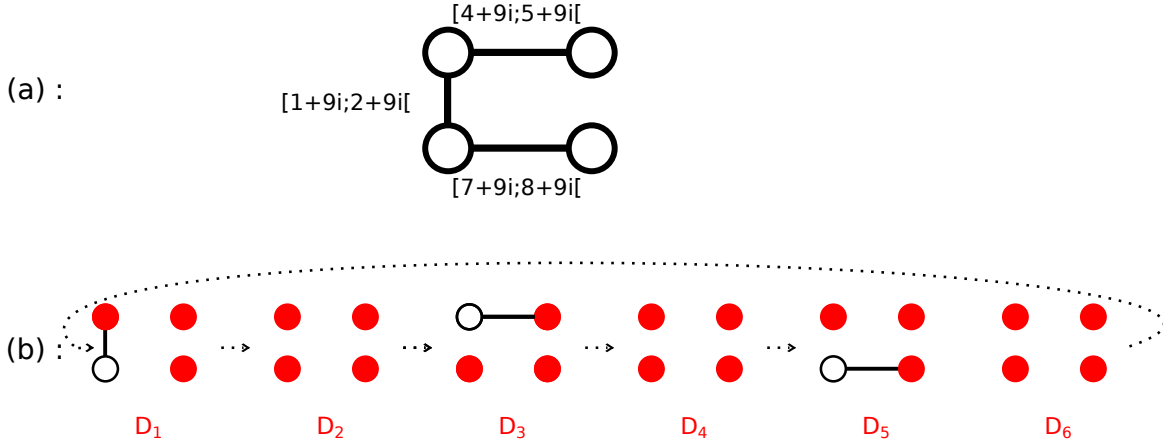


FIGURE 4.10 – Sur cet exemple, la figure 4.4(a) représente un $\mathcal{GVT}$ périodique faiblement connexe $\mathcal{G}$ où ζ et ϕ sont égales à 0. Les graphes de la figure 4.4(b) correspondent à la représentation de $\mathcal{G}$ par sa suite d'évolutions. La suite (D_i) des sommets pleins illustre un ensemble dominant évolutif sur $\mathcal{G}$. Comme $\mathcal{G}$ est faiblement connexe l'ensemble dominant calculé à chaque fois contient un nombre important de sommets. La réduction de l'ensemble des sommets V induite par le calcul de l'ensemble dominant D_i n'est pas importante (souvent $D_i \simeq V$). Par conséquent, l'utilité d'un ensemble dominant évolutif dans ces cas particuliers de $\mathcal{GVT}$ est limitée.

ne change pas. La stabilité de la solution que propose l'approche temporelle est un avantage (par rapport à l'approche évolutive).

Nous allons maintenant définir le problème de l'ensemble dominant et le problème du couplage selon l'approche temporelle.

Ensemble dominant temporel Un ensemble dominant temporel d'un $\mathcal{GVT}$ est un ensemble dominant sur l'empreinte globale du $\mathcal{GVT}$.

Définition 4.7 Dans un graphe variant dans le temps $\mathcal{G}$, un ensemble $D \subseteq V$ est un **ensemble dominant temporel** de $\mathcal{G}$ si et seulement si D est un ensemble dominant de l'empreinte globale $U_{\mathcal{G}}$ de $\mathcal{G}$.

Dans un graphe variant dans le temps $\mathcal{G}$, un ensemble $M \subseteq V$ est un **ensemble dominant minimal temporel** de $\mathcal{G}$ si et seulement si M est un ensemble dominant minimal de l'empreinte globale $U_{\mathcal{G}}$ de $\mathcal{G}$.

La figure 4.11 illustre l'exemple d'un ensemble dominant temporel sur un $\mathcal{GVT}$.

couplage temporel Le couplage temporel est un couplage basé sur l'évolution du graphe dans sa globalité. L'approche temporelle appliquée sur le couplage conduit à la définition suivante :

Définition 4.8 Un **couplage temporel** d'un $\mathcal{GVT}$ $\mathcal{G}$ est un ensemble d'arêtes $M \subseteq E$ tel que M est un couplage de l'empreinte globale $U_{\mathcal{G}}$ de $\mathcal{G}$.

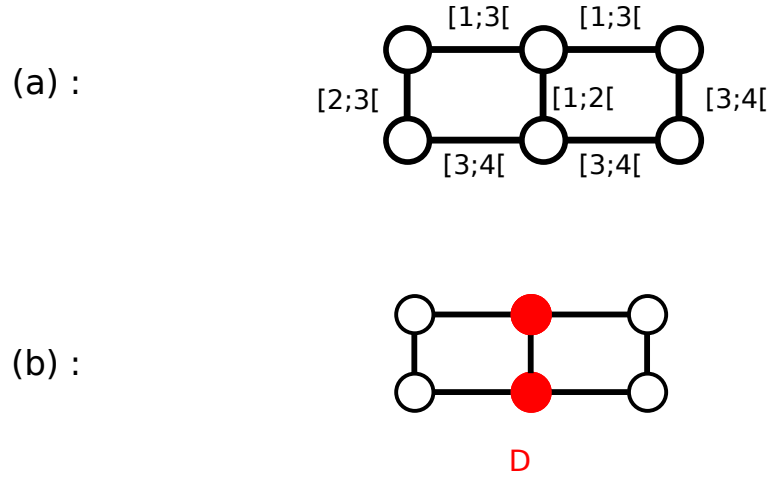


FIGURE 4.11 – Afin de souligner la différence entre les ensembles dominants temporels et les ensembles dominants évolutifs, nous prenons le même $\mathcal{GVT}$ pour exemple comme illustré par la figure 4.11(a). Le graphe de la figure 4.11(b) est l’empreinte globale de $\mathcal{G}$. Dans la figure 4.11(b), les sommets pleins représentent les sommets appartenant à un ensemble dominant minimal temporel D .

Un **couplage maximal temporel** d’un $\mathcal{GVT}$ $\mathcal{G}$ est un ensemble d’arêtes $M \subseteq E$ tel que M est un couplage maximal de l’empreinte globale $U_{\mathcal{G}}$ de $\mathcal{G}$.

La figure 4.12 illustre un exemple de couplage temporel maximal sur un $\mathcal{GVT}$.

Points faibles de l’approche temporelle. L’approche temporelle est souvent utilisée dans un contexte de $\mathcal{GVT}$ où les différents événements topologiques sont récurrents. Par conséquent, cette approche n’est pas pertinente si l’évolution du $\mathcal{GVT}$ n’est pas connue à l’avance ou si ce dernier contient des arêtes vouées à disparaître. En effet, en appliquant cette approche à un problème sur la famille $\mathcal{CTT}$ de $\mathcal{GVT}$, la solution calculée peut être sans intérêt comme l’illustre l’exemple de la figure 4.14 pour le problème du couplage maximal temporel.

En effet, dans cette approche une structure couvrante est calculée sur la base d’empreintes qui peuvent contenir des arêtes vouées à disparaître. De ce fait, l’hypothèse sur laquelle est basée cette approche qui consiste à satisfaire une certaine propriété à travers des liens récurrents est fortement compromise puisqu’une solution dépendant de l’empreinte peut exploiter une arête vouée à disparaître.

4.4 Approche ultime

Les approches précédentes comportent les limitations suivantes :

- une solution n’existe pas toujours avec l’approche permanente ;
- la solution calculée n’est pas stable avec l’approche évolutive ;

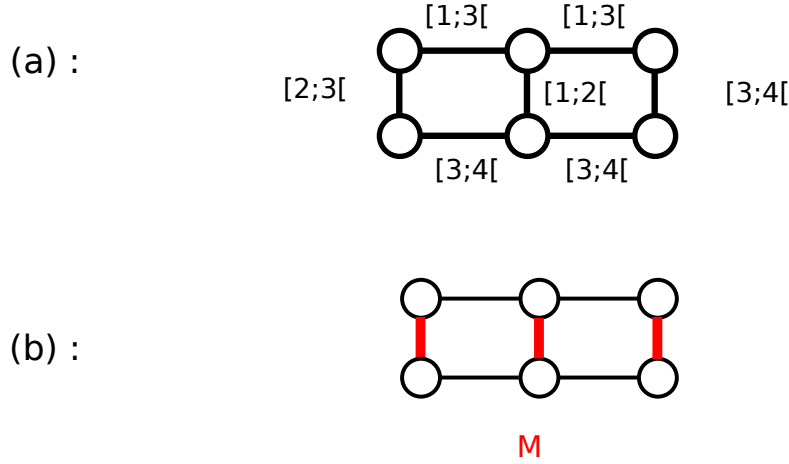


FIGURE 4.12 – Dans cet exemple, la figure 4.12(b) illustre l’empreinte globale du $\mathcal{GV}\mathcal{T}$ de la figure 4.12(a). Dans la figure 4.12(b), les arêtes **en gras** représentent les arêtes d’un couplage maximal temporel.

- la solution calculée comporte le risque d’être inutile ultimement avec l’approche temporelle.

À travers l’approche ultime, nous souhaitons proposer une nouvelle approche qui ne comporte pas ces problèmes. L’intuition était donc de proposer une approche dans laquelle une solution :

- existe toujours
- est stable (c’est-à-dire, une fois atteinte la solution calculée ne changera pas),
- est utile ultimement (c’est-à-dire, une solution satisfaite à travers les récurrences qu’il y a dans un $\mathcal{GV}\mathcal{T}$).

Ensemble dominant ultime. Si on applique l’approche ultime au problème de l’ensemble dominant, on obtient une définition dans laquelle l’ensemble dominant calculé est stable et dans laquelle tout sommet de l’ensemble dominé est infiniment souvent voisin d’au moins un sommet de l’ensemble dominant. En effet, nous imposons alors le calcul d’un ensemble dominant minimal de l’empreinte ultime du $\mathcal{GV}\mathcal{T}$.

Définition 4.9 (Ensemble dominant minimal ultime) *Un **ensemble dominant ultime** d’un $\mathcal{GV}\mathcal{T}$ $\mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta, \phi)$ est un ensemble de sommets $D \subseteq V$ tel que D est un ensemble dominant de l’empreinte ultime $U_{\mathcal{G}}^{\omega}$ de $\mathcal{G}$.*

*Un **ensemble dominant minimal ultime** d’un $\mathcal{GV}\mathcal{T}$ $\mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta, \phi)$ est un ensemble de sommets $D \subseteq V$ tel que D est un ensemble dominant minimal de l’empreinte ultime $U_{\mathcal{G}}^{\omega}$ de $\mathcal{G}$.*

La figure 4.13 illustre la différence entre un ensemble dominant minimal ultime et un ensemble dominant minimal temporel. Après avoir défini l’ensemble dominant ultime, nous spécifions maintenant le problème de construction de l’ensemble dominant minimal ultime comme suit :

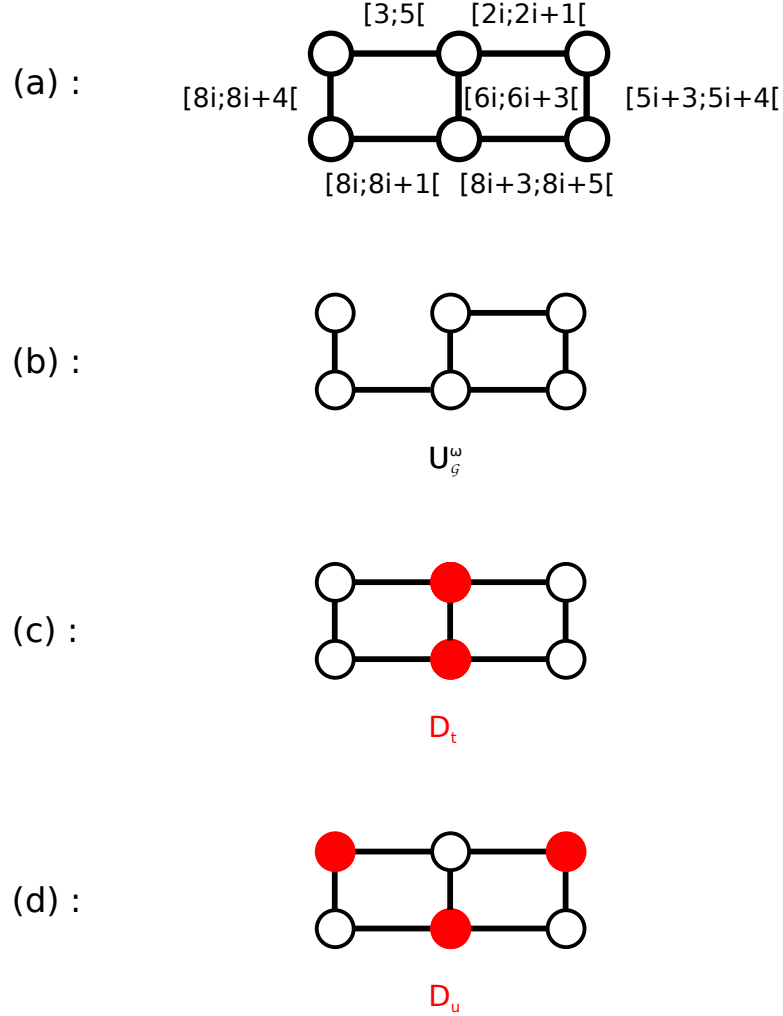


FIGURE 4.13 – Dans cet exemple, la figure 4.13(b) illustre l’empreinte ultime U_G^ω du $\mathcal{GVT}$ $\mathcal{G}$ de la figure 4.13(a). Dans les figures 4.13(c) les sommets pleins représentent les sommets appartenant à un ensemble dominant minimal temporel D_t . Tandis que les sommets pleins de la figures 4.13(d) définissent les sommets appartenant à un ensemble dominant minimal ultime D_u de $\mathcal{G}$. Notez que lorsque l’empreinte ultime est connexe alors un ensemble dominant minimal ultime est toujours un ensemble dominant minimal temporel et que la réciproque n’est pas forcément vrai.

Spécification 4.3 (Construction de l'ensemble dominant minimal ultime)

Un algorithme déterministe $\mathcal{A}$ satisfait la spécification de la construction d'un ensemble dominant minimal ultime pour une famille $\mathcal{C}$ de $\mathcal{GVT}$ si toute exécution $\mathcal{E} = \gamma_0, \gamma_1, \dots$ de $\mathcal{A}$ sur tout graphe variant dans le temps $\mathcal{G}$ de $\mathcal{C}$ a un suffixe $\mathcal{E}_i = \gamma_k, \gamma_{k+1}, \dots$ pour un $k \in \mathbb{N}$ donné tel que tout processus fournit en sortie une valeur booléenne constante $\mathcal{E}_i$ de telle sorte que l'ensemble des processus fournissant en sortie vrai est un ensemble dominant minimal ultime de $\mathcal{G}$.

Couplage. Comme pour l'ensemble dominant, nous proposons ici la définition résultante de l'application de l'approche ultime au couplage. Un couplage ultime est stable et chaque arête de ce couplage est infiniment souvent présente dans le $\mathcal{GVT}$.

Définition 4.10 (Couplage maximal ultime) Un **couplage ultime** dans un $\mathcal{GVT}$ $\mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta, \phi)$ est un ensemble d'arêtes $M \subseteq E$ tel que M est un couplage de l'empreinte ultime $U_{\mathcal{G}}^{\omega}$ de $\mathcal{G}$. Un **couplage maximal ultime** dans un $\mathcal{GVT}$ $\mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta, \phi)$ est un ensemble d'arêtes $M \subseteq E$ tel que M est un couplage maximal de l'empreinte ultime $U_{\mathcal{G}}^{\omega}$ de $\mathcal{G}$.

La figure 4.14 illustre la différence entre un couplage maximal ultime et un couplage maximal temporel. Maintenant que nous avons défini le couplage par l'approche ultime, nous spécifions le problème de construction du couplage maximal ultime comme suit :

Spécification 4.4 (Construction du couplage maximal ultime) Un algorithme déterministe $\mathcal{A}$ satisfait la spécification de construction d'un couplage maximal ultime pour une famille $\mathcal{C}$ de $\mathcal{GVT}$ si chaque exécution $\mathcal{E} = \gamma_0, \gamma_1, \dots$ de $\mathcal{A}$ sur tout $\mathcal{GVT}$ $\mathcal{G}$ de $\mathcal{C}$ a un suffixe $\mathcal{E}_i = \gamma_k, \gamma_{k+1}, \dots$ pour un $k \in \mathbb{N}$ donné tel que tout processus fournit en sortie une valeur booléenne constante sur $\mathcal{E}_i$ de telle sorte que l'ensemble des processus fournissant en sortie vrai est un couplage maximal ultime de $\mathcal{G}$.

Point faible d'une approche ultime. Bien que cette approche comble les points faibles des autres approches, à savoir que cette approche est compatible avec un $\mathcal{GVT}$ sous hypothèse faible de connexité admettant des arêtes vouées à disparaître et sur lequel les changements topologiques sont fréquents. Il comporte néanmoins les points faibles suivant :

- la structure couvrante construite sollicite les liens récurrents cependant aucune borne sur les délais de récurrence n'est émise. En d'autres termes, il est possible que la structure couvrante satisfaisant la propriété ultime le fasse sous des délais non-bornés ce qui peut aussi être synonyme d'une exploitation difficile dans certains contextes.
- il est parfois impossible de construire une structure couvrante pouvant satisfaire cette spécification. Ce que nous soulignerons à travers des exemples dans le chapitre suivant.

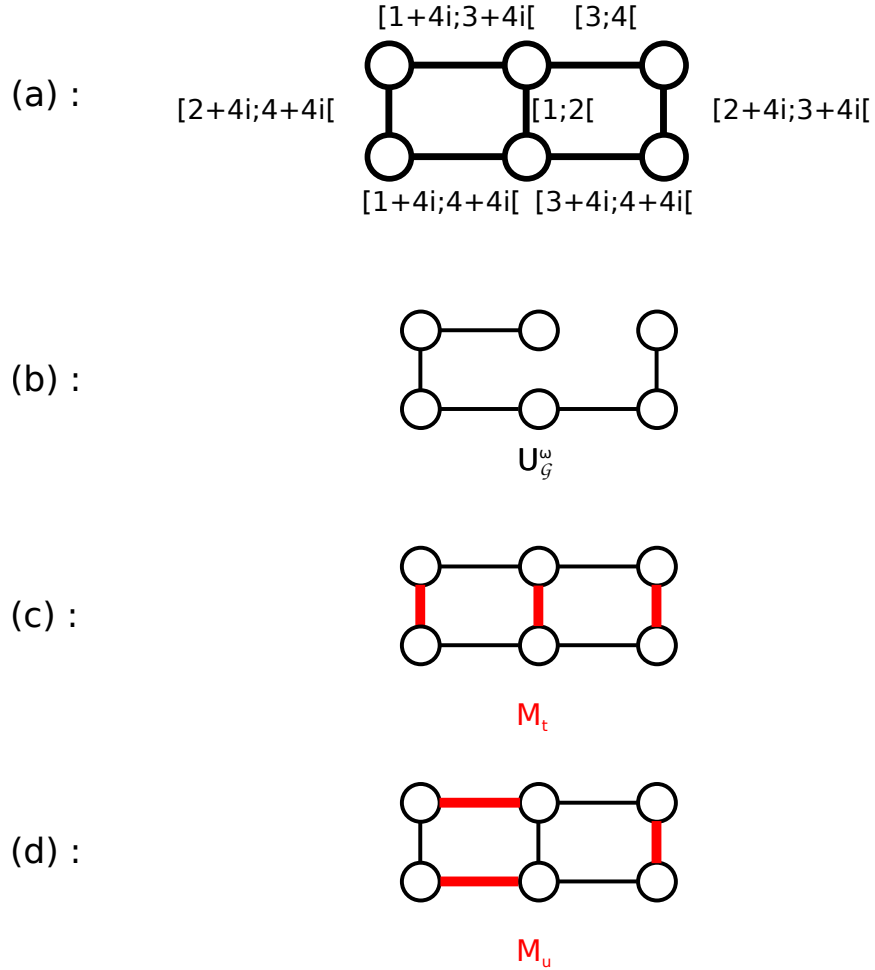


FIGURE 4.14 – Dans cet exemple, la figure 4.13(b) illustre l’empreinte ultime $U_{\mathcal{G}}^{\omega}$ du $\mathcal{GVT}$ $\mathcal{G}$ de la figure 4.13(a). Dans les figures 4.13(c) les arêtes en **gras** représentent les arêtes appartenant à un couplage maximal temporel M_t de $\mathcal{G}$. Tandis que les arêtes en **gras** dans la figures 4.13(d) correspondent les arêtes appartenant à un couplage maximal ultime M_u de $\mathcal{G}$. Notez que si l’empreinte ultime est connexe alors un couplage maximal ultime est toujours un couplage maximal temporel et que la réciproque n’est pas forcément vrai.

4.5 Conclusion

Dans ce chapitre nous avons présenté les approches existantes pour redéfinir les problèmes traditionnels dans le contexte des graphes variant dans le temps.

Afin de contourner les limitations de ces approches, nous avons proposé une nouvelle approche qui définit les problèmes en fonction de l’empreinte ultime du $\mathcal{GVT}$. Nous avons vu que cette approche prend tout son sens lorsqu’il s’agit d’étudier des problèmes sur la famille $\mathcal{CTT}$ de $\mathcal{GVT}$. Dans ce contexte où l’unique hypothèse est que l’empreinte ultime est connexe, les approches étudiées jusqu’à maintenant comporte des faiblesses que notre approche comble. En effet, l’approche ultime assure qu’il existe toujours une solution, que cette solution est stable et qu’elle satisfait infiniment souvent la spécification d’un problème. Nous avons défini les problèmes de l’ensemble dominant et du couplage selon cette nouvelle approche. Nous étudions dans les chapitres suivants la construction de ces ensembles. Nous montrons tout d’abord dans le chapitre 5 qu’il n’est pas toujours possible de construire de tels ensembles avec un algorithme déterministe (bien que les ensembles existent toujours par définition). Le chapitre 7 sera consacré à la présentation d’un algorithme de construction de l’ensemble dominant minimal.

Chapitre 5

Résultats d'impossibilité

Sommaire

5.1 Définitions	80
5.1.1 Sous-famille induite	80
5.1.2 Espace métrique et suite de Cauchy	82
5.1.3 Espace de graphes variant dans le temps	83
5.2 Outil de preuves d'impossibilité	85
5.3 Applications de l'outil	87
5.3.1 Empreinte ultime	87
5.3.2 Ensemble dominant minimal ultime	89
5.3.3 Mariage maximal ultime	94
5.4 Conclusion	97

Dans ce chapitre, nous nous intéressons aux preuves d'impossibilité dans les graphes variant dans le temps. Nous proposons en particulier un outil générique dont l'objectif est de faciliter les preuves formelles des résultats d'impossibilité dans les $\mathcal{GVT}$.

Nous verrons ensuite la pertinence de ce résultat en prouvant qu'aucun algorithme déterministe ne permet de résoudre le problème du calcul de l'empreinte ultime, du couplage ultime et de l'ensemble dominant ultime dans la famille $\mathcal{CTT}$ de $\mathcal{GVT}$.

Dans ce chapitre, nous allons définir une métrique qui permet de calculer une distance entre toute paire de graphes variant dans le temps $\mathcal{G}$ et $\mathcal{G}'$ (partageant la même empreinte globale). Cette distance est définie sur la longueur du plus long préfixe en commun temporel entre $\mathcal{G}$ et $\mathcal{G}'$.

Notre résultat principal consiste à montrer qu'étant donné un algorithme $\mathcal{A}$ conçu pour tout graphe variant dans le temps et une suite de graphes variant dans le temps $(\mathcal{G}_n)_n$ qui converge vers un certain graphe variant dans le temps $\mathcal{G}$, alors la suite d'exécutions de $\mathcal{A}$ sur chaque graphe variant dans le temps de la suite converge aussi. En outre, ils convergent vers l'exécution de l'algorithme $\mathcal{A}$ sur $\mathcal{G}$.

Ce résultat est pratique pour prouver des résultats d'impossibilité de la

manière suivante. Nous raisonnons par contradiction en supposons qu'il existe un algorithme $\mathcal{A}$ qui peut résoudre le problème étudié. Supposons maintenant qu'il est possible de construire une suite de graphes variant dans le temps partageant des préfixes communs qui ne cessent de croître de telle sorte que l'exécution de l'algorithme $\mathcal{A}$ n'arrive pas à converger sur ces préfixes. De ce fait, en appliquant notre résultat principal nous pouvons déduire que cette suite converge vers un graphe variant dans le temps $\mathcal{G}$ de telle sorte que l'exécution de l'algorithme $\mathcal{A}$ sur $\mathcal{G}$ ne converge jamais. En d'autres termes, notre résultat principal permet de construire formellement un contre-exemple en s'appuyant sur une limite de suite de graphes variant dans le temps.

5.1 Définitions

Dans cette section nous allons définir quelques notions dont nous avons besoin afin de présenter, expliquer et prouver notre théorème principal.

Avant de prouver notre résultat principal dans la section 5.2, notez que certains ensembles de graphes variant dans le temps et d'exécutions peuvent être considérés comme des espaces métriques dotés de propriétés topologiques utiles.

5.1.1 Sous-famille induite

Dans la pratique, il nous est globalement impossible de prédire les disponibilités des liens de communications dans un système dynamique. De plus, des hypothèses peuvent restreindre ou empêcher la communication entre certains processus. Néanmoins, l'empreinte globale d'un $\mathcal{GVT}$ est calculable d'où l'intérêt que nous lui portons. De part son accessibilité, l'empreinte globale d'un $\mathcal{GVT}$ octroie la possibilité d'étudier certains aspects d'un $\mathcal{GVT}$.

Dans ce qui suit nous nous intéressons à une famille spécifique de graphe connexe à travers le temps, la famille $\mathcal{CTT}$ (voir la définition 3.9, page 48 pour la définition de la famille $\mathcal{CTT}$). Plus particulièrement, nous nous focalisons sur des sous-familles comprenant tous les $\mathcal{GVT}$ pour lesquels l'empreinte globale appartient à un certain ensemble de graphes.

Définition 5.1 (Sous-famille induite) *Étant donné un ensemble de graphes statiques $\mathcal{F}$ et une famille de graphes variant dans le temps $\mathcal{C}$, la sous-famille de $\mathcal{C}$ induite par $\mathcal{F}$ (notée $\mathcal{C}|_{\mathcal{F}}$) est l'ensemble de tous les $\mathcal{GVT}$ de $\mathcal{C}$ pour lesquels l'empreinte globale appartient à $\mathcal{F}$.*

Les deux prochains résultats découlent directement des définitions 3.9 et 5.1.

Lemme 5.1 *Dans toute sous-famille induite $\mathcal{C}|_{\mathcal{F}}$, si un graphe variant dans le temps $\mathcal{G}$ de $\mathcal{C}$ admet $f \in \mathcal{F}$ comme empreinte globale, alors tout autre graphe variant dans le temps $\mathcal{G}' \neq \mathcal{G}$ de $\mathcal{C}$ pour lequel f est l'empreinte globale appartient à $\mathcal{C}|_{\mathcal{F}}$.*

Preuve. Soit $\mathcal{C}|_{\mathcal{F}}$ une sous-famille induite et soit $\mathcal{G}$ un $\mathcal{GVT}$ de $\mathcal{C}|_{\mathcal{F}}$ qui a pour empreinte $U_{\mathcal{G}} \in \mathcal{F}$. Nous devons prouver que, pour tout $\mathcal{G}' \in \mathcal{C}$ si $U_{\mathcal{G}'} = U_{\mathcal{G}}$ alors $\mathcal{G}' \in \mathcal{C}|_{\mathcal{F}}$.

Nous savons que chaque $\mathcal{G}'$ est un $\mathcal{GVT}$ de la famille $\mathcal{C}$. Puisque $U_{\mathcal{G}'} = U_{\mathcal{G}}$, alors nous savons aussi que $U_{\mathcal{G}'} \in \mathcal{F}$.

Par la définition 5.1, nous savons que la sous-famille induite $\mathcal{C}|_{\mathcal{F}}$ contient l'ensemble des $\mathcal{GVT}$ de la famille $\mathcal{C}$ pour lesquels l'empreinte globale appartient à $\mathcal{F}$.

De ce fait, nous pouvons déduire directement que pour tout $\mathcal{G}' \in \mathcal{C}$ tel que $U_{\mathcal{G}'} = U_{\mathcal{G}}$, $\mathcal{G}'$ appartient à la sous-famille induite $\mathcal{C}|_{\mathcal{F}}$. $\square$

Lemme 5.2 *Il existe un $\mathcal{GVT}$ de $\mathcal{CTT}|_{\mathcal{F}}$ qui admet une arête vouée à disparaître si et seulement si l'ensemble $\mathcal{F}$ ne contient pas que des arbres.*

Preuve. Dans un premier temps nous allons prouver l'implication suivante. S'il existe un $\mathcal{GVT}$ de $\mathcal{CTT}|_{\mathcal{F}}$ qui admet une arête vouée à disparaître alors l'ensemble $\mathcal{F}$ ne contient pas que des arbres.

Par l'absurde, supposons qu'il existe un $\mathcal{GVT}$ $\mathcal{G}$ dans $\mathcal{CTT}|_{\mathcal{F}}$ qui admet une arête vouée à disparaître tel que l'ensemble $\mathcal{F}$ ne contient que des arbres.

Par la définition 5.1 de la sous-famille induite, nous savons que l'empreinte $U_{\mathcal{G}}$ du $\mathcal{GVT}$ $\mathcal{G}$ appartient à l'ensemble $\mathcal{F}$.

Par la définition 3.9 de la famille de graphes connexe à travers le temps, nous savons que l'empreinte ultime $U_{\mathcal{G}}^{\omega}$ est connexe.

Puisque le $\mathcal{GVT}$ $\mathcal{G}$ admet une arête vouée à disparaître alors on en déduit que l'empreinte $U_{\mathcal{G}}$ est différente de l'empreinte ultime $U_{\mathcal{G}}^{\omega}$, à savoir que $U_{\mathcal{G}} \neq U_{\mathcal{G}}^{\omega}$ (voir la définition 3.5 page 37 de l'empreinte ultime pour plus de détails).

D'un autre côté, l'ensemble $\mathcal{F}$ ne contient que des arbres, ce qui implique que $U_{\mathcal{G}}$ est un arbre. Cependant, si $U_{\mathcal{G}}$ est un arbre et que le $\mathcal{GVT}$ $\mathcal{G}$ admet une arête vouée à disparaître alors $U_{\mathcal{G}}^{\omega}$ est non connexe car supprimer une arête d'un arbre donne un graphe non connexe. Cela contredit le fait qu'il puisse exister un $\mathcal{GVT}$ $\mathcal{G}$ dans $\mathcal{CTT}|_{\mathcal{F}}$ qui admet une arête vouée à disparaître tel que l'ensemble $\mathcal{F}$ ne contient que des arbres.

Pour compléter la preuve, il nous reste à prouver l'implication suivante. Si l'ensemble $\mathcal{F}$ ne contient pas que des arbres, alors il existe un $\mathcal{GVT}$ de $\mathcal{CTT}|_{\mathcal{F}}$ qui admet une arête vouée à disparaître.

Soit $\mathcal{F}$ un ensemble de graphes statiques qui contient au moins un graphe qui n'est pas acyclique. Soit f un graphe statique de $\mathcal{F}$ comptant au moins un cycle et e une arête de f qui appartient à l'un de ses cycles. Soit $\mathcal{G}$ un $\mathcal{GVT}$ qui a comme empreinte globale f tel que e est l'unique arête vouée à disparaître dans $\mathcal{G}$.

Si e est l'unique arête vouée à disparaître dans $\mathcal{G}$ alors l'empreinte ultime de $U_{\mathcal{G}}^{\omega}$ de $\mathcal{G}$ est connexe (puisque supprimer une arête d'un cycle ne déconnecte pas le graphe). Donc, par la définition 3.9 le $\mathcal{GVT}$ $\mathcal{G}$ appartient à la famille $\mathcal{CTT}$.

Maintenant que nous savons que $\mathcal{G}$ est un $\mathcal{GVT}$ de $\mathcal{CTT}$ et qu'il a pour empreinte globale f , nous pouvons déduire par le lemme 5.1 que $\mathcal{G}$ est un $\mathcal{GVT}$ de la

famille $\mathcal{CTT}|_{\mathcal{F}}$, ce qui achève la preuve. $\square$

5.1.2 Espace métrique et suite de Cauchy

Nous proposons dans cette section des rappels de topologie qui seront utiles à la preuve de notre outil. Un espace métrique est un ensemble au sein duquel la notion de distance entre les différents éléments de l'ensemble est formellement définie. Nous avons remarqué que les sous-familles de graphes variant dans le temps auxquels nous nous intéressons peuvent être définies comme des espaces métriques. De ce fait, nous verrons comment représenter un ensemble de graphes variant dans le temps d'une sous-famille induite par un espace métrique.

Distance. Une distance $d(x, y)$ est une application qui formalise la longueur que sépare deux éléments x et y appartenant à un ensemble X . Une **distance** sur un ensemble X est une application d définie sur X^2 tel que $\forall x, y \in X^2$, $d(x, y)$ est une valeur réelle satisfaisant les propriétés suivantes :

1. séparation : $d(x, y) = 0 \Leftrightarrow x = y$
2. symétrie : $d(x, y) = d(y, x)$
3. inégalité triangulaire : $\forall (x, y, z) \in X^3, d_X(x, z) \leq (d_X(x, y) + d_X(y, z))$

Espace métrique. (X, d) est un espace métrique, si d est une distance sur X .

Distance ultramétrique. Une application d est une distance **ultramétrique** sur un ensemble X si d est une distance sur X qui satisfait l'inégalité ultra-triangulaire $(\forall (x, y, z) \in X^3 : d_X(x, z) \leq \max(d_X(x, y), d_X(y, z)))$.

Espace ultramétrique Un espace métrique (X, d) est ultramétrique si d est une distance ultramétrique sur X .

Suite de Cauchy. Une suite de Cauchy est une suite d'éléments d'un espace métrique dont les termes se rapprochent à partir d'un certain rang de la suite. En d'autres termes, les éléments de la suite se rapprochent les uns des autres d'une certaine façon qui suggèrent que la suite doit avoir une limite. Cependant, la limite d'une suite de Cauchy n'est pas nécessairement un élément de son espace métrique. La Figure 5.1 illustre un exemple de suite de Cauchy dans $\mathbb{R}$.

Définition 5.2 (Suite de Cauchy) La suite $(x_n)_{n \in \mathbb{N}}$ est une suite de Cauchy si, pour tout réel positif $\epsilon > 0$, il existe un entier naturel k tel que pour tout couple d'entiers $p \geq k, q \geq k$, la distance $d(x_p, x_q)$ est inférieure à ϵ :

$$\forall \epsilon \in \mathbb{R}^{*+}, \exists k \in \mathbb{N}, \forall p, q \in \mathbb{N}^2, p \geq k, q \geq k, d(x_p, x_q) < \epsilon$$

ou plus synthétiquement

$$\lim_{n \rightarrow +\infty} \sup_{p, q \geq n} d(x_p, x_q) = 0$$

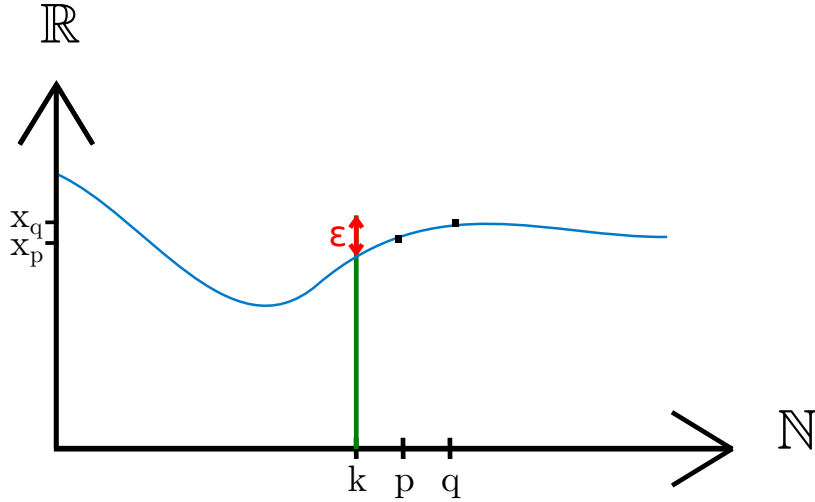


FIGURE 5.1 – L’axe des abscisses représente l’indice i dans $\mathbb{N}$ d’une suite, tandis que l’axe des ordonnées correspond aux valeurs des éléments de la suite qui sont dans $\mathbb{R}$. Dans cet exemple, nous pouvons constater que plus la valeur de l’indice k est grande, plus la distance entre les valeurs de deux éléments dont l’indice est supérieur à k se rapprochent. Ce qui fait de cette suite une suite de Cauchy.

Remarque 5.1 *Dans un espace métrique toute suite convergente est une suite de Cauchy. Cependant la réciproque n’est pas forcément vraie.*

La Remarque 5.1 concerne par exemple une suite de Cauchy dans un *espace vectoriel normé* (un espace vectoriel muni d’une norme) $\mathbb{Q}^n$ non complet qui peut converger vers une limite qui n’est pas dans $\mathbb{Q}^n$ mais dans $\mathbb{R}^n$. Comme exemple de suite de Cauchy convergente, on peut considérer la suite de rationnels des approximations décimales de $\sqrt{2}$, c’est-à-dire, $x_0 = 1, x_2 = 1.4, x_3 = 1.41, x_3 = 1.414\dots$, cette suite est de Cauchy car tous ses termes se rapprochent d’une limite, par contre elle n’est pas convergente. En effet, étant définie sur $\mathbb{Q}$, X_n ne tend vers aucun rationnel.

Définition 5.3 (Espace métrique complet) *Un **espace métrique complet** est un espace métrique dans lequel toute suite de Cauchy converge.*

5.1.3 Espace de graphes variant dans le temps

Étant donné un domaine temporel $\mathbb{T}$, un graphe statique (V, E) et une fonction de latence ζ , supposons que l’ensemble $\Gamma_{(V,E),\mathbb{T},\zeta}$ est l’ensemble de tous les graphes variant dans le temps dont l’empreinte globale est (V, E) et dont la fonction de latence est ζ . Par souci de clarté nous allons omettre la notation $\Gamma_{(V,E),\mathbb{T},\zeta}$ pour la remplacer par Γ .

Notons que deux graphes variant dans le temps distincts de Γ ne peuvent être différenciés que par leurs fonctions de présence.

Nous allons maintenant définir une distance entre les graphes variant dans le temps de cet ensemble de la manière suivante. Pour tout graphe variant dans le

temps $\mathcal{G}$ de Γ , notons sa fonction de présence par $\rho_{\mathcal{G}}$. Nous définissons l'application $d_{\Gamma} : \Gamma \times \Gamma \rightarrow [0, 1]$ par :

$$(\mathcal{G}, \mathcal{G}') \mapsto \begin{cases} 0 & \text{si } \mathcal{G} = \mathcal{G}' \\ 2^{-\lambda} & \text{sinon, avec } \lambda = \sup \{t \in \mathbb{T} \mid \forall t' \leq t, \forall e \in E, \rho_{\mathcal{G}}(e, t') = \rho_{\mathcal{G}'}(e, t')\} \end{cases}$$

Lemme 5.3 *L'application d_{Γ} est une ultramétrie.*

Preuve. Le premier point est satisfait directement par la définition de l'application d_{Γ} . En effet, $d_{\Gamma}(\mathcal{G}, \mathcal{G}') = 0$ si et seulement si $\mathcal{G} = \mathcal{G}'$.

Le deuxième point est aussi satisfait directement par la définition de l'application d_{Γ} . En effet, lorsque $\mathcal{G} = \mathcal{G}'$ nous avons $d_{\Gamma}(\mathcal{G}, \mathcal{G}') = d_{\Gamma}(\mathcal{G}', \mathcal{G}) = 0$ et quand $\mathcal{G} \neq \mathcal{G}'$ nous avons $d_{\Gamma}(\mathcal{G}, \mathcal{G}') = d_{\Gamma}(\mathcal{G}', \mathcal{G})$. En effet, on a alors $\sup \{t \in \mathbb{T} \mid \forall t' \leq t, \forall e \in E, \rho_{\mathcal{G}}(e, t') = \rho_{\mathcal{G}'}(e, t')\} = \sup \{t \in \mathbb{T} \mid \forall t' \leq t, \forall e \in E, \rho_{\mathcal{G}}(e, t') = \rho_{\mathcal{G}'}(e, t')\}$ car il s'agit simplement de l'instant à partir duquel les deux graphes ne partagent plus le même préfixe.

Il ne nous reste plus à prouver que le troisième point. Soient $\mathcal{G}$, $\mathcal{G}'$ et $\mathcal{G}''$ trois graphes variant dans le temps de Γ . Supposons que $d_{\Gamma}(\mathcal{G}, \mathcal{G}') = 2^{-\lambda'}$ et $d_{\Gamma}(\mathcal{G}', \mathcal{G}'') = 2^{-\lambda''}$. Soit $\lambda = \min(\lambda', \lambda'')$. De ce fait, par définition de l'application d_{Γ} , nous avons : $\forall t < \lambda', \forall e \in E, \rho_{\mathcal{G}}(e, t) = \rho_{\mathcal{G}'}(e, t)$ et $\forall t < \lambda'', \forall e \in E, \rho_{\mathcal{G}'}(e, t) = \rho_{\mathcal{G}''}(e, t)$. Nous pouvons en déduire que $\forall t < \lambda, \forall e \in E, \rho_{\mathcal{G}}(e, t) = \rho_{\mathcal{G}''}(e, t)$, c'est-à-dire que $d_{\Gamma}(\mathcal{G}, \mathcal{G}'') \leq 2^{-\lambda}$.

D'autre part, nous avons : $\max(d_{\Gamma}(\mathcal{G}, \mathcal{G}'), d_{\Gamma}(\mathcal{G}', \mathcal{G}'')) = \max(2^{-\lambda'}, 2^{-\lambda''}) = 2^{-\lambda}$. Pour conclure, $d_{\Gamma}(\mathcal{G}, \mathcal{G}'') \leq \max(d_{\Gamma}(\mathcal{G}, \mathcal{G}'), d_{\Gamma}(\mathcal{G}', \mathcal{G}''))$, ce qui marque la fin de la preuve. $\square$

Lemme 5.4 *L'espace métrique (Γ, d_{Γ}) est complet.*

Preuve. Soit $(\mathcal{G}_n)_{n \in \mathbb{N}}$ une suite de Cauchy dans Γ . Il suffit de prouver que la suite $(\mathcal{G}_n)_{n \in \mathbb{N}}$ converge dans Γ .

Par définition d'une suite de Cauchy, nous avons : $\forall \epsilon \in \mathbb{R}^{*+}, \exists k \in \mathbb{N}, \forall i \in \mathbb{N}, d_{\Gamma}(\mathcal{G}_k, \mathcal{G}_{k+i}) < \epsilon$. Nous avons en particulier que $\forall \lambda \in \mathbb{T}, \exists k \in \mathbb{N}, \forall i \in \mathbb{N}, d_{\Gamma}(\mathcal{G}_k, \mathcal{G}_{k+i}) < 2^{-\lambda}$.

D'autre part, par définition de d_{Γ} , nous savons que l'existence d'un $\lambda_{(k,i)} \in \mathbb{T}$ tel que $d_{\Gamma}(\mathcal{G}_k, \mathcal{G}_{k+i}) < 2^{-\lambda_{(k,i)}}$ pour $k \in \mathbb{N}$ et $i \in \mathbb{N}$ implique que $\forall t < \lambda_{(k,i)}, \forall e \in E, \rho_{\mathcal{G}_k}(e, t) = \rho_{\mathcal{G}_{k+i}}(e, t)$. Par conséquent, nous avons : $\forall \lambda \in \mathbb{T}, \exists k \in \mathbb{N}, \forall i \in \mathbb{N}, \forall t < \lambda, \forall e \in E, \rho_{\mathcal{G}_k}(e, t) = \rho_{\mathcal{G}_{k+i}}(e, t)$. Soit $\mathcal{G}_{\omega} \in \Gamma$ le graphe variant dans le temps défini par $\forall \lambda \in \mathbb{T}, \forall e \in E, \rho_{\mathcal{G}_{\omega}}(e, \lambda) = \rho_{\mathcal{G}_k}(e, \lambda)$.

Soit $\epsilon \in \mathbb{R}^{*+}$ et λ le plus petit entier tel que $2^{-\lambda} < \epsilon$. De ce fait, nous avons $\exists k \in \mathbb{N}, \forall i \in \mathbb{N}, \forall t < \lambda, \forall e \in E, \rho_{\mathcal{G}_{k+i}}(e, t) = \rho_{\mathcal{G}_k}(e, t) = \rho_{\mathcal{G}_{\omega}}(e, t)$. Nous pouvons déduire que : $\forall i \in \mathbb{N}, d_{\Gamma}(\mathcal{G}_k, \mathcal{G}_{\omega}) \leq 2^{-\lambda} < \epsilon$. En d'autres termes, $(\mathcal{G}_n)_{n \in \mathbb{N}}$ converge vers $\mathcal{G}_{\omega} \in \Gamma$, ce qui prouve que l'espace (Γ, d_{Γ}) est complet. $\square$

Espace de sortie. Étant donné un algorithme $\mathcal{A}$ et un graphe variant dans le temps $\mathcal{G}$, définissons la $(\mathcal{A}, \mathcal{G})$ -sortie comme la fonction qui associe à tout $t \in \mathbb{T}$ l'état de $\mathcal{G}$ au temps t en exécutant l'algorithme $\mathcal{A}$. Nous dirons que $\mathcal{G}$ est le $\mathcal{GVT}$ support de cette sortie. Soit $\mathcal{O}_{\mathcal{A}, \Gamma}$ l'ensemble de toutes les $(\mathcal{A}, \mathcal{G})$ -sorties pour tous les graphes variant dans le temps $\mathcal{G}$ de Γ . Dans un souci de simplicité, nous omettrons l'indice. $\mathcal{A}, \Gamma$ et nous noterons cet ensemble par $\mathcal{O}$. Remarquez que deux sorties distinctes de $\mathcal{O}$ ne peuvent être différenciées que par leur $\mathcal{GVT}$ support. Pour toute sortie o de $\mathcal{O}$, notons le graphe variant dans le temps support de o par $\mathcal{G}_o$.

Nous allons maintenant définir une distance entre les $(\mathcal{A}, \mathcal{G})$ -sorties.

Nous définissons l'application $d_{\mathcal{O}} : \mathcal{O} \times \mathcal{O} \longrightarrow [0, 1]$ par :

$$(o, o') \mapsto \begin{cases} 0 & \text{si } o = o' \\ 2^{-\lambda} & \text{sinon, avec } \lambda = \sup \{t \in \mathbb{T} \mid \forall t' \leq t, o(t') = o'(t')\} \end{cases}$$

En raison de l'analogie entre les définitions de d_{Γ} et $d_{\mathcal{O}}$, les lemmes suivants se prouvent de la même manière que le lemme 5.3 et le lemme 5.4 :

Lemme 5.5 *L'application $d_{\mathcal{O}}$ est une ultramétrie sur $\mathcal{O}$.*

Lemme 5.6 *L'espace métrique $(\mathcal{O}, d_{\mathcal{O}})$ est complet.*

5.2 Outil de preuves d'impossibilité

Nous sommes en mesure maintenant de présenter le résultat principal du chapitre 5. Intuitivement, ce théorème nous assure que si nous prenons une suite de graphes variant dans le temps partageant des préfixes en commun qui ne cessent de croître (donc convergente selon la distance introduite ci-dessus), alors la suite des sorties d'un algorithme s'exécutant sur cette suite (de $\mathcal{GVT}$) converge aussi.

Par ailleurs, la limite de cette suite est la sortie qui correspond au graphe variant dans le temps qui partage tous les préfixes communs de cette suite de $\mathcal{GVT}$. La Figure 5.2 correspond à une illustration du théorème principale du chapitre.

Ce résultat est très utile car il nous permet de construire, à travers un raisonnement par l'absurde, des contre-exemples sur l'hypothèse de l'existence d'un algorithme résolvant un problème donné. En effet, il suffit de construire une suite de graphes variant dans le temps (avec des préfixes communs toujours croissants) et de prouver que la sortie de l'algorithme sur chacun de ces $\mathcal{GVT}$ viole la spécification du problème en un temps fini. Cela permet d'exhiber une exécution qui transgresse infiniment souvent la spécification du problème (c'est-à-dire, la limite de la suite des sorties de l'algorithme).

Théorème 5.1 *Pour tout algorithme déterministe $\mathcal{A}$, si une suite $(\mathcal{G}_n)_{n \in \mathbb{N}}$ de Γ converge vers un certain $\mathcal{G}_{\omega} \in \Gamma$, alors la suite $(o_n)_{n \in \mathbb{N}}$ des $(\mathcal{A}, \mathcal{G}_n)$ -sorties converge vers $o_{\omega} \in \mathcal{O}$. De plus, o_{ω} est la $(\mathcal{A}, \mathcal{G}_{\omega})$ -sortie.*

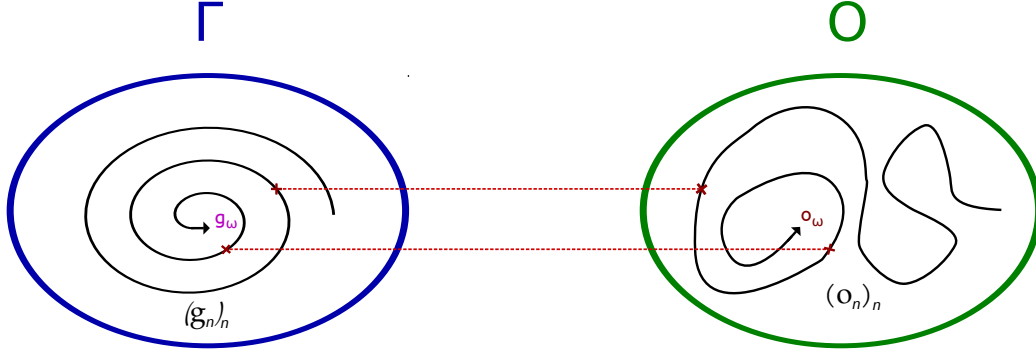


FIGURE 5.2 – Cette illustration correspond à l'idée du Théorème 5.1. De manière générale, cette illustration exprime que lorsqu'il existe une suite dans Γ qui converge vers un certain $\mathcal{G}_\omega$ alors une suite de sortie finit par converger aussi vers un certain o_ω .

Preuve. Soit $\mathcal{A}$ un algorithme déterministe et $(\mathcal{G}_n)_{n \in \mathbb{N}}$ une suite de Γ qui converge vers un certain $\mathcal{G}_\omega \in \Gamma$. Soit $(o_n)_{n \in \mathbb{N}}$ la suite des $(\mathcal{A}, \mathcal{G}_n)$ -sorties.

Dans un premier temps, nous allons prouver que $(o_n)_{n \in \mathbb{N}}$ converge dans $\mathcal{O}$. Étant donné que $\mathcal{O}$ est un espace complet (voir le lemme 5.6), il est suffisant de prouver que $(o_n)_{n \in \mathbb{N}}$ est une suite de Cauchy.

Soit $\epsilon \in \mathbb{R}^{*+}$. Puisque Γ est un espace métrique complet (voir le lemme 5.4), nous savons que $(\mathcal{G}_n)_{n \in \mathbb{N}}$ est une suite de Cauchy. Donc, nous avons par définition : $\exists k_\epsilon \in \mathbb{N}, \forall i \in \mathbb{N}, d_\Gamma(\mathcal{G}_{k_\epsilon}, \mathcal{G}_{k_\epsilon+i}) < \epsilon$.

D'un autre coté, par définition de la distance d_Γ , nous savons que l'existence de $\lambda_{(k,i)} \in \mathbb{T}$ tel que $d_\Gamma(\mathcal{G}_k, \mathcal{G}_{k+i}) = 2^{-\lambda_{(k,i)}}$ pour $k \in \mathbb{N}$ et $i \in \mathbb{N}$ implique que $\forall t < \lambda_{(k,i)}, \forall e \in E, \rho_{\mathcal{G}_k}(e, t) = \rho_{\mathcal{G}_{k+i}}(e, t)$. Étant donné que $\mathcal{A}$ est un algorithme déterministe, nous pouvons déduire que $\forall t < \lambda_{(k,i)}, o_k(t) = o_{k+i}(t)$ (puisque $\mathcal{G}_{o_n} = \mathcal{G}_n$ pour tout $n \in \mathbb{N}$ par construction de $(o_n)_{n \in \mathbb{N}}$). Ensuite, la définition de la distance $d_\mathcal{O}$ implique que $d_\mathcal{O}(o_k, o_{k+i}) \leq 2^{-\lambda_{(k,i)}}$. En d'autres termes, nous pouvons déduire que $\forall k \in \mathbb{N}, \forall i \in \mathbb{N}, d_\mathcal{O}(o_k, o_{k+i}) \leq d_\Gamma(\mathcal{G}_k, \mathcal{G}_{k+i})$.

Par conséquent, $\exists k_\epsilon \in \mathbb{N}, \forall i \in \mathbb{N}, d_\mathcal{O}(o_{k_\epsilon}, o_{k_\epsilon+i}) < \epsilon$. En conclusion, $(o_n)_{n \in \mathbb{N}}$ est une suite de Cauchy. Notez $o \in \mathcal{O}$ la limite de cette suite.

Soit o_ω la $(\mathcal{A}, \mathcal{G}_\omega)$ -sortie. Nous allons prouver que $o = o_\omega$. Étant donné que $(\mathcal{O}, d_\mathcal{O})$ est ultramétrique (voir le lemme 5.5), nous savons que $0 \leq d_\mathcal{O}(o, o_\omega) \leq \max(d_\mathcal{O}(o, o_n), d_\mathcal{O}(o_n, o_\omega))$ pour tout $n \in \mathbb{N}$. Le fait que la suite (o_n) converge vers 0 implique que la suite $(d_\mathcal{O}(o, o_n))_{n \in \mathbb{N}}$ converge vers 0. L'algorithme $\mathcal{A}$ étant déterministe et Γ et $\mathcal{O}$ étant des espaces complets, nous pouvons prouver en suivant un raisonnement similaire au précédant que $d_\mathcal{O}(o_n, o_\omega) \leq d_\Gamma(\mathcal{G}_n, \mathcal{G}_\omega)$ pour tout $n \in \mathbb{N}$. La convergence de la suite $(\mathcal{G}_n)_{n \in \mathbb{N}}$ vers $\mathcal{G}_\omega$ implique que la suite $(d_\Gamma(\mathcal{G}_n, \mathcal{G}_\omega))_{n \in \mathbb{N}}$ converge vers 0. On en déduit que la suite $(d_\mathcal{O}(o_n, o_\omega))_{n \in \mathbb{N}}$ converge aussi vers 0 (puisque $d_\mathcal{O}(o_n, o_\omega) \geq 0$ pour tout $n \in \mathbb{N}$). Ainsi, la suite $(\max(d_\mathcal{O}(o, o_n), d_\mathcal{O}(o_n, o_\omega)))_{n \in \mathbb{N}}$ converge vers 0 ce qui signifie que $d_\mathcal{O}(o, o_\omega) = 0$. Étant donné que $d_\mathcal{O}$ est une distance, nous pouvons conclure que $o = o_\omega$, ce qui marque la fin de la preuve. $\square$

5.3 Applications de l'outil

Dans cette section, nous nous intéressons à l'étude des conditions nécessaires à la faisabilité de certains problèmes distribués dans les familles de $\mathcal{GVT}$ à durée de vie non bornée. Plus particulièrement, nous portons notre intérêt à la famille $\mathcal{CTT}$ (voir la Propriété 3.9 pour la définition de la famille $\mathcal{CTT}$). En d'autres termes, nous nous sommes intéressés à identifier des conditions nécessaires sur la topologie du $\mathcal{GVT}$ pour résoudre des *problèmes de calcul de points fixes* (voir ici 6.2 page 102 pour la définition d'un problème de calcul de points fixes) dans les $\mathcal{GVT}$ de $\mathcal{CTT}$. Cet intérêt particulier pour cette famille vient du fait que dans la hiérarchie des familles de $\mathcal{GVT}$ de durée de vie non bornée (voir la Figure 3.25), la famille $\mathcal{CTT}$ comprend toutes les autres.

Nous allons étudier les trois problèmes suivants : la construction de l'empreinte ultime, la construction du couplage maximal et la construction de l'ensemble dominant minimal.

Nous référons le lecteur au chapitre 4 pour une discussion sur la spécification de ces problèmes dans un cadre dynamique.

5.3.1 Empreinte ultime

Cette section est consacrée à la construction de l'empreinte ultime dans les graphes variant dans le temps.

Nous adoptons la spécification du problème présentée dans le chapitre 4. Nous prouvons alors qu'il n'existe pas d'algorithme déterministe pour résoudre ce problème sur un $\mathcal{GVT}$ de $\mathcal{CTT}$ si le $\mathcal{GVT}$ admet une arête vouée à disparaître.

Nous verrons, dans cette section, qu'il est impossible de résoudre de manière déterministe le problème de construction de l'empreinte ultime dans les $\mathcal{GVT}$ de $\mathcal{CTT}$, hormis dans le cas trivial où l'empreinte globale est un arbre. En effet, si l'empreinte globale est un arbre alors l'empreinte globale est aussi l'empreinte ultime puisque que par définition de $\mathcal{CTT}$ l'empreinte ultime est connexe (or, la suppression d'une des arêtes d'un arbre déconnecte le graphe). Ce qui implique qu'il n'existe pas d'arête vouée à disparaître dans un $\mathcal{GVT}$ de $\mathcal{CTT}$ qui a un arbre comme empreinte globale.

Théorème 5.2 *Pour tout ensemble de graphes (statiques) $\mathcal{F}$ qui ne contient pas que des arbres, il n'existe aucun algorithme déterministe qui satisfait la spécification de la construction de l'empreinte ultime pour la sous-famille induite $\mathcal{CTT}|_{\mathcal{F}}$.*

Preuve. Pour tout graphe variant dans le temps $\mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta, \phi)$, nous définissons le graphe variant dans le temps $\mathcal{G} \oplus \{(e_1, \mathcal{T}_{e_1}), \dots, (e_k, \mathcal{T}_{e_k})\}$ (avec, pour tout $i \in \{0, \dots, k\}$, $e_i \in E$ et $\mathcal{T}_{e_i} \subseteq \mathcal{T}$) comme le graphe variant dans le temps $(V, E, \mathcal{T}, \rho', \zeta, \phi)$ tel que :

$$\rho'(e, t) = \begin{cases} 1 & \text{si } \exists i \in \{0, \dots, k\}, e = e_i \text{ et } t \in \mathcal{T}_{e_i} \\ \rho(e, t) & \text{autrement} \end{cases}$$

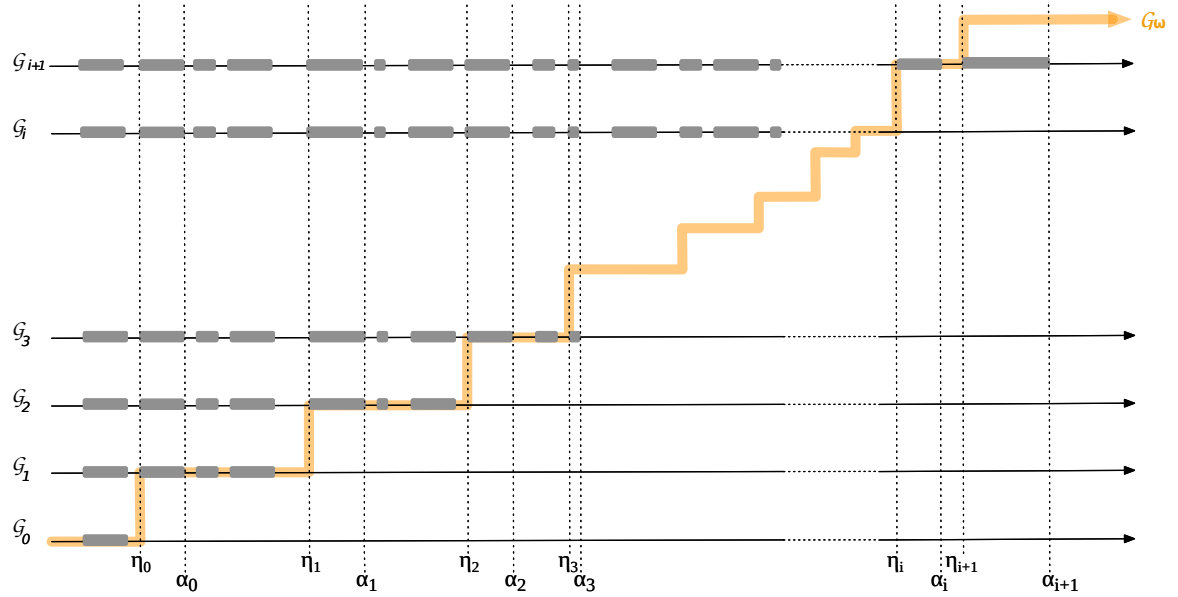


FIGURE 5.3 – Construction de la suite $(\mathcal{G}_n)_{n \in \mathbb{N}}$ dans la preuve du théorème 5.2. Les lignes en gris représentent les moments durant lesquels l'arête e appartient au graphe délivré en sortie par tous les processus de V .

Cette preuve sera construite par l'absurde. Nous supposons donc qu'il existe un algorithme déterministe $\mathcal{A}$ qui satisfait la spécification du calcul de l'empreinte ultime dans la sous-famille induite $\mathcal{CTT}|_{\mathcal{F}}$ tel qu'il existe un ensemble de graphes (statiques) $\mathcal{F}$ qui ne contient pas que des arbres. De ce fait, tout processus exécutant l'algorithme $\mathcal{A}$ délivre à tout moment en sortie un graphe (statique).

Par le lemme 5.2, nous savons qu'il existe $\mathcal{G} \in \mathcal{CTT}|_{\mathcal{F}}$ tel que $\mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta, \phi)$ admet au moins une arête vouée à disparaître e . Nous construisons alors une suite $(\mathcal{G}_n)_{n \in \mathbb{N}}$ de graphes variant dans le temps de la manière suivante. Nous fixons $\mathcal{G}_0 = \mathcal{G}$ et nous définissons $\mathcal{G}_i$ par induction pour tout $i \in \mathbb{N}$ de la façon suivante :

1. Considérons l'exécution de l'algorithme $\mathcal{A}$ sur $\mathcal{G}_i$ et soit $\eta_i \in \mathcal{T} \cup \{+\infty\}$ le plus grand temps durant lequel l'arête e appartient au graphe délivré en sortie par certains processus de V (remarquez que $\eta_i = +\infty$ si et seulement si e appartient infiniment souvent au graphe délivré en sortie par au moins un processus) ;
2. Soit $\mathcal{G}'_i = \mathcal{G}_i \oplus (e, \mathcal{T} \cap]\eta_i, +\infty[)$;
3. Considérons l'exécution de l'algorithme $\mathcal{A}$ sur $\mathcal{G}'_i$ et soit $\alpha_i \in \mathcal{T} \cup \{+\infty\}$ le plus petit temps strictement supérieur à η_i où e appartient au graphe délivré en sortie par tous les processus de V (remarquez que $\alpha_i = +\infty$ si et seulement si l'arête e n'a jamais simultanément appartenu au graphe délivré en sortie par tous les processus $\eta_i = +\infty$) ;
4. Soit $\mathcal{G}_{i+1} = \mathcal{G}_i \oplus (e, \mathcal{T} \cap]\eta_i, \alpha_i[)$.

Nous pouvons prouver que, pour tout $i \in \mathbb{N}$, si $\mathcal{G}_i$ appartient à $\mathcal{CTT}|_{\mathcal{F}}$ et si e est une arête vouée à disparaître dans $\mathcal{G}_i$ alors $\eta_i \neq +\infty$ et $\alpha_i \neq +\infty$. En effet,

supposons que e est une arête vouée à disparaître dans $\mathcal{G}_i \in \mathcal{CTT}|_{\mathcal{F}}$ pour un certain $i \in \mathbb{N}$. Par définition, e n'appartient pas à $U_{\mathcal{G}_i}^\omega$. Étant donné que l'algorithme $\mathcal{A}$ satisfait la spécification du calcul de l'empreinte ultime pour $\mathcal{CTT}|_{\mathcal{F}}$, nous savons donc que e ne peut appartenir infiniment souvent au graphe délivré en sortie par un processus durant l'exécution de l'algorithme $\mathcal{A}$ sur $\mathcal{G}_i$, c'est-à-dire que $\eta_i \neq +\infty$. Ensuite, comme l'arête e n'est pas une arête vouée à disparaître dans $\mathcal{G}'_i$ par construction, l'arête e appartient à $U_{\mathcal{G}'_i}^\omega$. Par le lemme 5.1, $\mathcal{G}'_i$ appartient à $\mathcal{CTT}|_{\mathcal{F}}$ puisque $\mathcal{G}_i$ et $\mathcal{G}'_i$ partagent la même empreinte globale $U_{\mathcal{G}}$. Comme l'algorithme $\mathcal{A}$ satisfait la spécification du calcul de l'empreinte ultime pour $\mathcal{CTT}|_{\mathcal{F}}$, nous déduisons donc que l'arête e appartient ultimement au graphe délivré en sortie par tous les processus de V , c'est-à-dire $\alpha_i \neq +\infty$.

Nous obtenons que, pour tout $i \in \mathbb{N}$, si $\mathcal{G}_i$ appartient à $\mathcal{CTT}|_{\mathcal{F}}$ et si e est une arête vouée à disparaître dans $\mathcal{G}_i$, alors $\mathcal{G}_{i+1}$ appartient à $\mathcal{CTT}|_{\mathcal{F}}$ et e est une arête vouée à disparaître dans $\mathcal{G}_{i+1}$. En effet, $\mathcal{G}_{i+1}$ appartient à $\mathcal{CTT}|_{\mathcal{F}}$ d'après le lemme 5.1 (puisque $\mathcal{G}_i$ et $\mathcal{G}_{i+1}$ partagent le même empreinte globale $U_{\mathcal{G}}$). Comme nous avons prouvé que $\eta_i \neq +\infty$ et $\alpha_i \neq +\infty$ lorsque e est une arête vouée à disparaître dans $\mathcal{G}_i$, $\mathcal{G}_{i+1}$ est obtenu par l'ajout de cette arête e dans $\mathcal{G}_i$ pendant une durée finie qui implique que e est une arête vouée à disparaître dans $\mathcal{G}_{i+1}$.

Par l'hypothèse, $\mathcal{G}$ appartient à $\mathcal{CTT}|_{\mathcal{F}}$ et par construction de $\mathcal{G}$, l'arête e est une arête vouée à disparaître dans $\mathcal{G}_0 = \mathcal{G}$. Ce qui implique que $(\mathcal{G}_n)_{n \in \mathbb{N}}$ est une suite de $\mathcal{CTT}|_{\mathcal{F}}$ telle que $\eta_i \neq +\infty$ et $\alpha_i \neq +\infty$ pour tout $i \in \mathbb{N}$. De plus, notez que, pour tout $i \in \mathbb{N}$, $\eta_i < \alpha_i$ (par construction) et $\alpha_i < \eta_{i+1}$ (puisque e appartient au graphe délivré en sortie par tout processus au temps α_i dans $\mathcal{G}_{i+1}$ tandis que l'arête e n'appartient pas au graphe délivré en sortie par tout processus au temps η_{i+1} dans $\mathcal{G}_{i+1}$).

Cela nous permet de définir le graphe variant dans le temps suivant : $\mathcal{G}_\omega = g \oplus \{(e, \mathcal{T} \cap]\eta_i, \alpha_i]) \mid i \in \mathbb{N}\}$. Notez que $U_{\mathcal{G}_\omega} = U_{\mathcal{G}}$ et que $\mathcal{G}_\omega$ appartient à $\mathcal{CTT}|_{\mathcal{F}}$ d'après le lemme 5.1. Enfin, pour tout $k \in \mathbb{N}^*$, nous avons $d_\Gamma(\mathcal{G}_k, \mathcal{G}_\omega) = 2^{-\eta_k}$ par construction de $(\mathcal{G}_n)_{n \in \mathbb{N}}$ et $\mathcal{G}_\omega$. Par conséquent, $(\mathcal{G}_n)_{n \in \mathbb{N}}$ converge dans $\mathcal{CTT}|_{\mathcal{F}}$ et sa limite est $\mathcal{G}_\omega$. Les figures 5.3 et 5.4 illustrent la construction de la suite $(\mathcal{G}_n)_{n \in \mathbb{N}}$ du théorème 5.2.

Par le théorème 5.1 page 85, la $(\mathcal{A}, \mathcal{G}_\omega)$ -sortie est la limite de la suite des $(\mathcal{A}, \mathcal{G}_n)$ -sorties. En d'autres termes, la $(\mathcal{A}, \mathcal{G}_\omega)$ -sortie partage un préfixe de longueur η_i avec la $(\mathcal{A}, \mathcal{G}_i)$ -sortie pour tout $i \in \mathbb{N}$. Cela signifie qu'il existe une infinité de configurations durant l'exécution de l'algorithme $\mathcal{A}$ sur $\mathcal{G}_\omega$ où e appartient au graphe calculé par tous les processus et qu'il existe une infinité de configurations durant l'exécution de l'algorithme $\mathcal{A}$ sur $\mathcal{G}_\omega$ où e n'appartient pas au graphe calculé par tous les processus, ce qui contredit le fait que l'algorithme $\mathcal{A}$ puisse satisfaire les spécifications du calcul de l'empreinte ultime pour $\mathcal{CTT}|_{\mathcal{F}}$ et marque ainsi la fin de la preuve. $\square$

5.3.2 Ensemble dominant minimal ultime

Dans cette section, nous allons aborder le problème de construction de l'ensemble dominant minimal ultime de manière déterministe. Nous employons la

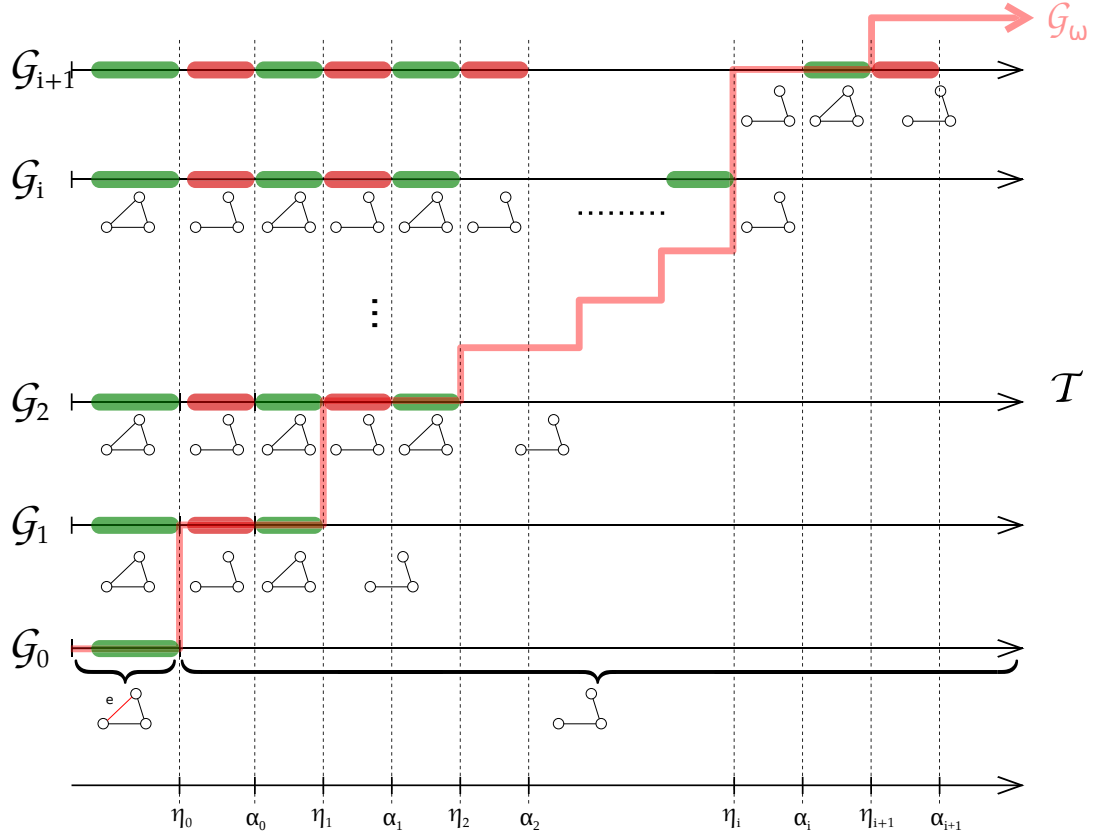


FIGURE 5.4 – Une illustration de la construction d'une suite de GVT dans la preuve du théorème 5.2 par l'exemple. Les lignes en vert représentent les moments durant lesquels l'arête e appartient au graphe délivré en sortie par tous les processus de V et les lignes en rouge représentent les moments durant lesquels l'arête e n'appartient pas au graphe délivré en sortie par tous les processus de V (délivré par quelques uns ou aucun). Le graphe sans l'arête e signifie que l'arête n'est pas présente dans la sortie de tous les processus quand sa présence indique qu'elle est délivrée en sortie par tous les processus. Notez que par construction de cette suite, les préfixes communs ne cessent de croître.

spécification du problème de l'ensemble dominant minimal ultime présentée dans le chapitre 4. Nous montrons donc qu'il n'existe pas d'algorithme déterministe pour résoudre ce problème dans tout $\mathcal{GVT}$ de $\mathcal{CTT}$.

Plus précisément, nous prouvons que tout algorithme déterministe ne peut satisfaire la spécification de l'ensemble dominant minimal ultime que s'il existe un ensemble dominant minimal pour tout graphe partiel connexe de l'empreinte du $\mathcal{GVT}$. Nous appelons cette ensemble dominant minimal particulier un *ensemble dominant minimal robuste*.

Définition 5.4 (Ensemble dominant minimal robuste) *Un ensemble dominant minimal robuste (EDMR) d'un graphe (statique) $g = (V, E)$ est un ensemble de processus $M \subseteq V$ tel que M est un ensemble dominant minimal de tout graphe partiel connexe de g .*

Le lemme suivant découle directement des définitions et justifie notre intérêt pour l'ensemble dominant minimal robuste.

Lemme 5.7 *Si l'empreinte globale d'un $\mathcal{GVT}$ $\mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta, \phi)$ admet un ensemble dominant minimal robuste M alors M est un ensemble dominant minimal ultime de $\mathcal{G}$.*

Preuve. Soit $G = (V, E, \mathcal{T}, \rho, \zeta)$ un $\mathcal{GVT}$ de $\mathcal{CTT}$ de telle sorte qu'il existe un EDMR M sur l'empreinte globale $U_{\mathcal{G}}$ de $\mathcal{G}$. Par définition, un EDMR sur $U_{\mathcal{G}}$ est un ensemble dominant minimal de tout graphe partiel connexe de $U_{\mathcal{G}}$. Selon la définition 4.9, un ensemble dominant minimal ultime de $\mathcal{G}$ est un ensemble dominant minimal de l'empreinte ultime de $\mathcal{G}$. Par définition l'empreinte ultime d'un $\mathcal{GVT}$ de $\mathcal{CTT}$ est un graphe partiel connexe de l'empreinte globale. Par conséquent, M est un ensemble dominant minimal ultime de G . $\square$

Le résultat décrit dans le lemme suivant nous fournit une caractérisation des graphes (statiques) qui admettent un EDMR. Nous allons utiliser cette caractérisation pour prouver notre prochain théorème.

Lemme 5.8 *Pour tout graphe $g = (V, E)$ et tout ensemble dominant minimal M de g , M est un ensemble dominant minimal robuste si et seulement si l'ensemble des arêtes $\{\{p, q\} | q \in M \cap \mathcal{N}_p\}$ est une coupe de g pour tout processus $p \in V \setminus M$.*

Preuve. Nous allons prouver d'abord que pour tout EDMR M d'un graphe $g = (V, E)$, l'ensemble des arêtes $\{\{p, q\} | q \in M \cap \mathcal{N}_p\}$ est une coupe de g pour tout processus $p \in V \setminus M$. Par l'absurde, supposons qu'il existe un EDMR M d'un graphe $g = (V, E)$ tel que l'ensemble des arêtes $\{\{p, q\} | q \in M \cap \mathcal{N}_p\}$ n'est pas une coupe de g pour tout processus $p \in V \setminus M$. Soit $s_g = (V, E')$ le sous graphe de g défini par $E' = E \setminus \{\{p, q\} | q \in M \cap \mathcal{N}_p\}$. Par l'hypothèse, s_g est un graphe connexe. De plus, dans s_g le processus p n'a aucun voisin dans M , cela signifie que M n'est pas un ensemble dominant minimal de g . Ce qui contredit le fait que M est un EDMR de g et prouve la nécessité de la condition.

Ensuite, nous allons prouver que, tout ensemble dominant minimal M d'un graphe g tel que l'ensemble des arêtes $\{\{p, q\} | q \in M \cap \mathcal{N}_p\}$ est une coupe dans g pour tout processus $p \in V \setminus M$ est un *EMDR* de g . Par l'absurde, supposons qu'il existe un ensemble dominant minimal M d'un graphe $g = (V, E)$ tel que l'ensemble des arêtes $\{\{p, q\} | q \in M \cap \mathcal{N}_p\}$ est une coupe dans g pour tout processus $p \in V \setminus M$ de telle sorte que M n'est pas un *EMDR* de g . Par définition d'un *EMDR*, il existe alors un graphe partiel connexe $s_g = (V, E')$ de g tel que M n'est pas un ensemble dominant minimal de s_g . Étudions maintenant les deux cas suivants.

1. M n'est pas un ensemble dominant minimal de s_g . Donc, il existe un processus p tel qu'il n'existe aucune voisin de p dans s_g qui appartient à M . Comme, s_g est connexe nous avons donc que l'ensemble $\{\{p, q\} | q \in M \cap \mathcal{N}_p\}$ n'est pas une coupe de g , ce qui est en contradiction avec l'hypothèse initiale sur M .
2. M est un ensemble dominant de s_g qui n'est pas minimal. Nous disons qu'un processus d'un ensemble dominant domine proprement l'un de ses voisins lorsqu'il est le seule processus dans son voisinage qui le domine. Par conséquent, nous savons qu'il existe, dans s_g , deux voisins $p \in M$ et $q \in M$ tel que p ne domine aucun de ses voisins proprement. Puisque M est un ensemble dominant minimal de g , nous déduisons que p domine proprement au moins un de ses voisins $r \in V \setminus M$ (notez que p et q sont des voisins dans g par construction). Cela signifie que l'ensemble des arêtes $\{\{r, s\} | s \in M \cap \mathcal{N}_r\} = \{\{p, r\}\}$ n'est pas une coupe de g (puisque cette arête n'appartient pas à s_g qui est connexe). Ce qui est en contradiction avec l'hypothèse initiale sur M .

Ces contradictions nous montrent la suffisance de la condition et marque la fin de la preuve. $\square$

L'intuition derrière ce résultat d'impossibilité vient des faits suivants. Nous savons qu'aucun processus n'est capable de détecter ou d'affirmer si une arête est vouée à disparaître. De ce fait, l'ensemble dominant minimal construit par tout algorithme déterministe doit être aussi un ensemble dominant minimal pour toute empreinte ultime. En d'autres termes, l'ensemble construit doit être un ensemble dominant minimal sur tout graphe partiel connexe de l'empreinte pour parer à toute évolution de l'empreinte ultime. La principale difficulté de la preuve de ce résultat réside dans la construction de la suite de graphes variant dans le temps adéquate pour l'application du théorème 5.1.

Théorème 5.3 *Pour tout ensemble de graphes statiques $\mathcal{F}$ qui contient au moins un graphe qui n'admet pas d'ensemble dominant minimal robuste, il n'existe pas d'algorithme déterministe qui satisfait la spécification de l'ensemble dominant minimal ultime pour la sous-famille induite $\mathcal{CTT}|_{\mathcal{F}}$.*

Preuve. Nous commençons par introduire quelques notations. Nous définissons, pour tout graphe variant dans le temps $\mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta, \phi)$, le graphe variant dans le temps $\mathcal{G} \odot \{(E_i, \mathcal{T}_i) | i \in I\}$ (avec $I \subseteq \mathbb{N}$ et pour tout $i \in I$, $E_i \subseteq E$ et $\mathcal{T}_i \subseteq \mathcal{T}$)

comme le graphe variant dans le temps $(V, E, \mathcal{T}, \rho', \zeta, \phi)$ tel que :

$$\rho'(e, t) = \begin{cases} 0 & \text{si } \exists i \in I, e \in E_i \text{ et } t \in \mathcal{T}_i \\ 1 & \text{si } \exists i \in I, e \in E \setminus E_i \text{ et } t \in \mathcal{T}_i \\ \rho(e, t) & \text{autrement} \end{cases}$$

Par l'absurde, supposons qu'il existe un algorithme déterministe $\mathcal{A}$ qui satisfait la spécification de l'ensemble dominant minimal ultime dans tout $\mathcal{GVT}$ de la sous-famille $\mathcal{CTT}|_{\mathcal{F}}$ de telle sorte à ce que l'ensemble de graphes statiques $\mathcal{F}$ comprenant au moins un graphe statique qui n'admet pas un ensemble dominant robuste minimal. De ce fait, tout processus exécutant l'algorithme $\mathcal{A}$ délivre une valeur booléenne en sortie à tout instant.

Soit $\mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta, \phi)$ un graphe connexe à travers le temps de la sous-famille induite $\mathcal{CTT}|_{\mathcal{F}}$ tel que l'empreinte globale $U_{\mathcal{G}}$ de $\mathcal{G}$ n'admet pas un ensemble dominant robuste minimal et qu'il existe un temps $t_0 \in \mathcal{T}$ tel que $\forall e \in E, \forall t \leq t_0, \rho(e, t) = 1$ ($\mathcal{G}$ existe par construction de $\mathcal{F}$ et par définition de $\mathcal{CTT}|_{\mathcal{F}}$). Nous construisons une suite $(\mathcal{G}_n)_{n \in \mathbb{N}}$ de graphes variant dans le temps comme suit. Nous fixons $\mathcal{G}_0 = g$. Supposons que nous avons déjà $\mathcal{G}_i = (V, E, \mathcal{T}, \rho', \zeta, \phi)$ pour un certain $i \in \mathbb{N}$ tel que $\mathcal{G}_i \in \mathcal{CTT}|_{\mathcal{F}}$, $U_{\mathcal{G}_i} = U_{\mathcal{G}}$, et que $\exists \alpha_{i-1} > t_0, \forall e \in E, \forall t > \alpha_i, \rho'(e, t) = \rho(e, t)$. Ensuite, nous définissons par induction $\mathcal{G}_{i+1}$ de la façon suivante :

1. Considérons l'exécution de l'algorithme $\mathcal{A}$ sur $\mathcal{G}_i$ et soit $\eta_i \in \mathcal{T}$ la plus petite date strictement supérieure à α_{i-1} durant lequel l'ensemble des processus fournissant en sortie "vrai" est constant (autrement dit, que l'ensemble dominant calculé par $\mathcal{A}$ est stable). Notez que η_i existe par l'hypothèse sur $\mathcal{A}$ puisque $\mathcal{G}_i \in \mathcal{CTT}|_{\mathcal{F}}$;
2. Soit M_i l'ensemble dominant minimal calculé par l'algorithme $\mathcal{A}$ sur $\mathcal{G}_i$ (représenté par l'ensemble des processus de $\mathcal{G}_i$ qui fournissent en sortie la valeur "vrai" après la date η_i). Étant donné que $U_{\mathcal{G}_i} = U_{\mathcal{G}}$, nous savons par l'hypothèse sur $U_{\mathcal{G}}$ que $U_{\mathcal{G}_i}$ n'admet pas d'ensemble dominant minimal robuste. Plus précisément, nous savons que M_i n'est pas un ensemble dominant minimal robuste de $U_{\mathcal{G}_i}$. De ce fait, il existe un processus p_i de $V \setminus M_i$ tel que l'ensemble des arêtes $E_i = \{\{p_i, q\} | q \in M_i \cap \mathcal{N}_{p_i}\}$ n'est pas une coupe de $U_{\mathcal{G}_i}$;
3. Soit $\mathcal{G}'_i = \mathcal{G}_i \odot \{(E_i, \mathcal{T} \cap]\eta_i, +\infty[)\}$.
4. Remarquez que $U_{\mathcal{G}'_i} = U_{\mathcal{G}_i} = U_{\mathcal{G}}$ (par construction de $\mathcal{G}'_i$ puisque $\eta_i > t_0$) et que $U_{\mathcal{G}'_i}^\omega$ est connexe (puisque $E_{U_{\mathcal{G}'_i}^\omega} = E_{U_{\mathcal{G}}} \setminus E_i$ par construction et E_i n'est pas une coupe de $U_{\mathcal{G}}$). Par conséquent, $\mathcal{G}'_i \in \mathcal{CTT}|_{\mathcal{F}}$ et nous pouvons considérer l'exécution de l'algorithme $\mathcal{A}$ sur $\mathcal{G}'_i$. Soit $\alpha_i \in \mathcal{T}$ la plus petite date strictement supérieure à η_i à partir de laquelle l'ensemble des processus qui délivrent en sortie "vrai" est constant. Soit M'_i l'ensemble dominant minimal calculé par l'algorithme $\mathcal{A}$ sur $\mathcal{G}'_i$ (à savoir, l'ensemble des processus de $\mathcal{G}'_i$ fournissant en sortie la valeur "vrai" après la date α_i). Notez que $M'_i \neq M_i$ puisque M_i n'est pas un ensemble dominant minimal de $U_{\mathcal{G}'_i}^\omega$ (rappelons que, dans $U_{\mathcal{G}'_i}^\omega$, p_i n'a aucun voisin dans M_i);

5. Soit $\mathcal{G}_{i+1} = \mathcal{G}_i \odot \{(E_i, \mathcal{T} \cap \eta_i, \alpha_i)\}$.

Cette construction assure que s'il existe $\mathcal{G}_i = (V, E, \mathcal{T}, \rho', \zeta, \phi)$ pour un i donné ($i \in \mathbb{N}$) tel que $\mathcal{G}_i \in \mathcal{CTT}|_{\mathcal{F}}$, que $U_{\mathcal{G}_i} = U_{\mathcal{G}}$ et que $\exists \alpha_{i-1} > t_0, \forall e \in E, \forall t > \alpha_i, \rho'(e, t) = \rho(e, t)$, alors $\mathcal{G}_{i+1}$ satisfait la même propriété. D'ailleurs, puisque $\mathcal{G}_0 = g$, cette propriété est naturellement satisfaite pour $i = 0$ pour tout $\alpha_{-1} > t_0$. Par conséquent, la suite $(\mathcal{G}_n)_{n \in \mathbb{N}}$ est bien définie. Notez que, pour tout $i \in \mathbb{N}$, $\eta_i < \alpha_i$ et $\alpha_i < \eta_{i+1}$ (par construction).

Ceci nous permet de définir le $\mathcal{GV}\mathcal{T}$ suivant : $\mathcal{G}_\omega = g \odot \{(E_i, \mathcal{T} \cap \eta_i, \alpha_i) | i \in \mathbb{N}\}$. Notez que $U_{\mathcal{G}_\omega} = U_{\mathcal{G}}$ et alors que $\mathcal{G}_\omega$ appartient à $\mathcal{CTT}|_{\mathcal{F}}$. Enfin, pour tout $k \in \mathbb{N}^*$, nous avons $d_\Gamma(\mathcal{G}_k, \mathcal{G}_\omega) = 2^{-\eta_k}$ par construction de la suite $(\mathcal{G}_n)_{n \in \mathbb{N}}$ et $\mathcal{G}_\omega$. Par conséquent, $(\mathcal{G}_n)_{n \in \mathbb{N}}$ converge dans $\mathcal{CTT}|_{\mathcal{F}}$ et sa limite est $\mathcal{G}_\omega$.

Nous sommes maintenant en mesure d'appliquer le théorème 5.1 et d'en déduire que la $(\mathcal{A}, \mathcal{G}_\omega)$ -sortie est la limite de la suite des $(\mathcal{A}, \mathcal{G}_n)$ -sorties. En d'autres termes, la $(\mathcal{A}, \mathcal{G}_\omega)$ -sortie partage un préfixe de longueur η_i avec la $(\mathcal{A}, \mathcal{G}_i)$ -sortie pour tout $i \in \mathbb{N}$. Cela signifie qu'il existe une infinité de configurations durant l'exécution de l'algorithme $\mathcal{A}$ sur $\mathcal{G}_\omega$ où l'ensemble des processus fournissant en sortie la valeur "vrai" est l'ensemble M_i (les configurations aux dates η_i) et qu'il existe une infinité de configurations durant l'exécution de l'algorithme $\mathcal{A}$ sur $\mathcal{G}_\omega$ où l'ensemble des processus fournissant en sortie la valeur "vrai" est l'ensemble M'_i (les configurations aux dates α_i). Comme nous savons que pour tout $i \in \mathbb{N}^*$ $M_i \neq M'_i$, nous aboutissons à la conclusion que l'ensemble des processus qui fournissent en sortie la valeur "vrai" n'est jamais stable dans $\mathcal{G}_\omega$, ce qui contredit l'hypothèse selon laquelle l'algorithme $\mathcal{A}$ satisfait la spécification du problème de l'ensemble dominant minimal ultime dans les $\mathcal{GV}\mathcal{T}$ de la famille $\mathcal{CTT}|_{\mathcal{F}}$. $\square$

5.3.3 Mariage maximal ultime

Dans cette section nous allons aborder le problème de la construction du mariage maximal ultime.

Dans cette section, nous exhiberons ensuite notre résultat d'impossibilité de construction d'un mariage maximal ultime par un algorithme déterministe sur la famille des $\mathcal{GV}\mathcal{T}$ s de $\mathcal{CTT}$. Plus particulièrement, nous allons prouver que le mariage maximal ultime ne peut être résolu de manière déterministe par un algorithme sur un $\mathcal{GV}\mathcal{T}$ de $\mathcal{CTT}$ que si l'empreinte globale de ce $\mathcal{GV}\mathcal{T}$ admet un mariage maximal ne contenant que des isthmes.

5.3.3.0.1 Isthme. Dans un graphe $g = (V, E)$, un **isthme** est une arête de E dans la suppression induit à un graphe avec plus de composantes connexes. En d'autres termes, une arête $e \in E$ est un isthme si et seulement si elle n'appartient pas à un cycle dans g .

Théorème 5.4 *Pour tout ensemble de graphes statiques $\mathcal{F}$ qui contient au moins un graphe qui n'admet pas de mariage maximal qui n'est composé que d'isthmes, il n'existe*

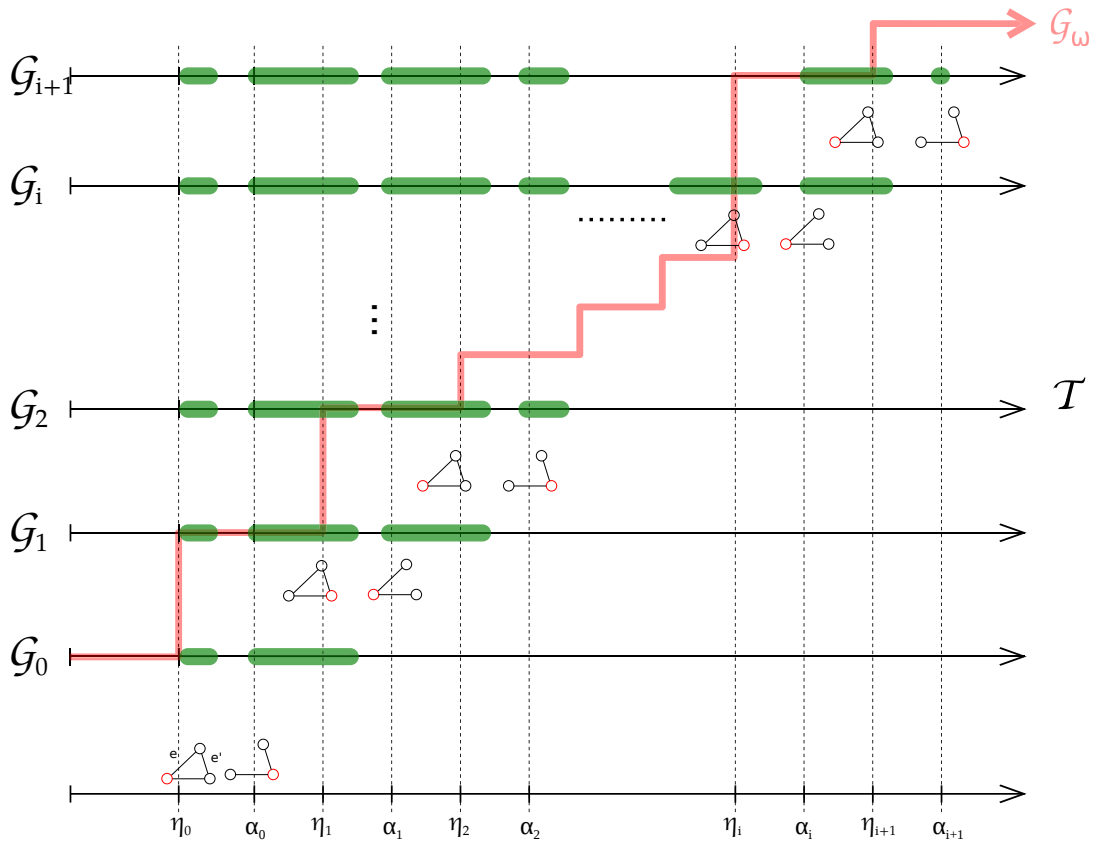


FIGURE 5.5 – Construction de la suite de $\mathcal{GV}\mathcal{T}$ dans la preuve du théorème 5.3 par l'exemple. Les lignes en vert représentent les moments durant lesquels l'ensemble dominant minimal fournit en sortie par tous les processus de V est stable. Les graphes de la figure permettent d'illustrer, à travers les sommets en rouge, un exemple d'ensembles dominants M_i et M'_i fournis en sortie par les processus durant la construction de la preuve.

aucun algorithme déterministe qui satisfait la spécification du mariage maximal ultime pour la sous-famille induite $\mathcal{CTT}|_{\mathcal{F}}$.

Preuve. Nous allons d'abord introduire quelques notations. Nous définissons pour tout graphe variant dans le temps $\mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta, \phi)$, le graphe variant dans le temps $\mathcal{G} \ominus \{(e_1, \mathcal{T}_{e_1}), \dots, (e_k, \mathcal{T}_{e_k})\}$ (avec, pour tout $i \in \{0, \dots, k\}$, $e_i \in E$ et $\mathcal{T}_{e_i} \subseteq \mathcal{T}$) comme le graphe variant dans le temps $(V, E, \mathcal{T}, \rho', \zeta, \phi)$ avec :

$$\rho'(e, t) = \begin{cases} 0 & \text{si } \exists i \in \{0, \dots, k\}, e = e_i \text{ et } t \in \mathcal{T}_{e_i} \\ \rho(e, t) & \text{autrement} \end{cases}$$

Par l'absurde, supposons qu'il existe un ensemble de graphes statiques $\mathcal{F}$ comprenant au moins un graphe statique qui n'admet pas un mariage maximal ne contenant que des isthmes et qu'il existe un algorithme déterministe $\mathcal{A}$ qui satisfait la spécification du mariage maximal ultime pour $\mathcal{CTT}|_{\mathcal{F}}$. Par définition, tout processus qui exécute l'algorithme $\mathcal{A}$ délivre en sortie une valeur booléenne à tout instant.

Soit $\mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta, \phi)$ un graphe variant dans le temps de $\mathcal{CTT}|_{\mathcal{F}}$ tel que $U_{\mathcal{G}}$ n'admet pas de mariage maximal ne contenant que des isthmes et que $\forall e \in E, \forall t \in \mathcal{T}, \rho(e, t) = 1$ ($\mathcal{G}$ existe par construction de $\mathcal{F}$ et par définition de $\mathcal{CTT}|_{\mathcal{F}}$). Nous construisons alors une suite $(\mathcal{G}_n)_{n \in \mathbb{N}}$ de graphe variant dans le temps comme suit. Nous fixons $\mathcal{G}_0 = g$. Supposons que nous avons déjà $\mathcal{G}_i = (V, E, \mathcal{T}, \rho', \zeta, \phi)$ pour un certain $i \in \mathbb{N}$ tel que $\mathcal{G}_i \in \mathcal{CTT}|_{\mathcal{F}}$, $U_{\mathcal{G}_i} = U_{\mathcal{G}}$, et que $\exists \alpha_{i-1} > 0, \forall e \in E, \forall t > \alpha_i, \rho'(e, t) = \rho(e, t)$. Ensuite, nous définissons par induction $\mathcal{G}_{i+1}$ de la manière suivante :

1. Considérons l'exécution de l'algorithme $\mathcal{A}$ sur $\mathcal{G}_i$ et soit $\eta_i \in \mathcal{T}$ la plus petite date strictement supérieure à α_{i-1} après laquelle l'ensemble des processus fournissant en sortie "vrai" est stable (η_i existe par l'hypothèse sur $\mathcal{A}$ puisque $\mathcal{G}_i \in \mathcal{CTT}|_{\mathcal{F}}$);
2. Soit M_i le mariage maximal calculé par l'algorithme $\mathcal{A}$ sur $\mathcal{G}_i$ (c'est-à-dire, l'ensemble des processus de $\mathcal{G}_i$ qui délivrent en sortie la valeur "vrai" après la date η_i). Puisque $U_{\mathcal{G}_i} = U_{\mathcal{G}}$, nous savons par l'hypothèse sur $U_{\mathcal{G}}$ que $U_{\mathcal{G}_i}$ n'admet pas de mariage maximal ne contenant que des isthmes. Plus précisément, il existe une arête e_i qui n'est pas un isthme dans M_i ;
3. Soit $\mathcal{G}'_i = \mathcal{G}_i \ominus \{(e_i, \mathcal{T} \cap]\eta_i, +\infty[)\}$.
4. Remarquez que $U_{\mathcal{G}'_i} = U_{\mathcal{G}_i} = U_{\mathcal{G}}$ (par construction de $\mathcal{G}'_i$ puisque $\eta_i > 0$) et que $U_{\mathcal{G}'_i}^{\omega}$ est connexe (puisque e_i n'est pas un isthme). Par conséquent, $\mathcal{G}'_i \in \mathcal{CTT}|_{\mathcal{F}}$ et nous pouvons considérer l'exécution de l'algorithme $\mathcal{A}$ sur $\mathcal{G}'_i$. Soit $\alpha_i \in \mathcal{T}$ la plus petite date strictement supérieure à η_i après laquelle l'ensemble des processus qui délivrent "vrai" est stable. Soit M'_i le mariage maximal calculé par l'algorithme $\mathcal{A}$ sur $\mathcal{G}'_i$ (c'est-à-dire, l'ensemble des processus de $\mathcal{G}'_i$ qui délivrent "vrai" après la date α_i). Notez que $M'_i \neq M_i$ puisque e_i n'appartient pas à $U_{\mathcal{G}'_i}^{\omega}$ par construction;
5. Soit $\mathcal{G}_{i+1} = \mathcal{G}_i \ominus \{(e_i, \mathcal{T} \cap]\eta_i, \alpha_i])\}$.

On peut facilement vérifier que cette construction assure que s'il existe $\mathcal{G}_i = (V, E, \mathcal{T}, \rho', \zeta, \phi)$ pour un certain $i \in \mathbb{N}$ tel que $\mathcal{G}_i \in \mathcal{CTT}|_{\mathcal{F}}$, $U_{\mathcal{G}_i} = U_{\mathcal{G}}$, et que

$\exists \alpha_{i-1} > 0, \forall e \in E, \forall t > \alpha_i, \rho'(e, t) = \rho(e, t)$, alors $\mathcal{G}_{i+1}$ satisfait la même propriété. De plus, comme $\mathcal{G}_0 = g$, cette propriété est naturellement satisfaite pour $i = 0$ avec n'importe quel $\alpha_{-1} > 0$. Par conséquent, la suite $(\mathcal{G}_n)_{n \in \mathbb{N}}$ est bien définie. Notez que, pour tout $i \in \mathbb{N}$, $\eta_i < \alpha_i$ et $\alpha_i < \eta_{i+1}$ (par construction).

Ceci nous permet de définir le $\mathcal{GVT}$ suivant : $\mathcal{G}_\omega = \mathcal{G} \ominus \{(e_i, \mathcal{T} \cap]\eta_i, \alpha_i]) \mid i \in \mathbb{N}\}$. Notez que $U_{\mathcal{G}_\omega} = U_{\mathcal{G}}$ et donc que $\mathcal{G}_\omega$ appartient à $\mathcal{CTT}|_{\mathcal{F}}$. Enfin, pour tout $k \in \mathbb{N}^*$, nous avons $d_\Gamma(\mathcal{G}_k, \mathcal{G}_\omega) = 2^{-\eta_k}$ par construction de la suite $(\mathcal{G}_n)_{n \in \mathbb{N}}$ et de $\mathcal{G}_\omega$. Par conséquent, $(\mathcal{G}_n)_{n \in \mathbb{N}}$ converge dans $\mathcal{CTT}|_{\mathcal{F}}$ et sa limite est $\mathcal{G}_\omega$.

Nous sommes maintenant en mesure d'appliquer le théorème 5.1 de déduire que la $(\mathcal{A}, \mathcal{G}_\omega)$ -sortie est la limite de la suite des $(\mathcal{A}, \mathcal{G}_n)$ -sorties. En d'autres termes, la $(\mathcal{A}, \mathcal{G}_\omega)$ -sortie partage un préfixe de longueur η_i avec la $(\mathcal{A}, \mathcal{G}_i)$ -sortie pour tout $i \in \mathbb{N}$.

En résumé, pour tout $i \in \mathbb{N}^*$, M_i est l'ensemble des processus qui délivrent "vrai" dans $\mathcal{G}_\omega$ à la date η_i et que M'_i est l'ensemble des processus qui fournissent en sortie "vrai" dans $\mathcal{G}_\omega$ à la date α_i lors de l'exécution de $\mathcal{A}$ sur $\mathcal{G}_\omega$. Puisque $M_i \neq M'_i$ pour tout $i \in \mathbb{N}$, nous aboutissons à la conclusion que l'ensemble des processus qui délivrent en sortie la valeur "vrai" n'est jamais stable dans $\mathcal{G}_\omega$, ce qui contredit l'hypothèse selon laquelle l'algorithme $\mathcal{A}$ satisfait la spécification du problème du mariage maximal dans les $\mathcal{GVT}$ de la famille $\mathcal{CTT}|_{\mathcal{F}}$. $\square$

5.4 Conclusion

Dans ce chapitre, nous avons présenté un outil général dont la motivation est de justifier les arguments informels basés sur la convergence de suites d'objets. Ce type de raisonnement est très souvent utilisé dans la preuve de résultats d'impossibilité ou la nécessité de conditions. Plus précisément, nous avons prouvé dans le théorème 5.1 qu'étant donné un algorithme $\mathcal{A}$ conçu pour tout graphe variant dans le temps et une suite $(\mathcal{G})_n$ de $\mathcal{GVT}$ qui converge vers un certain graphe variant dans le temps $\mathcal{G}$, alors la suite d'exécutions de $\mathcal{A}$ sur chaque graphe variant dans le temps de la suite converge aussi.

Dans la section 5.3 nous avons démontré l'utilité de cet outil générique au travers les preuves d'impossibilité de construction de l'empreinte ultime, du mariage maximal ultime et de l'ensemble dominant minimal ultime sur la famille $\mathcal{CTT}$ de $\mathcal{GVT}$.

Notez que nous prouverons que les conditions nécessaires que nous avons trouvées pour la construction de l'ensemble dominant minimal ultime sont suffisantes, dans le chapitre 7.

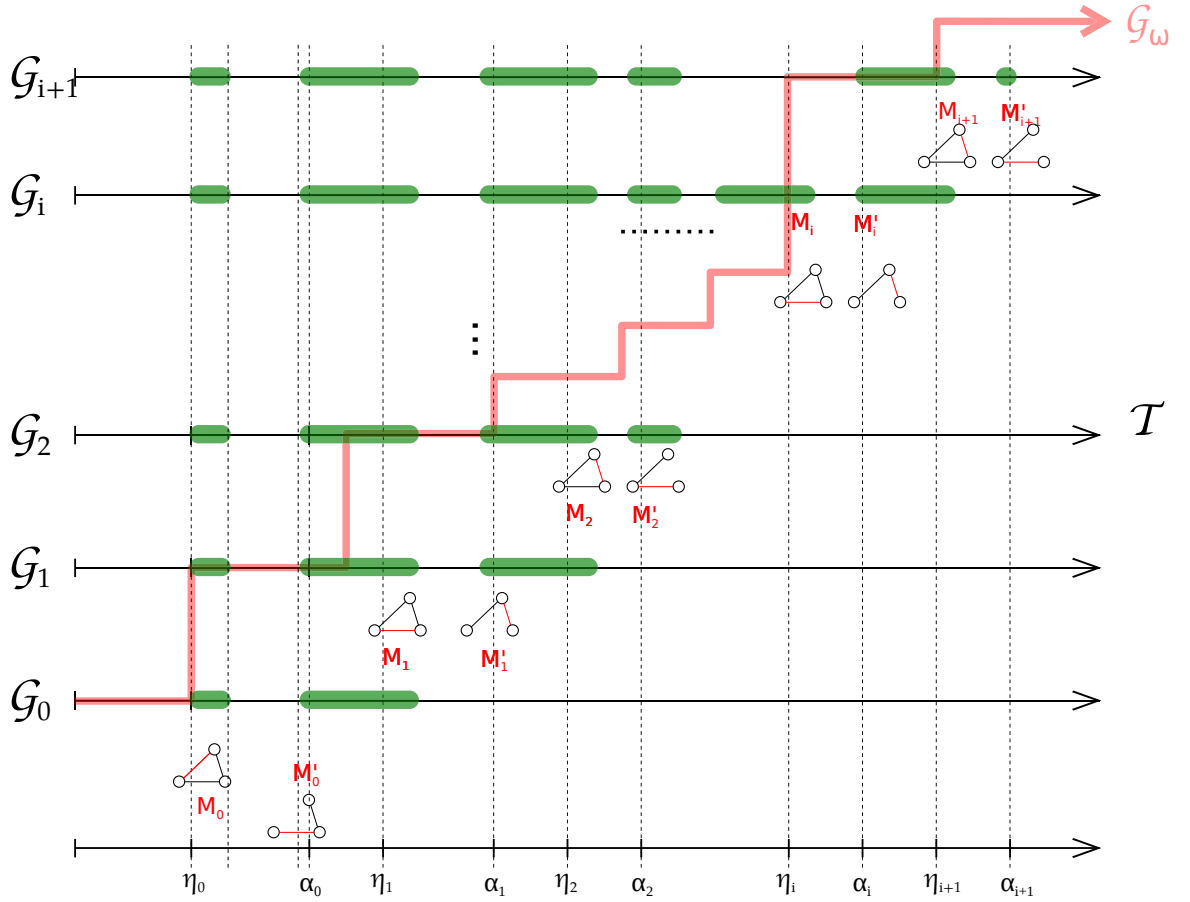


FIGURE 5.6 – Construction de la suite de $\mathcal{GVT}$ dans la preuve du théorème 5.4 par l'exemple. Les lignes en vert représentent les moments durant lesquels le couplage maximal fournit en sortie par tous les processus de V est stable. Les graphes de la figure permettent d'illustrer, à travers les arêtes en rouge, le couplage maximal stable M_i et M'_i que délivre les processus durant la construction de la preuve.

Chapitre 6

Complexité

Sommaire

6.1	Mesure de complexité existantes	100
6.1.1	Complexité en messages	100
6.1.2	Complexité en temps	100
6.2	Nouvelle mesure de complexité	101
6.3	Exemple : construction de l’empreinte globale	104
6.3.1	Algorithme	104
6.3.2	Correction	105
6.3.3	Complexité	107
6.3.4	Optimalité	108
6.4	Conclusion	109

Résoudre une tâche donnée dans un système distribué dynamique requiert un certain temps et un certain volume de communications. En algorithmique, une mesure de complexité permet de comparer les performances de différents algorithmes résolvant la même tâche. Dans la mesure du possible, celle-ci doit être équitable (c’est-à-dire, discriminer les coûts liés au système de ceux dus à l’algorithme).

Dans la littérature, il existe principalement deux approches pour mesurer la complexité d’algorithmes distribués dans des systèmes dynamiques, à savoir mesurer la complexité en temps et la complexité en messages.

Dans ce chapitre, nous présentons ces deux approches (voir section 6.1). Nous verrons en particulier que les mesures de complexités en temps existantes ne considèrent que des cas particuliers et ne conviennent pas au cadre des systèmes asynchrones hautement dynamiques que nous considérons dans cette thèse.

C’est pourquoi nous proposons dans la section 6.2 une nouvelle mesure de complexité qui étend la mesure de complexité classique en rondes utilisée pour les algorithmes distribués sur des systèmes statiques.

La section 6.3 est dédiée à un exemple pour illustrer cette nouvelle mesure de complexité. Nous donnons un algorithme de construction de l’empreinte globale d’un $\mathcal{GVT}$ et nous montrons que cet algorithme est optimal par rapport à cette mesure.

6.1 Mesure de complexité existantes

Dans la littérature, certains travaux se sont intéressés à la complexité dans les modèles de graphes variant dans le temps.

Nous allons décrire dans cette section les mesures de complexité existantes dans la littérature en distinguant les mesures s'intéressant au volume de communication puis au temps d'exécution.

6.1.1 Complexité en messages

La complexité en nombre de messages d'un algorithme $\mathcal{A}$ sur une famille de graphes variant dans le temps $\mathcal{C}$, est le nombre de messages échangés, dans le pire cas, durant l'exécution de l'algorithme $\mathcal{A}$ sur les $\mathcal{GVT}$ de $\mathcal{C}$.

Quelques travaux utilisent cette mesure de complexité pour les graphes variant dans le temps, comme dans [CFMS15, CFMS10], afin de mesurer la complexité en messages de trajets rapides, trajets terminant plus tôt et trajets courts. Dans [CFMS10], les auteurs ont montré que si le nombre de sommets n et/ou la borne de récurrence des arêtes Δ sont connus des sommets, il est alors possible de résoudre le problème de la diffusion terminant plus tôt avec une complexité en message de $O(m)$ (m étant le nombre d'arêtes). Les auteurs montrent aussi que s'il existe une borne sur la récurrences des arêtes et que n ou Δ est connu des sommets, la complexité en temps est de $O(n\Delta)$.

6.1.2 Complexité en temps

D'une manière générale, la complexité en temps d'un algorithme $\mathcal{A}$, dans une famille de graphes variant dans le temps $\mathcal{C}$, est le temps nécessaire, dans le pire cas, que requiert l'exécution de l'algorithme $\mathcal{A}$ pour satisfaire sa tâche sur les $\mathcal{GVT}$ de $\mathcal{C}$.

Cette définition se heurte au fait que la mesure du temps dans un système distribué (en particulier asynchrone) n'est pas toujours significative. C'est sans doute la raison pour laquelle cette mesure n'a été appliquée qu'à des modèles synchrones ou semi-synchrones dans le cadre des systèmes dynamiques jusqu'à présent.

Le modèle synchrone le plus utilisé dans la littérature suppose que le temps de vie du $\mathcal{GVT}$ est un domaine temporel discret (souvent $\mathbb{N}$). Par exemple, les auteurs dans [KMO11a] prennent pour modèle synchrone la famille constamment connexe de $\mathcal{GVT}$ (voir la Propriété 3.2.2) dans le cadre de l'étude du problème du consensus. Les auteurs montrent, entre autres, que le consensus éventuel (où les processus s'accordent en même temps sur une même sortie) ne peut être résolu en moins de $|V| - 1$ pas d'exécution (où $|V|$ est le nombre de sommets de V). Dans [KLO10], les auteurs montrent sur le même modèle, dans le cadre du calcul de la taille d'un système, que la complexité en temps est $O(n^2)$. Tandis que pour la famille T-intervalle connexe, pour tout $T > 1$, les auteurs montrent que la complexité en temps est de l'ordre de $O(n + n^2/T)$.

Les modèles semi-synchrones font l'hypothèse de l'existence d'une borne supérieure Δ sur la récurrence des arêtes, à l'image des travaux de [CFMS10] dans le cadre de l'étude des diffusions (les plus rapides, les plus courtes et celles terminant plus tôt) dans les graphes variant dans le temps des familles $F10$ et $F11$. Le modèle semi-synchrone des familles de $\mathcal{GVT}$ T-intervalle connexe est aussi utilisé dans [KO11], [IW13] dans le cadre de l'étude de l'exploration d'un système dynamique par des agents.

6.2 Nouvelle mesure de complexité

Les mesures de complexité en temps que nous avons décrit précédemment ne sont pas adaptées à notre modèle. En effet, ces mesures de complexité ne s'appliquent que sur des cas particuliers de systèmes dynamiques : les systèmes synchrones ou les systèmes dont les événements topologiques sont récurrents de telle sorte que cette récurrence est bornée. Or, dans notre modèle, les communications ne sont pas synchrones et les événements topologiques ne sont pas nécessairement récurrents.

Par conséquent, nous avons besoin de définir une méthode qui puisse mesurer la performance de nos algorithmes indépendamment des communications asynchrones et des changements topologiques. En effet, les caractéristiques de notre modèle peuvent introduire des délais dans le calcul de la solution qui ne sont pas imputables à l'algorithme mais plutôt à la dynamique du système. Un exemple typique d'un tel délai est l'attente d'une prochaine apparition d'une arête par un processus qui peut être arbitrairement longue.

Notre objectif est de définir une mesure de complexité en temps qui soit pertinente dans notre modèle. Nous proposons d'étendre la définition traditionnelle de la complexité de temps en rondes qui est largement adoptée dans le modèle asynchrone par passage de messages dans les systèmes statiques.

Cette mesure de complexité contourne le problème des délais dus aux communications asynchrones en considérant comme unité de temps d'une exécution le pire délai entre l'émission et la réception d'un message durant cette exécution (voir [AW04] pour plus de détails concernant cette approche). Par conséquent, en utilisant cette définition, il est possible de borner le temps de terminaison de toute exécution d'un algorithme indépendamment des délais de communication de cette exécution. En réalité, cette mesure capture les relations de causalité durant l'exécution de l'algorithme. Cette mesure de complexité en temps est donc une méthode équitable de comparaison entre algorithmes.

La contribution de ce chapitre est d'étendre cette mesure aux environnements dynamiques en prenant en compte les délais introduits par la dynamique du système. De manière intuitive, nous considérons comme unité de temps d'une exécution le pire délai entre l'invocation de la primitive **Send_retry** et la réception du message par cette primitive durant cette exécution.

Cette définition naturelle de l'unité de temps pour des systèmes dynamiques n'est pas suffisante. En effet, la dynamique du système peut introduire un autre délai

arbitrairement long que nous appelons *le délai initial*. Comme exemple de ce délai, prenons un problème comme le consensus qui nécessite que chaque processus propage une valeur initiale. Imaginons que la dynamique du système ait pour effet d'isoler un processus en le déconnectant durant un temps arbitrairement long. Cela amènera tous les processus à attendre la première apparition de ce processus. Intuitivement, ce délai est imputable à la dynamique du système et non à l'algorithme. De ce fait, notre mesure de complexité doit aussi ignorer ce délai initial.

Dans ce but, nous définissons pour chaque problème un *temps de départ* comme suit. Le temps de départ d'un problème est le plus petit instant de l'exécution où la dynamique du système a "montré" l'information topologique minimale aux processus pour résoudre le problème. Notez que ce temps de départ dépend uniquement du problème. Par exemple, pour les problèmes similaires au consensus, il s'agira de l'instant de la première connexion du dernier processus à se connecter à un de ses voisins. Notez aussi que dans un système statique, le temps de départ et le temps initial sont égaux puisque le système n'a à attendre l'apparition d'aucune information topologique.

Nous proposons donc de mesurer le temps de terminaison d'un algorithme, sur une famille de $\mathcal{GVT}$ $\mathcal{C}$, par le pire temps (exprimé à l'aide de l'unité de temps défini précédemment) entre le temps de départ et la terminaison de l'algorithme sur toutes les exécutions de cet algorithme sur les $\mathcal{GVT}$ de $\mathcal{C}$. Nous sommes convaincus que cette mesure de complexité permet de comparer de manière équitable les algorithmes définis pour toute famille de $\mathcal{GVT}$. En effet, cette méthode de calcul exhibe le coût de communication intrinsèque des algorithmes (c'est-à-dire, la longueur des chaînes causales) sans prendre en compte les délais introduits par les changements topologiques et les communications asynchrones.

Nous allons maintenant présenter cette mesure de complexité plus formellement. Dans un souci de simplification nous considérons uniquement les *problèmes de calcul de points fixe* dans une famille $\mathcal{C}$ de graphes variant dans le temps. Cette classe de problèmes est définie comme suit.

Problème de calcul de points fixe. Il s'agit des problèmes qui admettent une spécification de la forme suivante : toute exécution sur chaque $\mathcal{GVT}$ de la famille $\mathcal{C}$ atteint en un temps fini un suffixe où chaque processus fournit en sortie une valeur donnée de manière stable. La valeur requise dépend du problème considéré et n'est pas forcément identique pour chaque processus.

Selon cette définition, l'élection de leader ou la construction de structure couvrante sont des problèmes de calcul de points fixes tandis que l'exclusion mutuelle et la diffusion ne le sont pas.

Prenons maintenant un algorithme déterministe $\mathcal{A}$ qui satisfait la spécification d'un problème $\mathcal{P}$ de calcul de points fixes dans une famille de $\mathcal{GVT}$ $\mathcal{C}$. Soit e l'exécution de l'algorithme $\mathcal{A}$ sur un $\mathcal{GVT}$ donné de la famille $\mathcal{C}$.

Délai. Pour tout message m envoyé durant l'exécution e , nous appelons **délai** (de m) le temps entre l'invocation de la primitive "Send_retry" par l'émetteur de m et la

réception de m par son destinataire. Notez que ce délai peut être infini dans le cas où m n'est jamais délivré.

Ronde de communication. Nous appelons **ronde de communication** (ou simplement **ronde**) de l'exécution e , le grand délai parmi l'ensemble des messages qui sont délivrés durant l'exécution e . Notez que nous ne considérons pas ici les messages qui ne sont jamais délivrés dans e .

Fonction d'ensemble de présence nécessaire. Nous appelons **fonction d'ensemble de présence nécessaire** de $\mathcal{P}$, la fonction $EPN_{\mathcal{P}}$ (propre à $\mathcal{P}$) qui retourne pour tout $\mathcal{GVT}$ $(V, E, \mathcal{T}, \rho, \zeta, \phi)$ de $\mathcal{C}$, un ensemble de sous ensembles de E . Chaque élément de $EPN_{\mathcal{P}}(\mathcal{G})$ est un des ensembles d'arêtes dont l'apparition est nécessaire pour entamer la résolution du problème de manière effective (indépendamment de l'algorithme utilisé). Notez que la définition réelle de cette fonction dépend du problème lui même et non d'un $\mathcal{GVT}$ ou d'une exécution.

Par exemple, pour le problème du calcul de l'empreinte globale $\mathcal{U}_{\mathcal{G}}$ sur un $\mathcal{GVT}$ $\mathcal{G}$, nous avons $EPN_{\mathcal{U}_{\mathcal{G}}}(\mathcal{G}) = E$ puisque chaque arête de E doit apparaître dans la sortie émise par tout processus. Pour donner un autre exemple, prenons le problème de la diffusion $\mathcal{B}$, nous avons $EPN_{\mathcal{B}}(\mathcal{G}) = \{\{(p, q)\} | q \in \mathcal{N}_p\}$ (où le processus p est l'émetteur du message) puisque l'apparition de n'importe quelle arête adjacente à p est nécessaire pour commencer la diffusion du message par p .

Définition 6.1 (Temps de départ) *Le temps de départ d'une exécution e d'un algorithme $\mathcal{A}$ sur un $\mathcal{GVT}$ $\mathcal{G}$ est la plus petite date $t \in \mathcal{T}$ de telle sorte que chaque arête d'au moins un élément de $EPN_{\mathcal{P}}(\mathcal{G})$ est présente au moins une fois avant t .*

Notez que dans un système distribué statique, le temps initial et le temps de départ sont identiques pour tout problème $\mathcal{P}$, puisque, dans toute configuration initiale, toutes les arêtes de tous les éléments de $EPN_{\mathcal{P}}(\mathcal{G})$ sont présentes quelle que soit la définition de $EPN_{\mathcal{P}}$.

Définition 6.2 (Temps de convergence) *Le temps de convergence de l'exécution d'un algorithme $\mathcal{A}$ résolvant un problème $\mathcal{P}$ sur un graphe variant dans le temps $\mathcal{G}$ est le temps (exprimé en rondes de e) entre le temps de départ de e et le plus petit temps dans e où la spécification de $\mathcal{P}$ est satisfaite.*

Nous sommes maintenant en mesure de présenter la définition centrale de ce chapitre.

Définition 6.3 (Complexité en temps sur une famille de $\mathcal{GVT}$) *La complexité en temps d'un algorithme distribué $\mathcal{A}$ qui satisfait la spécification d'un problème de points fixes $\mathcal{P}$ dans une famille de $\mathcal{GVT}$ $\mathcal{C}$ est le pire temps de convergence de $\mathcal{A}$ sur l'ensemble des $\mathcal{GVT}$ de $\mathcal{C}$.*

Nous avons présenté cette définition dans le cadre des problèmes de calcul de points fixes mais notez qu'elle peut naturellement être étendue aux problèmes dits de service de la manière suivante. D'abord, nous prenons comme temps de départ le maximum entre le temps de départ définie précédemment et le temps d'arrivée d'une requête du service (par exemple, l'envoi d'un message pour un algorithme de diffusion, la requête d'accès à section critique pour une exclusion mutuelle). Deuxièmement, on remplace le temps de convergence d'un algorithme par le temps d'achèvement du service requis par l'algorithme (par exemple, la livraison du message à son destinataire pour un algorithme de diffusion, l'entrée en section critique pour une exclusion mutuelle).

6.3 Exemple : construction de l'empreinte globale

Dans cette section, nous présentons un algorithme de construction de l'empreinte globale (voir la section 6.3.1) et nous prouvons son optimalité en temps d'exécution par rapport à notre nouvelle mesure (voir la section 6.3.4). Cet algorithme sera utilisé comme composant de base pour la construction de notre algorithme de calcul de l'ensemble dominant minimal ultime dans le chapitre suivant.

6.3.1 Algorithme

De manière intuitive, l'idée de notre algorithme est la suivante. Chaque processus sauvegarde localement un graphe, initialement vide, qui contiendra ultimement toutes les arêtes de l'empreinte globale. Ce graphe constitue la sortie de chaque processus. Lors de la première apparition d'une arête, ses deux (processus) extrémités ajoutent cette arête à leur graphe. Ensuite, ils essaient de propager la dernière version de leur graphe local à tous les processus qu'ils ont eu comme voisin au moins une fois depuis le début de l'exécution. Lorsqu'un processus reçoit un tel message (qui contient le graphe d'un autre processus), il ajoute à son empreinte globale locale l'ensemble des arêtes du graphe reçu qu'il ignore. Si le graphe local a été modifié durant cette étape, alors le processus essaie de propager à nouveau son propre graphe local à tous les processus qu'il a eu pour voisin au moins une fois depuis le début de l'exécution.

Cet algorithme garantit que, lors de la première apparition de la dernière arête de l'empreinte globale, cette arête est ajoutée au graphe local de ses deux extrémités et qu'ensuite cette arête est propagée (au moins) à leur voisin de l'empreinte ultime en une seule ronde et ainsi de suite. Notez qu'il n'existe aucune garantie de diffusion à tous les voisins du graphe local car des arêtes vouées à disparaître peuvent exister. Ainsi, dans toute exécution, après au plus $diam(U_G^\omega)$ rondes (où $diam(U_G^\omega)$ est le diamètre de l'empreinte ultime), cette arête (ainsi que toutes les autres) apparaît dans le graphe fourni en sortie par chaque processus. Le pseudo-code de cet algorithme est fourni dans l'Algorithme 1.

Les sections suivantes sont dédiées à la preuve du résultat suivant :

Variables :

$g_p = (V_p, E_p)$: l'empreinte construite par p
 $\mathcal{N}_p$: l'ensemble des voisins de p

Initialisation :

$g_p := (\{x\}, \emptyset)$
 $\mathcal{N}_p := \emptyset$

Lors de l'apparition d'une arête $\{p, q\}$:

if $\{p, q\} \notin E_p$ **then**
 $\mathcal{N}_p := \mathcal{N}_p \cup \{q\}$
 $g_p := (V_p \cup \{q\}, E_p \cup \{\{p, q\}\})$
foreach $r \in \mathcal{N}_p$ **do**
 Send_retry($add(g_p), r$)

À la réception de $add(g_q)$ depuis q :

if $E_q \setminus E_p \neq \emptyset$ **then**
 $g_p := (V_p \cup V_q, E_p \cup E_q)$
foreach $r \in \mathcal{N}_p \setminus \{q\}$ **do**
 Send_retry($add(g_p), r$)

Algorithme 1 – L'algorithme du calcul de l'empreinte par un processus p .

Théorème 6.1 *L'Algorithme 1 satisfait la spécification de construction de l'empreinte globale pour tout $\mathcal{GVT}$ de la famille $\mathcal{CCT}$. De plus, son temps de convergence sur tout $\mathcal{GVT}$ $\mathcal{G}$ de la famille $\mathcal{CCT}$ est $\text{diam}(U_{\mathcal{G}}^{\omega})$ rondes, ce qui est optimal.*

6.3.2 Correction

Lemme 6.1 *L'Algorithme 1 satisfait la spécification de construction de l'empreinte globale pour tout $\mathcal{GVT}$ de la famille $\mathcal{CCT}$.*

Preuve. Pour prouver ce résultat, nous allons montrer que pour toute exécution ϵ de l'Algorithme 1 sur tout $\mathcal{GVT}$ $\mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta)$ de $\mathcal{CCT}$, il existe un instant $t_f \in \mathcal{T}$ tel que tout processus $p \in V$ fournira en sortie de manière stable un graphe qui correspond à l'empreinte globale de $\mathcal{G}$.

Soit $\mathcal{G}_0 = (V, E, \mathcal{T}, \rho, \zeta)$ un $\mathcal{GVT}$ de la famille $\mathcal{CCT}$ tel que $E = \emptyset$. Par définition de la famille $\mathcal{CCT}$, le $\mathcal{GVT}$ $\mathcal{G}_0$ a au plus un processus dans V . Supposons que p est l'unique processus de $\mathcal{G}_0$. Soit ϵ_0 l'exécution de l'Algorithme 1 sur $\mathcal{G}_0$. D'après l'Algorithme 1 à l'initialisation l'ensemble des processus du graphe construit par p est initialisé à $\{p\}$ et l'ensemble des arêtes E_p est initialisé à $\emptyset$. Puisque le processus p n'a aucun voisin dans $\mathcal{G}_0$ alors il existe un instant t durant l'exécution où le processus p fournit en sortie de manière stable le graphe (V, E) tel que $V = \{p\}$ et $E = \emptyset$. Cela correspond à l'empreinte globale de $\mathcal{G}_0$.

Nous allons maintenant prouver par induction que dans tout $\mathcal{GVT}$ $\mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta)$ de la famille $\mathcal{CCT}$, une arête $\{p, q\}$ qui est présente au moins une fois

durant le temps de vie $\mathcal{T}$ de $\mathcal{G}$ sera connu par tous les processus $p \in V$ de $\mathcal{G}$. Cette induction portera sur le nombre de processus connaissant $\{p, q\}$.

Soit $\mathcal{G}$ un $\mathcal{GV}\mathcal{T}$ de la famille $\mathcal{CCT}$ tel que $E \neq \emptyset$. Soit $\{p, q\} \in E$ une arête de $\mathcal{G}$. D'après l'Algorithme 1, lorsqu'une arête $\{p, q\}$ apparaît, à un instant $t_{app}(\{p, q\})$, pour la première fois durant l'exécution de l'algorithme, le processus p (respectivement q) ajoute $\{p, q\}$ à son ensemble E_p (respectivement E_q) à un instant $t_0(\{p, q\}) \geq t_{app}(\{p, q\})$. Nous pouvons dire que le processus p (respectivement le processus q) connaissent l'arête $\{p, q\}$ à partir de l'instant $t_0(\{p, q\})$. Soit $I_0(\{p, q\})$, l'ensemble des processus qui connaissent l'arête $\{p, q\}$ et les processus qui lui sont associés à l'instant $t_0(\{p, q\})$. Notez que $|I_0(\{p, q\})| \geq 2$, puisque au moins le processus p et q connaissent l'arête $\{p, q\}$.

Soit $I_i(\{p, q\})$ l'ensemble des processus qui connaissent l'arête $\{p, q\}$ à l'instant $t_i(\{p, q\})$. Mettons, $\bar{I}_i(\{p, q\}) = V \setminus I_i(\{p, q\})$ l'ensemble des processus qui ne connaissent pas l'arête $\{p, q\}$ à l'instant $t_i(\{p, q\})$.

Nous allons prouver qu'il existe un temps $t_{i+1}(\{p, q\})$ durant l'exécution tel que l'ensemble des processus $I_{i+1}(\{p, q\}) \not\supseteq I_i(\{p, q\})$ connaissant l'arête $\{p, q\}$ croît si $\bar{I}_i(\{p, q\}) \neq \emptyset$.

Puisque $\mathcal{G}$ appartient à la famille $\mathcal{CTT}$, il existe un temps $t > t_i$ où une arête liera un processus de $I_i(\{p, q\})$ avec un processus de $\bar{I}_i(\{p, q\})$. Soit $t'_i \geq t_i$ un instant tel qu'il existe au moins une arête qui lie au moins un processus de I_i à un autre processus de $\bar{I}_i$. Soit S_{i+1} le sous ensemble de processus de $\bar{I}_i$ impliqué dans une connexion entre I_i et $\bar{I}_i$ à la date t'_i . Soit $\{p_i, q_i\}$ une des arêtes qui apparaît à l'instant t'_i tel que $p_i \in I_i$ et $q_i \in \bar{I}_i$.

- Si le processus p_i ne connaît pas encore l'arête $\{p_i, q_i\}$ alors d'après l'Algorithme 1, le processus p_i partagera à travers un message son graphe local (V_{p_i}, E_{p_i}) avec son voisin q_i . À travers cette opération, le processus p_i demande au processus q_i d'ajouter l'arête $\{p, q\}$ (encore inconnu de q_i) dans le graphe local de q_i . Par l'Algorithme 1, il existe un instant $t_{i+1} > t'_i$ tel que le processus q_i ajoutera l'arête $\{p, q\}$ à l'ensemble E_{q_i} .
- Si l'arête $\{p_i, q_i\}$ est déjà connue du processus p_i (ce qui signifie que $\{p, q\} \in E_{p_i}$ et $\{p, q\} \notin E_{q_i}$ puisque $q_i \in \bar{I}_i$ ne connaît pas l'arête $\{p, q\}$), alors d'après l'Algorithme 1 le processus p_i partagera son graphe local (V_{p_i}, E_{p_i}) avec son voisin q_i à travers l'envoi d'un message à un instant $t'_i \geq t_i$ (par la primitive `Send_retry` qui va envoyer la mise à jour du graphe de p_i à q_i). Donc il existe un instant $t_{i+1} > t'_i$ où le processus q_i ajoute l'arête $\{p, q\}$ à l'ensemble E_{q_i} .

Par ces deux points on montrent qu'il existe un temps $t_{i+1}(\{p, q\})$ où l'ensemble des processus $I_{i+1}(\{p, q\})$ connaissant l'arête $\{p, q\}$ grandit par rapport à $I_i(\{p, q\})$, puisque $I_{i+1}(\{p, q\}) \not\supseteq I_i(\{p, q\})$ et $\bar{I}_{i+1}(\{p, q\}) = \bar{I}_i(\{p, q\}) \setminus I_{i+1}(\{p, q\})$. Cela implique que s'il existe encore des processus ne connaissant pas l'arête $\{p, q\} \in E$ au temps $t_i(\{p, q\})$ alors il existe un temps $t_{i+1}(\{p, q\}) > t_i(\{p, q\})$ tel que l'ensemble des processus connaissant $\{p, q\}$ grandit.

Puisque pour un i donné, l'ensemble $\bar{I}_i(\{p, q\})$ est fini, alors il existe un temps k tel que $I_k(\{p, q\}) = V$ et $\bar{I}_k = \emptyset$.

Pour conclure notre preuve, nous allons prouver qu'il n'existe aucune arête $e \notin E$ tel que $\forall k \geq 0, I_k(e) \neq \emptyset$. En d'autres termes, il n'existe aucun processus p qui connaisse une arête e qui n'existe pas dans E . Par l'Algorithme 1, chaque message d'un *Send_retry* envoyé ou pendant d'un processus $p \in V$ au cours de l'exécution de l'Algorithme 1 est provoqué par une apparition d'une arête. De ce fait, si une arête $\{p, q\}$ n'est pas dans E alors il n'y aura aucun événement d'apparition de e qui risque de se dérouler durant l'exécution de l'Algorithme 1 sur $\mathcal{G}$. Par conséquent, aucun processus $p \in V$ n'ajoute e dans son ensemble E_p et aucun message contenant e n'est envoyé pendant l'exécution de l'Algorithme 1 sur $\mathcal{G}$. $\square$

6.3.3 Complexité

Lemme 6.2 *L'Algorithme 1 converge sur tout $\mathcal{GVT}$ $\mathcal{G}$ de la famille $\mathcal{CCT}$ en au plus $\text{diam}(U_{\mathcal{G}}^{\omega})$ rondes.*

Preuve. Pour prouver ce résultat, nous allons montrer que, pour toute exécution ϵ de l'Algorithme 1, chaque arête $e \in E$ d'un $\mathcal{GVT}$ $\mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta)$ de la famille $\mathcal{CCT}$ sera "connue" par tout processus $p \in V$ de $\mathcal{G}$ (c'est-à-dire, apparaîtra de façon constante dans la sortie de p) au plus tard après $\text{diam}(U_{\mathcal{G}}^{\omega})$ rondes après le temps de départ de ϵ (noté t_s).

Notons $\Delta(e, p)$ la fonction qui retourne la plus petite distance entre p et chaque extrémité de l'arête e dans l'empreinte ultime $U_{\mathcal{G}}^{\omega}$ de $\mathcal{G}$.

Nous allons prouver par induction, que chaque arête $e \in E$ sera connue par un processus p en au plus $\Delta(e, p)$ rondes après t_s .

D'après l'Algorithme 1, chaque processus $p \in V$ ajoute à son graphe local les arêtes qui lui sont adjacentes dès leur première apparition. Puisque chaque arête de $\mathcal{G}$ est présente au moins une fois avant t_s par définition de $EPN_{\mathcal{U}}$, alors pour chaque arête e , chaque processus p tel que $\Delta(e, p) = 0$ connaîtra e en au plus 0 rondes après t_s . Soit e une arête de $\mathcal{G}$ et k un entier tel que $k \geq 0$. Supposons que, pour chaque processus p tel que $\Delta(e, p) = k$, e est connue de p en au plus k rondes après t_s . Nous allons prouver que chaque processus q tel que $\Delta(e, q) = k + 1$ connaîtra l'arête e en au plus $k + 1$ rondes après t_s .

D'après l'Algorithme 1, chaque processus qui met à jour son graphe local par l'ajout d'une nouvelle arête propage cette mise à jour par le biais de la primitive **Send_retry** à ses voisins. Or, tout processus q tel que $\Delta(e, q) = k + 1$, admet au moins un voisin p dans $U_{\mathcal{G}}^{\omega}$ tel que $\Delta(e, p) = k$. Par l'hypothèse de récurrence, le processus p va connaître e en au plus k rondes après t_s . Cela implique que p va envoyer son graphe au processus q dès l'ajout de e dans son graphe local.

Par définition de la ronde de communication et le fait que p et q soient voisins dans $U_{\mathcal{G}}^{\omega}$, q ajoutera e à son graphe en au plus $k + 1$ rondes après t_s .

La plus petite distance entre tout processus et l'une des extrémités de toute arête de $\mathcal{G}$ étant bornée par $\text{diam}(U_{\mathcal{G}}^{\omega})$, nous pouvons conclure que chaque processus va connaître chaque arête de $\mathcal{G}$ en au plus $\text{diam}(U_{\mathcal{G}}^{\omega})$ rondes après t_s . $\square$

6.3.4 Optimalité

Dans cette section, nous nous intéressons à une borne inférieure sur la complexité en temps pour la construction de l'empreinte globale. Nous nous sommes restreint aux algorithmes gloutons qui sont les plus naturels pour ce genre de problème.

Algorithme glouton Nous définissons un algorithme glouton pour la construction de l'empreinte globale comme un algorithme satisfaisant la propriété suivante. Initialement, la sortie de tout processus est un graphe vide et cette sortie ne peut que croître (au sens de l'inclusion) au fil du temps. En d'autres termes, un tel algorithme assure qu'une fois que le processus commence à émettre en sortie une certaine arête ou un processus, ce dernier apparaît toujours dans la sortie par la suite. Notez que l'Algorithme 1 est glouton selon cette définition.

Nous allons prouver qu'il n'existe aucun algorithme glouton pour le problème de construction de l'empreinte globale pour la famille de $\mathcal{GVT} \mathcal{CTT}$ qui réalise une meilleure complexité en temps que notre algorithme. En effet, nous allons prouver que, pour tout algorithme glouton, il existe un $\mathcal{GVT} \mathcal{G}$ de la famille $\mathcal{CTT}$ tel que cet algorithme nécessite $\text{diam}(U_g^\omega)$ rondes pour calculer l'empreinte globale de $\mathcal{G}$. Avant de démontrer ce résultat, nous avons besoin du lemme technique suivant :

Lemme 6.3 *Pour tout algorithme glouton $\mathcal{A}$ satisfaisant la spécification de la construction de l'empreinte globale pour la famille $\mathcal{CTT}$, pour tout $\mathcal{GVT} \mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta, \phi)$ de $\mathcal{CTT}$, pour toute arête $e \in E$ qui n'est pas une coupe de l'empreinte ultime U_g^ω de $\mathcal{G}$, pour tout processus $p \in V$ et pour tout $t \in \mathcal{T}$, l'arête e ne peut pas appartenir au graphe émis en sortie, à l'instant t , par p durant l'exécution de $\mathcal{A}$ sur $\mathcal{G}$ s'il n'existe aucun trajet entre une extrémité de e et p qui commence après la première apparition de e dans $\mathcal{G}$ et se termine avant t .*

Preuve. Par l'absurde, supposons qu'il existe un algorithme glouton $\mathcal{A}$ qui satisfait la spécification de construction de l'empreinte globale sur $\mathcal{CTT}$, un $\mathcal{GVT} \mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta, \phi)$ de $\mathcal{CTT}$, une arête $\hat{e} \in E$ qui n'est pas une coupe de U_g^ω , un processus $\hat{p} \in V$ et un temps $\hat{t} \in \mathcal{T}$ tel que $\hat{e}$ apparaît dans le graphe émis en sortie par $\hat{p}$ durant l'exécution de $\mathcal{A}$ sur $\mathcal{G}$ au temps $\hat{t}$ et qu'il n'existe aucun trajet de l'une des extrémités de $\hat{e}$ vers $\hat{p}$ qui commence après la première apparition de $\hat{e}$ dans $\mathcal{G}$ et se termine avant $\hat{t}$.

Considérons le $\mathcal{GVT} \mathcal{G}' = (V, E, \mathcal{T}, \rho', \zeta, \phi)$ avec :

$$\forall e \in E, \forall t \in \mathcal{T}, \rho'(e, t) = \begin{cases} 0 & \text{si } e = \hat{e} \\ \rho(e, t) & \text{sinon} \end{cases}$$

Notez que, par l'hypothèse selon laquelle $\hat{e}$ n'est pas une coupe de U_g^ω , $\mathcal{G}'$ appartient à la famille $\mathcal{CTT}$. Par conséquent, en raison de la construction de $\mathcal{G}$ et du déterminisme de l'algorithme $\mathcal{A}$, le processus $\hat{p}$ reçoit exactement les mêmes messages avant le temps $\hat{t}$ dans l'exécution de $\mathcal{A}$ sur $\mathcal{G}$ et sur $\mathcal{G}'$ (l'hypothèse de l'existence d'un trajet après la première apparition de $\hat{e}$ dans $\mathcal{G}$ nous garantit que le fait de retirer $\hat{e}$

dans $\mathcal{G}'$ n'est pas détecté par $\hat{p}$ avant le temps $\hat{t}$). En d'autres termes, le processus $\hat{p}$ ne peut pas distinguer l'exécution de $\mathcal{A}$ sur $\mathcal{G}$ et sur $\mathcal{G}'$ avant le temps $\hat{t}$. De ce fait, $\hat{e}$ apparaît dans le graphe émis en sortie par $\hat{p}$ durant l'exécution de $\mathcal{A}$ sur $\mathcal{G}'$ au temps $\hat{t}$. Puisque l'algorithme $\mathcal{A}$ est un algorithme glouton, alors cette arête ne disparaît jamais de la sortie de $\hat{p}$, après le temps $\hat{t}$, durant l'exécution de $\mathcal{A}$ sur $\mathcal{G}'$. Cela est contradictoire avec le fait que $\mathcal{A}$ satisfait la spécification de la construction de l'empreinte globale sur $\mathcal{CCT}$ puisque $\hat{e}$ n'appartient pas à $U_{\mathcal{G}'}$ et prouve par la même occasion ce lemme. $\square$

Nous allons maintenant prouver l'optimalité de notre algorithme.

Théorème 6.2 *Pour tout algorithme glouton $\mathcal{A}$ qui satisfait la spécification du problème de construction de l'empreinte globale sur $\mathcal{CCT}$, il existe un $\mathcal{GVT}$ $\mathcal{G}$ de $\mathcal{CCT}$ tel que le temps de convergence de $\mathcal{A}$ est d'au moins $\text{diam}(U_{\mathcal{G}}^{\omega})$ rondes.*

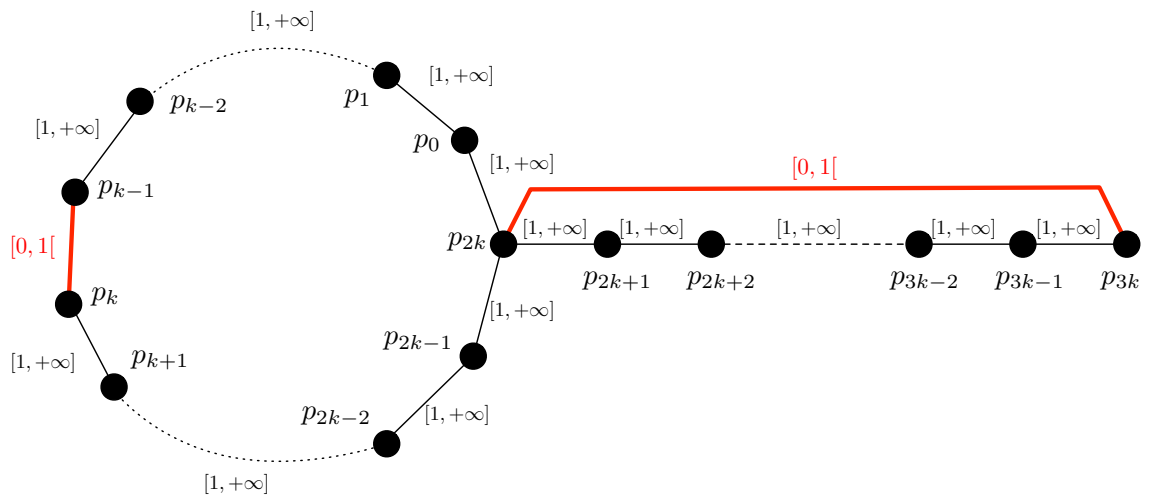
Preuve. Soit $\mathcal{A}$ un algorithme glouton qui satisfait la spécification du calcul de l'empreinte globale sur $\mathcal{CCT}$. Définissons la famille de $\mathcal{GVT}$ ($\mathcal{G}_{k \in \mathbb{N}^*}$ décrite par la Figure 6.1. Notez que, pour tout $k \in \mathbb{N}^*$, nous avons $\text{diam}(U_{\mathcal{G}_k}^{\omega}) = 2k$ (et $\text{diam}(U_{\mathcal{G}_k}) < \text{diam}(U_{\mathcal{G}_k}^{\omega})$ puisque $\text{diam}(U_{\mathcal{G}_k}) = k + 1$). Comme le graphe $U_{\mathcal{G}_k}^{\omega}$ est connexe, $\mathcal{G}_k$ appartient à $\mathcal{CCT}$. Par construction de $\mathcal{G}_k$, le temps de départ de l'exécution de $\mathcal{A}$ sur $\mathcal{G}_k$ est 1 pour tout $k \in \mathbb{N}^*$ (rappelons que $EPN_{\mathcal{UG}}(\mathcal{G}) = \{E\}$). Notez qu'en raison du choix de la fonction de latence, toute ronde de communication de l'exécution de $\mathcal{A}$ sur $\mathcal{G}_k$ prend exactement une unité de temps.

Considérons l'exécution e_k de $\mathcal{A}$ sur $\mathcal{G}_k$ pour $k \in \mathbb{N}^*$. D'après le lemme 6.3, nous savons que l'arête $\{p_{k-1}, p_k\}$ ne peut apparaître dans le graphe émis en sortie par p_{3k} dans e_k avant qu'il n'existe au moins un trajet de p_{k-1} ou p_k vers p_{3k} . Notez que la construction de $\mathcal{G}_k$ implique qu'un tel trajet (après le temps 1) nécessite au moins $2k$ rondes (la longueur du chemin de p_{k-1} ou p_k à p_{3k} puisque $\mathcal{G}_k$ est statique après le temps 1). Puisque l'arête $\{p_{k-1}, p_k\}$ doit finalement apparaître dans la sortie émise par tout processus de e_k par hypothèse sur $\mathcal{A}$, nous obtenons que le temps de convergence de $\mathcal{A}$ est d'au moins $\text{diam}(U_{\mathcal{G}_k}^{\omega})$ rondes. $\square$

6.4 Conclusion

Dans ce chapitre, nous avons proposé une nouvelle mesure de complexité en temps pour les algorithmes s'exécutant sur des systèmes dynamique asynchrones. Cette mesure est une extension de la mesure de complexité classiquement utilisée dans les systèmes distribués statiques par passage de messages. L'idée intuitive est de négliger les délais introduits par la dynamique du système au même titre que ceux introduits par l'asynchronisme des communications. Cela permet de capturer la longueur des chaînes de dépendance causale qui est la mesure la plus équitable pour comparer les performances de deux algorithmes.

Nous avons ensuite à travers l'exemple de la construction de l'empreinte globale illustré cette mesure.

FIGURE 6.1 – Une illustration de famille de $\mathcal{GVT}$ de la preuve du Théorème 6.2.

Chapitre 7

Construction de l'ensemble dominant ultime

Sommaire

7.1	L'algorithme EDMU	111
7.2	Conclusion	113

Dans ce chapitre, nous nous intéressons aux graphes variant dans le temps de la famille $\mathcal{CTT}$ dont l'empreinte globale admet un ensemble dominant minimal robuste (voir la définition 5.4 page 91 pour l'ensemble dominant minimal robuste).

Rappelons que nous avons montré dans le chapitre 5 que cela était une condition nécessaire à l'existence d'un algorithme déterministe pour le problème de la construction de l'ensemble dominant minimal ultime (voir la définition 4.9 page 73 pour l'ensemble dominant minimal ultime).

Dans ce contexte, nous proposons un algorithme qui répond à la spécification du problème de l'ensemble dominant minimal ultime pour la famille $\mathcal{CTT}$ de $\mathcal{GVT}$. De ce fait, nous obtenons que la condition nécessaire du chapitre 5 est aussi suffisante pour la famille $\mathcal{CTT}$.

7.1 L'algorithme EDMU

L'algorithme que nous proposons pour le calcul de l'ensemble dominant minimal ultime de tout graphe de $\mathcal{CTT}$ qui admet au moins un ensemble dominant minimal robuste repose sur deux phases. La première étape consiste à calculer l'empreinte globale du $\mathcal{GVT}$ à l'aide de l'algorithme présenté dans la section 6.3.

Une fois l'empreinte globale du $\mathcal{GVT}$ calculée par un processus, il est alors possible pour ce dernier de déterminer s'il appartient ou non à l'ensemble dominant ultime calculé par l'algorithme. En effet, il suffit que ce processus énumère localement et dans un ordre déterministe (par exemple, en se basant sur les identifiants des processus) tous les ensembles dominant minimaux de l'empreinte globale et de choisir le premier qui soit robuste. Il ne reste au processus qu'à vérifier s'il fait partie de

cet ensemble dominant ou non. En effet, le Lemme 5.7 permet d'affirmer que cet ensemble dominant robuste est un ensemble dominant minimal ultime. De plus, l'ordre déterministe commun à tous les processus nous assure que tous les processus calculent le même ensemble dominant minimal robuste.

Étant donné que la détection de la convergence du calcul de l'empreinte globale par l'Algorithme 1 est impossible, ces deux phases ne peuvent pas être séquentielles.

Notre algorithme procède comme suit. L'algorithme 2 exécute l'algorithme 1 et chaque mise à jour du graphe local calculé par l'algorithme 1 entraîne systématiquement la recherche d'un ensemble dominant minimal robuste sur ce dernier. Par conséquent, lorsque chaque processus du système a calculé l'empreinte globale du $\mathcal{GVT}$, si ce dernier admet un ensemble dominant minimal robuste alors l'énumération de tous les ensembles dominant minimaux de l'empreinte globale permettra d'en trouver un.

Le pseudo-code de cet algorithme est présenté dans l'Algorithme 2.

Théorème 7.1 *Pour tout ensemble $\mathcal{F}$ ne contenant que des graphes (statiques) qui admettent un ensemble dominant minimal robuste, l'Algorithme 2 satisfait la spécification de l'ensemble dominant minimal ultime pour la sous-famille de $\mathcal{GVT}$ $\mathcal{CTT}|_{\mathcal{F}}$.*

Preuve. Soit $\mathcal{F}$ un ensemble de graphes (statiques) qui admettent un ensemble dominant minimal robuste. Nous considérons une exécution de l'Algorithme 2 sur un $\mathcal{GVT}$ $\mathcal{G} = (V, E, \mathcal{T}, \rho, \zeta, \phi)$ de $\mathcal{CTT}|_{\mathcal{F}}$.

Nous allons prouver dans un premier temps que l'exécution de l'algorithme 2 ne bloque pas la convergence de l'algorithme 1 lancé lors de l'initialisation de l'algorithme 2.

Comme le nombre de sommets que peut contenir V_p est borné par le nombre de processus de V , le nombre de sous-ensemble de V_p énuméré à chaque mise à jour de g_p est borné. En d'autres termes, la boucle qui consiste à explorer tous les sous-ensembles de V_p s'exécute en un temps fini. Cela signifie que les calculs effectués par p à chaque mise à jour de g_p n'empêcherons pas la convergence de l'Algorithme 1.

Nous allons maintenant prouver que lorsqu'un processus p de $\mathcal{G}$ a fini de calculer l'empreinte globale alors, après un temps fini, le processus p fini par fournir en sortie une valeur booléenne stable et que l'ensemble des processus fournissant vrai est un ensemble dominant minimal ultime du $\mathcal{GVT}$ $\mathcal{G}$.

Par le Théorème 6.1, nous savons que l'algorithme 1 satisfait la spécification de construction de l'empreinte globale pour tout $\mathcal{GVT}$ de la famille $\mathcal{CCT}$. De ce fait, dans toute exécution de l'algorithme 1 par les processus sur $\mathcal{G}$, il existe un temps t_x à partir duquel le graphe g_p aura convergé vers l'empreinte globale pour tout processus p . D'après l'algorithme 2, tout processus qui met à jour sa copie de l'empreinte globale procède de manière déterministe à la recherche d'un ensemble dominant minimal robuste sur ce graphe. Par l'hypothèse sur $\mathcal{G}$, l'empreinte globale $U_{\mathcal{G}}$ de $\mathcal{G}$ admet au moins un ensemble dominant minimal robuste. Étant donné que tous les processus

parcourent de la même manière l'ensemble V , à la recherche d'un ensemble dominant minimal robuste, alors ils calculent le même ensemble M_p . De plus, d'après le Lemme 5.7, M est aussi un ensemble dominant minimal ultime. L'algorithme 2 satisfait donc la spécification du calcul de l'ensemble dominant minimal ultime pour toute exécution sur un $\mathcal{GVT}$ de $\mathcal{CTT}|_{\mathcal{F}}$. $\square$

Dans l'algorithme 2, on peut constater que les communications sont uniquement dues à l'exécution de l'algorithme 1. En effet, le reste de l'algorithme se contente de chercher un ensemble dominant minimal robuste dans la structure calculée par l'algorithme 1. Ceci a pour conséquence que la complexité en temps de l'algorithme 2 est la même que celle de l'algorithme 1 car on néglige le temps de calcul local des processus dans cette mesure. Cependant, signalons que cette recherche en local d'un ensemble dominant minimal robuste est exponentielle.

Lemme 7.1 *Pour tout ensemble $\mathcal{F}$ de graphes (statiques) qui admettent un ensemble dominant minimal robuste, l'algorithme 2 converge vers un ensemble dominant minimal ultime dans tout $\mathcal{GVT}$ $\mathcal{G}$ de la sous-famille $\mathcal{CTT}|_{\mathcal{F}}$ en $\text{diam}(U_{\mathcal{G}}^{\omega})$ rondes.*

Preuve. D'après l'algorithme 2, les communications sont exclusivement à l'initiative de l'algorithme 1. D'après le Théorème 6.1, la complexité en temps de l'algorithme 1 exécuté sur un $\mathcal{GVT}$ $\mathcal{G}$ de $\mathcal{CTT}$ est de $\text{diam}(U_{\mathcal{G}}^{\omega})$ rondes. Puisque $\mathcal{CTT}|_{\mathcal{F}}$ est une sous-famille de $\mathcal{CTT}$, la complexité de l'exécution de l'algorithme 2 dans tout $\mathcal{GVT}$ $\mathcal{G}$ est de $\text{diam}(U_{\mathcal{G}}^{\omega})$ rondes. $\square$

7.2 Conclusion

Dans le chapitre 5, nous avons donnée la condition nécessaire pour la construction de l'ensemble dominant minimal ultime par un algorithme sur tout graphe variant dans le temps de la famille $\mathcal{CTT}$: l'empreinte du $\mathcal{GVT}$ doit admettre un ensemble dominant minimal robuste.

Ce chapitre a montré que cette condition est suffisante. Nous avons en effet fourni un algorithme qui calcule un ensemble dominant minimal ultime pour tout $\mathcal{GVT}$ de $\mathcal{CTT}$ dont l'empreinte admet au moins un ensemble dominant minimal robuste.

Bien que la complexité de cet algorithme soit linéaire en fonction du diamètre de l'empreinte ultime du $\mathcal{GVT}$, il utilise une énumération de toutes les parties de l'ensemble des sommets, ce qui conduit à un temps de calcul local exponentiel.

Variables :

$g_p = (V_p, E_p)$: l'empreinte construite par p
 M_p : l'ensemble dominant minimal construit par p
 $\mathcal{N}_p$: l'ensemble des voisins de p

Variables de sortie :

D_p : variable booléenne représentant l'appartenance de p à l'ensemble dominant

Initialisation :

$g_p \leftarrow (\emptyset, \emptyset);$
 $M_p \leftarrow \emptyset;$
 $\mathcal{N}_p \leftarrow \emptyset;$
 $D_p \leftarrow faux;$
 Exécuter l'algorithme 1

Fonctions :

Fonction booléenne $\mathcal{SMDS}$ qui prend en paramètre un ensemble M et un graphe G
 si M est un ensemble dominant minimal robuste de G
 la fonction $\mathcal{SMDS}$ renvoie “vrai”
 sinon $\mathcal{SMDS}$ renvoie “faux”

Lors de la mise à jour de g_p par l'Algorithme 1 :

$M_p \leftarrow \emptyset;$
 $D_p \leftarrow faux;$
 Pour tout M sous ensemble de V_p (énuméré dans un ordre déterministe et commun à tous les processus)
 si ($\mathcal{SMDS}(M, g_p)$) **alors**
 $M_p \leftarrow M;$
 $D_p \leftarrow (p \in M_p);$

Algorithme 2 – L'algorithme EDMU permet de construire un ensemble dominant minimal ultime de tout $\mathcal{GVT}$ de $\mathcal{CTT}$ dont l'empreinte admet un ensemble dominant minimal robuste.

Chapitre 8

Conclusion

Dans cette thèse, nous avons étudié la faisabilité de certains problèmes traditionnels des systèmes distribués dans les systèmes hautement dynamiques.

La difficulté de ce type d'environnement réside sur l'impossibilité de prédire la présence de ses composants (processus, liens, utilisateurs, etc.). En effet, les différentes composantes d'un système dynamique sont susceptibles de rejoindre, de quitter ou de se déplacer dans le système à tout moment.

Après avoir décrit notre modèle de système distribué dynamique par un graphe variant dans le temps dans la première partie de cette thèse, nous avons comparé les différentes approches existantes permettant de redéfinir un problème distribué de couverture pour l'adapter à un système dynamique. Ensuite, nous proposons l'approche ultime (qui consiste à satisfaire ultimement la propriété d'un problème). Contrairement aux autres approches (permanente, évolutive, temporelle), elle est compatible avec les familles de $\mathcal{GVT}$ asynchrones, de durée de vie non bornée et hautement dynamiques, sous les hypothèses les plus faibles de connexité représenté par la famille de graphe connexe à travers le temps. Nous avons proposé un outil générique qui permet de faciliter les preuves des résultats d'impossibilité dans un graphe variant dans le temps. Nous avons ensuite utilisé ce résultat pour prouver plusieurs résultats d'impossibilité sur la famille de $\mathcal{GVT}$ connexe à travers le temps.

Sur les $\mathcal{GVT}$ connexe à travers le temps, nous avons notamment prouvé les trois résultats d'impossibilité suivant :

- il est impossible de calculer de façon déterministe le graphe sous-jacent ultime (c'est-à-dire, l'ensemble des arêtes qui sont infiniment souvent présentes) dès lors que le graphe sous-jacent n'est pas un graphe acyclique (arbre) ;
- il est impossible de construire de façon déterministe un couplage maximal ultime (défini comme un couplage maximal du graphe sous-jacent ultime) sur un $\mathcal{GVT}$ à moins que son empreinte ultime admette un mariage maximal ne contenant que des isthmes.
- il est impossible de calculer de manière déterministe un ensemble dominant minimal ultime (défini comme un ensemble dominant minimal du graphe sous-jacent ultime) sur un $\mathcal{GVT}$ si son graphe sous-jacent n'admet par un

ensemble dominant minimal robuste (défini comme un ensemble dominant minimal qui a la propriété d'être un ensemble dominant minimal de tout graphe partiel connexe du graphe sous-jacent) ;

Nous avons ensuite proposé d'étendre la mesure de complexité classique en temps de rondes des systèmes statiques sur des système distribués dynamiques. À travers cette nouvelle mesure de complexité, nous avons prouvé l'optimalité de notre algorithme de construction du graphe sous-jacent sur un $\mathcal{GVT}$.

Finalement, nous avons prouvé la condition nécessaire et suffisante pour résoudre le problème de l'ensemble dominant minimal ultime.

Cette thèse ouvre plusieurs questions dans le domaine des systèmes répartis hautement dynamiques. Par exemple, il serait intéressant de caractériser par exemple l'ensemble des topologies qui correspondent aux conditions nécessaires à la fois pour le problème du couplage maximal ultime et le problème de l'ensemble dominant minimal ultime. Il serait également intéressant d'étudier si notre résultat général est applicable à d'autres problèmes y compris des problèmes non couvrants.

D'un autre côté, il serait utile d'étudier les conditions nécessaires et suffisantes pour résoudre encore une fois le couplage maximal ultime et le problème de l'ensemble dominant minimal ultime sur des $\mathcal{GVT}$ connexe à travers le temps tout en tolérant des fautes (c'est-à-dire, assurer que certaines propriétés sont garanties même si certaines fautes touchent le système).

Index

Symboles

Δ -Récurent 51

A

Agent mobile 26
Apparition 35
Arbre 19
Arbre couvrant 19
Arêtes 18
 consécutives 18
 multiples 18

B

Boucle 18

C

Chemin 18
Chemin temporel 38
Configuration 59
Connecté 18
Connexe 18
Constamment connexe 52
Couplage 63
Couplage évolutif 69
Couplage maximal 63
Couplage maximal temporel ... 72
couplage maximal ultime 75
couplage permanent 66
Couplage temporel 71
couplage ultime 75

D

Date de fin 38
Date de départ 38
Dates caractéristiques 35

Dates caractéristiques d'une arête .
35

Délai 102
Diamètre 18
Diffusion 24
disparition 35
Distance 18
 temporelle 40
 topologique 39
Distance sur un ensemble 82
Distance ultramétrique 82

E

Empreinte globale 36
Empreinte Ultime 37
Ensemble dominant 63
Ensemble dominant ultime 73
Ensemble dominant évolutif ... 69
Ensemble dominant minimal ... 64
Ensemble dominant minimal ultime
73
Ensemble dominant permanent 66
espace métrique complet 83
Événement topologique 35
Exécution 59
Exploration 26
Extrémité 18

F

Faute 17
Fonction de latence des arêtes . 35
la fonction de latence des sommets
35
Fonction de présence des arêtes 35
Forêt 19

G

Graphe biparti	18
Graphe complet	18
Graphe non-orienté	18
Graphe partiel	19
Graphe simple	18
Graphe sous-jacent	36
Graphe Sous-Jacent Ultime	37
Graphe statique	18
Graphe Variant dans le Temps	34
Graphe variant dans le temps .	35

I

Isthme	94
--------------	----

L

Lien de communication	15
Longueur d'un chemin	18

M

Modèle à passage de messages .	54
Modèle à état	56
Modèle de registre	56
Modèle de registre partagé	56

P

Périodique	51
Présence	35

R

Ronde	103
-------------	-----

S

Send_retry	58
Sommet	18
Sous graphe	19
Sous graphe couvrant	19
Sous graphe temporel	39
Suite d'évolutions	36
Système distribué	15
Système distribué dynamique ..	16

T

Temps de convergence	103
Temps de départ	103
Temps de vie	35
Trajet	38
Trajet court	39
Trajet rapide	40
Trajet strict	39
Trajet terminant au plus tôt ...	40

V

Voisins	16
---------------	----

Bibliographie

- [ABL05] Mansoor Alicherry, Randeep Bhatia, and Li Erran Li. Joint channel assignment and routing for throughput optimization in multi-radio wireless mesh networks. In *Proceedings of the 11th annual international conference on Mobile computing and networking*, pages 58–72. ACM, 2005.
- [AD09] Matthew Andrews and Michael Dinitz. Maximizing capacity in arbitrary wireless networks in the sinr model : Complexity and game theory. In *INFOCOM 2009, IEEE*, pages 1332–1340. IEEE, 2009.
- [AE84] Baruch Awerbuch and Shimon Even. Efficient and reliable broadcast is achievable in an eventually connected network. In *PODC*, pages 278–281, 1984.
- [AEK⁺09] Chen Avin, Yuval Emek, Erez Kantor, Zvi Lotker, David Peleg, and Liam Roditty. Sinr diagrams : towards algorithmically usable sinr models of wireless networks. In *Proceedings of the 28th ACM symposium on Principles of distributed computing*, pages 200–209. ACM, 2009.
- [AH00] Stephen Alstrup and Jacob Holm. Improved algorithms for finding level ancestors in dynamic trees. In *Automata, Languages and Programming*, pages 73–84. Springer, 2000.
- [AKL08] Chen Avin, Michal Koucký, and Zvi Lotker. How to explore a fast-changing world (cover time of a simple random walk on evolving graphs). In *Automata, languages and programming*, pages 121–132. Springer, 2008.
- [AKM⁺12] Aris Anagnostopoulos, Ravi Kumar, Mohammad Mahdian, Eli Upfal, and Fabio Vandin. Algorithms on evolving graphs. In *ITCS*, pages 149–160, 2012.
- [AOPS02] Ravindra K Ahuja, James B Orlin, Stefano Pallottino, and Maria G Scutella. Dynamic shortest paths minimizing travel times and costs. *Networks*, 41(4) :197–205, 2002.
- [AW04] Hagit Attiya and Jennifer Welch. *Distributed Computing : Fundamentals, Simulations and Advanced Topics (2nd edition)*. John Wiley Interscience, 2004.
- [BA99] Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *science*, 286(5439) :509–512, 1999.

- [BCC⁺14] Matthieu Barjon, Arnaud Casteigts, Serge Chaumette, Colette Johnen, and Yessin M Neggaz. Un algorithme de test pour la connexité temporelle des graphes dynamiques de faible densité. *arXiv preprint arXiv :1405.0170*, 2014.
- [BCF11] Hervé Baumann, Pierluigi Crescenzi, and Pierre Fraigniaud. Parsimonious flooding in dynamic graphs. *Distributed Computing*, 24(1) :31–44, 2011.
- [BDTC05] Jeremy Blum, Min Ding, Andrew Thaeler, and Xiuzhen Cheng. Connected dominating set in sensor networks and manets. In Ding-Zhu Du and Panos M. Pardalos, editors, *Handbook of Combinatorial Optimization*, pages 329–369. Springer US, 2005.
- [Ber96] Kenneth Berman. Vulnerability of scheduled networks and a generalization of menger’s theorem. *Networks*, 28(3) :125–134, 1996.
- [BFFS03] Lali Barriere, Paola Flocchin, Pierre Fraigniau, and Nicola Santor. Can we elect if we cannot compare? In *Proceedings of the fifteenth annual ACM symposium on Parallel algorithms and architectures*, pages 324–332. ACM, 2003.
- [BFNO03] Lali Barriere, Pierre Fraigniaud, Lata Narayanan, and Jaroslav Opatrny. Robust position-based routing in wireless ad hoc networks with irregular transmission ranges. *Wireless Communications and Mobile Computing*, 3(2) :141–153, 2003.
- [BRS10] Douglas M Blough, Giovanni Resta, and Paolo Santi. Approximation algorithms for wireless link scheduling with sinr-based interference. *Networking, IEEE/ACM Transactions on*, 18(6) :1701–1712, 2010.
- [BSDKP15a] Nicolas Braud-Santoni, Swan Dubois, Mohamed-Hamza Kaaouachi, and Franck Petit. A generic framework for impossibility results in time-varying graphs. In *17th Workshop on Advances on Parallel and Distributed Processing Symposium (APDCM’15)*, pages 483–489, 2015.
- [BSDKP15b] Nicolas Braud-Santoni, Swan Dubois, Mohamed-Hamza Kaaouachi, and Franck Petit. The next 700 impossibility results in time-varying graphs. *International Journal of Networking and Computing (IJNC)*, page to appear, 2015.
- [BXFJ03] Binh-Minh Bui-Xuan, Afonso Ferreira, and Aubin Jarry. Computing shortest, fastest, and foremost journeys in dynamic networks. *International Journal of Foundations of Computer Science*, 14(02) :267–285, 2003.
- [BYGI92] Reuven Bar-Yehuda, Oded Goldreich, and Alon Itai. On the time-complexity of broadcast in multi-hop radio networks : An exponential gap between determinism and randomization. *Journal of Computer and System Sciences*, 45(1) :104–126, 1992.
- [CCD⁺15] Andrea Clementi, Pierluigi Crescenzi, Carola Doerr, Pierre Fraigniaud, Francesco Pasquale, and Riccardo Silvestri. Rumor spreading in random evolving graphs. *Random Structures & Algorithms*, 2015.

- [CCF09] Arnaud Casteigts, Serge Chaumette, and Afonso Ferreira. Characterizing topological assumptions of distributed algorithms in dynamic networks. In *Proc. of 16th Intl. Conference on Structural Information and Communication Complexity (SIROCCO'09)*, volume 5869 of *Lecture Notes in Computer Science*, pages 126–140, Piran, Slovenia, May 2009. Springer-Verlag.
- [CF04] Colin Cooper and Alan Frieze. Crawling on simple models of web graphs. *Internet Mathematics*, 1(1) :57–90, 2004.
- [CF13] Arnaud Casteigts and Paola Flocchini. Deterministic algorithms in dynamic networks : Problems, analysis, and algorithmic tools. Technical report, DRDC 2013-020, 2013.
- [CFMS10] Arnaud Casteigts, Paola Flocchini, Bernard Mans, and Nicola Santoro. Deterministic computations in time-varying graphs : Broadcasting under unstructured mobility. In CristianS. Calude and Vladimiro Sassone, editors, *Theoretical Computer Science*, volume 323 of *IFIP Advances in Information and Communication Technology*, pages 111–124. Springer Berlin Heidelberg, 2010.
- [CFMS14] Arnaud Casteigts, Paola Flocchini, Bernard Mans, and Nicola Santoro. Measuring temporal lags in delay-tolerant networks. *IEEE Transactions on Computers*, 63(2), 2014.
- [CFMS15] Arnaud Casteigts, Paola Flocchini, Bernard Mans, and Nicola Santoro. Shortest, fastest, and foremost broadcast in dynamic networks. *Int. Journal of Foundations of Computer Science*, 26(4) :499–522, 2015.
- [CFQS12] Arnaud Casteigts, Paola Flocchini, Walter Quattrociocchi, and Nicolas Santoro. Time-varying graphs and dynamic networks. *IJPEDS*, 27(5) :387–408, 2012.
- [CGMO06] Jérémie Chalopin, Emmanuel Godard, Yves Métivier, and Rodrigue Ossamy. Mobile agent algorithms versus message passing algorithms. In *Principles of Distributed Systems*, pages 187–201. Springer, 2006.
- [CHC⁺07] Augustin Chaintreau, Pan Hui, Jon Crowcroft, Christophe Diot, Richard Gass, and James Scott. Impact of human mobility on opportunistic forwarding algorithms. *Mobile Computing, IEEE Transactions on*, 6(6) :606–620, 2007.
- [Chr76] Nicos Christofides. Worst-case analysis of a new heuristic for the travelling salesman problem. Technical report, DTIC Document, 1976.
- [CKZ04] Colin Cooper, Ralf Klasing, and Michele Zito. Dominating sets in web graphs. In *Algorithms and Models for the Web-Graph*, pages 31–43. Springer, 2004.
- [CKZ05] Colin Cooper, Ralf Klasing, and Michele Zito. Lower bounds and algorithms for dominating sets in web graphs. *Internet Mathematics*, 2(3) :275–300, 2005.
- [CMM⁺08] Andrea EF Clementi, Claudio Macci, Angelo Monti, Francesco Pasquale, and Riccardo Silvestri. Flooding time in edge-markovian dyna-

- mic graphs. In *Proceedings of the twenty-seventh ACM symposium on Principles of distributed computing*, pages 213–222. ACM, 2008.
- [CMM11] Arnaud Casteigts, Bernard Mans, and Luke Mathieson. On the feasibility of maintenance algorithms in dynamic graphs. Technical report, arXiv – abs/1107.2722, 2011.
- [DFGV03] Henri Dubois-Ferriere, Matthias Grossglauser, and Martin Vetterli. Age matters : efficient route discovery in mobile ad hoc networks using encounter ages. In *Proceedings of the 4th ACM international symposium on Mobile ad hoc networking & computing*, pages 257–266. ACM, 2003.
- [DFPS02] Stefan Dobrev, Paola Flocchini, Giuseppe Prencipe, and Nicola Santoro. Searching for a black hole in arbitrary networks : optimal mobile agent protocols. In *Proceedings of the twenty-first annual symposium on Principles of distributed computing*, pages 153–162. ACM, 2002.
- [DFPS06] Stefan Dobrev, Paola Flocchini, Giuseppe Prencipe, and Nicola Santoro. Searching for a black hole in arbitrary networks : Optimal mobile agents protocols. *Distributed Computing*, 19(1) :1–99999, 2006.
- [DFSY07] Shantanu Das, Paola Flocchini, Nicola Santoro, and Masafumi Yamashita. Fault-tolerant simulation of message-passing algorithms by mobile agents. In *Structural Information and Communication Complexity*, pages 289–303. Springer, 2007.
- [Die91] Paul F Dietz. Finding level-ancestors in dynamic trees. In *Algorithms and Data Structures*, pages 32–40. Springer, 1991.
- [Dij74] Edsger W. Dijkstra. Self-stabilizing systems in spite of distributed control. *Communications of ACM*, 17(11) :643–644, 1974.
- [DIM93] Shlomi Dolev, Amos Israeli, and Shlomo Moran. Self-stabilization of dynamic systems assuming only read/write atomicity. *Distributed Computing*, 7(1) :3–16, 1993.
- [DKP15a] Swan Dubois, Mohamed-Hamza Kaaouachi, and Franck Petit. Dynamisme et domination. In *17èmes Rencontres Francophones pour les Aspects Algorithmiques des Télécommunications (AlgoTel’15)*, pages 1–4, 2015.
- [DKP15b] Swan Dubois, Mohamed-Hamza Kaaouachi, and Franck Petit. Enabling minimal dominating set in highly dynamic distributed systems. In *17th International Symposium on Stabilization, Safety, and Security of Distributed Systems (SSS’15)*, pages 51–66, 2015.
- [DMS11] Josep Díaz, Dieter Mitsche, and Paolo Santi. Theoretical aspects of graph models for manets. In *Theoretical aspects of distributed computing in sensor networks*, pages 161–190. Springer, 2011.
- [DRm14] Ilcinkas David, Klasing Ralf, and Wade Ahmed mouhamadou. Exploration of constantly connected dynamic graphs based on cactuses. In *SIROCCO*, pages 250–262, 2014.

- [ER60] Paul Erdős and Alfréd Rényi. On the evolution of random graphs. *PUBLICATION OF THE MATHEMATICAL INSTITUTE OF THE HUNGARIAN ACADEMY OF SCIENCES*, 5 :17–61, 1960.
- [Fal03] Kevin Fall. A delay-tolerant network architecture for challenged internets. In *SIGCOMM – CATAPCC*, pages 27–34, 2003.
- [Fer02] Afonso Ferreira. On models and algorithms for dynamic communication networks : The case for evolving graphs. In *In Proc. ALGOTEL*, 2002.
- [Fer04] Afonso Ferreira. Building a reference combinatorial model for manets. *Network*, 18(5) :24–29, 2004.
- [FGM07] Afonso Ferreira, Alfredo Goldman, and Julian Monteiro. On the evaluation of shortest journeys in dynamic networks. In *NCA*, pages 3–10, 2007.
- [FKMS09] Paola Flocchini, Matthew Kellett, Peter Mason, and Nicola Santoro. Map construction and exploration by mobile agents scattered in a dangerous network. In *Parallel & Distributed Processing, 2009. IPDPS 2009. IEEE International Symposium on*, pages 1–10. IEEE, 2009.
- [FKMS10] Paola Flocchini, Matthew Kellett, Peter C Mason, and Nicola Santoro. Mapping an unfriendly subway system. In *Fun with Algorithms*, pages 190–201. Springer, 2010.
- [FKMS12a] Paola Flocchini, Matthew Kellett, Peter C Mason, and Nicola Santoro. Fault-tolerant exploration of an unknown dangerous graph by scattered agents. In *Stabilization, Safety, and Security of Distributed Systems*, pages 299–313. Springer, 2012.
- [FKMS12b] Paola Flocchini, Matthew Kellett, Peter C Mason, and Nicola Santoro. Finding good coffee in paris. In *Fun with Algorithms*, pages 154–165. Springer, 2012.
- [FKMS12c] Paola Flocchini, Matthew Kellett, Peter C Mason, and Nicola Santoro. Searching for black holes in subways. *Theory of Computing Systems*, 50(1) :158–184, 2012.
- [FMS09] Paola Flocchini, Bernard Mans, and Nicola Santoro. Exploration of periodically varying graphs. In Yingfei Dong, Ding-Zhu Du, and Oscar Ibarra, editors, *Algorithms and Computation*, volume 5878 of *Lecture Notes in Computer Science*, pages 534–543. Springer Berlin Heidelberg, 2009.
- [GAS11] Fabíola Greve, Luciana Arantes, and Pierre Sens. What model and what conditions to implement unreliable failure detectors in dynamic networks ? In *Proceedings of the 3rd International Workshop on Theoretical Aspects of Dynamic Distributed Systems*, pages 13–17. ACM, 2011.
- [Gil61] Edward N Gilbert. Random plane networks. *Journal of the Society for Industrial & Applied Mathematics*, 9(4) :533–543, 1961.

- [GK99] Piyush Gupta and Panganamala R Kumar. Critical power for asymptotic connectivity in wireless networks. In *Stochastic analysis, control, optimization and applications*, pages 547–566. Springer, 1999.
- [GK00] Piyush Gupta and Panganamala R Kumar. The capacity of wireless networks. *Information Theory, IEEE Transactions on*, 46(2) :388–404, 2000.
- [GV03] Matthias Grossglauser and Martin Vetterli. Locating nodes with ease : Last encounter routing in ad hoc networks through mobility diffusion. In *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*, volume 3, pages 1954–1964. IEEE, 2003.
- [HG97] Frank Harary and Gopal Gupta. Dynamic graph models. *Mathematical and Computer Modelling*, 25(7) :79–87, 1997.
- [HH00] Frank Harary and Teresa W Haynes. Double domination in graphs. *Ars Combinatoria*, 55 :201–214, 2000.
- [Hol05] Petter Holme. Network reachability of real-world contact sequences. *Physical Review E*, 71(4) :046119, 2005.
- [IP11] R Krishnam Raju Indukuri and Suresh Varma Penumathsa. Dominating sets and spanning tree based clustering algorithms for mobile ad hoc networks. *IJACSA-International Journal of Advanced Computer Science and Applications*, 2(2), 2011.
- [IW11] David Ilcinkas and Ahmed Mouhamadou Wade. On the power of waiting when exploring public transportation systems. In *Principles of Distributed Systems*, pages 451–464. Springer, 2011.
- [IW13] David Ilcinkas and Ahmed Mouhamadou Wade. Exploration of the t-interval-connected dynamic graphs : the case of the ring. In *Structural Information and Communication Complexity*, pages 13–23. Springer, 2013.
- [JFP04] Sushant Jain, Kevin Fall, and Rabin Patra. Routing in a delay tolerant network. *SIGCOMM Comput. Commun. Rev.*, 34(4) :145–158, August 2004.
- [JLSW07] Evan PC Jones, Lily Li, Jakub K Schmidtke, and Paul AS Ward. Practical routing in delay-tolerant networks. *Mobile Computing, IEEE Transactions on*, 6(8) :943–959, 2007.
- [JPPQ05] Kamal Jain, Jitendra Padhye, Venkata N Padmanabhan, and Lili Qiu. Impact of interference on multi-hop wireless network performance. *Wireless networks*, 11(4) :471–487, 2005.
- [Kel12] Matthew Kellett. *Black Hole Search in the Network and Subway Models*. PhD thesis, Carleton University, 2012.
- [KK02] David Kempe and Jon Kleinberg. Protocols and impossibility results for gossip-based communication mechanisms. In *Foundations of Computer Science, 2002. Proceedings. The 43rd Annual IEEE Symposium on*, pages 471–480. IEEE, 2002.

- [KKK00] David Kempe, Jon Kleinberg, and Amit Kumar. Connectivity and inference problems for temporal networks. In *Proceedings of the thirty-second annual ACM symposium on Theory of computing*, pages 504–513. ACM, 2000.
- [KLO10] Fabian Kuhn, Nancy Lynch, and Rotem Oshman. Distributed computation in dynamic networks. In *STOC*, pages 513–522, 2010.
- [KMO11a] Fabian Kuhn, Yoram Moses, and Rotem Oshman. Coordinated consensus in dynamic networks. In *PODC*, pages 1–10, 2011.
- [KMO11b] Fabian Kuhn, Yoram Moses, and Rotem Oshman. Coordinated consensus in dynamic networks. In *Proceedings of the 30th annual ACM SIGACT-SIGOPS symposium on Principles of distributed computing*, pages 1–10. ACM, 2011.
- [KO09] Ari Keränen and Jörg Ott. Dtn over aerial carriers. In *Proceedings of the 4th ACM workshop on Challenged networks*, pages 67–76. ACM, 2009.
- [KO11] Fabian Kuhn and Rotem Oshman. Dynamic networks : models and algorithms. *ACM SIGACT News*, 42(1) :82–96, 2011.
- [KWZ03] Fabian Kuhn, Rogert Wattenhofer, and Aaron Zollinger. Ad-hoc networks beyond unit disk graphs. In *Proceedings of the 2003 joint workshop on Foundations of mobile computing*, pages 69–78. ACM, 2003.
- [Lam86] Leslie Lamport. On interprocess communication. *Distributed computing*, 1(2) :86–101, 1986.
- [LC91] Chih-Chung Lin and Ruei-Chuan Chang. On the dynamic shortest path problem. *Journal of Information Processing*, 13(4) :470–476, 1991.
- [LDS03] Anders Lindgren, Avri Doria, and Olov Schelén. Probabilistic routing in intermittently connected networks. *ACM SIGMOBILE mobile computing and communications review*, 7(3) :19–20, 2003.
- [LKF05] Jure Leskovec, Jon Kleinberg, and Christos Faloutsos. Graphs over time : densification laws, shrinking diameters and possible explanations. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 177–187. ACM, 2005.
- [LKF07] Jure Leskovec, Jon Kleinberg, and Christos Faloutsos. Graph evolution : Densification and shrinking diameters. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 1(1) :2, 2007.
- [LL09] Emmanuelle Lebhar and Zvi Lotker. Unit disk graph and physical interference model : Putting pieces together. In *Parallel & Distributed Processing, 2009. IPDPS 2009. IEEE International Symposium on*, pages 1–8. IEEE, 2009.
- [LMS99] Igor Litovsky, Yves Métivier, and Éric Sopena. Graph relabelling systems and distributed algorithms. *Handbook of graph grammars and computing by graph transformation*, 3 :1–56, 1999.

- [LP10] Zvi Lotker and David Peleg. Structure and algorithms in the sinr wireless model. *ACM SIGACT News*, 41(2) :74–84, 2010.
- [LW03] J Nicholas Laneman and Gregory W Wornell. Distributed space-time-coded protocols for exploiting cooperative diversity in wireless networks. *Information Theory, IEEE Transactions on*, 49(10) :2415–2425, 2003.
- [Lyn96] Nancy A Lynch. *Distributed Algorithms*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1996.
- [Mos07] Thomas Moscibroda. The worst-case capacity of wireless sensor networks. In *Information Processing in Sensor Networks, 2007. IPSN 2007. 6th International Symposium on*, pages 1–10. IEEE, 2007.
- [Pen97] Mathew D Penrose. The longest edge of the random minimal spanning tree. *The annals of applied probability*, pages 340–361, 1997.
- [PS98] Stefano Pallottino and Maria Grazia Scutella. Shortest path algorithms in transportation models : classical and innovative aspects. In *Equilibrium and advanced transportation modelling*, pages 245–281. Springer, 1998.
- [RBK07] Ram Ramanathan, Prithwish Basu, and Rajesh Krishnan. Towards a formalism for routing in challenged networks. In *Proceedings of the second ACM workshop on Challenged networks*, pages 3–10. ACM, 2007.
- [SQF⁺11] Nicola Santoro, Walter Quattrociocchi, Paola Flocchini, Arnaud Castegits, and Frederic Amblard. Time-varying graphs and social network analysis : Temporal indicators and metrics. *arXiv preprint arXiv :1102.0629*, 2011.
- [ST81] Daniel D Sleator and Robert Endre Tarjan. A data structure for dynamic trees. In *Proceedings of the thirteenth annual ACM symposium on Theory of computing*, pages 114–122. ACM, 1981.
- [SW09] Johannes Schneider and Roger Wattenhofer. Coloring unstructured wireless multi-hop networks. In *PODC*, pages 210–219, 2009.
- [SW10] Johannes Schneider and Roger Wattenhofer. An optimal maximal independent set algorithm for bounded-independence graphs. *Distributed Computing*, 22(5-6) :349–361, 2010.
- [SZP08] Hany Samuel, Weihua Zhuang, and Bruno R. Preiss. DTN based dominating set routing technique for mobile ad hoc networks. In *5th International ICST Conference on Heterogeneous Networking for Quality, Reliability, Security and Robustness, QSHINE 2008, Hong Kong, July 28-31, 2008*, page 9, 2008.
- [Tel00] Gerard Tel. *Introduction to distributed algorithms*. Cambridge university press, 2000.
- [TMML10] John Tang, Mirco Musolesi, Ccecilia Mascolo, and Vito Latora. Characterising temporal distance and reachability in mobile and online social networks. *ACM SIGCOMM Computer Communication Review*, 40(1) :118–124, 2010.

- [WDdACG12] John Whitbeck, Marcelo Dias de Amorim, Vania Conan, and Jean-Loup Guillaume. Temporal reachability graphs. In *MobiCom*, pages 377–388, 2012.
- [Zha06] Zhensheng Zhang. Routing in intermittently connected mobile ad hoc networks and delay tolerant networks : overview and challenges. *IEEE Communications Surveys & Tutorials*, 1(8) :24–37, 2006.
- [ZT87] Daniel J Zawack and Gerald Luther Thompson. A dynamic space-time network flow model for city traffic congestion. *Transportation Science*, 21(3) :153–162, 1987.

Une approche distribuée pour les problèmes de couverture dans les systèmes hautement dynamiques

Un système distribué est un système composé d'éléments de calcul autonomes dotés de capacité de communication. Il s'agit d'un modèle commun pour l'étude des réseaux. L'évolution rapide des réseaux sans fils et/ou mobiles aussi bien dans la vie quotidienne que dans la recherche amène progressivement à intégrer la dynamique (i.e. l'évolution dans le temps de la connectivité) dans les systèmes distribués. Concrètement, cela revient à ajouter l'hypothèse que les capacités de communication des éléments du système peuvent varier dans le temps.

De nombreux modèles considèrent ainsi la dynamique comme composante à part entière du système (et non pas comme une faute). De manière récente, une nouvelle approche, appelée graphe variant dans le temps, tente d'unifier tous ces modèles dans un formalisme commun qui permet de classifier les systèmes en fonction de leurs propriétés de connexité temporelle. Dans cette thèse, nous nous intéressons à des systèmes distribués hautement dynamiques dans lesquels les hypothèses de connexité sont minimalistes. Plus précisément, nous concentrons nos efforts sur les systèmes connexes à travers le temps dans lesquels la seule garantie est que tout élément du système peut infiniment souvent envoyer un message à tout autre (sans garantie sur la pérennité de la route utilisée ni sur le délai de communication). Nous nous intéressons plus particulièrement aux problèmes de couverture (par exemple, ensemble dominant minimal, couplage maximal, ensemble indépendant maximal, ...) dans ces systèmes distribués hautement dynamiques.

Les contributions de cette thèse dans ce contexte sont les suivantes. Nous proposons tout d'abord une nouvelle définition pour les problèmes de couverture qui est plus adaptée aux systèmes distribués hautement dynamiques que les définitions existantes. Dans un deuxième temps, nous fournissons un outil générique qui permet de faciliter les preuves de résultats d'impossibilité dans les systèmes distribués dynamiques. Nous appliquons cet outil pour prouver plusieurs résultats d'impossibilité à propos de problèmes de couverture. Ensuite, nous proposons une nouvelle mesure de complexité en temps qui permet de comparer équitablement les performances de protocoles dans les systèmes distribués dynamiques. Enfin, nous donnons un algorithme de construction d'un ensemble dominant minimal dans les systèmes distribués hautement dynamiques.