Parameter estimation and performance analysis of several network applications

Sara Alouf

Université de Nice - Sophia Antipolis – UFR Sciences
École Doctorale STIC

THÈSE

Présentée pour obtenir le titre de :
Docteur en Sciences de l'Université de Nice - Sophia Antipolis

Spécialité : INFORMATIQUE

par

Sara ALOUF

Équipe d'accueil : Mistral – INRIA Sophia Antipolis

Estimation de paramètres et analyse des performances de diverses applications réseaux

Soutenue publiquement à l'INRIA le 8 nov. 2002 à 14:30 devant le jury composé de :

Président :	Ernst	BIERSACK	Institut Eurécom
Directeur :	Philippe	NAIN	INRIA
Rapporteurs :	Patrick	THIRAN	EPFL
	Don	TOWSLEY	Université du Massachusetts
Examinateurs :	Walid	DABBOUS	INRIA
	Michel	RIVEILL	I3S - ESSI

Estimation de paramètres et analyse des performances de diverses applications réseaux

Sara Alouf

Titre de la thèse en anglais :

Parameter estimation and performance analysis of several network applications

Cette thèse est dédicacée
à mes parents
et à ma sœur

Remerciements

Plusieurs personnes ont contribué, de près ou de loin, au bon déroulement de cette thèse. Je les remercie du fond de mon cœur. Tout particulièrement, je souhaite exprimer ma gratitude envers mon directeur de thèse, Philippe Nain, pour son encadrement précieux et sa grande confiance en moi. Il m'a toujours traitée en égale et ceci m'a beaucoup touchée. Notre collaboration ne s'en est trouvée qu'enrichie. Mon respect pour lui n'a pas de bornes et je suis fière d'avoir été son étudiante.

J'ai vivement apprécié le séjour d'un mois que j'ai passé à l'Université du Massachusetts dans le groupe de recherche CNRG. L'accueil qui m'y a été réservé était des plus chaleureux, et j'en remercie Don Towsley ainsi que tous les membres du groupe.

Certains résultats présentés dans cette thèse sont issus de collaborations scientifiques avec Eitan Altman, Chadi Barakat, Fabrice Huet et Don Towsley. Je tiens particulièrement à les remercier pour le temps qu'ils m'ont consacré.

Je suis reconnaissante à Patrick Thiran et Don Towsley d'avoir accepté d'être rapporteurs de ma thèse et d'avoir lu et corrigé mon manuscrit de thèse.

Je remercie Ernst Biersack, d'avoir accepté de présider le jury, et Walid Dabbous et Michel Riveill d'avoir accepté d'être membres du jury.

Je remercie Sven Östring pour avoir lu et corrigé mon manuscrit de thèse.

Un grand remerciement va à l'assistante du projet Mistral, Ephie Deriche. Sa disponibilité, ses réponses à toutes les questions possibles et imaginables et sa bonne humeur permanente ont été une aide de tous les jours.

Je souhaite exprimer mon amitié pour tous les membres de Mistral, passés et présents. Pour tous les merveilleux moments passés ensemble, entre repas de midi, pots, soirées et pique-niques, je les remercie.

Je ne saurais assez exprimer ma gratitude envers Fabrice Huet pour m'avoir aidée, conseillée et encouragée durant ses trois années de thèse. Je lui en serai éternellement reconnaissante.

Enfin, une pensée chaleureuse va vers ma famille qui m'a toujours accompagnée, malgré les milliers de kilomètres qui nous séparaient, et qui a tenu à faire le voyage, pour assister à la soutenance de ma thèse et me soutenir en ce dernier jour de ma vie d'étudiante. Merci.

Contents

List of Figures

List of Tables

Présentation des travaux de thèse

1 Introduction

"Ce qui a débuté comme un exercice de paranoïa militaire est devenu un moyen de communications globales"[1] [2].

L'Internet est un réseau mondial reliant des réseaux d'ordinateurs et plusieurs centaines de millions d'utilisateurs de par le monde. La croissance exponentielle de l'Internet en nombre de réseaux ou d'utilisateurs pose de réels problèmes en termes de performance et de contrôle du réseau. L'un des principaux concepts de base qui ont contribué à la popularité de l'Internet et à son développement est connu sous le nom de "l'argument de bout-en-bout". Cette philosophie consiste à garder le réseau le plus simple possible et à déployer les fonctionnalités complexes aux extrémités de celui-ci [108, 37, 66]. Bien que ce principe de base fut en grande partie responsable du succès de l'Internet, il rend problématique d'obtenir des informations sur l'état interne du réseau.

Si jamais une fonctionnalité devait être déployée dans l'Internet, deux possibilités s'offrent au concepteur : soit de l'ajouter au cœur du réseau, soit de l'implémenter dans la couche application. Il est très coûteux de changer quoi que ce soit à l'intérieur du réseau, puisque ceci implique de modifier tous les routeurs de l'Internet. D'autre part, cette tâche devient encore plus ardue quand on sait que l'Internet est très hétérogène. Des réseaux câblés, sans fil et même satellitaires sont reliés entre eux, et forcément les couches les plus basses sont différentes d'un sous-réseau à l'autre. Pour toutes ces raisons, nous estimons qu'il est préférable de garder le réseau tel quel, et tout au long de cette thèse, nous n'allons nous intéresser qu'à des cas où les fonctionnalités désirées sont à implémenter au niveau applicatif.

1.1 Description des sujets étudiés

Tout au long de cette thèse, nous nous intéresserons à résoudre des problèmes propres à certains types d'applications, et pour chacun de ces problèmes, nous adopterons une approche de bout-en-bout.

Le premier sujet que nous aborderons est lié au problème de contrôle de congestion dans les applications point-à-points. Celles qui utilisent le protocole de transmission TCP

[1] En anglais dans le texte.

(pour Transmission Control Protocol [98]) pour assurer le transport des informations d'un site à un autre ne sont pas concernées par ce problème puisque le protocole TCP s'en charge déjà [67]. Les applications utilisant le protocole UDP (pour User Datagram Protocol [97]) pour le transport des données n'ont pas ce privilège et doivent contrôler leurs débits afin d'éviter de congestionner le réseau. Il est d'ailleurs souhaitable que ce type d'applications puisse se comporter de manière "civilisée" en évitant d'accaparer toutes les ressources disponibles, en d'autres termes, il faudrait que ces applications puissent être *TCP-friendly* [4].

Pour pouvoir adapter le débit d'envoi des données sur le réseau, l'application devrait avoir une idée sur l'état actuel du chemin utilisé. Une façon de faire consiste à estimer des caractéristiques propres au lien le plus congestionné de ce dernier. Par exemple, la bande passante disponible sur ce lien permet de faire un contrôle de congestion, alors que sa bande passante permet juste de contrôler le débit de l'application. Dans ce contexte, nous proposons une méthodologie de bout-en-bout qui fournit à l'application des estimations sur l'état du nœud le plus congestionné, rendant possible l'utilisation d'un mécanisme d'adaptation. À part le problème de contrôle de congestion, notre approche peut servir à équilibrer la charge des routeurs ou bien encore à la planification du réseau, puisque la méthodologie proposée permet d'identifier les nœuds congestionnés et ceux peu utilisés.

Le deuxième sujet que nous traiterons concerne les applications multipoints et, plus précisément, l'estimation de la taille des sessions multipoints en nombre de terminaux connectés à la session. Suite à l'apparition de matériels informatiques sophistiqués permettant de bien reproduire tant l'image que le son, les applications multimédia deviennent de plus en plus populaires et le trafic généré par ce type d'applications ne cesse d'augmenter [119]. Nous pensons que, tôt ou tard, les grands événements médiatiques finiront par être diffusés sur l'Internet via des sessions multipoints. De plus en plus de chaînes de télévision voudront diffuser leurs programmes via le multipoint. Il est alors essentiel pour ces sources d'information d'évaluer la popularité du contenu émis tant pour l'aménagement des programmes futurs que pour des raisons publicitaires. Il faudrait donc concevoir une méthode pour estimer le nombre de récepteurs dans une session multipoint à un instant donné. À noter que cette information peut également être utilisée par les fournisseurs d'accès dans l'établissement des factures : au lieu de faire payer les sources sur la base de leurs débits, le fournisseur d'accès peut choisir de les faire payer sur la base de leurs nombres de récepteurs.

Pour estimer la taille d'une session multiploint, la façon la plus directe serait de déployer une fonctionnalité de comptage dans le réseau lui-même. Toutefois, cette solution n'est pas très adaptée à la situation. Premièrement, elle impliquerait que chaque routeur maintienne un compteur par session multipoint, ce qui est évidemment coûteux en ressources et supporte mal le passage à l'échelle. Deuxièmement, cette modification devrait être répercutée sur tous les routeurs assurant le multipoint, ce qui est évidemment une tâche énorme. Troisièmement, l'estimation fournie serait toujours en dessous de la réalité puisque les routeurs d'accès des réseaux locaux n'ont aucun moyen de savoir le compte exact de terminaux connectés à une session multipoint donnée, la seule information disponible étant l'existence d'au moins un récepteur dans ladite session [43]. C'est pourquoi nous avons concentré nos

efforts sur l'estimation de la taille d'une session multipoint en utilisant une approche de bout-en-bout.

Le dernier sujet que nous étudierons est propre aux applications dites à code mobile. Un agent mobile est un programme qui peut changer de lieu d'exécution quand il le décide. Partant de cette définition, a été développé un paradigme appelé "code mobile" [117] qui fait de la mobilité une partie intégrante d'une application d'où le nom "application à code mobile". Les champs d'application de ce paradigme sont nombreux ; citons notamment l'équilibrage de charge, le commerce électronique, la recherche d'information et l'analyse de données [33]. Dans ce dernier cas, il s'agit d'analyser de grandes bases de données situées sur un site lointain ; le fait de déplacer le code vers le site en question permet d'économiser de la bande passante puisque le code est typiquement de petite taille contrairement aux données.

Les problèmes intrinsèques au paradigme de code mobile concernent les communications entre agents : il faudrait que ceux-ci puissent communiquer entre eux malgré la mobilité. Dans la plupart des cas, les librairies de code mobile fournissent, en sus de la mobilité, des mécanismes de communication entre agents. Parmi les mécanismes les plus répandus, citons (i) un mécanisme réparti qui fait usage d'objets appelés "répéteurs", chargés de transférer tout message à son destinataire ; (ii) un mécanisme centralisé qui s'appuie sur un serveur de localisation pour assurer les communications. Ce serveur garde à jour une base de données relative à l'emplacement de tous les agents mobiles. La question qui se pose tout naturellement est la suivante : quel mécanisme est le plus performant? C'est à fin de répondre à cette question que nous nous sommes intéressés au problème. À noter qu'aucune étude formelle dans ce cadre n'a été entreprise auparavant. Les concepteurs de librairies à code mobile, fournissant un ou plusieurs mécanismes de communication, ne justifient pas en général leur choix du mécanisme adopté et se basent souvent sur des raisonnements intuitifs (voir à titre d'exemple [84]). Nous verrons par la suite, lors de la comparaison formelle entre les deux mécanismes étudiés, que la réponse à la question "quel mécanisme est le plus performant?" n'est pas toujours évidente.

1.2 Contributions des travaux de thèse

Les contributions de cette thèse sont multiples et variées, allant de la théorie des files d'attentes, à l'environnement à code mobile, en passant par l'ingénierie des réseaux.

Dans le premier chapitre de cette thèse, nous proposons deux modèles de files d'attente basés sur les files $M/M/1/K$ et $M/D/1/K$. Pour chacune de ces files nous trouvons des expressions pour la probabilité de perte, l'utilisation du serveur, le temps de réponse, etc.. Pour la première fois, des quantités telles que la probabilité de perte conditionnelle et la probabilité de succès conditionnelle sont calculées et utilisées pour la file $M+M/M/1/K$. Par ailleurs, c'est aussi la première fois que l'analyse de la file $M/D/1/K$ est aussi détaillée. Des explications précises sont fournies pour le calcul de la distribution stationnaire de l'occupation de la file d'attente, ce qui permet le calcul de la probabilité de perte, de

l'utilisation du serveur et du temps de réponse de la file d'attente. À partir des modèles étudiés, nous proposons plusieurs schémas permettant l'estimation des caractéristiques internes du réseau. Les performances de ces divers schémas sont alors évaluées et comparées grâce à des simulations faites avec `ns-2` [83].

Dans le second chapitre de cette thèse, nous nous intéressons à l'estimation de la taille des groupes multipoints. Dans un premier temps, nous modélisons le groupe multipoint par une file d'attente $M/M/\infty$. Dans ce cas, le nombre de récepteurs dans le groupe est représenté par le nombre de serveurs occupés dans la file d'attente $M/M/\infty$. En trafic fort et après normalisation, le nombre de récepteurs dans le groupe converge en distribution vers le processus d'Ornstein-Ühlenbeck. La dynamique de ce processus étant linéaire, nous pouvons utiliser le filtre de Kalman pour résoudre le problème d'estimation. Notre objectif étant de trouver l'estimateur optimal sous les hypothèses les plus générales possibles, nous utilisons un filtre de Wiener pour trouver l'estimateur optimal dans le cas d'une file d'attente $M/M/\infty$ soumise à un trafic quelconque. À noter que la théorie de Wiener s'applique pour le cas plus général de la file $M/G/\infty$ mais le calcul des paramètres du filtre n'est malheureusement possible que pour le cas $M/M/\infty$. Par ailleurs, nous construisons un filtre linéaire d'ordre 1, qui est optimal parmi tous les filtres linéaires d'ordre 1, dans le cas où le groupe multipoint est modélisé par une file $M/H_L/\infty$ (H_L désigne une loi hyperexponentielle d'ordre L). L'estimateur résultant de ce filtre pour $L = 2$ et ceux découlant des filtres de Kalman et de Wiener sont comparés à des traces synthétiques et des traces réelles, en vue d'évaluer leurs performances et de déterminer le meilleur estimateur parmi ceux proposés. Malgré le fait que les hypothèses de base des modèles ne sont pas vérifiées dans les traces réelles, nous observons de bonnes performances : tous les estimateurs proposés reflètent bien l'évolution dans le temps de la taille des groupes multipoints. Nous insistons sur le fait que c'est la première fois que des traces réelles (audio et vidéo) sont utilisées pour la validation d'estimateurs de la taille du groupe, puisque jusqu'à présent seules des traces synthétiques ont été utilisées.

Les travaux présentés dans le troisième chapitre sont très originaux du fait même qu'ils se trouvent au croisement de deux domaines de recherche très différents : le monde de la modélisation et celui du code mobile. Les chercheurs travaillant dans le premier domaine n'ont souvent pas connaissance des problèmes rencontrés dans le deuxième domaine. De la même façon, les chercheurs venant du monde du code mobile ne sont souvent pas à l'aise avec des techniques de modélisation telles les chaînes de Markov. Comme conséquence, il n'y a presque eu aucune étude d'évaluation de performance de mécanismes relatifs au code mobile jusqu'à présent. Nous développons des modèles markoviens pour deux implémentations de mécanismes de communications entre agents. Le mécanisme à base de répéteurs est modélisé par une chaîne de Markov en temps continu et à espace d'états infini. Pour résoudre les équations d'équilibre et trouver la distribution stationnaire des états, nous utilisons des transformées en z qui rendent fini le nombre d'équations à traiter. Quant au mécanisme centralisé, nous le modélisons par une chaîne de Markov en temps continu et à espace d'états fini et les équations de Chapman–Kolmogorov sont résolues numériquement. Dans chacun des modèles, nous trouvons une expression pour l'espérance du temps de

communication, et dans le cas du mécanisme à base de répéteurs, nous trouvons également une expression pour le nombre moyen de répéteurs. Les deux modèles proposés sont alors validés à travers des simulations et des expérimentations conduites sur un réseau local et sur un réseau régional. Nous trouvons que les résultats analytiques et expérimentaux sont assez proches ce qui montre que nous pouvons utiliser les formules analytiques donnant les temps de réponse des deux mécanismes pour comparer formellement leur performances. Comme conséquence, nous pouvons tester une large gamme de conditions ce qui n'était pas possible lors des expérimentations. Les résultats de cette comparaison formelle ont révélés qu'aucun mécanisme n'est toujours le meilleur, ce qui rend notre analyse encore plus intéressante.

Dans les sections suivantes, nous allons traiter à part chacun des problèmes évoqués dans la section *1.1*.

2 Modèles d'inférence pour l'estimation de caractéristiques réseaux

Pour le transport de ses informations d'un point à un autre dans le réseau, une application s'appuie soit sur le protocole TCP, soit sur le protocole UDP. Ces deux protocoles de transport font partie de la famille de protocoles appelée "TCP/IP". Alors que TCP fournit un mécanisme de contrôle de congestion, le protocole UDP transmet les paquets de l'application sans aucun contrôle, au risque de congestionner le réseau. Une autre grande différence entre ces deux protocoles est que TCP retransmet les paquets perçus comme perdus, alors qu'UDP n'offre aucune garantie sur l'intégralité des informations transmises. Dans ce contexte, les applications, utilisant UDP pour le transport des informations et souhaitant avoir un minimum de garanties, devraient pouvoir régler elles-mêmes le débit avec lequel elles injectent des paquets dans le réseau. Afin d'obtenir un taux de perte faible, ces applications devraient veiller à ce que leur débit ne dépasse pas la bande passante disponible. Il faudrait aussi qu'elles aient un moyen de connaître la taille maximale des rafales de paquets qu'elles pourraient vouloir injecter dans le réseau. Ce qu'il faut donc c'est que les applications puissent avoir des estimations sur ces paramètres.

Nous supposons par la suite qu'il n'existe qu'un seul goulot d'étranglement le long d'une connexion. Nous proposons de modéliser une connexion par une simple file d'attente, ayant (i) une vitesse de traitement égale à celle du goulot d'étranglement de la connexion (qu'on notera μ) et (ii) un tampon de taille finie égale à K. Nous proposons également que l'application sonde le réseau avec des paquets de taille exponentiellement distribuée, à des intervalles de temps exponentiellement distribués, afin de déterminer l'état du réseau. Ce flux "sonde" est donc un processus de Poisson injecté dans la file d'attente décrite avant, et nous noterons son débit par γ. Les paquets de données de l'application, ainsi que les paquets provenant de tous les autres flux traversant le goulot d'étranglement de la connexion, seront représentés par un seul flux de débit λ et appelé "flux transverse".

Pour pouvoir s'adapter aux conditions du réseau, il faudrait donc que l'application

puisse avoir une estimation de la vitesse du serveur μ et de l'intensité du trafic transverse λ. La taille du tampon K étant inconnue, il faudrait l'estimer aussi. Nous allons supposer que le flux transverse peut être modéliser par un processus de Poisson de paramètre λ. Nous savons que cette hypothèse n'est pas vérifiée en général [95], nous ne l'avons considérée que pour la facilité mathématique qu'elle amène dans le calcul[2]. Au vu des hypothèses introduites jusqu'à présent et selon la notation de Kendall–Lee [72, 78]), nous pouvons dire que la connexion est modélisée par une file d'attente de type $M{+}M/G/1/K$ (voir la figure 1.2, page 9).

La méthodologie que nous proposons pour estimer μ, λ et K se résume en quatre points :

1. enregistrer des informations de base relatives aux paquets du flux sonde (instants d'entrée dans le réseau, instants d'arrivée à la destination, indication sur les paquets perdus, etc.) ;

2. utiliser ces informations pour estimer le taux de perte des paquets sonde, le délai moyen induit par le réseau, etc. ;

3. exprimer formellement les mesures de performance estimées en étape 2 en fonction des paramètres μ, λ et K, en utilisant le modèle de la connexion ;

4. inférer les valeurs des paramètres μ, λ et K en se basant sur les expressions trouvées en étape 3 et les mesures de performance estimées en étape 2.

2.1 La file d'attente $M{+}M/M/1/K$

Dans un premier temps, nous considérons que les temps de service sont indépendants entre eux et suivent une loi exponentielle de paramètre μ. La distribution stationnaire de l'occupation de la file d'attente est bien connue [75]. Nous pouvons alors facilement exprimer la probabilité de perte P_L, l'utilisation du serveur U et l'espérance du temps de réponse de la file d'attente R en fonction des paramètres μ, λ, γ et K (voir les équations (1.3), (1.5) et (1.10) respectivement, pages 10–11). Pour exprimer les probabilités conditionnelles de perte q_L et de succès q_N, nous calculons la probabilité conditionnelle d'avoir k paquets dans la file à l'instant t sachant qu'il y en avait i à l'instant 0 et étant donné qu'il n'y a pas de flux sonde ($\gamma = 0$). Le calcul de la transformée de Laplace de cette probabilité conditionnelle est détaillé dans la proposition 1.4.1. Les probabilités conditionnelles q_L et q_N s'expriment simplement en fonction de cette transformée de Laplace, et après calcul nous obtenons les équations (1.18) et (1.21).

[2] Au moment de valider nos modèles nous avons considéré des flux non-Poissonniens pour tester la sensibilité des modèles à cette hypothèse.

2.2 La file d'attente $M{+}M/D/1/K$

Nous considérons à présent que les temps de service sont déterministes et valent $\sigma = 1/\mu$. Pour trouver la distribution stationnaire de l'occupation de la file d'attente, nous nous basons sur l'analyse de la file $M/G/1/K$ qu'a faite J. W. Cohen dans [39]. Un calcul intermédiaire est nécessaire : il faut trouver les coefficients de la série de Taylor d'une fonction dépendant de la transformée de Laplace–Stieltjes de la distribution des temps de service. La distribution stationnaire de l'occupation de la file d'attente s'exprime alors très simplement en fonction de ces coefficients que nous noterons α_j (voir les équations (1.28)–(1.31)). Le lecteur intéressé est vivement encouragé à consulter la section 1.5.1 pour tous les détails sur le calcul de ces coefficients. Ayant exprimé la distribution stationnaire du nombre de paquets dans la file, il est facile de trouver des expressions pour la probabilité de perte P_L, l'utilisation du serveur U et l'espérance du temps de réponse R (voir les équations (1.37), (1.38) et (1.40) respectivement, pages 21–22).

2.3 Estimation des paramètres

Pour le modèle $M{+}M/M/1/K$, nous avons trouvé des expressions pour cinq mesures de performance (P_L, U, R, q_L et q_N) en fonction des paramètres de la file d'attente μ, λ, γ et K. Le débit du trafic sonde γ est connu de l'application, mais pas les trois autres paramètres. Pour les estimer, il suffit d'estimer trois mesures de performance et utiliser leurs expressions pour inférer les valeurs des paramètres μ, λ et K. Comme nous avons le choix entre cinq différentes mesures de performance, nous obtiendrons au total 10 schémas différents selon les trois mesures choisies. À noter toutefois, que le schéma qui utilise les trois mesures relatives aux pertes (P_L, q_L et q_N) n'est pas valide car ces dernières ne sont pas indépendantes. Nous nous retrouvons donc avec 9 schémas d'estimation.

Pour le modèle $M{+}M/D/1/K$, nous avons trouvé des expressions pour trois mesures de performance (P_L, U et R) en fonction des paramètres inconnus de la file d'attente μ, λ et K. Pour estimer ces derniers nous n'avons qu'un seul choix possible : utiliser les expressions de P_L, U et R et des estimations de ces mesures de performance, et inverser le système de trois équations à trois inconnues obtenu pour trouver les estimateurs $\hat{\mu}, \hat{\lambda}$ et $\hat{K}$. La table 1.1 (page 23) présente les équations à utiliser dans ce schéma, ainsi que celles à utiliser dans les neuf schémas obtenus précédemment.

Jusqu'à présent nous n'avons pas tenu compte d'un fait important : il existe plusieurs méthodes permettant l'estimation de la vitesse du goulot d'étranglement, autrement dit, de μ. Citons notamment [21, 31, 94, 77, 46]. Dans toutes ces références, les techniques utilisées pour l'estimation de μ se basent sur la dispersion observée sur une paire ou un train de paquets sonde (ou même sur les deux types d'envoi comme dans [46]). Si nous considérons maintenant que la vitesse du goulot d'étranglement μ peut être estimée à l'aide de *pathrate* [46], PBM [94] ou bien ROPP [77], il ne nous reste plus qu'à estimer λ et K et il est suffisant

dans ce cas d'utiliser deux mesures de performances. Pour le modèle $M{+}M/M/1/K$, nous aurons 10 schémas d'estimation possible mais pour le modèle $M{+}M/D/1/K$, nous n'en aurons qu'un seul (au lieu de 3). Nous avons bien trouvé des expressions pour 3 mesures de performances (P_L, U et R) ce qui offre 3 possibilités pour le choix de deux mesures parmi ces trois mesures, mais l'expression trouvée pour l'espérance du temps de réponse R comporte plusieurs inconnues et ne peut pas être utilisée que si celles de P_L et U le sont également. Les 11 schémas d'estimation ainsi obtenus sont présentés brièvement dans la table 1.2 (page 24).

Étant donné que les expressions trouvées pour les mesures de performances ne sont pas linéaires, il n'est pas certain qu'un système de deux ou trois de ces équations ait une solution unique. Nous avons pu formellement montrer l'unicité de la solution pour certains schémas uniquement. Toutefois, dans toutes nos expérimentations, nous avons toujours obtenu une solution unique pour un système donné.

2.4 Analyse des résultats

Nous avons réalisé une cinquantaine de simulations à l'aide de `ns-2`. Dans 20 de ces simulations, nous avons veillé à ce que le scénario simulé soit le plus proche possible du modèle $M{+}M/M/1/K$. Nous pouvons ainsi observer lesquelles parmi les cinq mesures de performance obtenues pour ce modèle sont le mieux estimées. Rappelons que ces cinq mesures sont estimées grâce aux informations collectées sur les paquets sonde (instants d'arrivée et de départ du réseau, indication des paquets perdus, indication des paquets ayant subi des délais d'attente; voir les équations (1.81)–(1.85), page 37). Nous avons trouvé que l'estimateur $\hat{q}_L$ est très bruité et que $\hat{P}_L$ a une faible variance mais une erreur relative pouvant dépasser la valeur 0.2. Quant aux estimateurs $\hat{U}$, $\hat{R}$ et $\hat{q}_N$, ils ont une faible variance et une faible erreur relative. À noter toutefois, que certains estimateurs convergent plus rapidement que d'autres selon la condition du réseau simulé. Ainsi, pour le cas congestionné, la majorité des paquets subit un délai d'attente, et la première estimation valide de l'utilisation U est assez tardive puisqu'elle coïncide avec le premier paquet n'ayant pas subi de délai (il faut que $\hat{U} < 1$).

La trentaine de simulations qui ne sont pas de type $M{+}M/M/1/K$ se divise entre des simulations où le service est déterministe pour chaque flux (il y a 100 flux exogènes et un flux sonde) ; des simulations où le trafic transverse est constitué de 100 ou de 250 flux de type On/Off dans lesquels la durée des périodes d'activité et d'inactivité sont de distribution Pareto ; et des simulations où le trafic transverse est constitué de 250 ou de 1000 flux FTP sur TCP.

Nous allons d'abord présenter les résultats des schémas dans le cas où la vitesse du goulot d'étranglement μ est connue. Sur toutes les simulations réalisées, nous avons observé que le schéma utilisant la probabilité de perte P_L et l'espérance du temps de réponse R retourne les meilleures estimations de λ (intensité du trafic transverse) et K (la taille de la

file d'attente). Nous observons que la qualité des estimations s'améliore quand le nombre de flux en arrière plan augmente, que ce soit pour les flux On/Off ou FTP sur TCP. Sur les 50 simulations et après une durée de simulation de 500 secondes, l'erreur relative sur λ est inférieure à 1% (respectivement 5% et 9%) dans 26 simulations (respectivement 38 et 49 simulations), alors que l'erreur relative sur K est inférieure à 1% (respectivement 5% et 9%) dans 20 simulations (respectivement 35 et 42 simulations).

Les résultats des schémas d'estimation quand la vitesse du goulot d'étranglement μ doit également être estimée ne sont pas aussi intéressants. Quand nous utilisons une estimation d'une mesure de performance à la place de celle-ci, nous induisons une erreur sur l'estimation finale. Il est alors évident que l'utilisation d'un nombre supérieur de mesures de performances conduit à une erreur plus importante en sortie. C'est ainsi que nous observons que le meilleur schéma, qui est celui utilisant P_L, U et R, ne donne de bons résultats que dans les simulations de type $M+M/M/1/K$.

Finalement, nous avons conduit six simulations dans lesquelles le flux sonde traversait quatre liens en cascade (voir la figure 1.17, page 59) au lieu de n'en traverser qu'un seul (cas des 50 simulations déjà présentées). Les résultats obtenus à l'aide du schéma utilisant P_L et R sont assez satisfaisants. Comme précédemment, nous avons considéré des flux transverses de type On/Off (une simulation) et FTP sur TCP (trois simulations). L'erreur relative sur l'intensité du trafic transverse est très faible quelque soit la position de goulot d'étranglement. En ce qui concerne l'estimation de la taille de la file d'attente, nous obtenons de bons résultats sauf pour le cas où le trafic transverse est de type TCP et que le goulot d'étranglement se situe près de la destination. Nous observons dans ce cas que la qualité de l'estimation s'améliore quand le nombre de flux en arrière plan augmente. Tous les détails sur ces simulations se trouvent dans les tables 1.11 et 1.12 (page 60).

2.5 Conclusion

Nous avons présenté deux modèles d'inférence qui permettent d'estimer simultanément l'intensité du trafic transverse et la capacité du tampon au nœud le plus congestionné d'une connexion. Nous avons trouvé que la meilleure façon de faire consiste à estimer le taux de perte et le temps de réponse de la connexion, et utiliser ces estimations pour inférer les valeurs des paramètres cités ci-dessus. Cette technique a été testée sur diverses simulations et les résultats obtenus sont satisfaisants. Enfin, ce travail a en partie été publié dans [16].

3 Estimation de la taille de groupes multipoints

Le protocole IP multipoint [43, 44] a été introduit pour permettre à plusieurs terminaux de recevoir les mêmes informations sans pour autant surcharger le réseau. Les applications pouvant tirer profit de ce protocole sont nombreuses et variées, telles la visioconférence, les cours à distance, la vidéo à la demande, la retransmission des événements

sportifs et les diffusions à grand public (cours des monnaies et de la bourse, télévisions, radios). Dans ces derniers exemples, une estimation du nombre de récepteurs peut être très utile aux fournisseurs de contenu qui peuvent ainsi adapter leurs services aux préférences de leur public.

Pour estimer le nombre de membres dans un groupe multipoint, la façon la plus simple (et la plus sûre) consiste à demander à tous les membres de signaler leur présence par l'envoi d'un acquittement à la source du groupe. Toutefois, ceci aura un impact négatif tant sur le réseau que sur la source qui sera submergée par les acquittements (nous nous positionnons dans le cas de groupes multipoints très larges). Il est donc préférable (et souhaitable) que les envois d'acquittement soient probabilistes. Pour ceci, nous proposons que chaque récepteur, à la demande de la source, envoie un acquittement avec une probabilité notée p à des intervalles de temps réguliers ; soit S cet intervalle. De cette façon, la source reçoit un certain nombre d'acquittements toutes les S unités de temps. Nous montrerons par la suite comment ces mesures bruitées peuvent être filtrées pour donner une estimation de la taille du groupe qui suit son évolution aussi efficacement que possible.

Un groupe multipoint peut être simplement modélisé par une file d'attente $G/G/\infty$. Les inter-arrivées des membres du groupe sont représentées par les inter-arrivées des paquets dans la file d'attente, les temps de séjour des récepteurs dans le groupe sont représentés par les temps de service dans la file d'attente, et enfin, le nombre de récepteurs dans le groupe, autrement dit la taille du groupe, est représenté par le nombre de serveurs occupés dans la file $G/G/\infty$. Nous noterons par $N(t)$ la taille du groupe multipoint. Dans les sections suivantes, nous nous appuyons sur la théorie des filtres adaptatifs pour estimer au mieux $N(t)$.

3.1 Estimation optimale à base de filtre de Kalman

Dans un premier temps, nous supposons que les inter-arrivées des membres du groupe suivent une loi exponentielle de paramètre λ, et que les durées de vie des récepteurs dans le groupe suivent également une loi exponentielle mais de paramètre μ. Dans ce cas, la taille du groupe $N(t)$ n'est autre que le nombre de serveurs occupés dans la file d'attente $M/M/\infty$. Nous savons donc que $N(t)$ est une variable poissonnienne de paramètre $\rho := \lambda/\mu$ [75].

Sous l'hypothèse d'un trafic fort (i.e., un groupe multipoint très large), la variable $N(t)$ tend vers l'infini. Mais par contre, si nous normalisons $N(t)$ par rapport à sa trajectoire limite, le processus obtenu (voir l'équation (2.8)) converge en distribution vers un processus de diffusion (voir l'équation (2.9)) [103]. Le processus de diffusion, dit également processus d'Ornstein-Ühlenbeck, est caractérisé par une dynamique linéaire : sa valeur à un instant donné n'est autre qu'une fraction de sa valeur à l'instant précédent à laquelle s'ajoute un bruit blanc.

Considérons maintenant le nombre d'acquittements $Y(t)$ reçus à un instant donné t.

Quand la taille du groupe multipoint tend vers l'infini, cette variable $Y(t)$ va elle-même tendre vers l'infini. Pour ceci, nous normalisons également la variable $Y(t)$ autour de sa trajectoire limite. Le processus obtenu (voir l'équation (2.16)) converge en distribution vers un processus qui est une fonction linéaire du processus de diffusion déjà mentionné (voir l'équation (2.27)). Ainsi, la valeur de ce processus à un instant donné n'est autre qu'une fraction de la valeur du processus de diffusion au même instant à laquelle s'ajoute un bruit blanc.

Le filtre de Kalman est un filtre linéaire d'ordre 1 qui est optimal parmi tous les filtres mesurables [107]. L'optimalité réside dans le fait que la variance de l'erreur quadratique est minimisée. Pour pouvoir utiliser ce filtre, il est nécessaire que la dynamique de l'état du système et des mesures soit linéaire, ce que nous venons juste de montrer pour le cas normalisé. Trois équations décrivent le filtre de Kalman ; ce sont (*i*) l'équation de Riccati qui permet de calculer la variance de l'erreur quadratique (qui est minimale, rappelons-le), (*ii*) l'équation donnant le gain du filtre et (*iii*) l'équation donnant l'estimation de l'état du système qui, dans notre cas, est normalisé. Cette dernière équation met en jeu deux termes dont le premier prend en compte l'estimation précédente et le deuxième l'actualise grâce à la nouvelle mesure normalisée. Pour obtenir l'estimation de la taille du groupe multipoint, il suffit de prendre l'équation (*iii*) et dénormaliser les variables représentant le système et les mesures (nombre d'acquittements reçus à la source). L'équation finale est donnée en (2.38) (page 79).

Nous avons testé l'estimateur donné en (2.38) et noté $\hat{N}_n$ sur des traces synthétiques et réelles. Dans les deux cas, nous observons de très bonnes performances (voir à titre d'exemples les figures 2.3–2.6, pages 81–82 et 85–86). L'estimateur semble être très robuste puisque les résultats sont satisfaisants malgré le fait que les distributions des traces utilisées ne sont pas exponentielles. Nous avons effectivement utilisé des lois Pareto pour générer les traces synthétiques et avons observé que les lois des inter-arrivées et des temps de séjour dans les traces réelles sont sous-exponentielles (loi de Weibull, loi Lognormale). Nous obtenons également de bons résultats quand le groupe est de petite taille (une quarantaine de récepteurs). Rappelons que cet estimateur de la taille du groupe multipoint a été obtenu sous des hypothèses de lois exponentielles et de trafic fort. Nous savons qu'il est optimal sous ces conditions et avons observé de bonnes performances dans des cas plus généraux. Nous aimerions toutefois construire un estimateur optimal sous des hypothèses plus générales. C'est ce que nous essayons de faire dans les prochaines sections.

3.2 Estimation optimale à base de filtre de Wiener

Nous considérons à présent que les temps de séjour des récepteurs dans le groupe suivent une loi de distribution générale à l'exception des lois dites à queue lourde. Autrement dit, la somme des autocovariances du processus $N(t)$ est finie. L'autocovariance de la version stationnaire de $N(t)$ est donnée en (2.39) [41, équation (5.39)]. Dans la théorie de Wiener,

le processus du signal à estimer et celui des mesures bruitées ont une espérance nulle. Nous allons donc centrer les variables $N(t)$ et $Y(t)$ autour de leurs moyennes respectives qui sont ρ et $p\rho$. Les variables ainsi obtenues sont notées $\nu(t)$ et $y(t)$.

Par la suite, nous considérons ces variables à l'état stationnaire en temps discret. La réponse impulsionnelle du filtre optimal qui transforme y_n en $\hat{\nu}_n$ vérifie l'équation de Wiener-Hopf donnée en (2.43) [61]. Une alternative consiste à calculer la fonction de transfert du filtre optimal. Pour ceci, il faut procéder aux étapes suivantes:

1. calculer $S_y(z)$, le spectre de puissance du processus des mesures y_n ;
2. factoriser $S_y(z)$ de la façon suivant $S_y(z) = \sigma^2 G(z)G(z^{-1})$ où $G(z)$ est la partie de $S_y(z)$ ayant tous ses zéros et pôles dans le cercle unité ;
3. calculer $S_{\nu y}(z)$, le spectre de puissance croisé des processus ν_n et y_n ;
4. écrire le rapport $S_{\nu y}(z)/G(z^{-1})$ sous forme de somme de fractions et isoler les fractions ayant tous ses zéros et pôles dans le cercle unité ;
5. prendre le résultat obtenu en étape 4 et le diviser par $\sigma^2 G(z)$; ceci donne la fonction de transfert du filtre optimal.

L'utilisation du filtre de Wiener est conditionnée par l'aptitude à factoriser $S_y(z)$ comme présenté dans l'étape 2. Malheureusement, cette factorisation canonique n'est possible que pour le cas où la distribution des temps de séjour des récepteurs est exponentielle. Nous nous retrouvons donc avec un modèle $M/M/\infty$ pour le groupe multipoint, mais nous nous sommes affranchis de l'hypothèse de trafic fort considérée dans la section *3.1.* Effectuant les étapes 1 à 5 décrite ci-dessus et inversant la fonction de transfert pour retrouver la réponse impulsionnelle du filtre, nous obtenons une équation aux différences d'ordre 1 donnant l'estimateur centré $\hat{\nu}_n$ en fonction de sa valeur à l'instant précédent et du processus centré des mesures y_n. Il ne nous reste plus qu'à remplacer les processus centrés par ceux non-centrés pour obtenir l'équation donnant $\hat{N}_n$ (voir (2.47)).

L'estimateur obtenu grâce au filtre de Wiener est identique à celui obtenu dans la section *3.1* obtenu grâce au filtre de Kalman. Ce résultat s'explique par le fait que les deux approches considérées minimisent la variance de l'erreur quadratique. Le filtre de Kalman est toujours linéaire d'ordre 1, celui de Wiener est toujours linéaire, et dans le cas exponentiel, il s'est avéré qu'il est d'ordre 1 également. Il est donc tout naturel que les estimateurs fournis par ces deux méthodes de filtrage soient les mêmes pour le modèle $M/M/\infty$. À noter toutefois que l'utilisation du filtre de Kalman nécessite une hypothèse de trafic fort, ce qui n'est pas le cas dans la théorie de Wiener.

Utilisant les spectres de puissance $S_y(z)$ et $S_{\nu y}(z)$ et la fonction de transfert du filtre optimal, nous calculons la variance de l'erreur quadratique (qui est minimale). De toute évidence, l'expression trouvée est la même que celle donnée par l'équation de Riccati de la théorie de Kalman.

3.3 Estimation efficace à base de filtre linéaire d'ordre 1

Nous considérons à nouveau le modèle $M/G/\infty$, autrement dit, les temps de séjour sont généralement distribués, et comme précédemment, nous excluons les lois à queue lourde. Dans cette section, nous nous donnons une équation aux différences d'ordre 1 de la forme $\hat{\nu}_n = A\hat{\nu}_{n-1} + By_n$ et calculons les paramètres A et B qui minimisent la variance de l'erreur quadratique donnée par $\epsilon := \mathbf{E}[(\nu_n - \hat{\nu}_n)^2]$. Il suffit donc de résoudre le système d'équations donné donné par : $\partial\epsilon/\partial A = 0$ et $\partial\epsilon/\partial B = 0$. Nous montrons que ce système possède une solution unique, vérifiant $0 < A < 1$, ce qui garantit la stabilité du filtre donné ci-haut. Il est évident que si les temps de séjour sont exponentiellement distribués, alors l'estimateur obtenu sera identique à celui donné par le filtre de Wiener ou de Kalman. Nous considérons par la suite que les temps de séjour suivent une loi hyperexponentielle ayant L stages. Quand $L = 2$, une solution numérique est facilement obtenue. Nous noterons l'estimateur obtenu par $\hat{N}_n^{H_2}$ par opposition à $\hat{N}_n^E$ qui dénotera l'estimateur de la taille du groupe quand les temps de séjour des récepteurs sont exponentiellement distribués. À noter que $\hat{N}_n^E$ est optimal parmi tous les estimateurs dans le cas où les lois des inter-arrivées et des temps de séjour sont exponentielles, et que $\hat{N}_n^{H_2}$ est optimal parmi tous les estimateurs d'ordre 1 dans le cas où les inter-arrivées suivent une loi exponentielle et les temps de séjour sont hyperexponentiellement distribués.

3.4 Validation des modèles

Nous avons testé les estimateurs $\hat{N}_n^E$ et $\hat{N}_n^{H_2}$ sur des traces réelles de sessions vidéos. Pour chaque trace, les valeurs de la probabilité d'acquittement p et de l'intervalle d'acquittement S sont choisies de façon à obtenir une faible variance de l'erreur quadratique et un petit volume d'acquittements générés (se référer à la section 2.7 pour tous les détails). Sur les quatre traces considérées (voir à ce titre les figures 2.8–2.11, pages 105–106), nous observons de bonnes performances pour les deux estimateurs. Le niveau de qualité de l'estimation est presque la même si on utilise $\hat{N}_n^E$ ou $\hat{N}_n^{H_2}$ (les détails sont dans les tables 2.8 et 2.9, pages 103–104). Toutefois nous remarquons que l'estimateur $\hat{N}_n^E$ donne de meilleurs résultats quand la taille du groupe multipoint change abruptement, à l'inverse de l'estimateur $\hat{N}_n^{H_2}$ qui donne de meilleurs résultats quand il y a peu de changements dans le groupe. Mais ce qui avantage clairement l'utilisation de $\hat{N}_n^E$, c'est qu'elle nécessite la connaissance *a priori* de deux paramètres seulement : la taille moyenne du groupe et la valeur moyenne des temps de séjour des récepteurs. Quant à l'estimateur $\hat{N}_n^{H_2}$, son utilisation nécessite l'identification de quatre termes (la taille moyenne du groupe, les valeurs moyennes des temps de séjour des récepteurs dans les deux stages de la loi hyperexponentielle et la probabilité d'un des stages de cette loi) ce qui est plus difficile à assurer.

L'estimateur $\hat{N}_n^E$ a été testé sur des traces où les lois sont sous-exponentielles (voir la

section *3.1*), mais qu'en est-il de l'estimateur $\hat{N}_n^{H_2}$? Ce dernier a été testé sur des traces de sessions vidéos uniquement. Nous avons identifié les lois des inter-arrivées et des temps de séjour de ces traces comme étant soit Lognormales soit de Weibull (voir la table 2.10 page 107). Ceci indique que l'estimateur $\hat{N}_n^{H_2}$ est assez robuste puisque l'estimation donnée, dans des cas autres que celui où il a été développé (modèle $M/H_2/\infty$), est très satisfaisante.

Enfin, pour estimer la taille moyenne du groupe ρ et le temps moyen de séjour $1/\mu$ (les paramètres pré-requis pour utiliser $\hat{N}_n^E$), nous proposons trois méthodes différentes qui nécessitent l'envoi d'acquittements supplémentaires : (*i*) soit les récepteurs envoient des acquittements probabilistes au moment de leurs arrivées dans le groupe, ce qui permet d'estimer λ ; (*ii*) soit les récepteurs envoient des acquittements au moment de quitter le groupe, ce qui permet d'estimer μ ; (*iii*) soit les deux ensembles. Nous proposons au lecteur intéressé de consulter la section 2.9 pour plus de détails et un test sur une trace réelle.

3.5 Conclusion

Nous avons proposé plusieurs estimateurs de la taille du groupe multipoint en suivant successivement trois approches distinctes. Nous avons testé les estimateurs sur des traces synthétiques et réelles et avons observé de bonnes performances. Nous pensons que le meilleur de ces estimateurs est celui développé pour le modèle $M/M/\infty$ et qui nécessite la connaissance de la taille moyenne du groupe et du temps moyen de séjour. Ainsi, nous avons indiqué comment la source pourrait estimer ces deux paramètres. Ce travail a en partie été publié dans [12].

4 Analyse de deux mécanismes de communication dans un environnement à code mobile

Dans un environnement à code mobile, il ne suffit pas qu'une librairie de code fournisse les éléments nécessaires à la mobilité. Il est nécessaire de fournir également des moyens pour assurer les communications entre les divers agents d'une application. Deux mécanismes de communication sont très répandus, or jusqu'à présent, le choix d'utilisation de l'un ou de l'autre de ces mécanismes n'a pas été formellement justifié. Nous proposons de choisir le mécanisme le plus rapide, autrement dit, celui qui délivre un message à l'agent mobile le plus rapidement possible. Nous allons décrire et modéliser chacun de ces deux mécanismes dans les deux prochaines sections.

Nous considérons par la suite une application à code mobile dans laquelle un objet immobile qu'on désignera par "source" essaie de communiquer avec un objet mobile qu'on désignera indifféremment par "agent mobile" ou simplement "agent". Les communications entre objets sont avec *Rendez-vous*, autrement dit, tout objet ayant envoyé un message à un autre objet se retrouve bloqué tant que le message n'a pas atteint sa destination.

4.1 Mécanisme distribué à base de répéteurs

Le premier mécanisme considéré met en jeu des objets spéciaux appelés “répéteurs”. Quand un agent mobile migre, il laisse sur l’ancien site un objet – un répéteur – chargé de lui transmettre tout message lui étant destiné. Au fur et à mesure que l’agent migre, une chaîne de répéteurs se forme entre le site initial de l’agent et le site où l’agent se trouve actuellement. La source possède une référence sur le site initial de l’agent et, à chaque fois qu’un message parvient à l’agent après avoir été transmis par un répéteur, l’agent envoie un message de mise à jour à la source. Quand ceci arrive, la chaîne de répéteurs n’est plus utilisée lors de l’envoi de futurs messages. La chaîne est dite “court-circuitée” et, de proche en proche, ses répéteurs sont éliminés car plus référencés.

Pour modéliser ce mécanisme, nous devons prendre en considération la distance séparant le message (ou la source en l’absence de celui-ci) de l’agent en nombre de sauts, l’état de l’agent (inactif/migrant) et celui de la source (inactive/communiquant). Nous obtenons ainsi un triplet qui définit l’état du système. Sous des hypothèses d’indépendance et de lois exponentielles pour les variables en jeu (durée de migration, délai inter-sites, etc.), l’état du système est représenté par une chaîne de Markov en temps continu (voir la figure 3.3, page 121). L’espace d’états de cette chaîne est infini à cause de la première composante : la distance séparant la source de l’agent peut potentiellement atteindre l’infini s’il n’y a pas de limite sur le nombre de hôtes que l’agent peut visiter.

Pour trouver la condition de stabilité de cette chaîne de Markov et exprimer la distribution stationnaire des probabilités d’état, nous commençons par écrire les équations d’équilibre des flux en entrée et en sortie de chaque état. Nous introduisons ensuite une transformée en z de la distribution stationnaire. En utilisant cette transformée et en manipulant judicieusement les équations d’équilibre, nous nous retrouvons avec un nombre fini d’équations facile à résoudre. La condition de stabilité implique que l’agent ne doit pas se déplacer plus rapidement que le message qui essaie de le joindre. Ceci est évident puisque la distance entre le message (ou la source) et l’agent croît quand ce dernier migre, et décroît quand un répéteur transmet le message au site en aval le long de la chaîne des répéteurs.

Pour quantifier la performance de ce mécanisme, nous nous intéressons à deux mesures : le temps moyen de communication, noté T_F et donné en équation (3.33) (page 128), et le nombre moyen de répéteurs. Dans ce dernier cas, nous faisons la distinction entre le nombre moyen de répéteurs entre la source (qui est immobile) et l’agent, noté N_s et donné en équation (3.37) (page 129), et le nombre de répéteurs entre le message et l’agent, noté N et donné en équation (3.50) (page 131). Si l’agent ne migre pas entre le moment d’émission d’un message et le moment où le message lui parvient, nous obtiendrons $N_s = N$. Or un agent mobile peut migrer entre ces deux moments, ce qui fait que le message aura à traverser un nombre plus important de répéteurs. C’est pour cette raison que nous avons $N > N_s$. À noter que N_s peut être utilisé pour évaluer la tolérance aux pannes du mécanisme [20].

4.2 Mécanisme centralisé à base de serveur de localisation

Le deuxième mécanisme considéré se base sur une entité centralisée pour assurer les communications. À la fin de chaque migration, l'agent envoie un message de mise à jour à un serveur de localisation. Celui-ci maintient une base de donnée ayant les positions actuelles de tous les agents mobiles. Quand une source désire communiquer avec l'agent, elle utilise la dernière position connue de celui-ci et envoie le message au site correspondant. Si l'agent a migré depuis la dernière communication, l'essai de communication échoue. La source s'adresse alors au serveur de localisation pour obtenir la position actuelle de l'agent, qui peut bien ne pas être la bonne : pendant l'envoi de la réponse par le serveur, l'agent peut avoir migré.

Le serveur de localisation n'a pas le même fonctionnement d'une librairie à une autre. Nous allons décrire celui adopté dans la librairie de code mobile *ProActive* [101]. Dans cette librairie, le tampon du serveur est partitionné afin que chaque paire source-agent ait son propre espace de mémoire. Ainsi, les interactions entre les différentes paires sont éliminées. Un ordonnanceur cyclique est utilisé et une seule requête est servie à chaque fois. Quand plusieurs requêtes sont en attente dans le même tampon, le serveur de localisation traite en priorité les messages de mise à jour de l'agent. La politique de service est non-préemptive puisque *ProActive* est une libraire Java. Quand le serveur finit de traiter une requête de la source et qu'une requête de mise à jour de l'agent est en attente, le serveur ne va pas envoyer la position de l'agent trouvée dans la base de donnée (car devenue caduque) à la source mais va plutôt remettre la requête dans le tampon pour la retraiter ultérieurement. Pour finir, les requêtes de mise à jour d'un agent remplacent toute requête de mise à jour du même agent qui serait en attente.

Pour modéliser ce mécanisme, nous nous restreignons au cas d'une seule paire source-agent et montrons en section *4.4* comment étendre notre modèle au cas plus général de multiples paires source-agent. L'état du système ainsi simplifié est entièrement défini par l'état du serveur (notamment le type de requêtes s'y trouvant), celui de l'agent (inactif/migrant) et celui de la source (inactive/communiquant avec l'agent/communiquant avec le serveur). Sous des hypothèses d'indépendance et de lois exponentielles des variables en jeu (temps de service, latence réseau, inter-migrations, etc.), le système est représenté par une chaîne de Markov en temps continu et espace d'états fini (voir la figure 3.7, page 136).

Pour trouver la distribution stationnaire des probabilités d'état, nous écrivons les équations d'équilibre pour chaque état et l'équation de normalisation (la somme des probabilités vaut 1). La résolution du système matriciel ainsi obtenu se fait numériquement. De plus, nous exprimons le temps moyen de communication, noté T_S et donné en équation (3.51) (page 137).

4.3 Validation des modèles et évaluation des performances

Les modèles développés en sections *4.1* et *4.2* considèrent que les lois des variables

régissant le fonctionnement des mécanismes sont exponentielles et indépendantes entre elles. Or, des expérimentations conduites sur un réseau local et un autre régional montrent que ces variables ont plutôt une loi de distribution de Weibull, à l'exception des temps de services qui sont relativement constants. À noter que les temps d'inactivité de la source et de l'agent dépendent de l'application uniquement et peuvent donc suivre des lois arbitraires.

Dans un premier temps, nous validons les modèles a l'aide de simulations. Nous considérons plusieurs scénarios dans lesquels toutes les variables sauf une sont exponentiellement distribuées. Dans ces simulations, nous observons une très bonne robustesse des modèles à des temps d'inactivité déterministes de l'agent, à des délais inter-sites ayant une loi de distribution de Weibull et à des temps de services déterministes. Les modèles semblent être plus sensibles aux distributions des temps d'inactivités de la source et des durées de migration. Nous obtenons ainsi une erreur relative allant jusqu'à 17% quand les temps d'inactivité de la source sont déterministes, et allant jusqu'à 16% quand les durées de migration suivent une loi de Weibull. Pour terminer, nous considérons des scénarios où toutes les variables ne sont pas exponentiellement distribuées. Les résultats sont acceptables puisqu'en moyenne l'erreur relative est de 14%. Pour plus de détails, se référer à la table 3.3 (page 140).

Dans les simulations effectuées, les hypothèses d'indépendance étaient satisfaites. Pour tester la robustesse des modèles aux cas où celles-ci ne le sont pas, nous menons des expérimentations sur un réseau local et sur un réseau régional. En effet, dans ces conditions, les durées de migration et les délais inter-sites sont particulièrement corrélés. Malgré cela, nous observons que les résultats prédits par les modèles sont assez proches des résultats expérimentaux, à l'exception des expérimentations sur un réseau régional où le taux de migration de l'agent est élevé (voir les figures 3.8 et 3.9, pages 3.8 et 3.9). Dans ces cas, nous avons rencontré des difficultés lors de l'estimation du délai moyen inter-sites (latence du réseau régional), or ce paramètre a une grande influence sur les résultats théoriques. D'autre part, nous observons que le temps de communication offert par le mécanisme centralisé est plus court que celui offert par le mécanisme réparti sur un réseau local. C'est exactement l'inverse qui se produit sur un réseau régional où le mécanisme des répéteurs est plus performant (voir à ce titre la figure 3.10, page 145).

Les modèles proposés dans les sections *4.1* et *4.2* sont valides, puisque les résultats théoriques sont proches des résultats expérimentaux. Nous pouvons donc utiliser les modèles pour prédire les performances des mécanismes dans des cas généraux. En étudiant le signe de la différence entre les temps moyens de communication offerts par les deux mécanismes, plus précisément le signe de $\Delta T = T_F - T_S$, nous pouvons prédire lequel des deux mécanismes sera le plus performant. Ainsi quand ΔT est positif, le mécanisme centralisé est plus performant, et quand ΔT est négatif, c'est le mécanisme réparti qui est meilleur. Le signe de ΔT est représenté dans la figure 3.11 (page 147) pour plusieurs valeurs des paramètres des modèles. Nous retrouvons notamment les observations faites lors les expérimentations (le mécanisme centralisé est plus performant sur un réseau local et le mécanisme distribué est plus performant sur un réseau régional) et observons plusieurs comportements imprévisibles que l'intuition seule n'aurait pu prédire. Ainsi, il apparaît que, pour un taux de migration donné, le mécanisme centralisé est préférable sauf pour des taux de communication de

valeurs moyennes (dans ce cas, $\Delta T < 0$; voir les figures 3.11(a), (b) et (d), page 147). Une étude plus approfondie montre que pour des taux de communication de valeurs moyennes, les deux mécanismes offrent des performances presque égales, avec toutefois un temps moyen de communication légèrement plus faible pour le mécanisme des répéteurs (la différence est de l'ordre des dixièmes de millisecondes).

4.4 Extension au cas de multiples paires source-agent

La question du passage à l'échelle ne se pose que pour le modèle du mécanisme centralisé. Pour modéliser le mécanisme réparti, nous avons pris en compte la distance séparant une source d'un agent et les états desdits source et agent. Ceci revient à étudier la dynamique d'une seule chaîne de répéteurs. Or il n'y a pas d'interaction entre les différentes chaînes puisque, par définition, une chaîne est exclusive a une paire source-agent. Ceci n'est pas le cas du mécanisme centralisé. La politique de service élimine les interactions entre les requêtes quand celles-ci sont en attente, mais il n'empêche qu'il n'y a qu'un seul serveur pour traiter toutes les requêtes. Ainsi, quand le serveur traite une requête relative à une certaine paire source-agent, il sera vu comme étant en vacances par les requêtes générées par d'autres paires source-agent. Une alternative consiste à considérer que le serveur est en fait plus lent à traiter les requêtes. Ainsi le temps passé à traiter les requêtes d'autres paires sera comptabilisé comme faisant partie du temps de service des requêtes de la paire considérée.

Dans cette optique, nous allons remplacer dans le modèle la vitesse réelle du serveur par une vitesse réduite tenant compte du nombre de paires source-agent. Le temps moyen de communication trouvé n'est alors qu'une approximation. Cette simplification n'est toutefois plus valable quand le délai inter-sites est important ou que le taux de communication est très élevé. Le fait est que, dans ces cas, le serveur reste bloqué souvent (grand nombre de réponses à envoyer) et pendant un long laps de temps à chaque fois (grande latence du réseau). Pour évaluer la qualité de cette approximation, nous avons conduit quatre séries de simulations (une centaine de simulations dans chaque série, correspondant à certaines valeurs des taux de communication et de migration) dans lesquelles nous avons tenu à garder des variables de loi exponentielle. Le but est d'isoler l'effet de l'approximation dans les résultats théoriques.

Nous trouvons ainsi que la qualité de l'approximation dépend des valeurs choisies pour les taux de communication et de migration. Ainsi nous observons que l'erreur relative sur le temps moyen de communication reste en dessous de 10% tant que l'utilisation du serveur reste en dessous de 0.7 pour un taux de migration égal à 1, et en dessous de 0.5 pour un taux de migration égal à 10 (pour plus de détails, se référer à la table 3.7, page 153). À noter qu'un nombre de paires source-agent donné n'induit pas la même utilisation du serveur, celle-ci dépendant également des valeurs choisies pour les taux de communication et de migration. Ainsi, quand le taux de migration vaut 1 et que le taux de communication vaut 1 aussi, 92 paires source-agent sont nécessaires pour que le serveur soit utilisé à 70%,

alors qu'une cinquantaine suffit si le taux de communication vaut 10.

4.5 Conclusion

Nous avons proposé deux chaînes de Markov pour modéliser deux mécanismes de communication entre agents dans un environnement à code mobile. Grâce à ces modèles, nous avons pu trouver des expressions pour le temps moyen de communication donné par chacun des mécanismes et pour le nombre moyen de répéteurs utilisé dans le mécanisme réparti. Nous avons validé nos modèles à l'aide de simulations et d'expérimentations sur un réseau local et un autre régional. Ayant observé une bonne concordance entre les résultats théoriques et expérimentaux, nous avons utilisé les formules théoriques pour prédire la performance des mécanismes dans des cas plus généraux. Nous avons ainsi développé un moyen sûr pour choisir le meilleur mécanisme selon l'environnement de l'application.

5 Conclusions et perspectives

Nous avons, dans ce travail, considéré la même philosophie pour résoudre certains problèmes propres à certaines applications. Cette philosophie consiste d'abord à proposer des modèles mathématiques du système étudié ; ensuite, des estimateurs ou des mesures de performance sont dégagés des modèles proposés ; et finalement, les résultats analytiques sont comparés à des résultats simulés ou expérimentaux.

Dans la section *2*, nous proposons deux modèles basés sur les files d'attente $M/M/1/K$ et $M/D/1/K$ et considérons onze schémas différents permettant l'estimation desdits paramètres. Nous testons ces schémas sur des simulations et trouvons que la meilleure façon d'estimer l'intensité du trafic transverse et la taille de la mémoire au nœud le plus congestionné repose sur l'utilisation du taux de perte et du temps moyen de réponse. En ce qui concerne l'estimation de la vitesse du goulot d'étranglement, nous avons proposé des schémas permettant l'estimation des trois paramètres simultanément, mais les résultats montrent que les schémas d'estimation ne sont pas robustes. Ainsi, il est préférable d'estimer la vitesse du goulot d'étranglement en utilisant *pathrate* [46], PBM [94] ou ROPP [77], et ensuite estimer l'intensité du trafic transverse et la taille de la mémoire au nœud le plus congestionné. Toutefois, nous n'avons pas étudié l'impact d'utiliser une estimation de la vitesse du goulot d'étranglement au lieu d'utiliser la valeur correcte comme nous l'avons fait. Toute erreur sur ce paramètre va forcément influer sur les résultats donnés en section *2*. La quantification de cette influence n'a toujours pas été faite et reste une question ouverte.

Dans la section *3*, nous modélisons le groupe multipoint par une file d'attente $M/M/\infty$ et résolvons le problème d'estimation à l'aide d'un filtre de Kalman (cas d'un trafic fort) et un filtre de Wiener (cas général). Ensuite, nous construisons un estimateur dans le cas d'un modèle $M/H_2/\infty$. Nous testons ces estimateurs sur des traces synthétiques et sur des traces réelles (audio et vidéo) et observons de bonnes performances. Toutefois, d'autres modèles,

plus réalistes, pourront être proposés. Nous pensons notamment à des modèles $M/W/\infty$ ou $M/L/\infty$ puisque les lois de distributions observées sur les traces réelles (audio et vidéo) sont soit de Weibull soit Lognormales. Une façon de procéder consiste à s'imposer une équation aux différences d'une forme donnée (d'ordre 1 par exemple) reliant l'estimateur au processus de mesure, et à calculer les coefficients de cette équation de façon à minimiser la variance de l'erreur quadratique. Une autre perspective de travail concerne le choix de la probabilité d'acquittement et de l'intervalle d'acquittement. Un critère possible consiste à limiter le volume d'acquittements générés pendant un laps de temps fixe (dans cette thèse nous avons limité le volume d'acquittements générés à chaque intervalle d'acquittement).

Dans la section *4*, nous modélisons deux mécanismes de communication entre objets à l'aide de chaînes de Markov et évaluons leurs temps de réponse. Nous validons nos modèles par des simulations et des expérimentations sur des réseaux réels (local et régional). De nos expérimentations, il est apparu que le mécanisme centralisé est le plus performant sur un réseau local alors que le mécanisme réparti est le plus performant sur un réseau régional. Les expressions des temps moyen de communication sont en fait un outil de prédiction qui permettent de sélectionner le mécanisme le plus performant étant donné le contexte d'utilisation. Le modèle du mécanisme centralisé pourrait bien être revu et amélioré pour mieux tenir compte du passage à l'échelle. D'autre part, il existe plusieurs autres mécanismes de communication qui n'ont toujours pas été modélisés ni évalués. Citons notamment un mécanisme mixte qui met en jeu un serveur de localisation et des répéteurs à durée de vie limitée. Deux variantes existent selon qui parmi les répéteurs et les agents envoient des messages de mise à jour au serveur. Il n'est effectivement pas nécessaire que répéteurs et agents mettent à jour le serveur. Concernant la conception de ces mécanismes, plusieurs questions sont sans réponse, telles "quelle est la durée de vie optimale des répéteurs?" et "au bout de combien de migrations l'agent doit-il mettre à jour le serveur?" Cette dernière question ne se pose que si les répéteurs ne font pas de mise à jour avant de devenir inactifs.

Introduction

"What began as an exercise in military paranoia has become a method of global communication" [2].

The Internet is a rapidly-growing worldwide network of computer networks that connects millions of users in more than 170 countries. The recent exponential growth of the Internet poses challenging problems in terms of control of the network. One of the basic concepts which contributed to its rapid growth and popularity is the so-called "end-to-end argument". The idea behind this argument is: keep the network simple and put the intelligence in the edges of the network [108, 37, 66]. Even though this concept contributed to the success of the Internet, it made it very difficult to have any information on the internal state of the network.

Whenever a function is to be added to the Internet, one has the choice of adding it either to the network or to applications. Adding a single function to the core of the network is highly difficult as all the routers in the world would have to be updated, which is not very tractable due to the huge size of the Internet.

Another component of the success of the Internet is its heterogeneity and its ability to connect wired to wireless networks, as well as terrestrial to satellite networks. This heterogeneity presents yet another difficulty when a certain function is to be added to the network. For all of these reasons, we believe it is much more attractive to add the required functions to applications.

Description of the subjects studied

In this thesis, we are interested in solving problems related to different types of applications and in each case we will adopt an end-to-end approach.

First, we looked at *unicast* applications wishing to perform congestion control. Applications using TCP (Transmission Control Protocol [98]) as a transport protocol are not concerned with such control as this function is implemented in TCP [67]. Applications using UDP (User Datagram Protocol [97]) will have to control their sending rate in order to avoid congestion and/or to be TCP-friendly [4]. To enable such a control in such applications, one must provide the application with the state of the path connecting the sender to the sink. For instance, the available bandwidth on a path is well suited for congestion control and the

bottleneck bandwidth is well suited for transmission rate control. We propose an end-to-end methodology which supplies the application with estimates on network-internal characteristics triggering some adaptation algorithm. Another kind of application that could benefit from this work is load balancing in routers and capacity planning as our methodology allows one to identify congested and underutilized links.

In the second part of the thesis, we are concerned with *multicast* applications and, more precisely, with the estimation of the membership of multicast sessions. With the emergence of multimedia applications [119] and dedicated sophisticated hardware such as high-quality video and audio cards, one is expecting to see major events and live sports being multicast. It will then be of significant importance to the sender to have an estimate of the number of receivers. This will allow the sender to determine the popularity of emitted content and even to advertise when the number of receivers is high. Not only can the sender benefit from such estimates: Internet Service Providers (ISPs) may take advantage of this information and charge the source based on the number of receivers instead of charging on an input-rate basis. One may think that the easiest way to provide membership estimates to the source would be to implement this function in the network. However, this solution is not convenient for multiple reasons. First, it requires that each multicast router maintain in memory one counter per multicast session, which is clearly not scalable. Second, it requires all of multicast routers to be changed. Third, such estimates will not be precise as gateway routers will have no way of determining the number of receivers within Local Area Networks (LANs), the only information available to them being the possible existence of at least one receiver [43]. Such an approach will rather return a lower bound on the number of receivers. Here again, we believe that an end-to-end approach is more suitable.

The third and last part of the thesis is concerned with applications in a *mobile agent environment*. In such applications, components of a running program can move from one host to another and proceed with their execution on the new host. This is often called the "code mobility paradigm" [117] and is very useful to load balancing. Another possible application is data mining [33] in which the data to be processed is situated on a remote machine. Moving the code to the location of the data will save bandwidth as the code is usually smaller in size. Inherent problems to this paradigm are insuring communications between the different components of the applications. Usually, mobile code libraries supply the applications with communication mechanisms. There are two widely used mechanisms, and we will, in this thesis, analyze one implementation of each. The first mechanism is decentralized and relies on special objects called "forwarders" that will forward the messages to the moving components. The second mechanism is centralized and relies on a location server which records the current positions of all of moving components. The question which arises then is: which mechanism performs better? This Quality of Service (QoS) issue has motivated our work on the subject. The mobile code libraries implementing one or the other mechanism have not justified their choice formally (see for instance [84]), and there is no formal comparison of the performance of both mechanisms.

Thesis contributions

This thesis makes several contributions to the queueing theory, to the networking field and to the code mobility field.

In the first chapter, we propose two queueing models based on the $M/M/1/K$ and $M/D/1/K$ queues, and derive several QoS metrics of interest in each model, such as the loss probability, the server utilization, the response time and some conditional probabilities. For the first time, metrics such as the conditional loss probability and the conditional non-loss probability are derived for the $M+M/M/1/K$ queue. Furthermore, our detailed analysis of the $M/D/1/K$ queue has not been carried out before. Explicit information is provided on how to compute the probability distribution of the queue length, which triggers the computation of the loss probability, the server utilization and the response time of the $M/D/1/K$ queue. Based on these models, we propose several schemes for estimating network-internal characteristics, and evaluate their performance on simulations conducted with `ns-2` [83].

In the second chapter, we embark towards estimating the membership in multicast sessions. We first propose to model the multicast group as an $M/M/\infty$ queue under a heavy traffic regime. In this case, the scaled membership weakly converges to an Ornstein-Ühlenbeck process which is known to have linear dynamics. The fact that both the dynamics of the system and the measurements are linear enables the use of the powerful Kalman filter theory. The estimator derived from this theory is ensured to be optimal under the considered assumptions. The linearization of the problem and the use of the Kalman filter in this context is a very original contribution. Motivated to derive an estimator under more general assumptions, we build on Wiener filter theory which returns the best filter, in steady-state, among all of linear filters. The model considered in this case is the $M/M/\infty$ queue under a general traffic regime. The last estimator we derive in this chapter, is based on an $M/H_L/\infty$ queue model where H_L denotes an L-stage hyperexponential distribution. All of these estimators have been applied on real audio and video traces, and they appear to perform very well despite the fact that the assumptions considered in the different models were violated in the considered traces. Note that it is the first time that membership estimators are applied to real traces as usually they are only tested on synthetic traces.

The material presented in the third chapter is very original as it falls within the intersection of two very different fields: the modeling world and the code mobility world. Researchers from the first field are often not familiar with the problems encountered in the second field, and researchers from the code mobility world are often not familiar with modeling techniques such as Markov chains. This fact has impaired the performance evaluation of mechanisms related to code mobility. We model two implementations of location mechanisms using Markov chains. The forwarders mechanism is modeled as an infinite state-space Markov chain, and the state probability distribution is expressed using z-transforms. Expressions for the average number of forwarders and the average response time of the mechanism are then found. The centralized mechanism is modeled as a finite state-space

Markov chain and the Chapman–Kolmogorov equations are solved numerically. Here also we find an expression for the average response time of the mechanism. After validating both Markovian models through simulations and experiments over both a LAN and a MAN, the expressions of the expected response times returned by both models are compared under a wide variety of conditions, showing that no mechanism is uniformly better than the other, thereby justifying even more this research.

Thesis organization

The following chapters are self-contained, since each problem is treated individually and in sequential. Chapter 1 is concerned with adaptive unicast applications which transmission rate is to be adjusted to avoid congestion. Chapter 2 is concerned with multicast applications in which the sender is interested in tracking the dynamics of the group membership. Chapter 3 is concerned with the performance evaluation and comparison of two agent location mechanisms in a mobile code environment. Chapter 4 summarizes the thesis and discusses some perspectives. A glossary and the bibliography complete the report.

Notation in use

Here are some notes on the notation used in the following. Vectors and matrices are represented by bold letters. The Kendall–Lee notation [72, 78] is used to classify the queueing models. C_n^p will denote a combination of n objects arranged in groups of size p ($n > 0$ and $0 \leq p \leq n$). Row 1 in a table will denote the first line of the table, i.e., the one having the names assigned to the columns. When counting the number of rows, every line is considered even if it is almost empty. Similarly, column 1 will denote the first column as illustrated in this example:

	column 1	column 2	...
row 1			...
row 2			...
⋮	⋮	⋮	⋱

Chapter 1

Inference models to estimate network characteristics

The first chapter of the thesis is concerned with adaptive unicast applications that adjust their sending rate in order to avoid congestion in the network. One way of achieving this involves estimating both the available bandwidth and the available storage capacity along a connection. The research presented herein builds on simple inference models based on finite capacity single server queues to estimate the buffer size, the intensity of cross traffic and the service rate at the bottleneck link of a path between two application-level hosts. Throughout the chapter, several groups of moment-based estimators are proposed to estimate these quantities. The best scheme, identified as the one achieving the best performance and the fastest convergence, is then identified through simulation. An application to routing in content distribution networks is discussed.

Keywords: Queueing, $M/M/1/K$ queue, $M/D/1/K$ queue, measurement, monitoring, simulation.

Note: Part of the material presented in this chapter is published in [16].

1.1 Introduction

The huge expansion of the Internet coupled with the emergence of new (in particular, multimedia) applications pose challenging problems in terms of performance and control of the network. These include the design of efficient congestion control and recovery mechanisms, and the ability of the network to offer good Quality of Service (QoS) to the users. In the current Internet, there is a single class of best effort service which does not promise anything to the users in terms of performance guarantees. The forthcoming deployment in the Internet of differentiated services (known as DiffServ [1]) will be a first (long awaited) step towards the support of various types of applications and business requirements. It is however doubtful that DiffServ – which will mark each packet to receive a particular forwarding treatment, or per-hop behavior, at each network node – or the RED mechanism for congestion avoidance in gateways [53] alone will solve all QoS issues raised by real-time applications. Diffserv and RED are two instances of a general approach that aims at adding more intelligence in the *network*. A more ambitious component of this approach is captured in the concept of *active networking* [122] that aims at exploiting mobile code and programmable infrastructure to provide rapid and specialized service introduction.

A complementary approach for providing QoS guarantees is to add intelligence to the *applications*. The idea is to provide applications with enough knowledge of the network so that they can use this information to adapt their transmission rates to current network conditions. Since it is impossible to monitor every link on the Internet, static (e.g. bandwidth of a link) and dynamic (e.g. available bandwidth on a path) network internal characteristics have to be estimated from measurements delivered by the network (e.g. packet losses in RTCP feedback). Our work falls into the latter category.

In this chapter, we are interested in the *available* bandwidth on a route. Under the single bottleneck link assumption, we expect this quantity to be better estimated if one has estimates of the bottleneck bandwidth, the cross traffic intensity and the buffer size at the bottleneck link. The latter quantities are not provided by *packet pair* algorithms [73, 21, 31, 94, 77]). We believe that the available bandwidth will be better estimated when taking explicitly into account the cross traffic and the buffer size at the bottleneck link. Not only can the application adapt its sending rate and/or encoding in response to network congestion, but it can estimate the maximum size of a burst of packets as well. We propose to perform active measurements by probing the network, and model the end-to-end path as a simple single server queueing model with two input streams: the probe stream and the cross traffic stream (also called the background traffic stream). The input probe traffic will be a Poisson process and the input cross traffic will be *assumed* to be a Poisson process. Two models will be considered depending on whether the service times are assumed to be exponentially distributed or deterministic. Based on the single bottleneck link assumption, the inference models will allow us to simultaneously estimate the bandwidth capacity, the background traffic intensity and the buffer size at the bottleneck link. Potential applications for this work are: adapting the transmission for congestion control [21, 111], load balancing

in routers and analysis of IP network usage. In fact, ISPs need bandwidth monitoring tools to plan their capacity upgrades, and to detect congested or underutilized links [26].

The chapter is organized as follows. Section 1.2 briefly discusses some related works and Section 1.3 introduces the methodology followed in this work. Two inference models based on the $M/M/1/K$ and the $M/D/1/K$ queues are introduced in Sections 1.4 and 1.5, respectively. In each case, several QoS metrics of interest are identified and expressed in terms of the parameters to be estimated (buffer size, cross traffic intensity, server capacity). In Section 1.6, we propose ten schemes that can be used to estimate these quantities, and under the assumption that the service capacity is known (provided for instance by a packet pair technique), we propose a total of eleven schemes that can be used to compute estimates of the intensity of the cross traffic and the buffer size. Section 1.7 reports simulation results obtained with the `ns-2` simulator [83] from which we were able to select the best scheme out of the proposed schemes. Extensions to our work are given in Section 1.8.

1.2 Related work

Inference models have been widely used for characterizing internal network behavior from end-to-end multicast measurements. This methodology is captured in the MINC project [3]. Among the network characteristics which have been estimated (refer to [3] for a list of references), are the internal loss rates in the MBone [29, 28, 30, 51], the packet delay variance at interior network links [50], the network-internal delay [100, 49], and the multicast topology [27, 48, 47] (these lists are not exhaustive).

The estimation of network characteristics from measurements has been carried out in a number of cases. For instance, the arrival rates of interfering traffic and the service rates of customers on the route of a CAC (call acceptance controller) probe stream can be estimated for a product form Kelly network [110]. The steady-state throughput of a bulk transfer TCP flow can be estimated as a function of loss rate and round-trip time using the so-called TCP-friendly formula [93, 80, 82]. The packet pair technique can be used to estimate both the bottleneck bandwidth [21, 31, 94, 77] and the available bandwidth along a path connecting two hosts on the Internet [67, 73, 31]. Although the previous estimates have been devised under the assumption that a single bottleneck link exists on a path connecting two hosts, experiments reported in the previous references indicate that these estimates still perform reasonably well when this assumption is violated.

The estimation of the bottleneck bandwidth has been carefully investigated by Paxson [94, Chapter 14]. Paxson proposes a robust procedure called "packet bunch mode" (PBM) and compares the results returned by PBM with other techniques such as receiver-based packet pair (RBPP), sender-based packet pair (SBPP) and peak rate (PR). It is seen in [94] that PR performs poorly, that RBPP performs as well as PBM if some requirements are observed and that SRPP does not work especially well (the estimators studied in [21, 31] are both sender-based). As for PBM, Paxson observes that "*it produces many bandwidth*

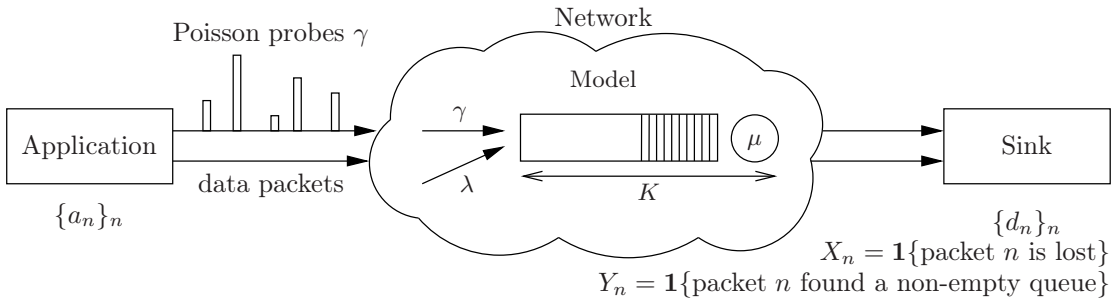
estimates that accord with known link speeds, and produces few erroneous results, except for a tendency to misdiagnose a multiple-channel bottleneck link in presence of considerable delay noise". Note that PBM has a large component of heuristics, as it has to deal with the multimodal distribution of bandwidth measurements to further identify and select the capacity estimate from these modes.

Subsequent to our work, Lai and Baker [77] propose another technique called "receiver only packet pair" (ROPP), which is yet another variant of the packet pair algorithm. The authors compare the performance of ROPP and SRPP and RBPP (but not PBM). The results reported in [77] suggest that ROPP can achieve an accuracy close to that of RBPP, while maintaining the ease of deployment of SBPP. Recently, Dovrolis, Ramanathan and Moore [46] develop a bandwidth estimation methodology that is robust to cross traffic effects. This methodology, based on end-to-end measurements, is implemented in a tool called *pathrate*. There are potentially two phases in *pathrate*. In the first phase, a packet pair probing is done. If the distribution of bandwidth measurements is unimodal, then the unique mode is the bandwidth estimator and the measurement process terminates. Otherwise (multimodal distribution), the second phase is initiated and a packet train probing is done. The authors make use of a heuristic to estimate the bottleneck bandwidth. When evaluating the heuristic rule, they observe that the methodology is quite accurate as long as the path capacity is lower than about 40 Mbps. For higher capacities, the estimator underestimates the capacity, in some cases by a factor of almost two. However, the authors show that the performance of their estimator is enhanced when increasing the *histogram bin width* ω. This parameter tunes the resolution of the estimator, for instance, if $\omega = 2$ then *pathrate* will output estimates at 2 Mbps interval (refer to [46] for more details).

1.3 Methodology

Consider the following scenario: an application sends a probe stream into the network together with its data packets. All packets are assumed to follow the same path to the destination. At the source, the arrival times to the network (denoted by a_n) are available. At the destination, the departure times from the network (denoted by d_n) are available as well as the information on lost packets and on packets that find the network non-empty. Assume that all these variables are available at the source by using some feedback mechanism. The source is then capable of estimating some QoS related metrics such as the loss probability or the expected response time of the network.

Given a model for the network (such as a simple single server queue representing the bottleneck node along the path source-destination), one is able to express the QoS related metrics in terms of the buffer size, the cross traffic intensity and the server capacity. Given estimates of these QoS related metrics, the source can simply infer the values of the buffer size, the cross traffic intensity and the server capacity, by using the network model. Figure 1.1 illustrates the estimation approach just described.



General methodology:

- collect at the source samples of the variables a_n, d_n, X_n, Y_n,
- estimate some performance metrics using these variables (e.g. loss probability),
- given a model for the connection, express these metrics in terms of the parameters to estimate,
- infer the unknown parameters.

Figure 1.1: The methodology

1.4 The $M+M/M/1/K$ queue

1.4.1 The model

We model an Internet connection by a single server queue representing its bottleneck node, following [22]. The buffer is finite with room for K customers ($K \geq 1$) including the customer in service. The incoming traffic at the bottleneck is modeled as two independent Poisson sources: the probe traffic generated by a foreground source with rate γ, and the cross traffic generated by a background source with rate λ. This background source can be seen as the superposition of many heterogeneous sources. We model the service times as i.i.d. random variables with exponential distribution with mean $1/\mu$, furthermore, independent of the arrival processes. This assumption represents the variability of the packet sizes. We are aware of the strength of this assumption, but we assume it for the tractability it gives to the study of the model.

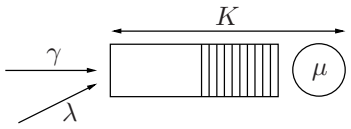


Figure 1.2: The inference model

The traffic intensity is defined as

$$\rho = \frac{\lambda + \gamma}{\mu}. \tag{1.1}$$

We are interested in the behavior of the system from the perspective of the foreground customers. This includes stationary measures such as expected delay, loss probability, server

occupancy and a number of additional statistics associated with the foreground loss process, namely, the probability of two consecutive losses and the probability of two consecutive successes. It is important to observe that these stationary metrics do not pertain exclusively to the foreground source, but to the background source as well, due to the Poisson assumption and its memoryless property.

Let $\{Q_n\}_{n=1}^{\infty}$ be the process of the number of packets in the buffer at time of the nth arrival from foreground source, and let $Q = \lim_{n\to\infty} Q_n$[1]. The distribution of Q is $\pi_i = P(Q = i)$. It is known that [75]

$$\pi_i = \frac{(1-\rho)\rho^i}{1-\rho^{K+1}}, \qquad i = 0, 1, \ldots, K. \tag{1.2}$$

1.4.2 The loss probability

We focus here on the loss process. We define $X_n = \mathbf{1}\{Q_n = K\}$ and $X = \lim_{n\to\infty} X_n$. A customer is lost whenever it arrives to a full buffer. In other words, customer n is lost whenever $X_n = 1$ and is not lost otherwise. Let $\{a_n\}_{n=1}^{\infty}$ and $\{d_n\}_{n=1}^{\infty}$ be the arrival times to the system and the departure times from the system, respectively, of the nth foreground customer, $n = 1, 2, \ldots$. When a packet is lost, it never reaches the destination. We shall assume that $d_n = \infty$ if $X_n = 1$.

Using the PASTA property [17, page 137], the probability that a foreground customer arrives to find the system full and is lost is

$$\begin{aligned} P_L := P(X = 1) &= P(Q = K) \\ &= \frac{(1-\rho)\,\rho^K}{1-\rho^{K+1}}. \end{aligned} \tag{1.3}$$

Observe that the expression for P_L can be used to give the following expression for K in terms of ρ and P_L,

$$K = \frac{1}{\log\rho} \log\left(\frac{P_L}{1-\rho\,(1-P_L)}\right). \tag{1.4}$$

1.4.3 The server utilization

The second metric of interest is the utilization U of the server, defined as the probability of a non-empty queue as seen by a foreground customer. In order to express U, we

[1] A word on the notation in use: let $\{Z_n\}_n$ be a sequence of random variables taking values in $[0, \infty)$. Assume that $\lim_{n\to\infty} P[Z_n \le x]$ exists for all $x \ge 0$. Then $Z = \lim_{n\to\infty} Z_n$ designates any random variable such that $P[Z \le x] = \lim_{n\to\infty} P[Z_n \le x]$

introduce the following indicator $Y_n = \mathbf{1}\{Q_n > 0\}$ and define $Y = \lim_{n\to\infty} Y_n$. The server utilization is then

$$\begin{aligned} U := P(Y=1) &= P(Q>0) \\ &= \rho\left(\frac{1-\rho^K}{1-\rho^{K+1}}\right) \qquad (1.5) \\ &= \rho\,(1-P_L). \qquad (1.6) \end{aligned}$$

Again, we can derive from the expression for U the following expression for K in terms of ρ and U,

$$K = \frac{1}{\log\rho}\log\left(\frac{1-U/\rho}{1-U}\right). \qquad (1.7)$$

1.4.4 The expected response time

When available, the expected response time is also a relevant metric. To express this quantity, we first define T_n as the response time of the nth foreground packet. It follows that $T_n = d_n - a_n$. Again, let $T = \lim_{n\to\infty} T_n$, then, the expected response time is

$$R := E[\,T\,|\,X=0\,] = \frac{\sum_{i=0}^{K-1}(i+1)\,\pi(i)}{\mu(1-\pi_K)} \qquad (1.8)$$

since a customer waits for an average time $(i+1)/\mu$ if there were already i customers in the queue; $1-\pi_K$ is the probability of a success. Using (1.2) we can write

$$R = \frac{1}{\mu\,(1-\pi_K)}\left(\frac{1-\rho}{1-\rho^{K+1}}\sum_{i=0}^{K-1}(i+1)\,\rho^i\right).$$

Hence,

$$\begin{aligned} R &= \frac{1}{\mu\,(1-\pi_K)}\left[\frac{1-\rho^K}{(1-\rho)\,(1-\rho^{K+1})} - \frac{K\,\rho_K}{1-\rho^{K+1}}\right] \\ &= \frac{1}{\mu\,(1-\rho)} - \frac{K}{\mu}\,\frac{\pi_K}{(1-\rho)\,(1-\pi_K)}. \qquad (1.9) \end{aligned}$$

The last expression for R, which was derived using (1.2), will prove useful. We can express R only in terms of ρ and K by replacing π_K given by (1.3) in (1.9). This gives

$$R = \frac{1}{\mu\,(1-\rho)} - \frac{K}{\mu}\,\frac{\rho^K}{1-\rho^K}. \qquad (1.10)$$

1.4.5 The conditional loss probability

The next metric we are going to study is the conditional loss probability or, in other words, the probability that two consecutive losses occur. It is expressed as follows

$$q_L := P(Q_n = K \,|\, Q_{n-1} = K). \tag{1.11}$$

In order to be able to derive a closed form expression for q_L, we define $N(t)$ to be the queue length of the system at time $t \geq 0$ with the foreground source removed ($\gamma = 0$). Let $P_{i,k}(t) = P(N(t) = k \,|\, N(0) = i)$. Hence (1.11) rewrites

$$q_L = \gamma \int_0^\infty e^{-\gamma t} P_{K,K}(t)\, \mathrm{d}t = \gamma\, P^*_{K,K}(\gamma) \tag{1.12}$$

where $P^*_{K,K}(\gamma)$ is the Laplace transform of $P_{K,K}(t)$.

Proposition 1.4.1

$$P^*_{i,K}(\gamma) = \frac{a^{i+1}\,(1-a)^{-1} - b^{i+1}\,(1-b)^{-1}}{\lambda\,(b^{K+1} - a^{K+1})}$$

for $i = 0, 1, \ldots, K$, *where*

$$a = \frac{\lambda + \mu + \gamma + \sqrt{(\lambda+\mu+\gamma)^2 - 4\lambda\mu}}{2\lambda}$$
$$b = \frac{\lambda + \mu + \gamma - \sqrt{(\lambda+\mu+\gamma)^2 - 4\lambda\mu}}{2\lambda}.$$

♦

Proof. Recall the definition $P_{i,k}(t) = P(N(t) = k \,|\, N(0) = i)$ where $N(t)$ is the queue length of the system with the foreground source removed ($\gamma = 0$) at time $t \geq 0$. We have the following differential equations:

$$\frac{\mathrm{d}}{\mathrm{d}t} P_{i,0}(t) = -\lambda P_{i,0}(t) + \mu P_{i,1}(t) \tag{1.13}$$
$$\frac{\mathrm{d}}{\mathrm{d}t} P_{i,K}(t) = -\mu P_{i,K}(t) + \lambda P_{i,K-1}(t) \tag{1.14}$$
$$\frac{\mathrm{d}}{\mathrm{d}t} P_{i,k}(t) = -(\lambda+\mu) P_{i,k}(t) + \lambda\, P_{i,k-1}(t) + \mu\, P_{i,k+1}(t), \text{for } k = 1, \ldots, K-1, \tag{1.15}$$

for $i = 0, \ldots, K$. Define $P_i(z,t) = \sum_{k=0}^K P_{i,k}(t) z^k$. Then,

$$z\frac{\mathrm{d}}{\mathrm{d}t} P_i(z,t) \;=\; z\frac{\mathrm{d}}{\mathrm{d}t} \sum_{k=0}^K P_{i,k}(t) z^k \;=\; z \sum_{k=0}^K \frac{\mathrm{d}}{\mathrm{d}t} P_{i,k}(t) z^k.$$

Using (1.13) – (1.15) and after some algebra we get

$$\frac{z}{1-z}\frac{\mathrm{d}}{\mathrm{d}t}P_i(z,t) = (\mu-\lambda z)P_i(z,t) - \mu P_{i,0}(t) + \lambda z^{K+1}P_{i,K}(t). \tag{1.16}$$

Now we consider the Laplace transform of $P_i(z,t)$, $P_i^*(z,s) = \int_0^\infty e^{-st}P_i(z,t)\,\mathrm{d}t$. Replacement of this in (1.16) along with the use of the following relation

$$\int_0^\infty e^{-st}\frac{\mathrm{d}}{\mathrm{d}t}\,P_i(z,t)\,\mathrm{d}t = sP_i^*(z,s) - P_i(z,0)$$

and some algebraic manipulations yields

$$P_i^*(z,s) = \frac{z^{i+1} - \mu(1-z)P_{i,0}^*(s) + \lambda(1-z)z^{K+1}P_{i,K}^*(s)}{sz - (1-z)(\mu-\lambda z)} \tag{1.17}$$

where $P_{i,k}^*(s) = \int_0^\infty e^{-st}P_{i,k}(t)\mathrm{d}t$, $k = 0, 1, \ldots, K$. The denominator of the right-hand side of (1.17) contains two zeros,

$$z_1(s) = \frac{\lambda+\mu+s-\sqrt{(\lambda+\mu+s)^2-4\lambda\mu}}{2\lambda}$$
$$z_2(s) = \frac{\lambda+\mu+s+\sqrt{(\lambda+\mu+s)^2-4\lambda\mu}}{2\lambda}$$

for $\Re(s) \geq 0$.

As $P_i^*(z,s)$ is analytic, the zeros of the denominator must also be zeros of the numerator. More precisely, the numerator must satisfy

$$z_k^{i+1}(s) - [1-z_k(s)][\mu P_{i,0}^*(s) - \lambda z_k(s)^{K+1}P_{i,K}^*(s)] = 0.$$

These two equations for $k = 1, 2$ can be solved to yield

$$P_{i,0}^*(s) = \frac{z_2(s)^{i+1}z_1(s)^{K+1}/[1-z_2(s)] - z_1(s)^{i+1}z_2(s)^{K+1}/[1-z_1(s)]}{\mu\left(z_1(s)^{K+1} - z_2(s)^{K+1}\right)}$$
$$P_{i,K}^*(s) = \frac{z_2(s)^{i+1}/[1-z_2(s)] - z_1(s)^{i+1}/[1-z_1(s)]}{\lambda\left(z_1(s)^{K+1} - z_2(s)^{K+1}\right)}.$$

Let a and b be defined as $a = z_2(\gamma)$ and $b = z_1(\gamma)$, we get

$$P_{i,K}^*(\gamma) = \frac{a^{i+1}\,(1-a)^{-1} - b^{i+1}\,(1-b)^{-1}}{\lambda\,(b^{K+1} - a^{K+1})}$$

and the proof is concluded. ■

From Proposition 1.4.1 we get that

$$P^*_{K,K}(\gamma) = \frac{(1-a)^{-1} - (b/a)^{K+1}(1-b)^{-1}}{\lambda\left((b/a)^{K+1} - 1\right)}$$

which in turn implies from (1.12) that

$$q_L = \left(\frac{\gamma}{\lambda}\right)\left(\frac{(1-a)^{-1} - (b/a)^{K+1}(1-b)^{-1}}{(b/a)^{K+1} - 1}\right). \tag{1.18}$$

Expression (1.18) for q_L can be inverted to give

$$K = \frac{\log\left(\frac{\gamma}{1-a} + q_L\lambda\right) - \log\left(\frac{\gamma}{1-b} + q_L\lambda\right)}{\log b - \log a} - 1. \tag{1.19}$$

1.4.6 The conditional non-loss probability

Another metric can also be calculated. It is the conditional probability that an arriving foreground packet finds room in the buffer given that the previous foreground customer was also admitted. We shall refer to this probability as the *c*onditional non-loss probability and will denote it by q_N. We have

$$\begin{aligned}
q_N &:= P(Q_{n+1} \neq K \,|\, Q_n \neq K) \\
&= \sum_{i=0}^{K-1} \frac{P(Q_{n+1} \neq K, Q_n = i, Q_n \neq K)}{P(Q_n \neq K)} \\
&= \sum_{i=0}^{K-1} \frac{P(Q_{n+1} \neq K \,|\, Q_n = i)\,\pi(i)}{1 - \pi(K)} \\
&= \sum_{i=0}^{K-1} \frac{(1 - P(Q_{n+1} = K \,|\, Q_n = i))\,\pi(i)}{1 - \pi(K)}.
\end{aligned}$$

Recall the definition of $P_{i,k}(t) = P(N(t) = k \,|\, N(0) = i)$, where $N(t)$ is the queue-length at time t when $\gamma = 0$ (no foreground customers). Since the nth foreground customer is accepted in the system when $Q_n = i < K$, we have

$$\begin{aligned}
P(Q_{n+1} = K \,|\, Q_n = i) &= \int_0^\infty P_{i+1,K}(t)\,\gamma\, e^{-t\gamma}\,\mathrm{d}t \\
&= \gamma\, P^*_{i+1,K}(\gamma)
\end{aligned}$$

for $i = 0, 1, \ldots, K-1$, with $P^*_{j,k}(s) = \int_0^\infty e^{-st} P_{j,k}(t)\mathrm{d}t$. Therefore,

$$q_N = 1 - \gamma \sum_{i=0}^{K-1} \frac{P^*_{i+1,K}(\gamma)\,\pi(i)}{1 - \pi(K)}. \tag{1.20}$$

Using Proposition (1.4.1) which gives an expression for $P^*_{j,K}(\gamma)$, (1.20) rewrites

$$q_N = 1 - \left(\frac{\gamma}{\lambda}\right)\left(\frac{1-\rho}{1-\rho^K}\right)\left(\frac{1}{b^{K+1}-a^{K+1}}\right)\left[\frac{a^2\left(1-(\rho a)^K\right)}{(1-a)(1-\rho a)} - \frac{b^2\left(1-(\rho b)^K\right)}{(1-b)(1-\rho b)}\right]. \quad (1.21)$$

Since

$$P(Q_{n+1}=K) = P\Big(Q_{n+1}=K|Q_n=K\Big)P(Q_n=K)+P\Big(Q_{n+1}=K|Q_n\neq K\Big)P(Q_n\neq K)$$

we deduce that P_L, q_L and q_N are linked by the following relationship

$$P_L\left(1-q_L\right) = \left(1-P_L\right)\left(1-q_N\right). \quad (1.22)$$

1.5 The $M+M/D/1/K$ queue

1.5.1 The model

We still consider the model introduced in Section 1.4.1 but we now change the assumption that the service times are exponentially distributed. Instead we will assume that the service times are constant and all equal to $1/\mu$. In Section 1.4.1, we motivated our choice for exponentially distributed service times by the fact that various packet lengths are possible. Taking into consideration that packet lengths may not be so variable to justify the choice of an exponential distribution, we study here another case: the service times are taken to be constant (i.e. all packets have the same length) with value $\sigma = 1/\mu$. Recall the definition of the traffic intensity given in (1.1).

Again, let $\{Q_n\}_{n=1}^{\infty}$ be the process of the number of packets in the queue at time of the nth arrival from foreground source and let $Q = \lim_{n\to\infty} Q_n$. Some preliminary results must be introduced before computing the stationary distribution of Q.

Let $\mathcal{F}(\theta) = E[\exp(-\theta\sigma)]$ ($\Re(\theta) \geq 0$) be the Laplace–Stieltjes transform (LST) of the service time distribution. Since we consider a constant service time, this transform rewrites as $\mathcal{F}(\theta) = \exp(-\theta\sigma)$. For $\rho > 0$ and $|z| \leq 1$, define

$$\begin{aligned}\mathcal{G}_\rho(z) &= \mathcal{F}\left(\frac{\rho\left(1-z\right)}{\bar{\sigma}}\right) - z \\ &= e^{-\rho\left(1-z\right)} - z. \end{aligned} \quad (1.23)$$

For $\rho > 0$, we denote by $z_0(\rho)$ the zero of $\mathcal{G}_\rho(z)$ having the smallest modulus. Also, we denote by $[z^n]f$ the coefficient of z^n in the Taylor series expansion of f.

To express the stationary distribution of Q, we base our calculus on Cohen's analysis of the $M/G/1$ queue with finite waiting room ($K > 1$) [39, Chapter III.6]. Introduce the

parameter B defined as

$$B = 1 + \frac{\rho}{2\pi i} \oint_{D_r} \left(\frac{1}{\mathcal{G}_\rho(z)} \right) \frac{\mathrm{d}z}{z^{K-1}} \tag{1.24}$$

with D_r any circle in the complex plane with center 0 and radius strictly less than $|z_0(\rho)|$. According to the results obtained by Cohen [39, page 575], we have

$$P(Q=0) = \frac{1}{B}, \tag{1.25}$$

$$P(Q=j) = \frac{1}{2\pi i B} \oint_{D_r} \left(\frac{1-z}{\mathcal{G}_\rho(z)} - 1 \right) \frac{\mathrm{d}z}{z^j}, \text{ for } j = 1, \ldots, K-1, \tag{1.26}$$

$$P(Q=K) = \frac{1}{2\pi i B} \oint_{D_r} \left(\frac{\rho - 1}{\mathcal{G}_\rho(z)} + \frac{1}{1-z} \right) \frac{\mathrm{d}z}{z^{K-1}}. \tag{1.27}$$

The integrals in the r.h.s. of (1.24), (1.26) and (1.27) can be evaluated using the theorem of residues [118]. More precisely, the formula which is to be used is

$$\oint_C f(z)\mathrm{d}z = 2\pi i \sum_{k=1}^{n} \mathrm{Res}[f, z_k]$$

where $(z_1, \ldots, z_n)$ are the poles of the meromorphic function[2] f inside the circle C. To calculate a residue, use the identity

$$\mathrm{Res}[f, z_k] = \frac{1}{(m-1)!} \left(\frac{\mathrm{d}^{(m-1)}}{\mathrm{d}z^{m-1}} (z - z_k)^m f(z) \right)_{z=z_k} = [z^{m-1}](z - z_k)^m f(z)$$

where m is the multiplicity of pole z_k. It is easily seen that

$$\frac{1}{2\pi i} \oint_{D_r} \frac{1}{z^j}\, \mathrm{d}z = \mathrm{Res}\left[\frac{1}{z^j}, 0 \right] = [z^{j-1}]1 = \begin{cases} 1, & \text{if } j = 1 \\ 0, & \text{if } j = 2, 3, \ldots \end{cases}$$

$$\frac{1}{2\pi i} \oint_{D_r} \frac{1}{(1-z)\, z^{K-1}}\, \mathrm{d}z = \mathrm{Res}\left[\frac{1}{(1-z)z^{K-1}}, 0 \right] = [z^{K-2}] \frac{1}{1-z} = 1,$$

$$\frac{1}{2\pi i} \oint_{D_r} \frac{1}{\mathcal{G}_\rho(z)\, z^{j-1}}\, \mathrm{d}z = \mathrm{Res}\left[\frac{1}{\mathcal{G}_\rho(z)\, z^{j-1}}, 0 \right] = \begin{cases} 0, & \text{if } j = 1 \\ [z^{j-2}] \frac{1}{\mathcal{G}_\rho(z)}, & \text{if } j = 2, 3, \ldots \end{cases}$$

Finally, from (1.24)–(1.27) and with the help of the previous identities, the distribution of

[2] A meromorphic function is a rational function, i.e. it has no essential singular points; for instance $f(z) = e^{(1/z)}$ is not a meromorphic function since 0 is a singular essential point.

Q is given by

$$\pi_0 = \frac{1}{1+\rho\,\alpha_K(\rho)}, \tag{1.28}$$

$$\pi_1 = \frac{\alpha_2(\rho)-1}{1+\rho\,\alpha_K(\rho)}, \tag{1.29}$$

$$\pi_j = \frac{\alpha_{j+1}(\rho)-\alpha_j(\rho)}{1+\rho\,\alpha_K(\rho)}, \quad j=2,\ldots,K-1, \tag{1.30}$$

$$\pi_K = \frac{1+(\rho-1)\,\alpha_K(\rho)}{1+\rho\,\alpha_K(\rho)}, \tag{1.31}$$

where we have defined

$$\alpha_j(\rho) := [z^{j-2}]\frac{1}{\mathcal{G}_\rho(z)} = \frac{1}{(j-2)!}\left(\frac{\mathrm{d}^{(j-2)}}{\mathrm{d}z^{j-2}}\,\frac{1}{\mathcal{G}_\rho(z)}\right)_{z=0} = \frac{1}{2\pi i}\oint_{D_r}\frac{1}{\mathcal{G}_\rho(z)\,z^{j-1}}\,\mathrm{d}z. \tag{1.32}$$

When the service times are constant, an analytical expression for $\alpha_j(\rho)$ can be derived. To this end, start from

$$\frac{1}{\mathcal{G}_\rho(z)} = \frac{1}{e^{-\rho\,(1-z)}-z} = e^{\rho\,(1-z)}\sum_{i\geq 0}(ze^{\rho\,(1-z)})^i$$

where the last equality is true for z such that $|ze^{\rho\,(1-z)}|<1$, which yields

$$\begin{aligned}\frac{1}{\mathcal{G}_\rho(z)} &= \sum_{i\geq 0} z^i e^{\rho\,(i+1)}\,e^{-\rho\,(i+1)z} = \sum_{i\geq 0}\sum_{k\geq 0} z^i e^{\rho\,(i+1)}\,\frac{(-\rho\,(i+1)z)^k}{k!}\\ &= \sum_{n\geq 0}\left(\sum_{i+k=n}\frac{e^{\rho\,(i+1)}\,(-1)^k\,\rho^k\,(i+1)^k}{k!}\right)z^n.\end{aligned}$$

Hence,

$$\alpha_j(\rho) = \sum_{i+k=j-2}\frac{e^{\rho\,(i+1)}(-1)^k\rho^k(i+1)^k}{k!} = \sum_{k=0}^{j-2}\frac{e^{\rho\,(j-k-1)}\,(-1)^k\,\rho^k\,(j-k-1)^k}{k!},\ j\geq 2. \tag{1.33}$$

1.5.1.1 Alternative computation of $\alpha_j(\rho)$

Computing $\alpha_j(\rho)$ as in (1.33) becomes rapidly intractable because of the factorial in the denominator. An alternative calculation consists of computing the integral in (1.32). The key point is to determine $r<r_0:=|z_0(\rho)|$, the radius of the circle over which the integral is to be computed. Recall that $z_0(\rho)$ is the zero of $\mathcal{G}_\rho(z)$ having the smallest modulus. For general service times, it is known from the Lemma of Takàcs [114, pages 47-49] that:

- $z_0(\rho)$ is a continuous function of ρ,
- $z_0(\rho) = 1$ if $\rho \leq 1$,
- $z_0(\rho) < 1$ if $\rho > 1$,
- $z_0(\rho)$ is the only zero inside the unit circle if $\rho > 1$,
- for $\rho = 1$, the multiplicity of the zero $z_0(1) = 1$ is 2.

When the service times are deterministic, the zeros of $\mathcal{G}_\rho(z)$ are simply a Lambert W series. For later use, we briefly review some of the properties of the Lambert W function [40]. By definition, the Lambert W function satisfies:

$$W(x)e^{W(x)} = x.$$

As this equation has an infinite number of solutions for each (non-zero) value of x, $W(x)$ has an infinite number of branches. Denote the kth branch by $W_k(x)$. If x is real, then for $-1/e \leq x < 0$, there are two possible (negative) *real* values of $W(x)$. The branch satisfying $-1 \leq W(x) < 0$ is denoted by $W_0(x)$ and referred to as the principal branch of the W function; and the branch satisfying $W(x) \leq -1$ is denoted by $W_{-1}(x)$. Note that $W_0(x)$ and $W_{-1}(x)$ are the only branches that take on real values and that $W_0(-1/e) = W_{-1}(-1/e) = -1$.

Let us now return to the function $\mathcal{G}_\rho(z)$. Rewriting it as follows:

$$\mathcal{G}_\rho(z) = -\frac{e^{\rho z}}{\rho}\Big(- \rho e^{-\rho} + \rho z e^{-\rho z}\Big),$$

it becomes clear that the zeros of $\mathcal{G}_\rho(z)$ satisfy the following equation:

$$-\rho z\, e^{-\rho z} = -\rho e^{-\rho}.$$

Thus, the zeros of $\mathcal{G}_\rho(z)$ are

$$z(\rho) := -\frac{1}{\rho} W_k\left(-\rho e^{-\rho}\right), \text{ for } k \in \mathbb{Z}. \tag{1.34}$$

Figure 1.3 depicts the modulus of the zeros $z(\rho)$ for $k \in [-4, 3] \cap \mathbb{Z}$ as a function of the load ρ. The zero having the smallest modulus is given by the principal branch of the Lambert function (drawn in red in Figure 1.3) which is real for any $\rho > 0$ (Note: $-\rho \exp(-\rho)$ is real and $-1/e \leq -\rho \exp(-\rho) < 0$ hence $W_0(-\rho \exp(-\rho))$ is real and negative), thus

$$r_0 = z_0(\rho) = -\frac{1}{\rho} W_0\left(-\rho e^{-\rho}\right)$$

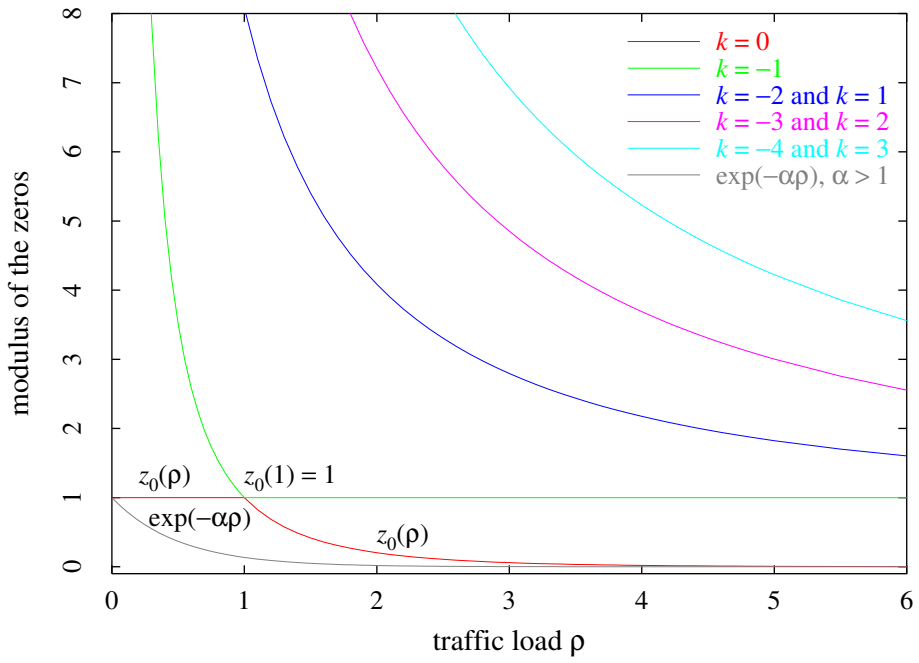


Figure 1.3: Modulus of the zeros of $\mathcal{G}_\rho(z)$, $-\frac{1}{\rho}W_k\left(-\rho e^{-\rho}\right)$, vs. the traffic load ρ

As expected, we find that $z_0(\rho) = 1$ if $\rho \leq 1$; $z_0(\rho) < 1$ if $\rho > 1$; $z_0(1) = 1$ has a multiplicity 2 (because $W_0(-1/e) = W_{-1}(-1/e) = -1$); and for $\rho > 1$, $z_0(\rho)$ is the only zero inside the unit circle. Furthermore, we have that $z_0(\rho)$ is a continuous real function of ρ.

We want to find now a real number r which is smaller than r_0 for any ρ. It can be easily proved that the function $\exp(-\alpha\rho)$ for $\alpha > 1$ serves well, as illustrated in Figure 1.3. Indeed, to have $\exp(-\alpha\rho) < z_0(\rho)$ for $\rho > 0$, it suffices to have $W_0(-\rho\exp(-\rho)) < \rho(\alpha - 1)$. The latter condition is always satisfied for $\alpha > 1$, since $W_0(-\rho\exp(-\rho)) < 0$.

Now that we have found a possible value for the radius of the circle D_r (take $r = \exp(-2\rho)$ for instance), letting $z = r\exp(i\theta) = r\cos\theta + ir\sin\theta$ in (1.32) yields

$$\alpha_j(\rho) = \frac{1}{2\pi r^{j-2}} \int_0^{2\pi} \frac{e^{-i\theta(j-2)}\overline{\mathcal{G}_\rho(\theta)}}{\left|\mathcal{G}_\rho(\theta)\right|^2}\,\mathrm{d}\theta, \tag{1.35}$$

where $\overline{\mathcal{G}_\rho(\theta)}$ denotes the conjugate of $\mathcal{G}_\rho(\theta)$ in the complex plane and $|\mathcal{G}_\rho(\theta)|$ denotes its modulus. We can write:

$$\begin{aligned}
\mathcal{G}_\rho(\theta) &= e^{-\rho(1-r\exp(i\theta))} - re^{i\theta} \\
\overline{\mathcal{G}_\rho(\theta)} &= e^{-\rho(1-r\exp(-i\theta))} - re^{-i\theta} \\
\left|\mathcal{G}_\rho(\theta)\right|^2 &= r^2 + e^{-2\rho(1-r\cos\theta)} - 2re^{-\rho(1-r\cos\theta)}\cos[\theta - \rho r\sin\theta]
\end{aligned}$$

$$\begin{aligned} e^{-i\theta(j-2)}\overline{\mathcal{G}_\rho(\theta)} &= e^{-\rho(1-r\cos\theta)}\, e^{-i(\rho r\sin\theta+\theta(j-2))} - re^{-i\theta(j-1)} \\ &= \mathrm{Real}_j(\theta) + i\,\mathrm{Imag}_j(\theta), \end{aligned}$$

where

$$\begin{aligned} \mathrm{Real}_j(\theta) &:= e^{-\rho(1-r\cos\theta)}\cos[\rho r\sin\theta+\theta(j-2)] - r\cos[\theta(j-1)] \\ \mathrm{Imag}_j(\theta) &:= -e^{-\rho(1-r\cos\theta)}\sin[\rho r\sin\theta+\theta(j-2)] + r\sin[\theta(j-1)]. \end{aligned}$$

It is seen that $\mathrm{Real}_j(\theta)$ and $|\mathcal{G}_\rho(\theta)|^2$ are even functions in θ and that $\mathrm{Imag}_j(\theta)$ is odd. Equation (1.35) reduces then to

$$\alpha_j(\rho) = \frac{1}{\pi r^{j-2}}\int_0^\pi \frac{\mathrm{Real}_j(\theta)}{|\mathcal{G}_\rho(\theta)|^2}\mathrm{d}\theta. \tag{1.36}$$

The integral in (1.36) is easily computed using a numerical procedure provided by the NAG[3] C library [91].

1.5.1.2 General remarks

Instead of taking $r = \exp(-\alpha\rho)$ with $\alpha > 1$, it is possible to assign to r any value strictly less than 1 when $\rho \leq 1$ (letting $r = 0.5$ rather than $r = 0.9$ leads to fast convergence when computing numerically the integral). Let us now investigate the case where $\rho > 1$. Notice that $\mathcal{G}_\rho(z)$ has a unique negative real minimum at $1 - \log\rho/\rho$ and that $\mathcal{G}_\rho(0) = \exp(-\rho)$ is positive. Hence, we are sure that r_0 is in the interval $[0, 1 - \log\rho/\rho]$ and may compute r iteratively, starting at $1 - \log\rho/\rho$, and decrementing r until $\mathcal{G}_\rho(r) > 0$.

Since the stationary distribution exists, the coefficients α_j for $j = 2, \ldots, K$ satisfy the following conditions:

- $1 + \rho\,\alpha_K(\rho) > 1$ (existence of π_0),
- $0 < \alpha_2(\rho) - 1 < 1 + \rho\,\alpha_K(\rho)$ (existence of π_1),
- $0 < \alpha_{j+1}(\rho) - \alpha_j(\rho) < 1 + \rho\,\alpha_K(\rho)$ for $j = 2, \ldots, K-1$ (existence of π_j),
- $0 < 1 + (\rho - 1)\alpha_K(\rho) < 1 + \rho\,\alpha_K(\rho)$ (existence of π_K),

which are summarized by:

- $1 < \alpha_2(\rho) < \cdots < \alpha_j(\rho) < \alpha_{j+1}(\rho) < \cdots < \alpha_K(\rho)$ for $j = 2, \ldots, K-1$,
- $\alpha_K(\rho)(1 - \rho) < 1$,

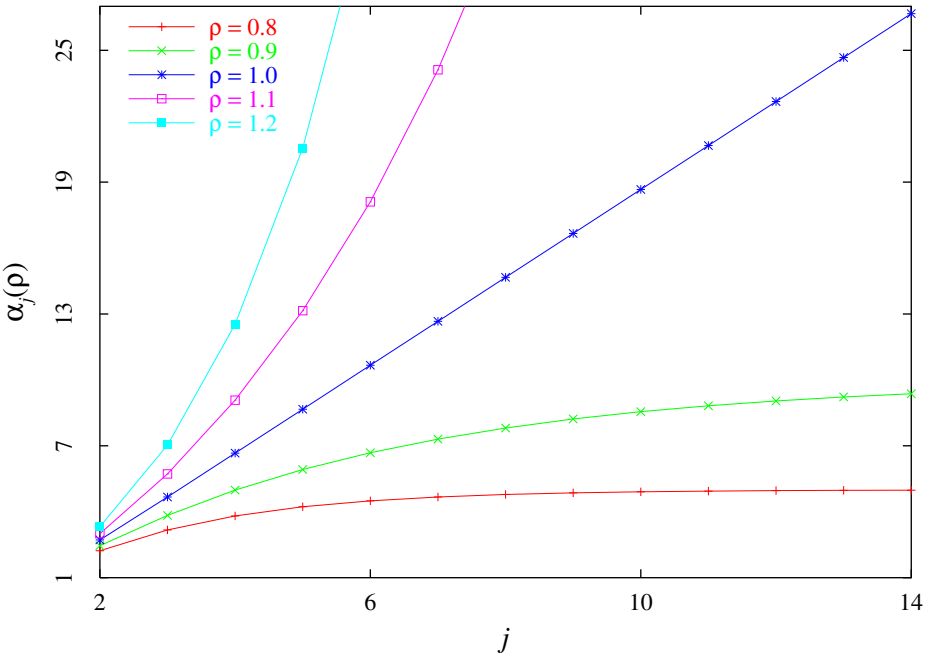


Figure 1.4: The parameter $\alpha_j(\rho)$ for $j = 2, \ldots, K$ and for several values of ρ

- $\alpha_{j+1}(\rho) - \alpha_j(\rho) < 1 + \rho\,\alpha_K(\rho)$ for $j = 2, \ldots, K-2$,

Last, we observe that the function $j \mapsto \alpha_j(\rho)$ is concave increasing for $\rho < 1$, convex increasing for $\rho > 1$ and linear increasing for $\rho = 1$ as plotted in Figure 1.4.

1.5.2 The loss probability

Recall the definition of X_n introduced in Section 1.4.2, $X_n = \mathbf{1}\{Q_n = K\}$ and $X = \lim_{n\to\infty} X_n$. A customer n is lost whenever $X_n = 1$ and is not lost otherwise.

The probability that a foreground customer is lost is the probability that it finds the system full upon arrival, namely,

$$\begin{aligned} P_L &:= P(X = 1) = P(Q = K) \\ &= \frac{1 + (\rho - 1)\,\alpha_K(\rho)}{1 + \rho\,\alpha_K(\rho)}. \end{aligned} \tag{1.37}$$

Notice that in order to have $0 < P_L < 1$, the following condition must be satisfied:

$$0 < \frac{\alpha_K(\rho)}{1 + \rho\,\alpha_K(\rho)} < 1.$$

[3]NAG is a copyright of The Numerical Algorithms Group Ltd

The first inequality says that $\alpha_K(\rho)$ and $1 + \rho\,\alpha_K(\rho)$ must have the same sign. This is always true since both quantities are greater than 1.

The second inequality says that $\alpha_K(\rho) < 1/(1-\rho)$ in the case $\rho < 1$ and $\alpha_K(\rho) > 1/(1-\rho)$ otherwise. The latter is always true, as $\alpha_K(\rho) > 1$ and $1/(1-\rho) < 0$. When $\rho < 1$ and for large values of K, the coefficients α_K become close to each others and very close to $1/(1-\rho)$. A lack of precision during the computation of $\alpha_K(\rho)$ may lead to a negative value for P_L instead of a negligible positive value.

1.5.3 The server utilization

The utilization U of the server was defined as the probability of a non-empty queue. The server utilization is

$$\begin{aligned} U &:= P(Q > 0) \\ &= \frac{\rho\,\alpha_K(\rho)}{1 + \rho\,\alpha_K(\rho)}. \end{aligned} \tag{1.38}$$

Notice that the condition

$$0 < U = \frac{\rho\,\alpha_K(\rho)}{1 + \rho\,\alpha_K(\rho)} < 1$$

is always satisfied.

1.5.4 The expected response time

Applying Little's formula and the PASTA property to the queue, we find that the expected response time R is given by

$$R = \frac{\sum_{j=1}^{K} j\,\pi(j)}{(\lambda+\gamma)(1-\pi_K)}. \tag{1.39}$$

Using (1.29), (1.30) and (1.31), (1.39) rewrites

$$R = \frac{K(1+\rho\alpha_K(\rho)) - \left[1 + \sum_{j=2}^{K} \alpha_j(\rho)\right]}{\mu\,\rho\,\alpha_K(\rho)} \tag{1.40}$$

with $\alpha_j(\rho)$ defined in (1.32).

1.6 Using the inference models

1.6.1 An inference question

Until now we have introduced two models for a connection. In the first model, we were able to identify five metrics describing the quality of service provided to the foreground source, P_L, U, R, q_L and q_N, given in (1.3), (1.5), (1.10), (1.18) and (1.21). In the second model, we were able to identify three QoS metrics, P_L, U and R, given in (1.37), (1.38) and (1.40). Since $\rho = (\lambda + \gamma)/\mu$, all these equations are expressed in terms of the parameters λ, μ and K (the probing rate γ is known).

The problem is therefore the following: How can we infer estimates $\hat{\lambda}_n$, $\hat{\mu}_n$ and $\hat{K}_n$ of parameters λ, μ and K, respectively, from the observations collected from the first n probes?

If the parameters λ, μ and K are unknown, then (1.3), (1.5), (1.10), (1.18) and (1.21) leave us with nine schemes to compute these three constants in the $M/M/1/K$ case (there are $C_5^3 = 10$ possible schemes but relation (1.22) reduces that number to nine); only one scheme is available in the $M/D/1/K$ case. These ten schemes are listed in Table 1.1, where the notation X_Y_Z denotes the scheme obtained by using the metrics X, Y and Z in the $M/M/1/K$ case and where $P_L\_U\_R\_D$ denotes the scheme obtained by using the metrics P_L, U and R in the $M/D/1/K$ case.

Table 1.1: Schemes for estimating λ, μ and K

Scheme	Reference	Equations to use
$P_L\_U\_R$	*1*	(1.3), (1.5), (1.10)
$P_L\_U\_q_L$	*2*	(1.3), (1.5), (1.18)
$P_L\_U\_q_N$	*3*	(1.3), (1.5), (1.21)
$P_L\_R\_q_L$	*4*	(1.3), (1.10), (1.18)
$P_L\_R\_q_N$	*5*	(1.3), (1.10), (1.21)
$U\_R\_q_L$	*6*	(1.5), (1.10), (1.18)
$U\_R\_q_N$	*7*	(1.5), (1.10), (1.21)
$U\_q_L\_q_N$	*8*	(1.5), (1.18), (1.21)
$R\_q_L\_q_N$	*9*	(1.10), (1.18), (1.21)
$P_L\_U\_R\_D$	*10*	(1.37), (1.38), (1.40)

If we now assume that only λ and K are unknown (μ being estimated for instance, using *pathrate* [46], PBM [94] or ROPP [77], see Section 1.2), then $C_5^2 = 10$ schemes in the $M/M/1/K$ case and one scheme in the $M/D/1/K$ case (Equation (1.40) can be used only if (1.37)–(1.38) are also used) can be used to compute these two constants, resulting in a total of eleven schemes. These eleven schemes are listed in Table 1.2, where the notation X_Y denotes the scheme obtained by using the metrics X and Y in the $M/M/1/K$ case

and where $P_L\_U\_D$ denotes the scheme obtained by using the metrics P_L and U in the $M/D/1/K$ case.

Table 1.2: Schemes for estimating λ and K (μ assumed to be already known/estimated)

Scheme	Reference	Equations to use
P_L_U	**I**	(1.3), (1.5)
P_L_R	**II**	(1.3), (1.10)
$P_L\_q_L$	**III**	(1.3), (1.18)
$P_L\_q_N$	**IV**	(1.3), (1.21)
U_R	**V**	(1.5), (1.10)
U_q_L	**VI**	(1.5), (1.18)
U_q_N	**VII**	(1.5), (1.21)
R_q_L	**VIII**	(1.10), (1.18)
R_q_N	**IX**	(1.10), (1.21)
$q_L\_q_N$	**X**	(1.18), (1.21)
$P_L\_U\_D$	**XI**	(1.37), (1.38)

1.6.2 Solving for the equations

Up to now, we have defined two groups of schemes. The schemes, denoted by the italic numbers *1*,...,*10*, have three QoS metrics as inputs and can be used to simultaneously estimate parameters λ, μ and K. The schemes, denoted by the roman numbers **I**,...,**XI**, have two QoS metrics as inputs and can be used to estimate parameters λ and K solely, parameter μ being known/estimated beforehand by using some other technique (the *pathrate* tool for instance). Assume that the QoS metrics involved in a scheme can be evaluated from measurements collected at the sender/receiver (cf. Section 1.6.3). Then, estimators for λ, μ and K (or for λ and K only) are obtained by "solving" the scheme with respect to (w.r.t.) the variables λ, μ and K (or w.r.t. the variables λ and K).

If we want to apply a certain scheme to estimate the buffer size, the intensity of the cross traffic, and possibly the server capacity, we must establish existence and uniqueness of its solution (λ, μ, K) or (λ, K). To be more precise, consider for instance scheme **I** that involves the loss probability P_L and the server utilization U (μ is known). Then, for any (measured) values of P_L and U with $0 < P_L < 1$ and $0 < U < 1$, we want to find a single pair (λ, K) satisfying the set of equations defined by (1.3) and (1.5). This existence and uniqueness property holds for scheme *1*, as shown below, as well as for schemes *10*, **I**, **II**, **V** and **XI**. As for the other schemes we have not been able to show that property, but in all experiments that have been carried out, and that are reported in Section 1.7, each scheme always gave us a unique solution. We now discuss the solution to scheme *1*, and indicate how the solution of scheme *10* can be derived. We next discuss the solution to scheme **I**,

show the existence and uniqueness of the solution for schemes **II** and **V**, and indicate how the solution of scheme **XI** can be found.

1.6.2.1 Solving for scheme *1*

The equations involved here are (1.3), (1.5) and (1.10), namely,

$$P_L = \frac{(1-\rho)\,\rho^K}{1-\rho^{K+1}},$$
$$U = \rho\left(\frac{1-\rho^K}{1-\rho^{K+1}}\right) \;=\; \rho\,(1-P_L),$$
$$R = \frac{1}{\mu\,(1-\rho)} - \frac{K}{\mu}\,\frac{\rho^K}{1-\rho^K} \;=\; \frac{1}{\mu\,(1-\rho)}\left(1-\frac{K\,P_L}{1-P_L}\right).$$

These equations produce the following expressions to ρ and μ,

$$\rho = \frac{U}{1-P_L} \qquad \text{i.e.} \qquad \lambda \;=\; \frac{\mu\,U}{1-P_L} - \gamma \tag{1.41}$$
$$\mu = \frac{1}{R\,(1-\rho)}\left(1-\frac{K\,P_L}{1-P_L}\right) \;=\; \frac{1-P_L(K+1)}{R\,(1-P_L-U)}. \tag{1.42}$$

Combining (1.41) and (1.7) yields an additional expression to K,

$$K = \frac{\log(P_L/(1-U))}{\log(U/(1-P_L))}. \tag{1.43}$$

1.6.2.2 Solving for scheme *10*

This scheme still involves P_L, U and R, but this time these quantities have to be computed for the $M{+}M/D/1/K$ queue. More precisely, cf. (1.37), (1.38) and (1.40),

$$P_L = \frac{1+(\rho-1)\,\alpha_K(\rho)}{1+\rho\,\alpha_K(\rho)}, \tag{1.44}$$
$$U = \frac{\rho\,\alpha_K(\rho)}{1+\rho\,\alpha_K(\rho)}, \tag{1.45}$$
$$R = \frac{K(1+\rho\alpha_K(\rho)) - \left[1+\sum_{j=2}^{K}\alpha_j(\rho)\right]}{\mu\,\rho\,\alpha_K(\rho)} \tag{1.46}$$

with $\alpha_K(\rho)$ given in (1.32).

Recall that we want to solve the system of equations (1.44)–(1.46) with respect to the variables λ, μ and K. We readily observe from (1.44)–(1.46) that

$$\rho = \frac{U}{1-P_L} \qquad \text{i.e.} \qquad \lambda = \frac{\mu\, U}{1-P_L} - \gamma, \tag{1.47}$$

$$\alpha_K(\rho) = \frac{1-P_L}{1-U}, \tag{1.48}$$

$$\mu = \frac{K - (1-U)\left[1 + \sum_{j=2}^{K} \alpha_j(\rho)\right]}{RU}. \tag{1.49}$$

Since all coefficients $\{\alpha_j(\rho), j \geq 2\}$ in the Taylor series expansion of $1/\mathcal{G}_\rho(z)$ are different (see Section 1.5.1.2), then (1.47)–(1.48) returns a unique solution (ρ, K).

For a given ρ, we computed the coefficients $\alpha_j(\rho)$ for a certain range of j and compared the results with the r.h.s. of (1.48); then K was chosen as the integer j for which $\alpha_j(\rho)$ was the closest to (the measured value of) $(1-P_L)/(1-U)$.

Having K at our disposal, as well as the coefficients $\{\alpha_j(\rho), 2 \leq j \leq K\}$, it is then possible to infer μ using (1.49), and subsequently λ using (1.47).

1.6.2.3 Solving for scheme I

The equations involved here are (1.3) and (1.5), namely,

$$P_L = \frac{(1-\rho)\,\rho^K}{1-\rho^{K+1}},$$

$$U = \rho\left(\frac{1-\rho^K}{1-\rho^{K+1}}\right) = \rho\,(1-P_L).$$

These yield the following expressions to ρ and K,

$$\rho = \frac{U}{1-P_L}, \quad \text{i.e.} \quad \lambda = \frac{\mu\, U}{1-P_L} - \gamma, \tag{1.50}$$

$$K = \frac{\log(P_L/(1-U))}{\log(U/(1-P_L))}, \tag{1.51}$$

where the last equation is obtained by combining (1.50) and (1.7). Therefore, the set of equations (1.3) and (1.5) in the variables λ and K has a unique solution given in (1.50) and (1.51), respectively.

It is interesting to investigate the sensitivity of λ and K with respect to the variables P_L and U. To do so, let us compute the differentials of λ and K considered as functions of

P_L and U. From (1.50) and (1.51) we find

$$\mathrm{d}\lambda = \frac{\mu}{1-P_L}\left(\mathrm{d}U + \frac{U}{1-P_L}\,\mathrm{d}P_L\right),$$
$$\mathrm{d}K = \frac{1}{\log^2(U/(1-P_L))}\Big(A\,\mathrm{d}U + B\,\mathrm{d}P_L\Big),$$

where

$$A = \frac{\log(U/(1-P_L))}{1-U} - \frac{\log(P_L/(1-U))}{U},$$
$$B = \frac{\log(U/(1-P_L))}{P_L} - \frac{\log(P_L/(1-U))}{1-P_L}.$$

Using (1.3) and (1.5), A and B are rewritten as follows:

$$A = \frac{\log\rho(1-\rho^{K+1})}{\rho^K(1-\rho)(1-\rho^K)}\Big(\rho^K(1-\rho^K) - K\rho^{K-1}(1-\rho)\Big),$$
$$B = \frac{\log\rho(1-\rho^{K+1})}{\rho^K(1-\rho)(1-\rho^K)}\Big(1-\rho^K - K\rho^K(1-\rho)\Big).$$

We find that $A > 0$ and $B < 0$ for any value of ρ. Comparing $|A|$ and $|B| = -B$, we obtain $|A| > |B|$ for $\rho > 1$ and $|A| < |B|$ for $\rho < 1$. We deduce that K follows primarily U's variations for $\rho > 1$ (since $|A| > |B|$) and it is more influenced by P_L's variations than by U's variations for $\rho < 1$ (since $|A| < |B|$). As for λ, it is more sensitive to the variations of P_L (resp. U) than to the variations of U (resp. P_L) whenever $\rho = U/(1-P_L) > 1$ (resp. $\rho < 1$). When $\rho = 1$ ($U = 1 - P_L$), both λ and K are equally influenced by P_L and U's variations since in this case

$$\mathrm{d}\lambda = \frac{\mu}{1-P_L}\Big(\mathrm{d}U + \mathrm{d}P_L\Big),$$
$$\mathrm{d}K = \frac{1}{2}\,(K+1)^2(\mathrm{d}U - \mathrm{d}P_L).$$

In other words, when $\rho > 1$ (resp. $\rho < 1$), fast convergence in the estimation of P_L (resp. of U) leads to fast convergence in the estimation of λ while slow convergence in the estimation of U (resp. of P_L) leads to slow convergence in the estimation of K.

1.6.2.4 Solving for scheme II

Assume that one knows R and P_L and that ρ and K are unknown. The expression

for P_L (1.3) yields the following expression for K in terms of ρ and P_L,

$$K = \frac{\log(P_L/(1-\rho\,(1-P_L)))}{\log\rho}. \tag{1.52}$$

Substituting this value of K into (1.9) yields

$$R = \frac{1}{\mu\,(1-\rho)} - \frac{P_L\,\log(P_L/(1-\rho\,(1-P_L)))}{\mu\,(1-\rho)\,(1-P_L)\,\log\rho}.$$

Observe that necessarily $\rho < 1/(1-P_L)$. For $0 < x < 1/(1-P_L)$, we introduce the mapping

$$f(x) := \frac{1}{\mu\,(1-x)} - \frac{P_L\,\log(P_L/(1-x\,(1-P_L)))}{\mu\,(1-x)\,(1-P_L)\,\log x} - R. \tag{1.53}$$

If one can show that the equation $f(x) = 0$ has a unique solution in $(0, 1/(1-P_L))$, then this solution yields ρ, hence λ, and subsequently K by using (1.52). Proposition 1.6.1 shows that this is indeed the case.

Proposition 1.6.1 *For any constants $\mu > 0$, $P_L \in (0,1)$ and $R \geq 1/\mu$, the equation $f(x) = 0$ has a unique solution in $[0, 1/(1-P_L)]$.* ♦

Proof. Define

$$g(x) := (1-P_L)\,(1-R\mu\,(1-x))\log x - P_L\,\log P_L + P_L\,\log(1-x\,(1-P_L)) \tag{1.54}$$

for $0 < x < 1/(1-P_L))$. Hence, cf. (1.53),

$$f(x) := \frac{g(x)}{\mu\,(1-P_L)(1-x)\log x}. \tag{1.55}$$

Denote by $g^{(1)}(x)$ (resp. $g^{(2)}(x)$) the first (resp. second) order derivative of $g(x)$. We find

$$g^{(1)}(x) = (1-P_L)\,R\mu\left(\log x - \frac{(1-P_L)(x-1)(x+a)}{x\,(1-x(1-P_L))}\right) \tag{1.56}$$

with

$$a := \frac{R\mu - 1}{R\mu\,(1-P_L)} \geq 0, \tag{1.57}$$

and

$$g^{(2)}(x) = \frac{(1-P_L)R\mu\,h(x)}{x^2\,(1-x(1-P_L))^2} \tag{1.58}$$

with

$$\begin{aligned} h(x) &:= (1-P_L)^2\,x^3 + ((1+a)\,P_L^2 + (1-2a)\,P_L - (2-a))x^2 \\ &\quad + (1-2a\,(1-P_L)^2)\,x + a(1-P_L). \end{aligned} \tag{1.59}$$

From the identities

$$g(1) = 0, \qquad g^{(1)}(1) = 0 \qquad \text{and} \qquad g^{(2)}(1) = \left(\frac{1-P_L}{P_L}\right)(2R\mu P_L - 1)$$

we see, by applying l'Hôpital's rule to the r.h.s. of (1.55), that $f(1)$ is well-defined, with value

$$f(1) = -\frac{g^{(2)}(1)}{2\mu(1-P_L)} = -\frac{2R\mu P_L - 1}{2\mu P_L}. \tag{1.60}$$

Therefore, unless $g^{(2)}(1) = 0$, the zeros of $f(x)$ in $(0, 1/(1-P_L))$ are the zeros of $g(x)$ in $(0, 1/(1-P_L)) \setminus \{1\}$; if $g^{(2)}(1) = 0$, then $f(x)$ and $g(x)$ have the same zeros in $(0, 1/(1-P_L))$.

We now show that $g(x)$ has a unique zero in $(0, 1/(1-P_L)) \setminus \{1\}$ when $g^{(2)}(1) \neq 0$ and that $g(x)$ has a unique zero in $(0, 1/(1-P_L))$, located at the point $x = 1$, when $g^{(2)}(1) = 0$, which will conclude the proof.

Assume first that $a \neq 0$. It is then seen from the identities

$$\begin{array}{ll} h(0) = a\,(1-P_L) > 0, & \lim_{x\to-\infty} h(x) = -\infty, \\ h(1/(1-P_L)) = -\frac{1}{(1-P_L)R\mu} < 0, & \lim_{x\to+\infty} h(x) = +\infty, \end{array} \tag{1.61}$$

that the polynomial $h(x)$ of degree 3 in the variable x has exactly one root $x = \rho(a)$ in $(0, 1/(1-P_L))$. Combining (1.61) and (1.58) yields

$$\begin{array}{ll} g^{(2)}(x) > 0, \text{ for } 0 < x < \rho(a), & \lim_{x\to 0} g^{(2)}(x) = \infty, \\ g^{(2)}(x) < 0, \text{ for } \rho(a) < x < 1/(1-P_L), & \lim_{x\to 1/(1-P_L)} g^{(2)}(x) = -\infty, \\ g^{(2)}(\rho(a)) = 0. & \end{array}$$

Assume now that $a = 0$. Then, cf. (1.58)–(1.59),

$$g^{(2)}(x) = \frac{(1-P_L)\,R\mu}{x\,(1-x(1-P_L))^2}\,k(x) \tag{1.62}$$

with

$$k(x) := (1-P_L)^2\,x^2 + (P_L^2 + P_L - 2)x + 1. \tag{1.63}$$

From

$$k(0) = 1, \qquad k(1/(1-P_L)) = -P_L, \qquad \text{and} \qquad \lim_{x\to\infty} k(x) = \infty,$$

we deduce that the polynomial $k(x)$, of degree 2 in the variable x, has exactly one zero, $x = \rho(0)$, in $(0, 1/(1 - P_L))$. This in turn implies from (1.62) that

$$\begin{array}{ll} g^{(2)}(x) > 0, \text{ for } 0 < x < \rho(0), & \lim_{x\to 0} g^{(2)}(x) = \infty, \\ g^{(2)}(x) < 0, \text{ for } \rho(0) < x < 1/(1 - P_L), & \lim_{x\to 1/(1-P_L)} g^{(2)}(x) = -\infty, \\ g^{(2)}(\rho(0)) = 0, & \end{array}$$

In summary, we have shown that for all $a \geq 0$, $g^{(2)}(x)$ has a unique zero $x = \rho(a)$ in $(0, 1/(1 - P_L))$; furthermore, $g^{(2)}(x) > 0$ for $0 < x < \rho(a)$ and $g^{(2)}(x) < 0$ for $\rho(a) < x < 1/(1 - P_L))$.

From the above we may now determine the variations of the function $g(x)$ in $(0, 1/(1 - P_L))$. This is done in Figures (1.5)–(1.7) by distinguishing the cases when (i) $0 < \rho(a) < 1$, (ii) $1 < \rho(a) < 1/(1 - P_L)$ and (iii) $\rho(a) = 1$.

- $0 < \rho(a) < 1$

x	0		$\rho(a)$		1	$(1-P_L)^{-1}$
$g^{(2)}(x)$	$+\infty$	$+$	0	$-$	$-$	$-\infty$
$g^{(1)}(x)$	$-\infty$	$-$	0	$+$	0	$-$ $-\infty$
$g(x)$	$+\infty$	$+$ 0		$-$	0	$-$ $-\infty$
$1-x$	$+$		$+$		0	$-$
$\log x$	$-$		$-$		0	$+$
$f(x)$	$-$	0	$+$	$f(1)$	$+$	

Figure 1.5: $0 < \rho(a) < 1$

We conclude from Figure 1.5 that $g(x)$ has a unique zero in $(0, 1/(1 - P_L)) \setminus \{1\}$ when $0 < \rho(a) < 1$. This zero is located in $(0, \rho(a))$

- $1 < \rho(a) < 1/(1 - P_L)$

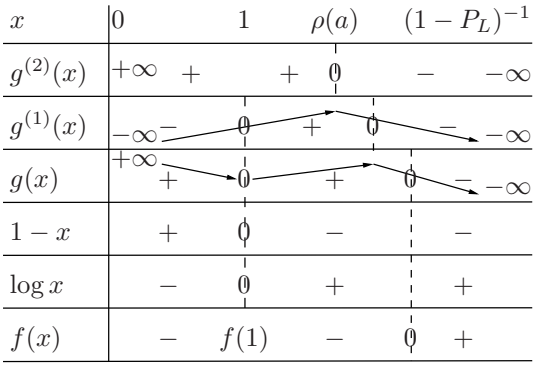


Figure 1.6: $1 < \rho(a) < 1/(1 - P_L)$

We conclude from Figure 1.6 that $g(x)$ has a unique zero in $(0, 1/(1-P_L)) \setminus \{1\}$ when $0 < \rho(a) < 1$. This zero is located in $(\rho(a), 1/(1 - P_L))$.

- $\rho(a) = 1$

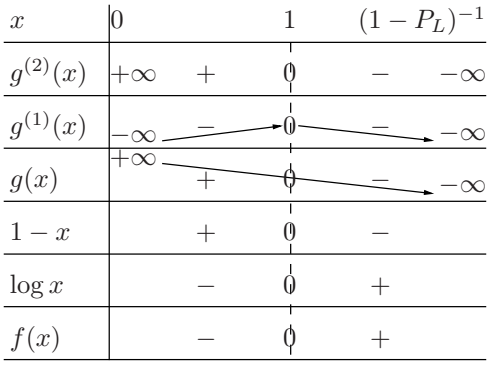


Figure 1.7: $\rho(a) = 1$

In this case, $x = 1$ in the only zero of $g(x)$ in $(0, 1/(1 - P_L))$ (cf. Figure 1.7). This is a zero of multiplicity 3.

■

It is interesting to investigate the sensitivity of λ and K with respect to the variables R and P_L. To do so, let us compute the differentials of λ and K considered as functions of R and P_L. From (1.10) and (1.3) we have

$$\mathrm{d}R = C\,\mathrm{d}\lambda + D\,\mathrm{d}K \tag{1.64}$$

$$\mathrm{d}P_L = E\,\mathrm{d}\lambda + F\,\mathrm{d}K \tag{1.65}$$

with

$$C = \frac{\partial R}{\partial \rho}\frac{\partial \rho}{\partial \lambda} = \frac{1}{\mu^2(1-\rho)^2} - \frac{K^2\rho^{K-1}}{\mu^2(1-\rho^K)^2},$$
$$D = \frac{\partial R}{\partial K} = -\frac{\rho^K(1+K\log\rho-\rho^K)}{\mu(1-\rho^K)^2},$$
$$E = \frac{\partial P_L}{\partial \rho}\frac{\partial \rho}{\partial \lambda} = \frac{\rho^{K-1}(K(1-\rho)-\rho(1-\rho^K))}{\mu(1-\rho^{K+1})^2},$$
$$F = \frac{\partial P_L}{\partial K} = \frac{\rho^K(1-\rho)\log\rho}{(1-\rho^{K+1})^2}.$$

From (1.64) and (1.65), we can write

$$d\lambda = \frac{F}{CF-DE}\mathrm{d}R - \frac{D}{CF-DE}\mathrm{d}P_L$$
$$dK = -\frac{E}{CF-DE}\mathrm{d}R + \frac{C}{CF-DE}\mathrm{d}P_L.$$

It is seen that $C \geq 0$, $D > 0$, $E > 0$ and $F < 0$ for any $\rho > 0$ and any $K \geq 1$. To identify which parameter affects the most λ (resp. K), one should compare $-F$ to D (resp. E to C). Unfortunately the sign of $D+F$ (resp. $C-E$) depends on the values of μ, ρ and K, and it is impossible to formally identify which estimator among R and P_L has the biggest impact on the estimation of λ and K.

1.6.2.5 Solving for scheme V

Scheme **V** involves Equations (1.5) and (1.10)

$$U = \rho\left(\frac{1-\rho^K}{1-\rho^{K+1}}\right)$$
$$R = \frac{1}{\mu\,(1-\rho)} - \frac{K}{\mu}\,\frac{\rho^K}{1-\rho^K}. \tag{1.66}$$

The expression for R in (1.9) is useful here

$$R = \frac{1}{\mu\,(1-\rho)} - \frac{K}{\mu}\,\frac{\pi_K}{(1-\rho)\,(1-\pi_K)}. \tag{1.67}$$

Using (1.6), π_K can be expressed as

$$\pi_K = 1 - U/\rho, \tag{1.68}$$

to yield from (1.67)

$$R = \frac{1}{\mu\,(1-\rho)} - \frac{K}{\mu}\,\frac{\rho - U}{(1-\rho)\,U}. \tag{1.69}$$

Recall the relation (1.7) for K

$$K = \frac{1}{\log \rho} \cdot \log\left(\frac{1 - U/\rho}{1-U}\right).$$

Substituting Equation (1.7) for K into (1.69) yields

$$R = \frac{1}{\mu\,(1-\rho)} - \frac{\rho - U}{\mu U\,(1-\rho)\,\log \rho}\,\log\left(\frac{\rho - U}{\rho\,(1-U)}\right). \tag{1.70}$$

Observe that necessarily $\rho \geq U$. For $\rho \geq U$, introduce the mapping

$$f(x) = \frac{1}{\mu\,(1-x)} - \frac{x - U}{\mu U\,(1-x)\,\log x}\,\log\left(\frac{x - U}{x\,(1-U)}\right) - R. \tag{1.71}$$

If one can show that the equation $f(x) = 0$ has a unique solution in $[U, \infty)$, then this solution yields ρ, and subsequently K by using (1.7). Proposition 1.6.2 shows that this is indeed the case.

Proposition 1.6.2 *For any constants μ, U and R such that $\mu > 0$, $0 < U < 1$ and $R \geq 1/\mu$, the equation $f(x) = 0$ has a unique solution in $[U, \infty)$.* ♦

Proof. Define

$$g(x) = U\,\log x - (x - U)\,\log\left(\frac{x - U}{x\,(1-U)}\right) - \mu R U\,(1-x)\,\log x$$

for $x \geq U$. Hence (1.71) rewrites as

$$f(x) = \frac{g(x)}{\mu U\,(1-x)\,\log x}. \tag{1.72}$$

Denote by $g^{(1)}(x)$ (resp. $g^{(2)}(x)$) the first (resp. second) order derivative of $g(x)$. We find

$$g^{(1)}(x) = \mu R U\,\log x - \mu R U\,\frac{1-x}{x} - \log\left(\frac{x - U}{x\,(1-U)}\right) \tag{1.73}$$

and

$$\begin{aligned} g^{(2)}(x) &= \frac{\mu R U}{x} + \frac{\mu R U}{x^2} - \frac{U}{x\,(x-U)} \\ &= \frac{U\,h(x)}{x^2\,(x-U)} \end{aligned} \tag{1.74}$$

with

$$h(x) := \mu R x^2 - [1 - \mu R(1-U)]\, x - \mu R U. \tag{1.75}$$

The function $h(x)$ has two zeros

$$x_1 = \frac{1 - \mu R(1-U) - \sqrt{[1-\mu R(1-U)]^2 + 4\mu^2 R^2 U}}{2\mu R}$$
$$x_2 = \frac{1 - \mu R(1-U) + \sqrt{[1-\mu R(1-U)]^2 + 4\mu^2 R^2 U}}{2\mu R}.$$

It is clear that $x_1 \leq 0$. As for x_2, observe that $h(U) = -U \leq 0$. Since $h(x_2) = 0$ and $\lim_{x\to\pm\infty} h(x) = +\infty$, it follows that $x_2 \geq U$. Hence $h(x)$ has only one zero in $[U, \infty)$.

Looking at expression (1.72), we can say that, unless $x = 1$, the zeros of $g(x)$ are the zeros of $f(x)$. For $x = 1$, we have $g(1) = 0$, $g^{(1)}(1) = 0$ and $g^{(2)}(1) = 2\mu RU - U/(1-U)$. By applying l'Hôpital's rule to the right-hand side of (1.72), we see that $f(1)$ is well-defined, with value

$$\begin{aligned} f(1) &= \frac{g^{(2)}(1)}{-2\mu U} \\ &= \frac{1 - 2\mu R\,(1-U)}{2\mu\,(1-U)}. \end{aligned} \tag{1.76}$$

$f(1) = 0$ if and only if $g^{(2)}(1) = 0$ i.e. if $1 - 2\mu R\,(1-U) = 0$. Therefore, unless this condition is satisfied, the zeros of $f(x)$ in $[U, \infty)$ are the zeros of $g(x)$ in $[U, \infty) \setminus \{1\}$; for $1 - 2\mu R\,(1-U) = 0$, the zeros of $f(x)$ in $[U, \infty)$ are the zeros of $g(x)$ in $[U, \infty)$. In that case, (1.75) rewrites as

$$h(x) = \mu\, R\,(x^2 - (1-U)x - U)$$

and its zeros become $x_1 = -U$ and $x_2 = 1$.

Three cases are to be considered depending on whether x_2 is less than, equal to or greater than 1 (we have seen that $x_2 = 1$ if $1 - 2\mu R(1-U) = 0$). In each case, the variations of the functions $g(x)$ and $f(x)$ are studied. We aim at showing that $g(x)$ has a unique zero in $[U, \infty) \setminus \{1\}$ when $g^{(2)}(1) \neq 0$ and a unique zero in $[U, \infty)$, located at the point $x = 1$, when $g^{(2)}(1) = 0$, which will conclude the proof. The following limits are easily calculated

$$\lim_{x\to+\infty} g^{(2)}(x) = 0, \qquad \lim_{x\to\pm\infty} g^{(1)}(x) = +\infty,$$
$$\lim_{x\to-\infty} g^{(2)}(x) = -\infty, \qquad \lim_{x\to+\infty} g(x) = +\infty.$$

- $U \leq x_2 \leq 1$

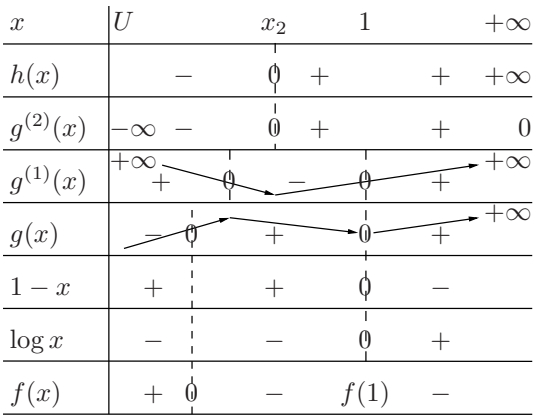


Figure 1.8: $U \leq x_2 \leq 1$

We conclude from Figure 1.8 that $g(x)$ has a unique zero in $[U, \infty) \setminus \{1\}$ when $U \leq x_2 \leq 1$. This zero is located in $[U, x_2)$

- $x_2 \geq 1$

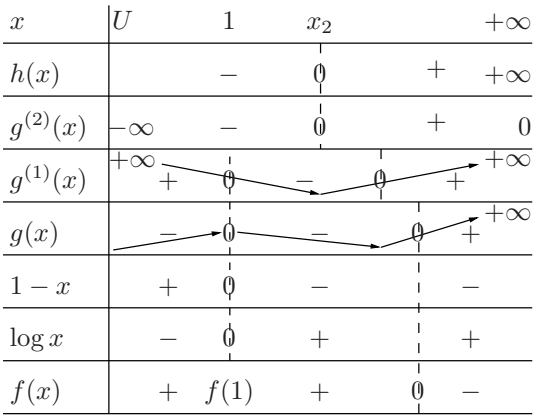


Figure 1.9: $x_2 \geq 1$

We conclude from Figure 1.9 that $g(x)$ has a unique zero in $[U, \infty) \setminus \{1\}$ when $x_2 \geq 1$. This zero is located in (x_2, ∞)

- $x_2 = 1$ (i.e. $1 - 2\mu R\,(1 - U) = 0$)

x	U		$x_2 = 1$		$+\infty$
$h(x)$		$-$	0	$+$	$+\infty$
$g^{(2)}(x)$	$-\infty$	$-$	0	$+$	0
$g^{(1)}(x)$	$+\infty$	$+$	0	$+$	$+\infty$
$g(x)$		$-$	0	$+$	$+\infty$
$1-x$		$+$	0	$-$	
$\log x$		$-$	0	$+$	
$f(x)$		$+$	0	$-$	

Figure 1.10: $x_2 = 1$

In this case (cf. Figure 1.10), $g(x)$ has only one zero in $[U, \infty)$ which is $x_2 = 1$.

■

Concerning the sensitivity of λ and K with respect to the variables U and R, a similar analysis as the one in Section 1.6.2.4 can be performed, leading to the same conclusion: $\mathrm{d}\lambda$ and $\mathrm{d}K$ depend on the values of μ, ρ and K and it is rather impossible to formally identify which estimator among U and R has the biggest impact on the estimations of λ and K.

1.6.2.6 Solving for scheme XI

This scheme has P_L and U as inputs, but this time these quantities have to be computed for the $M{+}M/D/1/K$ queue. More precisely, cf. (1.37) and (1.38),

$$P_L = \frac{1 + (\rho - 1)\,\alpha_K(\rho)}{1 + \rho\,\alpha_K(\rho)} \tag{1.77}$$

$$U = \frac{\rho\,\alpha_K(\rho)}{1 + \rho\,\alpha_K(\rho)} \tag{1.78}$$

with $\alpha_K(\rho)$ given in (1.33).

Recall that we want to solve the system of equations (1.77)–(1.78) with respect to the variables λ and K. We readily observe from (1.77)–(1.78) that

$$\rho = \frac{U}{1 - P_L}, \quad \text{i.e.} \quad \lambda = \frac{\mu\,U}{1 - P_L} - \gamma \tag{1.79}$$

$$\alpha_K(\rho) = \frac{1 - P_L}{1 - U}. \tag{1.80}$$

Since all coefficients $\{\alpha_j(\rho), j \geq 2\}$ in the Taylor series expansion of $1/\mathcal{G}_\rho(z)$ are different (see Section 1.5.1.2), then (1.79)–(1.80) will return a unique solution (ρ, K).

For a given ρ, we computed the coefficients $\alpha_j(\rho)$ for a certain range of values for j and compared the results with the r.h.s. of (1.80); then K was chosen as the integer j for which $\alpha_j(\rho)$ was the closest to (the measured value of) $(1 - P_L)/(1 - U)$.

1.6.3 Calculating the moment-based estimators

We have at our disposal the first n samples of $\{X_i\}_i$, $\{Y_i\}_i$, $\{a_i\}_i$, $\{d_i\}_i$ for the probing traffic, and we know γ and μ. Let $\hat{U}(n)$, $\hat{P}_L(n)$, $\hat{R}(n)$, $\hat{q}_L(n)$ and $\hat{q}_N(n)$ denote the estimators of U, P_L, R, q_L and q_N, respectively. They are defined as $(n = 1, 2, \ldots)$

$$\hat{P}_L(n) := \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}\{X_i = 1\}, \tag{1.81}$$

$$\hat{U}(n) := \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}\{Y_i = 1\}, \tag{1.82}$$

$$\hat{R}(n) := \frac{\sum_{i=1}^{n} \mathbf{1}\{X_i = 0\}\,(d_i - a_i)}{\sum_{i=1}^{n} \mathbf{1}\{X_i = 0\}}, \text{ for } \sum_{i=1}^{n} \mathbf{1}\{X_i = 0\} > 0, \tag{1.83}$$

$$\hat{q}_L(n) := \frac{\sum_{i=1}^{n-1} \mathbf{1}\{X_i = 1, X_{i+1} = 1\}}{\sum_{i=1}^{n-1} \mathbf{1}\{X_i = 1\}}, \text{ for } \sum_{i=1}^{n-1} \mathbf{1}\{X_i = 1\} > 0, \tag{1.84}$$

$$\hat{q}_N(n) := \frac{\sum_{i=1}^{n-1} \mathbf{1}\{X_i = 0, X_{i+1} = 0\}}{\sum_{i=1}^{n-1} \mathbf{1}\{X_i = 0\}}, \text{ for } \sum_{i=1}^{n-1} \mathbf{1}\{X_i = 0\} > 0. \tag{1.85}$$

The estimates $\hat{P}_L(n)$ and $\hat{U}(n)$ are obtained using all observations, $\hat{R}(n)$ and $\hat{q}_N(n)$ are obtained using observations corresponding to successful packets, and $\hat{q}_L(n)$ is obtained using observations corresponding to lost packets. We expect $\hat{q}_L(n)$ to converge slowly to q_L, hence, intuitively, we can say that all schemes involving this metric will not perform well.

1.6.4 Desirable properties of an estimator

If a comparison is to be made among several estimators, it is useful to identify the main properties expected of a good estimator. Namely, an estimator is preferably unbiased and consistent. Unbiasedness has been proven for $\hat{P}_L(n)$, $\hat{U}(n)$ and $\hat{R}(n)$, while $\hat{q}_L(n)$ and $\hat{q}_N(n)$ turn out to be biased (see next Section). Consistency[4] for each metric is much more complicated to show. Establishing such a property implies computing both the bias and the

[4]A consistent estimator is one that concentrates completely on its target as the sample size increases

variance of an estimator. The major difficulty in this computation is due to the fact that the random variables $\{X_i\}_i$ are correlated, since the queue is finite and since the samples are taken from consecutive foreground packets, rather than random packets.

1.6.4.1 Study of the mean values

Using the identity $\mathbf{E}[\mathbf{1}\{A\}] = P(A)$ that holds for any event A, we find

$$\mathbf{E}[\hat{P}_L(n)] = P_L,$$
$$\mathbf{E}[\hat{U}(n)] = U,$$
$$\mathbf{E}[\hat{R}(n)] = R,$$
$$\mathbf{E}[\hat{q}_L(n)] = q_L - \frac{\mathrm{cov}[\hat{q}_L(n), \hat{P}_L(n-1)]}{P_L},$$
$$\mathbf{E}[\hat{q}_N(n)] = q_N + \frac{\mathrm{cov}[\hat{q}_N(n), \hat{P}_L(n-1)]}{P_L}.$$

The last two equalities follow from (1.84) and (1.85) when expressed as follows

$$\hat{q}_L(n) = \frac{\sum_{i=1}^{n-1} \mathbf{1}\{X_i = 1, X_{i+1} = 1\}}{(n-1)\,\hat{P}_L(n-1)},$$
$$\hat{q}_N(n) = \frac{\sum_{i=1}^{n-1} \mathbf{1}\{X_i = 0, X_{i+1} = 0\}}{(n-1)\,(1-\hat{P}_L(n-1))}.$$

Clearly, $\hat{P}_L(n)$, $\hat{U}(n)$ and $\hat{R}(n)$ are unbiased estimators, whereas $\hat{q}_L(n)$ and $\hat{q}_N(n)$ are biased, but the bias depends on the size of the samples. Moreover, if $\hat{P}_L(n)$ is consistent then the bias approaches 0 as $n \to \infty$ for both estimators.

1.6.4.2 Distribution of $\hat{P}_L(n)$

In this section, we want to express the distribution of $\hat{P}_L(n)$, or in other words, $P(\hat{P}_L(n) = i/n)$. This allows the computation of the variance of $\hat{P}_L(n)$. We have seen that this estimator is unbiased. If its variance approaches zero as the number of probes goes to infinity, then $\hat{P}_L(n)$ is consistent, or equivalently, $\hat{P}_L(n) \to P_L$ for $n \to \infty$.

The estimation of P_L at the nth probe sample is simply the ratio of the number of lost probes, i, over the total number of probes, n. For n samples, one may construct the

indefinitely. In the limiting case, as the sample size becomes infinite, a consistent estimator will provide a perfect point estimate of the target. In other words, an estimator is said to be consistent if and only if its bias and variance *both* approach zero, as $n \to \infty$.

vector $(X_1, \ldots, X_n)$ where $X_i = \mathbf{1}\{Q_i = K\}$. This vector takes on values from the set space $\{0,1\}^n$. For i lost packets, there are $C_n^i := C(n,i)$ possible values for $(X_1, \ldots, X_n)$, denote by $\mathcal{A}_i$ the set of these values. We can now write

$$\begin{aligned}
P\Big(\hat{P}_L(n) = i/n\Big) &= \sum_{(x_1,\ldots,x_n)\in\mathcal{A}_i} P\Big((X_1,\ldots,X_n) = (x_1,\ldots,x_n)\Big) \\
&= \sum_{(x_1,\ldots,x_n)\in\mathcal{A}_i} P(X_1 = x_1, \ldots, X_n = x_n) \\
&= \sum_{(x_1,\ldots,x_n)\in\mathcal{A}_i} P(X_n = x_n | X_1 = x_1, \ldots, X_{n-1} = x_{n-1}) \\
&\qquad \times P(X_{n-1} = x_{n-1} | X_1 = x_1, \ldots, X_{n-2} = x_{n-2}) \\
&\qquad \times \ldots \times P(X_2 = x_2 | X_1 = x_1) \times P(X_1 = x_1) \\
&= \sum_{(x_1,\ldots,x_n)\in\mathcal{A}_i} P(X_n = x_n | X_{n-1} = x_{n-1}) \times P(X_{n-1} = x_{n-1} | X_{n-2} = x_{n-2}) \\
&\qquad \times \ldots \times P(X_2 = x_2 | X_1 = x_1) \times P(X_1 = x_1) \qquad (1.86)
\end{aligned}$$

where exactly i variables x_j are equal to 1 and the remaining $n-i$ variables are equal to 0. To derive the last equality, we have used the memoriless property of the Markovian process $\{Q_n\}_n$. In order to simplify (1.86), introduce the following:

$$\begin{aligned}
a &:= \text{number of occurrences of two consecutive lost packets} \\
&= \text{number of occurrences of the sequence '11' in } (x_1,\ldots,x_n) \\
b &:= \text{number of occurrences of two consecutive successful packets} \\
&= \text{number of occurrences of the sequence '00' in } (x_1,\ldots,x_n) \\
c &:= \text{number of occurrences of a lost packet followed by a successful packet} \\
&= \text{number of occurrences of the sequence '10' in } (x_1,\ldots,x_n) \\
d &:= \text{number of occurrences of a successful packet followed by a lost packet} \\
&= \text{number of occurrences of the sequence '01' in } (x_1,\ldots,x_n)
\end{aligned}$$

and

$$P_0 = \begin{cases} P_L, & \text{if } x_1 = 1, \\ 1 - P_L, & \text{if } x_1 = 0. \end{cases}$$

Obviously, the parameters a, b, c and d satisfy:

$$\begin{cases} a + b + c + d + 1 = n, \\ a + d + \mathbf{1}\{P_0 = P_L\} = i, \\ c = d, & \text{if } x_1 = x_n, \\ c = d + \mathbf{1}\{x_1 = 1, x_n = 0\} - \mathbf{1}\{x_1 = 0, x_n = 1\}, & \text{if } x_1 \neq x_n. \end{cases}$$

Notice that

$$\begin{aligned}
P(Q_k \neq K | Q_{k-1} = K) &= 1 - P(Q_k = K | Q_{k-1} = K) = 1 - q_L, \\
P(Q_k = K | Q_{k-1} \neq K) &= 1 - P(Q_k \neq K | Q_{k-1} \neq K) = 1 - q_N.
\end{aligned}$$

Equation (1.86) can now be rewritten as follows

$$P\Big(\hat{P}_L(n) = i/n\Big) = \sum_{(x_1,\ldots,x_n)\in\mathcal{A}_i} P_0(q_L)^a (q_N)^b (1-q_L)^c (1-q_N)^d. \tag{1.87}$$

At this point, we must distinguish between four different cases, according to whether the first packet and/or the last packet are lost or not.

- $x_1 = 0$ and $x_n = 0$. We have $P_0 = 1 - P_L$

$$i = 0 \Rightarrow \begin{cases} a = c = d = 0, \\ b = n - 1. \end{cases}$$

$$1 \le i \le n-2 \Rightarrow \begin{cases} a = i - d, \\ b = n - i - d - 1, \\ c = d, \\ \text{with } d = 1, \ldots, \min(i, n-i-1). \end{cases}$$

  For each value of d, there are $C^d_{n-i-1} C^{d-1}_{i-1}$ different possible values for $(x_2, \ldots, x_{n-1})$ (Note: there are C^d_{n-i-1} ways of putting the d '0' of the sequence '01' in $n-i-1$ positions, and there are C^{d-1}_{i-1} possibilities for choosing $d-1$ cuts among all $i-1$ possible cuts in order to separate all the i 1's into d groups '01...1').

- $x_1 = 0$ and $x_n = 1$. We have $P_0 = 1 - P_L$

$$1 \le i \le n-1 \Rightarrow \begin{cases} a = i - d, \\ b = n - i - d, \\ c = d - 1, \\ \text{with } d = 1, \ldots, \min(i, n-i), \\ \text{or } c = 0, \ldots, \min(i-1, n-i-1). \end{cases}$$

  For each value of c, there are $C^c_{n-i-1} C^c_{i-1}$ different possible values for $(x_2, \ldots, x_{n-1})$ (Note: there are C^c_{n-i-1} ways of putting the c '0' of the sequence '10' in $n-2-(i-1)$ positions, and there are C^c_{i-1} possibilities for choosing c cuts among all $i-1$ possible cuts in order to separate all the i 1's into c groups '1...10' and the right-most group '1...1').

- $x_1 = 1$ and $x_n = 0$. We have $P_0 = P_L$

$$1 \le i \le n-1 \Rightarrow \begin{cases} a = i - d - 1, \\ b = n - i - d - 1, \\ c = d + 1, \\ \text{with } d = 0, \ldots, \min(i-1, n-i-1). \end{cases}$$

For each value of d, there are $C_{n-i-1}^{d} C_{i-1}^{d}$ different possible values for $(x_2, \ldots, x_{n-1})$ (Note: there are C_{n-i-1}^{d} ways of putting the d '0' of the sequence '01' in $n-2-(i-1)$ positions, and there are C_{i-1}^{d} possibilities for choosing d cuts among all $i-1$ possible cuts in order to separate all the i 1's into d groups '01...1' and the left-most group '1...1').

- $x_1 = 1$ and $x_n = 1$. We have $P_0 = P_L$

$$2 \leq i \leq n-1 \Rightarrow \left\{ \begin{array}{l} a = i-d-1, \\ b = n-i-d, \\ c = d, \\ \text{with } d = 1, \ldots, \min(i-1, n-i). \end{array} \right.$$

$$i = n \Rightarrow \left\{ \begin{array}{l} a = n-1, \\ b = c = d = 0. \end{array} \right.$$

For each value of c $(= d)$, there are $C_{n-i-1}^{d-1} C_{i-1}^{d}$ different possible values for $(x_2, \ldots, x_{n-1})$ (Note: there are C_{i-1}^{c} ways of putting the c '1' of the sequence '10' in $n-1-(n-i)$ positions, and there are C_{n-i-1}^{c-1} possibilities for choosing $c-1$ cuts among all $n-i-1$ possible cuts in order to separate all the $n-i$ 0's into c groups '10...0').

Finally, and after a factorization of the cases $x_1 \neq x_n$, (1.87) becomes

$$\begin{aligned} P\Big(\hat{P}_L(n) = i/n\Big) \;=\; & \mathbf{1}\{i=0\}(1-P_L)(q_N)^{n-1} + \mathbf{1}\{i=n\}P_L(q_L)^{n-1} \\ & + \mathbf{1}\{1 \leq i \leq n-2\} \sum_{d=1}^{\min(i,n-i-1)} C_{n-i-1}^{d} C_{i-1}^{d-1} (1-P_L)(q_L)^{i-d}(q_N)^{n-i-d-1}(1-q_L)^{d}(1-q_N)^{d} \\ & + \mathbf{1}\{1 \leq i \leq n-1\} 2P_L(1-q_L) \\ & \qquad \times \sum_{d=0}^{\min(i-1,n-i-1)} C_{n-i-1}^{d} C_{i-1}^{d} (q_L)^{i-d-1}(q_N)^{n-i-d-1}(1-q_L)^{d}(1-q_N)^{d} \\ & + \mathbf{1}\{2 \leq i \leq n-1\} \sum_{d=1}^{\min(i-1,n-i)} C_{n-i-1}^{d-1} C_{i-1}^{d} P_L(q_L)^{i-d-1}(q_N)^{n-i-d}(1-q_L)^{d}(1-q_N)^{d}. \end{aligned}$$

The variance of the estimator $\hat{P}_L(n)$ is simply

$$\mathrm{Var}\left[\hat{P}_L(n)\right] = \mathbf{E}\left[\left(\hat{P}_L(n)\right)^2\right] - \mathbf{E}\left[\hat{P}_L(n)\right]^2 = \sum_{i=0}^{n} (i/n)^2 \, P\left(\hat{P}_L(n) = i/n\right) - (P_L)^2.$$

It is quite hard to formally compute $\lim_{n\to\infty} \mathrm{Var}[\hat{P}_L(n)]$. Thus, we will rely on simulated results to compute the empirical variance of $\hat{P}_L(n)$ and further know whether or not this estimator is consistent. This is addressed in the next section.

1.6.4.3 Overall performance of the estimators

The overall performance of the estimators is presented in Table 1.3. In order to have a fair comparison between the estimators, we report their performance in $M{+}M/M/1/K$ simulations only. In these simulations, all the assumptions considered in Section 1.4.1 are satisfied. The performance of the estimators is thus affected by the speed of convergence solely.

Table 1.3: Overall performance of the estimators for 50000 probes and $M{+}M/M/1/K$ simulations: sample mean and percentiles of the relative error (expressed in percentage) and the empirical variance

	Relative error (%)					
Metric	Mean	25	50	75	90	95
P_L	5.07	0.52	2.86	6.46	11.9	21.9
U	0.17	0.002	0.09	0.27	0.51	0.68
R	0.46	0.03	0.07	0.59	1.43	2.43
q_L	10.7	0.55	5.76	11.4	18.8	57.8
q_N	0.20	0.01	0.11	0.24	0.60	0.90
	Empirical variance					
Metric	Mean	25	50	75	90	95
P_L	0.20	0.01	0.09	0.23	0.59	0.88
U	0.04	$9\,10^{-4}$	0.02	0.07	0.09	0.16
R	1.39	$3\,10^{-6}$	$2\,10^{-5}$	10^{-4}	0.004	13.1
q_L	19.6	0.13	1.06	15.2	91.1	115
q_N	0.35	0.003	0.05	0.30	1.37	2.25

For each experiment (20 different $M{+}M/M/1/K$ experiments have been conducted; see details in Section 1.7), we have computed the relative error (expressed in percentage) between each estimator in (1.81)–(1.85) and its corresponding theoretical value, as well as the variance of the estimato. This computation has been performed after the generation of 50000 probes. For a given estimator, we therefore have a collection of 20 values for the relative error as well as 20 values for the variance of the estimator.

Rows 3–7 in Table 1.3 report the mean and the percentiles of the relative error (expressed in percentage) using the collection of 20 values computed from the simulations, and rows 10–14 in the same table report the mean and the percentiles of the empirical variance of the estimators.

Estimator $\hat{q}_L$ is the least efficient estimator[5] since it has the largest empirical variance (see row 13 in Table 1.3). In other words, the estimator $\hat{q}_L$ is not good because it exhibits high variance. The distribution of $\hat{q}_L$ is not concentrated and the estimates of $\hat{q}_L$ vary too

[5]An estimator is said to be more efficient if it has a smaller variance. The distribution of an efficient estimator is highly concentrated.

much over time. We expect all schemes using $\hat{q}_L$ to perform badly. The rest of the estimators is more or less efficient, their distribution being highly concentrated. For $\hat{P}_L$, $\hat{U}$ and $\hat{R}$, we already now that they are unbiased (see Section 1.6.4.1), hence their distributions are said to be on target.

Looking now at rows 3–7 in Table 1.3, we see that the relative error of estimators $\hat{U}$, $\hat{R}$ and $\hat{q}_N$ is low, unlike the relative errors of $\hat{P}_L$ and $\hat{q}_L$. Thus, we expect all schemes using $\hat{P}_L$ to perform worse than the schemes using $\hat{R}$ for instance. However, this might not be the case due to the unexpected effect of combining several "noisy" estimators within the same scheme.

Remark 1.6.1 *Estimators $\hat{P}_L$, $\hat{q}_L$ and $\hat{q}_N$ are all loss-related and based on single-sided measurements, as these estimators are "naturally" available at the destination. Estimator $\hat{R}$ is based on an end-to-end measurement, as well as $\hat{U}$. One might expect that schemes involving both kind of metrics will perform better than schemes involving single-sided or end-to-end metrics solely, regardless of the performance of the metrics at hand.*

1.7 Simulation results and analysis

1.7.1 Trace generation

The data sets $\{a_i\}_i$, $\{d_i\}_i$, $\{X_i\}_i$ and $\{Y_i\}_i$ were extracted from traces generated by simulation models in `ns-2`. In this section, we report simulations in which there is a single queue to better observe the behavior of both inference models under various background traffic patterns. Simulations with multiple links are reported later on in Section 1.7.5. Overall, 50 simulations have been performed in which several types of background traffic are considered:

(T1) A Poisson flow of packets with exponentially distributed packet size. The average packet size is 100 Bytes.

(T2) A superposition of 100 Poisson-like flows. The inter-arrivals within each flow is exponentially distributed. The packet length is constant for each flow and its value is taken from an exponential distribution with average 100 Bytes. Therefore, different flows have different packet lengths.

(T3) An aggregation of 100 On/Off flows, where the On and Off times were taken from a Pareto distribution with shape 1.5. The packet length is constant for each flow and its value is taken from an exponential distribution with average 100 Bytes.

(T4) An aggregation of 250 On/Off flows, where the On and Off times were taken from a Pareto distribution with shape 1.5. The packet length is constant for each flow and its value is taken from an exponential distribution with average 100 Bytes.

(T5) An aggregation of 250 long-lived FTP over TCP flows.

(T6) An aggregation of 1000 long-lived FTP over TCP flows.

In all experiments foreground packets arrive according to a Poisson process and have exponentially distributed packet sizes (average = 100 Bytes) except in the case when the background traffic is of type (T2); in the latter case, the foreground source is also of type (T2).

It is important to specify, for each type of simulations, which assumptions are satisfied and which ones are violated.

- The set of simulations where the cross traffic is of type (T1) corresponds to the $M+M/M/1/K$ queue model. The interest of such simulations is to identify which scheme among schemes **I-X** converges faster.

- When both foreground and background traffic are of type (T2), the only assumption that is violated is the one concerning the service times. In these simulations, the service times are the same for each flow, but differ from flow to flow. There are a total of 101 different values coming all from an exponential distribution (the service time is proportional to the packet size). If the number of background flows is increased, the service time distribution will get closer to the exponential distribution. In this set of simulations, we are testing the robustness of both queue models against violation of the assumption on the service times.

- When the cross traffic is of type (T3) or (T4), the Poisson assumption for the background traffic is violated. The assumption on the distribution of the service times is also violated, but as the number of flows increases, the service time distribution will get closer to the exponential distribution (cf. previous item). However, when the number of flows grows to infinity, it is known that the aggregation of such On/Off flows will exhibit correlations across several time scales (see [116]). So, when the number of flows is increased, on one hand the assumption on the cross traffic will be *more* violated, and on the other hand the assumption on the service times will be *less* violated (case of the $M+M/M/1/K$ queue model). A comparison of the performance of the estimators when the background traffic is either (T3) or (T4) will help identify which assumption is more essential to the $M+M/M/1/K$ queue model.

- In the last set of assumptions (cross traffic of type (T5) or (T6)), the Poisson assumption for the background traffic is violated as well as the assumption of exponential service times. The FTP/TCP flows generate two kinds of packets: 512-Bytes data packets, and 40-Bytes acknowledgements. Thus, the service times of the background traffic take on two distinct values, whereas the service times of the foreground traffic are exponentially distributed. When the number of background flows is increased, the resulting distribution of the service times is expected to get farther from the exponential distribution. Going back to the background traffic assumption (which has

been made only for mathematical tractability), it is known from [95] that the arrivals within a single FTP flow are not Poisson. However, what we do not know is whether the distribution of the arrivals from an *aggregation* of several FTP flows approaches the exponential distribution or not. We can say that, *a priori*, the assumption on the background arrivals is violated, and we are tempted to add: as the number of exogenous flows increases, the distribution of the resulting inter-arrival process gets closer to the exponential distribution.

Let us now return to the description of the traces generated. The rate of the foreground traffic (the probes) was equal to 250 pkts/s in all experiments (except in case (T1) where values of 125, 250 and 500 pkts/s were retained). On the network side, the server rate was either equal to 1500 pkts/s or to 6500 pkts/s and the buffer size was either equal to 10, 30, 65, 100, 150 or to 1000 packets (recall that in `ns-2` the size of the buffer is defined in number of packets regardless of their size).

Below, we give the ranges of values obtained for the five metrics over all 50 experiments:

- P_L ranged from 1.7×10^{-4} to 0.637
- U ranged from 0.892 to 1
- R ranged from 0.0007 to 0.66 seconds
- q_L ranged from 0.105 to 0.64
- q_N ranged from 0.367 to 0.9998.

As for the rate of exogenous traffic intensity λ, measured as the number of background packets arriving to the bottleneck link over the run time, its value ranged from 1593.2 to 17437 packets per second, giving the range 0.965 – 2.758 for the traffic intensity ρ. Notice that in all the simulations, the single link was (highly) congested.

1.7.2 Estimating cross traffic rate, buffer size and (possibly) server capacity

Having at our disposal $\{a_i\}_i$, $\{d_i\}_i$, $\{X_i\}_i$ and $\{Y_i\}_i$ for the n first probing packets, the moment-based estimators are computed according to formulas (1.81), (1.82), (1.83), (1.84) and (1.85). At this point, $\hat{P}_L(n)$, $\hat{U}(n)$, $\hat{R}(n)$, $\hat{q}_L(n)$ and $\hat{q}_N(n)$ are substituted into (1.3), (1.5), (1.10), (1.18), (1.21), (1.37), (1.38) and (1.40). The ten triples of equations referred to as schemes *1* through *10* and the eleven pairs of equations referred to as schemes **I** through **XI** are then solved numerically using a C program including the NAG[6] C library [90, 91]. Results are reported in Tables 1.3–1.9 and Figures 1.11–1.15.

[6]NAG is a copyright of The Numerical Algorithms Group Ltd

1.7.3 Analysis of the results in case μ is known

In this section, we restrict ourselves to the estimation of parameters λ and K, thereby assuming that μ is known. We briefly discuss the results returned by schemes **I** through **XI** (each involving two QoS metrics) and then focus on the performance of the "best" scheme.

Our findings can be summarized as follows:

- Schemes **III**, **IV** and **X** give almost identical results. This is not surprising, though, since the three of them involve loss-related metrics which are related by (1.22) as seen in Section 1.4.6. We further observed that their solutions always under-estimate the correct values. We believe that this is due to a lack of information as these schemes count on loss-related measurements solely (variables $\{X_n\}_n$ introduced in Section 1.4.2).

- Schemes $\mathbf{I} = P_L\_U$ and $\mathbf{VII} = U\_q_N$ return similar results as do schemes $\mathbf{II} = P_L\_R$ and $\mathbf{IX} = R\_q_N$. Notice that both pairs of schemes have a common metric (U for the first pair and R for the second one) and the second metric is either P_L (schemes **I** and **II**) or q_N (schemes **VII** and **IX**). It appears that interchanging P_L and q_N has little impact on the performance of a scheme, even though the latter metric is much better estimated than the former one (cf. Section 1.6.4.3).

- Schemes **VI** and **VIII** performed poorly, either returning bad estimates $\hat{\lambda}$ and $\hat{K}$ or even not returning results at all. This is due to the fact that these schemes involve q_L, which was seen to be badly estimated (cf. Section 1.6.4.3).

- The solution returned by scheme **V** always over-estimates the true values. This scheme involve metrics U and R which can both be seen as end-to-end measurements. These metrics provide somehow redundant informations which can explain the bad performance of the scheme.

- All schemes including U saw their performance degrade as the network become more congested. When the single queue is congested, its buffer does not empty often. As a result the first estimates of U are $\hat{U} = 1$, which is not a valid value in either model. As soon as a probe packet finds an empty queue, the first *valid* estimate of U is computed, and it is then and only then that the schemes using U can return a solution. Since in our simulations we considered only congested cases, all schemes including U suffered from this discrepancy and their results were not the best ones.

- Scheme **XI** returns the same estimate $\hat{\lambda}$ as does scheme **I**. Both schemes involve metrics P_L and U, but the first one derive from the $M/D/1/K$ model whereas the second one come from the $M/M/1/K$ model. However, the formulas giving $\hat{\lambda}$ are identical (see Equations (1.50) and (1.79)). As for the buffer size K, it is much better estimated using scheme **I**. The cause behind the misestimation of K when using scheme **XI**

is the misestimation of U. Even though scheme **I** involves U as well, scheme **XI** is much more sensitive to the value of $\hat{U}$. Recall that K is estimated as the integer j for which $\alpha_j(\rho)$ is the closest to $(1-\hat{P}_L)/(1-\hat{U})$ (cf. Section 1.6.2.6). In congested cases, $\hat{U}$ is almost 1 and a small deviation from the true value leads to a large error in $(1-\hat{P}_L)/(1-\hat{U})$ (e.g. if $\hat{U} = 1-10^{-5}$ and $U = 1-10^{-4}$ then $\alpha_{\hat{K}}(\rho) = 10\alpha_K(\rho)$, resulting in $\hat{K} > K$).

Now that we have discussed the averall results for all eleven schemes, we will focus on the best scheme, scheme **II** $= P_L\_R$. We have mentioned before that schemes P_L_R and R_q_N gave similar results but that the former scheme performed slightly better. Notice that both schemes combine the use of a single-sided measurement (P_L or q_N) and an end-to-end measurement which is R. These schemes contain the least redundant information, (as opposed to scheme $P_L\_q_N$, for instance). From now on, only results pertaining to scheme P_L_R are presented.

Tables 1.4, 1.5, 1.6 and 1.8 report the relative error (expressed in percentage) between the estimate of parameter λ (resp. K), denoted as $\hat{\lambda}$ (resp. $\hat{K}$), returned by scheme P_L_R and the measured value λ (resp. the true value K), for various cross traffic patterns (Poisson traffic as defined in (T1) in Table 1.4, Poisson-like flows as defined in (T2) in Table 1.5), a superposition of On/Off sources with Pareto On and Off time distribution as defined in (T3)-(T4) in Table 1.6, a superposition of FTP/TCP flows as defined in (T5)-(T6) in Table 1.8.

Table 1.4: Relative error (expressed in percentage) of the estimates for 50000 probes returned by the scheme P_L_R, when the cross traffic is a single Poisson source

		Simulation parameters			
		$\lambda = 6600$	$\lambda = 6600$	$\lambda = 6600$	$\lambda = 17068$
		$\gamma = 124$	$\gamma = 248$	$\gamma = 496$	$\gamma = 125$
		$\mu = 6968$	$\mu = 6968$	$\mu = 6967$	$\mu = 6962$
Buffer size	Estimator	$\rho = 0.965$	$\rho = 0.983$	$\rho = 1.019$	$\rho = 2.470$
$K = 10$	$\hat{\lambda}$	0.6	0.004	0.1	0.1
	$\hat{K}$	0.7	0.9	1.4	0.4
$K = 30$	$\hat{\lambda}$	0.1	0.1	0.3	0.9
	$\hat{K}$	0.7	1.6	0.6	0.05
$K = 65$	$\hat{\lambda}$	0.04	0.2	0.7	1.0
	$\hat{K}$	1.1	1.0	0.3	0.1
$K = 100$	$\hat{\lambda}$	0.04	0.1	0.1	0.5
	$\hat{K}$	2.8	2.7	0.2	0.01
$K = 150$	$\hat{\lambda}$	0.1	0.1	0.02	0.2
	$\hat{K}$	8.3	4.1	0.4	0.03

Table 1.4 shows excellent results regarding the estimation of λ (obtained after 50000 probes). The relative error on $\hat{\lambda}_{50000}$, computed as $|\hat{\lambda}_{50000} - \lambda|/\lambda$, is always below 1.0% in all the simulations reported in Table 1.4 (see rows 6, 8, 10, 12 and 14). As for the estimation of K, the results are very good for moderate values of K: the relative error on $\hat{K}_{50000}$, computed as $|\hat{K}_{50000} - K|/K$, is always below 1.6% when $K \leq 65$ (see rows 7, 9 and 11 in Table 1.4). For larger values of K, the speed of convergence depends on the load ρ. As ρ increases from 0.965 (third column) to 2.470 (column 6), the relative error on K decreases for $K \geq 65$ (see rows 11, 13 and 15). Notice that in any case the estimates should converge to the true values as the number of probes increases since the experiments considered in Table 1.4 simulate at best the $M/M/1/K$ queue.

The speed of convergence of $\hat{\lambda}$ can be observed in Figure 1.11, where the evolution of the estimated cross traffic intensity is plotted against the number of probes for several values of the load ρ. For ρ around 1, the estimation converges after approximately 10000 probes (see Figures 1.11(a)-(c)) whereas 5000 probes are enough for $\rho = 2.470$ (see Figure 1.11(d)).

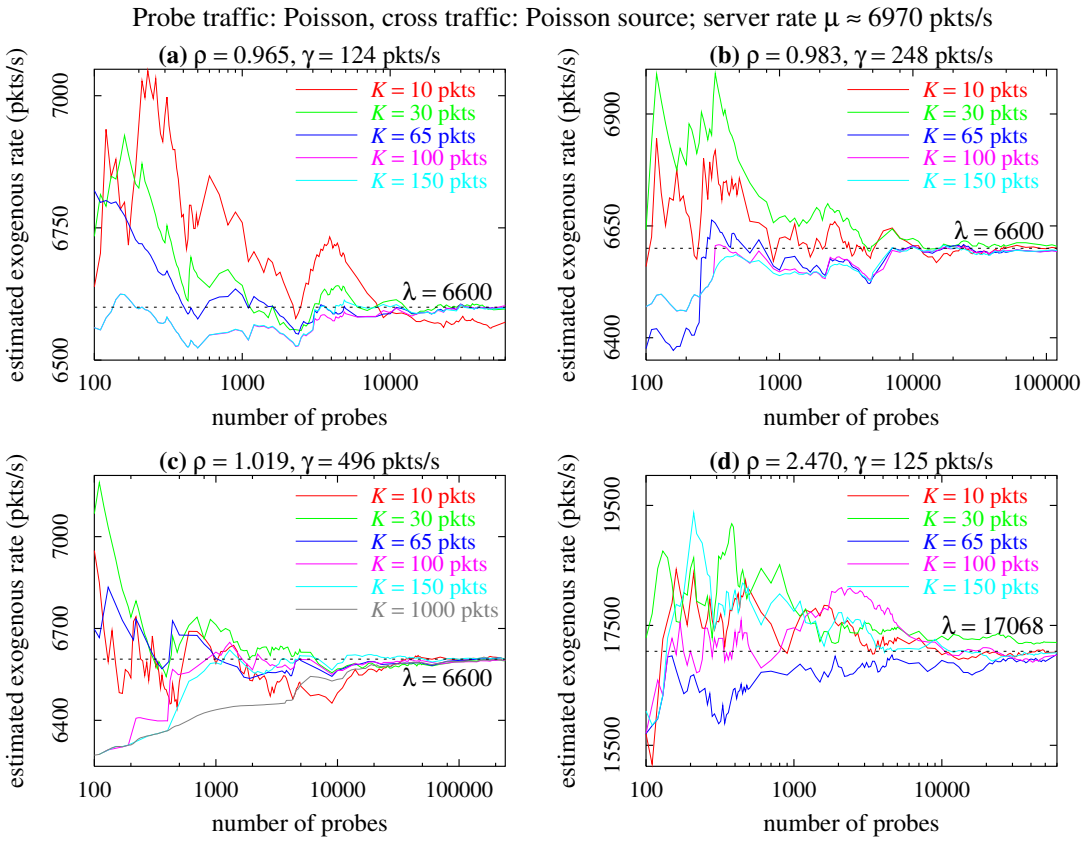


Figure 1.11: Evolution of the estimated cross traffic intensity vs. the number of probes

The slower convergence of $\hat{K}$ for smaller values of ρ can be observed in Figure 1.12

where the evolution of the estimated buffer size is plotted against the number of probes for several values of K. It is clearly visible in Figures 1.12(c)-(d) that $\hat{K}$ converges faster to K as ρ increases. Notice also the fast convergence of $\hat{K}$ to the true value for $\rho = 2.470$ in all four cases shown in Figure 1.12 ($K = 10, 30, 65, 100$).

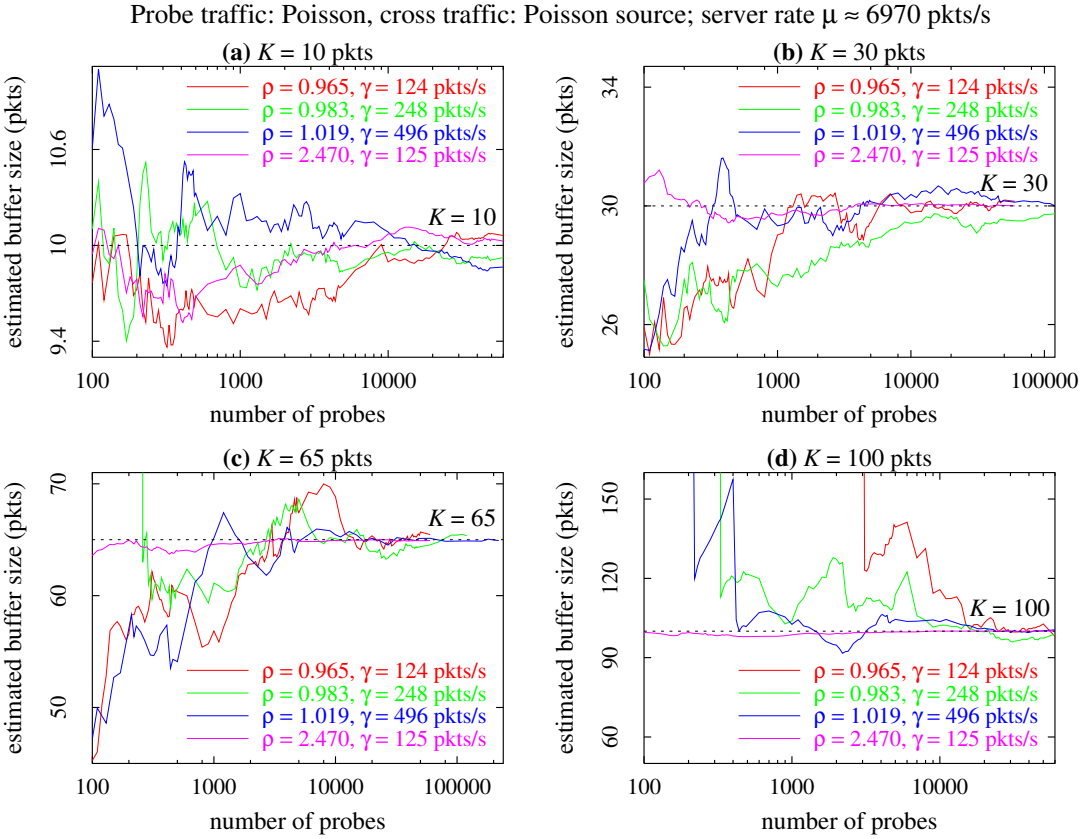


Figure 1.12: Evolution of the estimated buffer size vs. the number of probes

Consider now the set of simulations where all sources (i.e. foreground *and* background sources) are Poisson-like (type (T2)). We have seen before that this set of simulations undertakes the robustness of both models to violation of the assumption on the distribution of the service times. The relative error (expressed in percentage) is reported in Table 1.5. We observe that both λ and K are better estimated as K increases. In all cases, $\hat{\lambda}$ stays

Table 1.5: Relative error (expressed in percentage) of the estimates (scheme $P_{L\_}R$) for 120000 probes: Poisson-like flows, $\lambda = 6677$, $\gamma = 250$, $\mu = 6374$ and $\rho = 1.087$

Estimator	$K = 10$	$K = 30$	$K = 65$	$K = 100$	$K = 150$
$\hat{\lambda}$	3.1	1.1	0.04	0.1	0.1
$\hat{K}$	9.2	8.5	6.9	5.4	4.0

within 3.1% of the true value (cf. row 2 in Table 1.5) such as λ is overestimated, whereas $\hat{K}$ stays within 9.2% of the true value (cf. row 3 in Table 1.5) such as K is underestimated.

The estimation of K is not as satisfactory as the estimation of λ. It is clear that the quality of $\hat{\lambda}$ and $\hat{K}$ is influenced by the quality of estimators $\hat{P}_L$ and $\hat{R}$. Our next step is to look at the performance of these estimators. We will compute the loss probability P_L and the expected response time R as expected by the $M{+}M/M/1/K$ model using the true values of λ and K. The resulting values of P_L and R are then compared to the estimated values. A small error between both pairs of values implies a small error in $\hat{\lambda}$ and $\hat{K}$ as predicted by scheme P_L_R. Our findings can be summarized as follows.

- The loss probability predicted by the model, P_L, is smaller than the estimated $\hat{P}_L$. The absolute deviation of the predicted P_L (defined as $|P_L - \hat{P}_L|$) decreases from 0.025 (16.3% of relative deviation[7]) for $K = 10$ down to 0.002 (2.8% of relative deviation) for $K = 150$.
- The expected response time predicted by the model, R, is slightly larger than the estimate $\hat{R}$. The deviation of the predicted R increases from $57\mu s$ (6.3% of mismatch) for $K = 10$ up to $871\mu s$ (4.2% of mismatch) for $K = 150$.
- The $M{+}M/M/1/K$ model predicts much better the response time R than the loss probability P_L.

It seems that the error in estimating K is due to the error between P_L and $\hat{P}_L$, and that both the error on K and the (small) one on λ vary as does the error on $\hat{P}_L$. It is also possible that the errors at hand are also affected by the error on $\hat{R}$, but this impact (if it exists) is negligible both because of the small error on $\hat{R}$ and because of the large relative error on $\hat{P}_L$.

Remark 1.7.1 *The preceding observations describe the sensitivity of $\hat{\lambda}$ and $\hat{K}$ to variations in $\hat{P}_L$ and $\hat{R}$.*

The values that are given in Table 1.5 were computed for 120000 probes. The evolution of the estimates over the number of probes is plotted in Figure 1.13. Observe in Figure 1.13(a) how the estimate $\hat{\lambda}$ for $K = 10$ converges above the dashed line which represents the true value to be estimated. The same is true for $K = 30$. For $K = 65, 100, 150$, the estimate $\hat{\lambda}$ converges to the correct value. The reason why, for small K, $\hat{\lambda}$ overestimates λ, is that $\hat{P}_L$ is larger than the P_L predicted by the model ($\hat{R}$ being close to R for small

[7]The relative deviation between two values X and $\hat{X}$ is defined as $|X - \hat{X}|/\hat{X}$ and is sometimes called *mismatch*.

K). Recall that in the $M/M/1/K$ queue, when the load increases/decreases, both the loss probability and the response time increase/decrease. For large values of K, we have seen that $\hat{P}_L$ is slightly larger than P_L (2.8% of mismatch) and that $\hat{R}$ is slightly smaller than R (4.2% of mismatch). Each mismatch induces an error on $\hat{\lambda}$, but since the induced errors are in opposite directions, the final result is that $\hat{\lambda}$ is on target (cf. plots for $K = 65, 100, 150$ in Figure 1.13(a)).

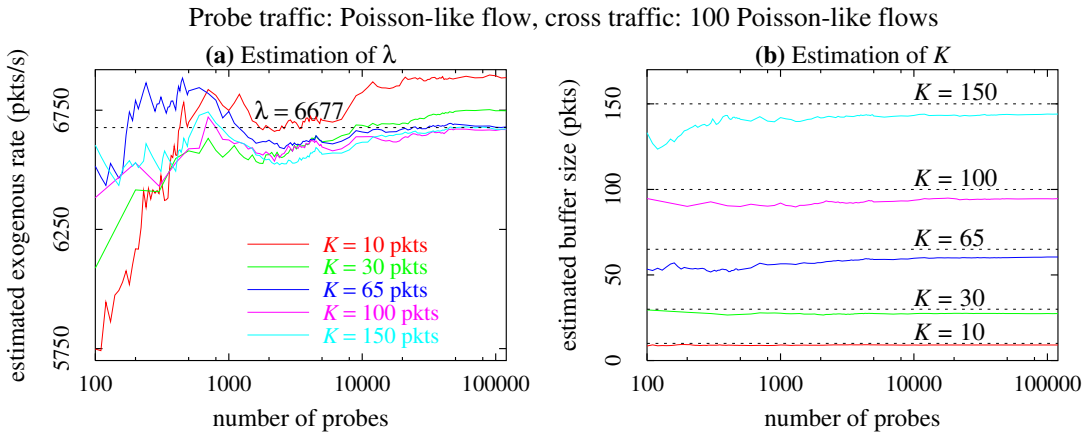


Figure 1.13: Evolution of the estimates vs. the number of probes

Figure 1.13(b) depicts the variations of the estimates for K as a function of the number of probes. Even though the absolute error ($|\hat{K}-K|$) between each plot and its corresponding dashed line increases as K increases, the relative error ($|\hat{K}-K|/K$) *decreases*. Observe that K is always underestimated, which is expected since $\hat{P}_L > P_L$ and $\hat{R} < R$. If the size of the buffer in the $M/M/1/K$ queue decreases, the loss probability P_L increases whereas the response time decreases. The mismatch of both metrics induces a larger error on K than the one induced on λ, as both mismatches result in underestimating K. Notice how "flat" are the plots in Figure 1.13(b), especially when compared to the bursty plots in Figure 1.12. Finally, convergence takes place with approximately 10000 probes when estimating λ, and even fewer when estimating K.

Of more interest are the results in Tables 1.6 and 1.8, since they have been obtained when the assumption that the cross traffic is Poisson is violated. We see that the quality of the estimates increases as the number of sources increases, or equivalently, when the load ρ increases (results obtained after 120000 probes). We will discuss first the results obtained when the cross traffic is the aggregation of On/Off flows (cf. Table 1.6) and come back later on to the case where the cross traffic is the aggregation of TCP flows (cf. Table 1.8).

For 100 On/Off sources (see rows 3–4 in Table 1.6), all estimates for λ are within 6% of the correct value, which is a satisfactory result. The relative error on $\hat{\lambda}$ seems to increase for $K \leq 65$ and decrease for $K > 65$ whereas the relative error on $\hat{K}$ definitely increases as

Table 1.6: Relative error (expressed in percentage) of the estimates (scheme P_L_R) for 120000 probes: cross traffic is of type (T3) and (T4) (On/Off flows)

	100 sources: $\lambda = 6812$, $\gamma = 247$, $\mu = 6663$ and $\rho = 1.059$				
Estimator	$K = 10$	$K = 30$	$K = 65$	$K = 100$	$K = 150$
$\hat{\lambda}$	3.8	5.2	6.0	5.4	4.7
$\hat{K}$	7.5	17.7	29.5	31.4	33.2
	250 sources: $\lambda = 17437$, $\gamma = 246$, $\mu = 6532$ and $\rho = 2.707$				
Estimator	$K = 10$	$K = 30$	$K = 65$	$K = 100$	$K = 150$
$\hat{\lambda}$	1.0	0.6	0.6	0.4	0.05
$\hat{K}$	1.1	0.5	0.4	0.4	0.1

the buffer size increases. For 250 On/Off sources (see rows 7–8 in Table 1.6), the results are much more satisfactory with all estimates for λ (resp. K) lying within 1% (resp. 1.1%) of the correct value. Notice that the relative error on both $\hat{\lambda}$ and $\hat{K}$ decreases as K increases. Searching for an explanation on these observations, we investigated on the performance of $\hat{P}_L$ and $\hat{R}$ in these particular simulations, and observed the following.

- For 100 On/Off sources ($\rho = 1.059$), the measured loss probability decreases from 0.16 for $K = 10$ down to 0.09 for $K = 150$, whereas the loss probability predicted by the model decreases from 0.13 to 0.05 as K increases and is smaller than the measured $\hat{P}_L$. The absolute deviation on P_L increases from 0.03 up to 0.05 for $10 \leq K \leq 65$ and decreases from 0.05 to 0.02 for $65 \leq K \leq 150$. The mismatch between the two values is not negligible (between 16.7% and 47.6%). As K increases, the response time predicted by the model increases from $0.94ms$ to $19.7ms$; $\hat{R}$ is smaller than the predicted R, increasing from $0.91ms$ to $13.5ms$. The deviation between the two values grows from $34\mu s$ (3.8% of mismatch) up to $6.2ms$ (45.9% of mismatch).

- For 250 On/Off sources ($\rho = 2.707$), both the predicted loss probability and the measured one are approximately constant and equal to 0.63, having the former being slightly larger than the latter (0.56% of maximum mismatch). The negligible deviation on P_L is always decreasing as K increases. As for the response time, the prediction returned by the $M{+}M/M/1/K$ model increases from $1.45ms$ to $22.90ms$ as K increases. As for the measured $\hat{R}$, it increases from $1.43ms$ to $22.87ms$ as K increases. The deviation between the two values is really negligible, of the order of tens of microseconds.

We can deduce from the above observations that for 100 On/Off sources ($\mu = 6663$), the error on $\hat{\lambda}$ varies as does the deviation between $\hat{P}_L$ and P_L; and that the error on $\hat{K}$ varies as does the deviation between $\hat{R}$ and R. For 250 On/Off sources ($\mu = 6532$), the errors on $\hat{\lambda}$ and $\hat{K}$ follow mainly the variations of the deviation between $\hat{P}_L$ and P_L.

Figure 1.14 depicts the evolution of the estimates as a function of the number of probes. Looking at Figure 1.14(a) (resp. Figure 1.14(b)), it appears that, for all K's, the estimator $\hat{\lambda}$ converges after 1000 probes (resp. 3000 probes) when the number of background flows is 100 (resp. 250). We observe in Figure 1.14(a) that the estimates deviate from the true value as K goes from 10 to 65, and get closer to $\lambda = 6812$ as K goes from 65 to 1000. The estimates for K are quite bursty (see Figures 1.14(c)-(f)). The estimates corresponding to 250 sources stay close to the true values (dashed line in each graph), while the ones corresponding to 100 sources considerably underestimate the true values as K increases. The causes of such underestimation (31.4% of error for $K = 100$) are twofold: first, $\hat{R} < R$, and second, $\hat{P}_L > P_L$. Recall that in a $M/M/1/K$ queue, decreasing the buffer size increases the loss probability and decreases the response time, which explains why such deviations in the values of P_L and R lead to an underestimation of K.

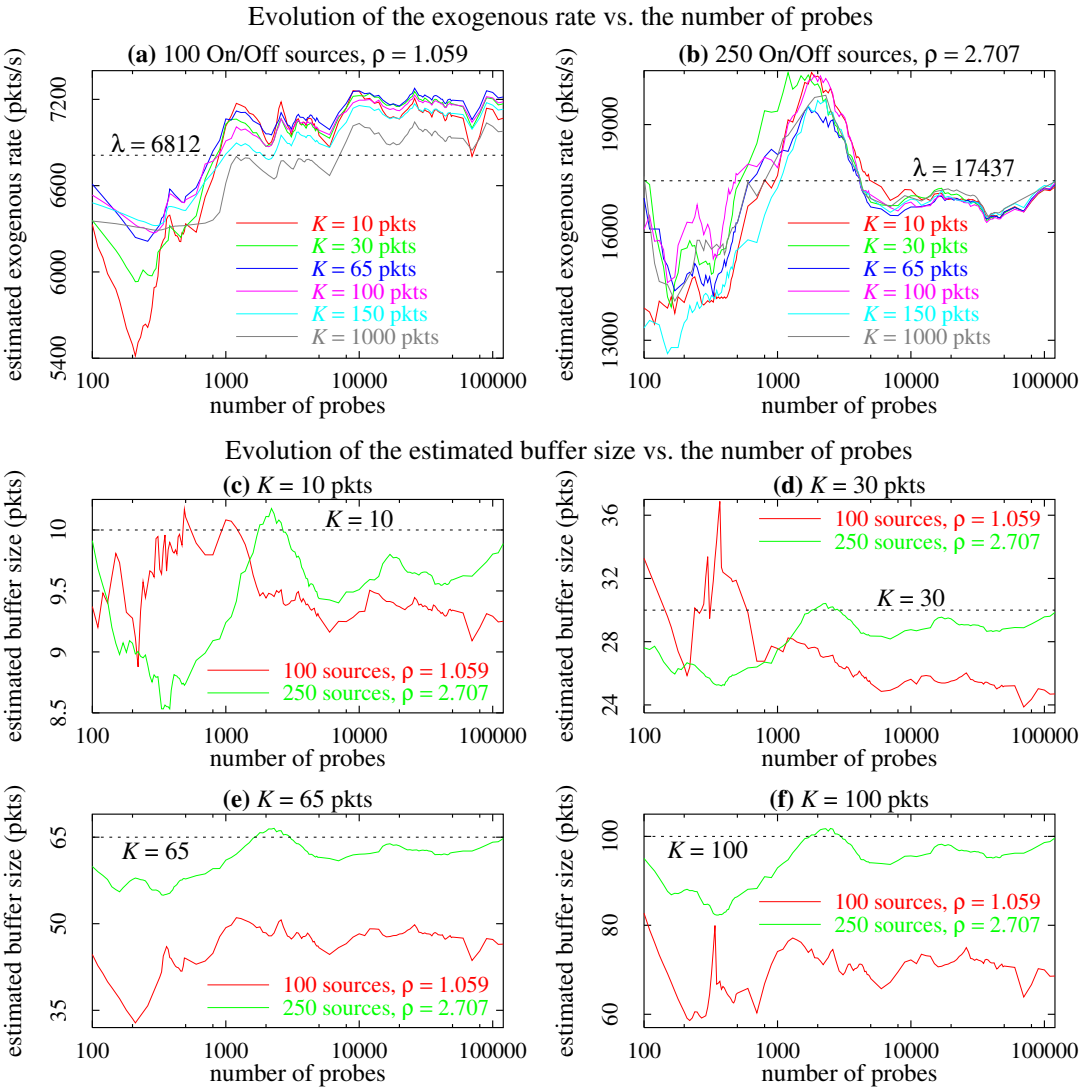


Figure 1.14: Probe traffic: Poisson, cross traffic: On/Off flows, Pareto On/Off times

To complete the discussion of the set of simulations where the cross traffic is of type (T3) or (T4), we will try to explain why scheme P_L_R behaves much better when there are 250 background sources rather than just 100 sources (for large K, the difference in performance between both cases is even more important). For each type of cross traffic, we have computed the Hurst parameter as well as the autocorrelation for both the interarrival sequence of the cross traffic and the service times sequence[8]. We have also tested the exponential distribution hypothesis of both sequences, using the goodness of fit test[9] [120]. Table 1.7 summarizes our findings. We have seen that the mismatch between the measured metrics and the predictions returned by the model drastically decreases when the number of background sources grows from 100 to 250 (see rows 12–13 in Table 1.7). We have not found any important difference in the characteristics of the cross traffic interarrivals sequence nor in the service times sequence when the number of sources increases (see rows 4–6, 8–10 in Table 1.7), which is not the case of the load (see second row in Table 1.7). We believe that the large value of ρ is behind the good performance of scheme P_L_R when there are 250 background sources. When the load is large, the queue does not empty often leading to a high number of lost packets and to a response time which is approximately the same

Table 1.7: Summary

Criterion	100 On/Off sources	250 On/Off sources
Rate	1.059	2.707
Cross traffic interarrivals sequence		
Distribution	slightly closer to an exponential when 250 sources	
Autocorrelation	slightly less autocorrelation when 250 sources	
Hurst parameter	0.805849	0.811553
Service times sequence		
Distribution	slightly closer to an exponential when 250 sources	
Autocorrelation	less autocorrelation when 250 sources	
Hurst parameter	0.837137	0.831835
Mismatch (in %) at 120000 packets	min, avg, max	min, avg, max
between P_L and $\hat{P}_L$	16.7, 37.1, 47.6	0.04, 0.26, 0.6
between R and $\hat{R}$	3.8, 28.4, 45.9	0.07, 0.45, 1.3

[8]In [35], the author provides practical instructions on how to compute the autocorrelation and the Hurst parameter using C programs, which are available at [34]

[9]For a descriptive derivation of the test, the N observed values of a sequence are ordered from smallest to largest. Denote as $x_{(i)}$ the observed value coming in the ith position. If the data really do come from an exponential distribution with parameter r, then we would expect the points $(1-\exp(-rx_{(i)}), i/(N+1))$ to be close to the diagonal line $y = x$; conversely, strong deviation from this line is evidence that the distribution did not produce the data.

for all packets. We believe that, in this case, the measured values of the metrics are close to the actual values (which are close to those predicted by the model, even though some assumptions are violated), which is not the case for 100 sources.

We will now analyze the results of the set of simulations in which the cross traffic is of type (T5) and (T6) (FTP over TCP flows). The relative error of the estimates are reported in Table 1.8. For 250 FTP/TCP flows, we observe the same thing as for 100 On/Off sources: the relative error on $\hat{\lambda}$ increases for $K \leq 65$ and decreases for $K > 65$, but here, the relative error on $\hat{K}$ is *decreasing* as the buffer size increases (refer to rows 4–5 in Table 1.8). However, in the case where there are 1000 FTP/TCP flows (see rows 9–10 in Table 1.8), both estimators present a relatively constant error. Furthermore, all estimates for λ (resp. K) are within 3.2% (resp. 0.6%) of the correct values, which is a good result.

Remark 1.7.2 *One characteristic of the set of simulations where the cross traffic is of type (T5) or (T6) is that the cross traffic intensity λ depends on the buffer size K. This is due to the fact that TCP is a closed-loop protocol. The cross traffic intensity is reported in rows 3 and 8 in Table 1.8.*

Table 1.8: Relative error (expressed in percentage) of the estimates (scheme P_L_R) for 120000 probes: cross traffic is of type (T5) and (T6) (FTP/TCP flows)

	250 sources: $\gamma = 248$, $\mu = 1502$ and $\rho = 1.258$				
	$K = 10$	$K = 30$	$K = 65$	$K = 100$	$K = 150$
Estimator	$\lambda = 1655$	$\lambda = 1661$	$\lambda = 1656$	$\lambda = 1642$	$\lambda = 1593$
$\hat{\lambda}$	7.0	7.6	8.3	7.6	6.7
$\hat{K}$	5.9	2.2	1.0	1.4	1.3
	1000 sources: $\gamma = 248$, $\mu = 1491$ and $\rho = 1.421$				
	$K = 10$	$K = 30$	$K = 65$	$K = 100$	$K = 150$
Estimator	$\lambda = 1883$	$\lambda = 1881$	$\lambda = 1882$	$\lambda = 1881$	$\lambda = 1820$
$\hat{\lambda}$	2.8	3.2	3.0	3.2	3.0
$\hat{K}$	0.5	0.04	0.4	0.6	0.6

We will try as before to identify what lies behind these observations. We looked at the performance of $\hat{P}_L$ and $\hat{R}$ in all the simulations in which the cross traffic is the aggregation of FTP over TCP flows, and found that:

- For 250 FTP/TCP sources, the absolute deviation between the measured $\hat{P}_L$ and the predicted P_L increases for $K \leq 65$ and decreases for $K \geq 65$, staying in the range 0.041 – 0.054 (mismatch in the range 15.3% – 20.2%), such that $\hat{P}_L > P_L$. The response time predicted by the model is quite close to the measured $\hat{R}$, the absolute deviation is in the range $0.11ms$ – $0.59ms$ and the mismatch is in the range 0.47% – 2.3%.

- For 1000 FTP/TCP sources, both metrics have a relatively stable mismatch, the one on P_L (resp. on R) being around 5.96% (resp. 0.57%). As before, $\hat{P}_L > P_L$.

In this set of simulations, it appears that the error on $\hat{\lambda}$ varies as does the error on $\hat{P}_L$; and that the error on $\hat{K}$ varies as does the error on $\hat{R}$. When the mismatch of $\hat{P}_L$ (resp. of $\hat{R}$) does not vary much for different values of K (given the same number of probes), the estimates of λ (resp. of K) will be approximately constant.

To observe the speed of convergence of the estimates, the reader is invited to look at Figure 1.15. In Figures 1.15(a)-(b), the evolution of the estimates of λ is plotted against the number of probes. Surprisingly enough, the estimates converge rather quickly. The convergence occurs around 2000 probes for all values of K and for both types of cross traffic ((T5) and (T6)), it is even faster than in the case of Poisson background traffic (see Figure

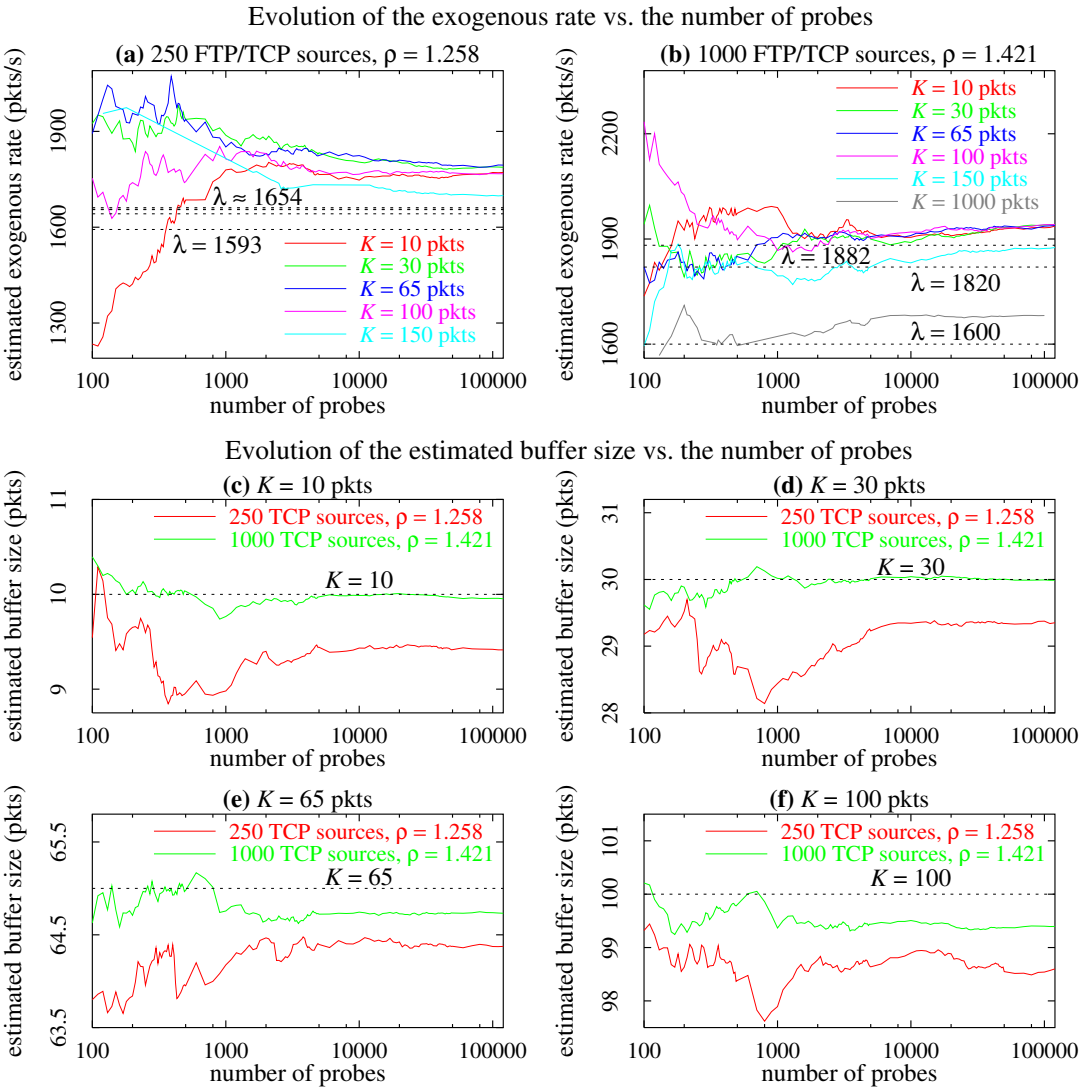


Figure 1.15: Probe traffic: Poisson, cross traffic: FTP/TCP sources

1.11). We observe the same speed of convergence of the estimates for K (between 2000 and 5000 probes, see 1.15(c)-(f)). We can also observe how the estimates are right on target when there are 1000 TCP flows. In the case where there are 250 TCP flows, $\hat{K}$ slightly underestimates K. The reason is that $\hat{P}_L > P_L$. Another effect of such mismatch is the overestimation of λ as observed in Figures 1.15(a)-(b).

As in the previous set of simulations, the predicted P_L returned by the model approaches the measured $\hat{P}_L$ when the load increases, resulting in a better performance for scheme PL_R.

Finally, we report the performance of scheme P_L_R over all 50 simulations (each simulation lasts exactly 500 seconds). Figure 1.16 displays the complementary cumulative distribution function (CCDF) of the relative error returned by estimates $\hat{\lambda}$ and $\hat{K}$. We observe that estimate $\hat{\lambda}$ returns more accurate results compared to $\hat{K}$, as the tail of the CCDF of the error on K is longer than the tail on the CCDF of the error on λ.

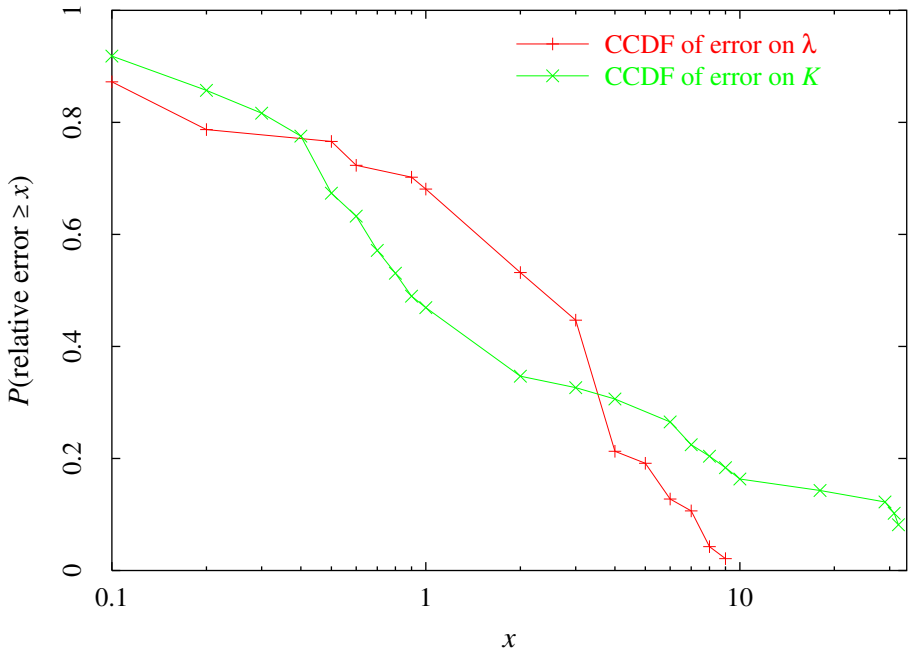


Figure 1.16: The complementary cumulative distribution function of the relative error returned by estimates $\hat{\lambda}$ and $\hat{K}$

In Table 1.9 we classify the results according to some specific performance criterion. For instance, we see from this table that the estimate for λ (resp. K) is within 9% of the exact value in 98% (resp. 84%) of the experiments. Only for large values of K ($K \geq 1000$) the scheme works poorly and may return no value for $\hat{K}$.

From this experimental study we conclude that the inference model $M+M/M/1/K$

Table 1.9: Percentage of hits for scheme P_L_R over the simulations

Criterion	Percentage of hits over the simulations
$\hat{\lambda}$, error within 1 %	52 %
$\hat{K}$, error within 1 %	40 %
$\hat{\lambda}$, error within 5 %	76 %
$\hat{K}$, error within 5 %	70 %
$\hat{\lambda}$, error within 9 %	98 %
$\hat{K}$, error within 9 %	84 %
bad estimation for λ	2 %
bad estimation for K	10 %
no estimation for K	6 %

returns reasonably good results even when the Poisson assumption on the background traffic and the assumption on the service times are violated. We have seen that, in general, the scheme P_L_R performs better under high load, returning good estimates for both λ and K.

1.7.4 Analysis of the results in case μ is unknown

We were not surprised to see that scheme *1* $= P_L\_U\_R$ was the "best" scheme in simultaneously estimating the cross traffic intensity λ, the server rate μ and the buffer size K. Even though this scheme was the "best", it returned results only in 31 simulations among all 50. Since this scheme uses an estimate of the utilization U, all simulations where the queue never emptied (there were 19 such simulations) did not lead to any valid estimate of U. (Recall that the value $U = 1$ is not valid in both models.) Table 1.10 lists the simulations in which the queue emptied at least once.

Table 1.10: The 31 simulations where $\hat{U} < 1$

Probe traffic		Cross traffic			
Type	Rate (pkts/s)	Type	Rate (pkts/s)	Buffer size K (pkts)	Simulations
(T1)	$\gamma = 124$	(T1)	$\lambda = 6600$	10, 30, 65, 100, 150	5
(T1)	$\gamma = 248$	(T1)	$\lambda = 6600$	10, 30, 65, 100, 150	5
(T1)	$\gamma = 496$	(T1)	$\lambda = 6600$	10, 30, 65, 100, 150, 1000	6
(T1)	$\gamma = 125$	(T1)	$\lambda = 17068$	10	1
(T2)	$\gamma = 250$	(T2)	$\lambda = 6677$	10, 30, 65, 100, 150	5
(T1)	$\gamma = 247$	(T3)	$\lambda = 6812$	10, 30, 65, 100, 150, 1000	6
(T1)	$\gamma = 246$	(T4)	$\lambda = 17437$	10	1
(T1)	$\gamma = 248$	(T5)	$\lambda = 1655$	10	1
(T1)	$\gamma = 248$	(T6)	$\lambda = 1883$	10	1

We will not discuss much the results returned by scheme $P_L\_U\_R$, as the latter does not perform especially well. The relative error (computed after 60000 probes) on λ (resp. on μ, on K) was below 4.98% (resp. below 4.72%, below 3.23%) in 17 simulations: the ones listed in rows 3–6 (except for $K = 1000$) and row 9 in Table 1.10. The relative error on each parameter is not negligible in the other 14 simulations (even after 120000 probes). The smallest error on λ (resp. on μ, on K) is 18.3% (resp. 18.4%, 22.1%), the average error on λ (resp. on μ, on K) is 75.0% (resp. 68.9%, 65.2%) and the largest error on λ (resp. on μ, on K) is 151.5% (resp. 126.4%, 102.9%)! In other words, the scheme behaves well almost only in $M+M/M/1/K$ simulations (rows 3–6 in Table 1.10). This statement is somehow biased since we have simulated congested cases solely ($\rho \in [0.965, 2.758]$). It is likely that for moderate values of ρ, the utilization will be better estimated, which will imply a better performance of scheme $P_L\_U\_R$.

1.7.5 Simulations with several links

We have seen that scheme P_L_R is the most promising one in estimating the cross traffic intensity λ and the buffer size K. Until now, we have tested this scheme on simulations in which there was only one single link. In this section, we briefly present the results returned by scheme P_L_R when applied on simulations with multiple links.

We have simulated six different scenarios all having the same network topology as illustrated in Figure 1.17. In each simulation, the probe traffic is Poisson. In two simulations, the bottleneck link is located between nodes 2 and 5, and in the other four simulations, it is located between nodes 5 and 7. Table 1.11 reports details on the cross traffic in each scenario. In each simulation, there was only one type of cross traffic considered (provided in first column of Table 1.11). Columns 2 and 3 give the total number of flows considered and the number of flows per route, respectively. There are six possible routes in the case that the cross traffic type is either Poisson or On/Off, and there are twelve routes when the cross

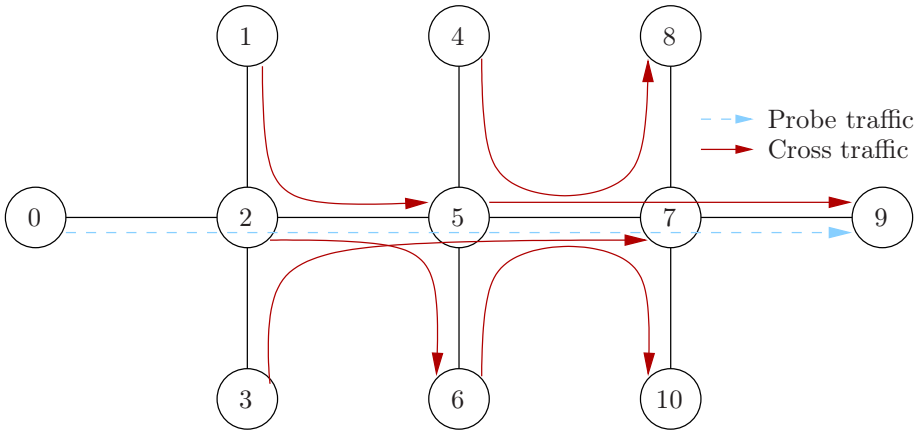


Figure 1.17: The simulated network

Table 1.11: Details on the cross traffic in the simulated scenarios

Along the path			Flows via bottleneck		
Type	Flows	Flows/route	link 2–5	link 5–7	Routes for cross traffic
Poisson	6	1	3	4	$(i, i+4), i = 1, \ldots, 6$
On/Off	750	125		500	$(i, i+4), i = 1, \ldots, 6$
FTP/TCP	1020	85	510	680	$(i, i+4), (i+4, i), i = 1, \ldots, 6$
FTP/TCP	1500	125		1000	$(i, i+4), (i+4, i), i = 1, \ldots, 6$

traffic is the aggregation of 1500 FTP/TCP flows (see column 6). Column 4 (resp. column 5) gives the number of cross traffic flows going through the bottleneck when the latter is the link 2–5 (resp. link 5–7).

We observed in our experiments that the propagation delay is crucial in simulations in which the cross traffic is transported by TCP. That is, of course, because TCP is a closed-loop protocol, as opposed to UDP which is used to transport Poisson or On/Off flows. Table 1.12 reports the relative error (expressed in percentage) returned by scheme P_L_R after 5000 probes only. We notice that $\hat{\lambda}$ performs very well (see column 5 in Table 1.12), which is not the case of $\hat{K}$, as the latter misestimates the buffer size K when the cross traffic is the aggregation of FTP/TCP flows (cf. rows 6–8 in Table 1.12, column 4). Observe that the error on K decreases from 27.6% to 16.8% when the number of FTP/TCP flows increases from 680 to 1000.

Table 1.12: Relative error of $\hat{K}$ and $\hat{\lambda}$ (expressed in %) after 5000 probes

	Flows via bottleneck		Relative error		Propagation
Bottleneck	Type	Flows	$\hat{K}$	$\hat{\lambda}$	delay considered?
link 2–5	Poisson	3	4.6	1.4	No
link 5–7	Poisson	4	5.6	0.6	No
link 5–7	On/Off	500	0.7	2.2	No
link 2–5	FTP/TCP	510	9.5	2.7	Yes
link 5–7	FTP/TCP	680	27.6	0.7	Yes
link 5–7	FTP/TCP	1000	16.8	5.5	Yes

Note that when the cross traffic is transported by TCP, it is necessary to account for the propagation delay in order for the scheme to behave well, thereby, suggesting an extension to the $M{+}M/M/1/K$ queue model in which the end-to-end delay is the sum of the response time of the queue and a propagation delay. The latter might be constant or a random variable.

1.8 Extensions

1.8.1 From simulation to reality

Till now, we have always assumed to have access to the first n samples of $\{a_i\}_i$, $\{d_i\}_i$, $\{X_i\}_i$ and $\{Y_i\}_i$. In this work, we have carried out simulations to generate traffic traces, and the samples were extracted from the traces. However, ultimately, we must extract the samples from the real network (e.g. the Internet). How can we do this?

Usually, real-time applications use the Real-time Transport Protocol (RTP) [109] together with UDP and IP. RTP provides end-to-end network transport functions suitable for this kind of application, over multicast or unicast network services. RTP consists of two parts, a data part and a control part referred to as RTCP, the RTP Control Protocol. The feedback information is carried in RTCP packets referred to as Receiver Reports (RRs).The rate at which they are multicast is controlled so that the load created by the control information is a small fraction of that created by data traffic. The RR sent by a destination includes several pieces of information: the highest sequence number received, the number of packets lost, the estimated packet interarrival jitter, and timestamps. At the sender, the $\{a_i\}_i$ are available, the $\{d_i\}_i$ are retrieved from the timestamps present in the RR and the $\{X_i\}_i$ are retrieved from the highest sequence number received at the destination, also given in the RR. As for the $\{Y_i\}_i$, they are hard to obtain. In [86], the authors propose an algorithm to estimate the clock skew in network delay measurements. This algorithm can be adapted to estimate the $\{Y_i\}_i$ (mainly, $Y = 0$ if the measured delay is minimal) but this estimation will be weak and uncertain, as the service times are considered constant. Fortunately we found that $P_{L\_}R$ is the best scheme and hence we will not use U. The work assumes a perfect knowledge of the bottleneck capacity. In some cases, μ is known exactly (we know the routes and we know the routers) but sometimes μ is to be estimated and this introduces further error. Its impact still needs to be investigated.

1.8.2 Example of a possible application

An interesting application for the methods proposed in this chapter is routing in content distribution networks. The goal would be to infer the available bandwidth and buffer size of the bottleneck queue on a path between two application layer routers so as to determine how to route new traffic. It should be possible to probe at a sufficiently high rate to quickly obtain good estimates. This could be done by embedding a Poisson stream within the data traffic and/or adding a (relatively) low bandwidth probe stream. Furthermore, between two application layer routers, we expect there to be one bottleneck node residing at a peering point between the two backbone networks within which the routers reside.

1.9 Conclusion

In this work, we have proposed two simple models for a connection, based on a single server queue with finite waiting room, to infer the *buffer size* and the *intensity of cross traffic* at the bottleneck link of a path between two hosts. We have quantified several parameters of both models and obtained eleven pairs of *moment-based* estimators. Using traces generated by the network simulator `ns-2`, estimated values for both parameters have been calculated according to the characteristics of the *a priori* models. Pairs of estimators have been discarded while others have proved to give good results. However, the pair of estimators we have "elected" as the best one need to be tested on an experimental network, or even better, on the Internet, in order to evaluate its performance under realistic network traffic conditions.

Chapter 2

Estimation of multicast membership

The second chapter of the thesis concerns multicast applications that are interested in the evolution of their membership over time. The chapter covers optimal on-line estimation algorithms for determining the membership of a multicast group. Throughout the chapter, three distinct methodologies are detailed and analyzed. The first one builds on Kalman filter theory to derive an optimal estimator that is tested both on synthetic traces and real audio traces. Under more general assumptions, the second approach relies on Wiener filter theory to achieve the same result, whereas the third approach develops the best first-order linear filter from which an estimator that holds for any lifetime distribution is derived. This methodology is illustrated in the case where the distribution of the receivers lifetime is hyperexponential. The chapter also provides guidelines on how to tune the parameters involved in the schemes in order to achieve high quality estimation while simultaneously avoiding feedback implosion.

Keywords: on-line estimation, multicast, $M/G/\infty$ queue, diffusion, Kalman filter, Wiener filter, simulation, validation, distribution fit.

Note: Parts of the material presented in this chapter are published in [9, 10, 11, 12].

2.1 Introduction

Since its introduction, IP multicast [43, 44] has seen slow deployment in the Internet. As stated in [45], the service model and architecture do not efficiently provide or address many features required for a robust implementation of multicast. However, the fact remains that IP multicast is very appealing in offering scalable point-to-multipoint delivery specially in satellite communications. Current research efforts tend to propose alternatives to IP multicast like the so-called "application layer multicast" [36, 56, 68, 96, 104], the idea being to deploy multicast at the application layer. Also, new models to support multicast communications in a more effective way have been proposed, such as the EXPRESS multicast [63]. The latter is an extension to IP multicast that provides explicit support for large-scale multicast applications such as real-time stock quote dissemination, live sports video feeds or Internet radio and TV. EXPRESS provides as well a *best-effort* count of the number of subscribers.

This work is motivated by the conviction that large-scale multicast applications will be widely deployed in the future as soon as the capability becomes available. We believe that membership estimates will be an essential component of this widespread deployment as they can be very useful for scalable multicast. The membership of a session can be used for feedback suppression as it is the case in current protocols such as RTP [109] and SRM [54]. In order to regulate the amount of session/control messages sent by receivers – the idea being not to exceed 5% of overall session bandwidth – these protocols use delay timers that are tuned based on the membership estimates.

The membership of a multicast session can be used for charging the sources in large-scale applications. ISPs traditionally charge their customers on an input-rate basis. An alternative pricing scheme would be to charge sources based on their audience size which is more profitable in the case of millions of subscribers.

Estimating the size of a multicast session can be quite useful to many applications. Bolot, Turletti and Wakeman [23] use membership estimation to further estimate the proportion of congested receivers as needed in their videoconference system IVS [65]. Future Internet radios and TVs will need to characterize their audience preferences and to follow the fluctuations of the audience size over time. Dutta, Schulzrinne and Yemini proposed an architecture for Internet radio and TV called MarconiNet [52] that relies on RTCP [105, 109]. Even though RTCP provides an easy mechanism for collecting statistics on the size of the audience, it does not scale well to large multicast sessions. In such applications, sampling-based techniques are more appropriate.

There has been a significant research effort in devising sampling-based schemes for the estimation of the membership in multicast sessions [23, 57, 79, 87]. The feedback algorithms presented in these references are all *at-least-one* scenarios in the sense that the membership estimation is based on at least one acknowledgement coming from the receivers. In these probabilistic schemes, the receivers send ACKs to the source as a reply to a specific request, either with a certain probability as in [23], or after some random time like in [57, 79, 87]. But

what is common to these schemes – except the one used in [87] – is that they all assume that the size of the group does not change during the estimation process. Whenever an estimation of the population size is needed, the application re-runs the estimation algorithm without taking into account previous estimates.

In this work, we propose a novel sampling-based technique which is not an *at-least-one* scenario. Whenever a source is interested in knowing how many recipients are connected to the multicast session (or are actively following some application that is being broadcast), it could ask all of the connected members to send an acknowledgment (ACK). But this is undesirable in case of large populations as the ACKs could overload the network. To avoid this, the source could alternatively ask each active connected receiver to send an ACK with some *small* probability p. Yet, p needs to be chosen carefully because if it is too small, the estimation would be inaccurate.

If the source wishes to further have the possibility of tracking the population size, it should ask the receivers to repeatedly send ACKs say every S seconds. Occasionally, the source re-issues this request to insure that newly arrived receivers participate in the polling. If S is not too large then the population size at two consecutive estimation instants would present statistical dependence. The engineering question we pose in this chapter is how can we benefit from this dependence in order to be able to get better estimation, or alternatively, to get a given quality of estimation with a smaller required volume of ACKs (i.e. decreasing p or increasing S).

Throughout the chapter, we will address the issue of tracking the membership of a multicast group. Considering the sampling scheme proposed earlier, we build on adaptive filter theory to derive the estimator. Three distinct approaches are successively analyzed, starting with Kalman filter theory, proceeding with Wiener filter theory and ending with least square estimation given a particular filter structure.

The Kalman filter is introduced for the heavy traffic regime under which the membership process weakly converges to an Ornstein-Ühlenbeck process. Due to the linearity of the latter, the Kalman filter is optimal. Note that the Kalman filter gives a linear, unbiased, and minimum error variance recursive algorithm, and under normality assumptions, this filter is optimal, not only among all linear filters based on a set of observations, but among all measurable filters [107, Section 2.2].

The Wiener filter is introduced next for a general traffic regime. The heavy traffic assumption is relaxed at the cost of the restriction to a class of filters which is linear and optimal only among all linear filters. In Wiener filter theory, the minimum mean-square error criterion is used to optimize the filter, and unlike Kalman filters which are applied in linear systems in general, Wiener filters are time invariant and are only applied in linear time invariant systems.

The 'Least Mean Squares' algorithm is introduced for a general traffic regime and a general distribution of the receivers lifetime. A first-order linear filter, which is optimal among the class of all first-order linear filters, is derived and its parameters are computed numerically in the case that the distribution of the receivers lifetime is hyperexponential.

The chapter is organized as follows. First, we will briefly overview the sampling-based estimation schemes studied in the literature (Section 2.2) and discuss the motivations behind our work (Section 2.3). Our first approach, based on Kalman filter theory, is developed in Section 2.4 and validated via simulations, driven by both synthetic and real traces. Section 2.5 builds on Wiener filter theory to design the membership estimator, whereas a least square estimation method is adopted in Section 2.6. Before undertaking the robustness of the latter pair of estimators in Section 2.8, we propose some guidelines on how to choose parameters p and S (Section 2.7). Finally, ongoing research is discussed in Section 2.9.

2.2 Related work

In the previous section, we briefly introduced the different feedback mechanisms proposed in the literature. Throughout this section, we will review each technique separately, following their chronological conception.

The feedback mechanism proposed by Bolot, Turletti and Wakeman in [23] (**BTW** mechanism) consists of a series of probabilistic polling rounds, each with a higher reply probability than in the previous one, until feedback is obtained. Each round begins when the sender (or source) multicasts a polling request in which the reply probability p_n is specified. In the first round, $p_1 = 2^{-16}$, and for each subsequent round n, $n \geq 1$, we have $p_n = 2^{n-2}/(2^{16} - 2^{n-2})$. After issuing a polling request, the sender sets a timer at twice the largest round trip time in the receiving group. When the timer timeouts, the sender initiates a new round with a higher probability. The polling rounds keep going until either a reply has been received, or the round in which the reply probability is 1 has been reached, in which case any receiver will send a response. This ends the ongoing series of rounds, or epoch. This probabilistic scheme is used to estimate the number of receivers in a multicast group and further estimate the proportion of congested receivers in the group. The authors of [23] map the number of receivers N to the average round $\mathbf{E}[\text{First}]$ in which the first reply is received as follows:

$$N \sim e^{16.25 - \mathbf{E}[\text{First}]/1.4} = \hat{N}_{BTW}. \tag{2.1}$$

The **BTW** mechanism relies on probabilistic arguments for scalability and it avoids feedback implosion unless N is an order of magnitude greater than 2^{16}. However, the estimator deriving from (2.1) does not fully render the variations in the membership, and as time goes on, $\mathbf{E}[\text{First}]$, and therefore $\hat{N}_{BTW}$, converges to its average value over the entire multicast session duration. The **BTW** estimator, $\hat{N}_{BTW}$, has been analyzed in [57], and an extension to the mechanism is therein proposed.

The notion of timer-based schemes for multicast feedback was first mentioned in [124], but Nonnenmacher and Biersack were the first to deeply analyze this family of mechanisms [87, 88]. They have evaluated the performance of timer-based schemes, in which the timer is either uniformly distributed, beta distributed or exponentially distributed. The main

concern of the authors is the scalability to groups as large as 10^6 receivers. The feedback implosion problem is handled at the receivers: each participant multicasts his response unless he receives one from another participant, in which case he will suppress his own feedback. The analysis performed in [87] revealed that, from the set considered, the best distribution in terms of feedback latency, sensitivity to poor estimates of the number of receivers and feedback suppression is the exponential distribution. The timer-based feedback proposed by Nonnenmacher and Biersack (**NB** mechanism) is intended for reliable multicast, but it can be used as well for membership estimation.

The **NB** feedback mechanism is round-based and works as follows. Based on an estimation of N, the number of receivers, the sender computes λ and T, the parameters of a truncated exponential distribution (each timer coming from this distribution is in the interval $[0, T]$). At the beginning of each round n, the source multicasts a request for feedback (n, λ, T). The receivers set their timers accordingly and send a feedback message when the timer times out, unless it is suppressed by another message. On the receipt of the feedback messages, the sender estimates N using the timer settings of all of the receivers that returned feedback, which triggers the computation of λ and T for the subsequent round. To express the estimator of N, let $F(z)$ be the distribution of the truncated exponential timer z, c be the *constant* delay between receivers and between any receiver and the sender, m be the minimal timer among the feedback returned and Y be the amount of feedback returned. It is therefore shown [87] that

$$F(z) = \frac{e^{\lambda z/T} - 1}{e^{\lambda} - 1},$$

$$\hat{N}_n = Y \frac{1 - F(m)}{F(m + c) - F(m)} = Y \frac{e^{\lambda(1-m/T)} - 1}{e^{\lambda c/T} - 1}, \tag{2.2}$$

$$\hat{N}_{n,\alpha} = \begin{cases} 1, & n = 1, \\ \alpha \hat{N}_{n-1,\alpha} + (1 - \alpha)\hat{N}_n, & n > 1. \end{cases} \tag{2.3}$$

The exponential weighted moving average expression $\hat{N}_{n,\alpha}$, with $\alpha = 0.8$, is used as membership estimator, whereas the expression of $\hat{N}_n$ is used to compute λ and T to react faster to changes in the population size. It readily comes from (2.3) that $\mathbf{E}[\hat{N}_{n,\alpha}] = \mathbf{E}[\hat{N}_n]$ in steady-state, but it is seen that $\mathbf{E}[\hat{N}_n] \neq N$, and the bias on the estimator depends on the membership N as illustrated in [57]. Another issue is the choice of the parameter α in (2.3). Nonnenmacher and Biersack suggest the use of $\alpha = 0.8$ to achieve a fast convergence and a reasonably smooth estimate. However, it is not known whether this choice is optimal or not.

To the best of our knowledge, Friedman and Towsley were the first ones to investigate the estimation of the membership size as a whole. In [57], they base their analysis upon a mapping of the polling mechanisms to the problem of estimating the parameter N of the binomial (N, p) distribution, they derive an interval estimator for N and bounds for the

amount of feedback as well as the polling probability in order to achieve specific requirements. They apply their results on both mechanisms introduced in [23, 87] which have point estimators and further add some contributions to each. Friedman and Towsley propose an extension to the **BTW** mechanism that achieves a desired measure of quality. In this new feedback scheme, as soon as a reply reaches the sender, the polling continues at the same probability until a minimum amount of feedback h_{min} is reached. The interval estimator proposed in [57] is centered on $\mathbf{E}[Y]/p$ where Y denotes the amount of feedback received in each round (only the last k rounds having a reply probability of p are considered) and p denotes the reply probability in the last k rounds. It is shown that

$$\tilde{N} = \frac{\mathbf{E}[Y]}{p} \pm \frac{z_{\alpha/2}}{2kp}\left((1-p)z_{\alpha/2} + \sqrt{4k(1-p)\mathbf{E}[Y] + (1-p)^2 z_{\alpha/2}^2}\right). \tag{2.4}$$

where $z_{\alpha/2}$ is the value for which the CDF of the standard normal distribution is $\Phi(z_{\alpha/2}) = (1-\alpha/2)$. Note that an estimate for N is computed only after several polling rounds, which was not the case for the estimator proposed by Nonnenmacher and Biersack which returns an estimate at each polling round (see (2.3)). Another weakness of (2.4) is that it does not take advantage of previous estimates as does (2.3).

Concerning the **NB** mechanism, Friedman and Towsley map the timer-based scheme to their binomial model. The reply probability is then

$$p = \frac{F(m+c) - F(m)}{1 - F(m)}$$

and (2.2) can be rewritten $\hat{N}_n = Y/p$. This point estimator is known to be bursty and in order to have a more stable estimator, Nonnenmacher and Biersack have introduced the estimator $\hat{N}_{n,\alpha}$ defined in (2.3). The latter does not fit in the binomial model presented in [57] as the reply probability in the timer-based scheme changes from round to round, and it is not possible to rely on any two polling rounds resulting in the same probability, as required in the binomial model. Thus, the authors have restricted themselves to estimation over a single polling round, and propose a new point estimator $\hat{N}_{FT} = 1 + (Y-1)/p$ which revealed to be unbiased, whereas the point estimator Y/p is biased (recall that Y denotes the amount of feedback received in a single polling round). The interval estimator proposed in [57] is centered on $\hat{N}_{FT}$ and given by

$$\tilde{N}_{FT} = 1 + \frac{Y-1}{p} \pm \frac{z_{\alpha/2}}{2kp}\left((1-p)z_{\alpha/2} + \sqrt{4k(1-p)(Y-1) + (1-p)^2 z_{\alpha/2}^2}\right). \tag{2.5}$$

One major contribution of [57] is that it removes the assumption on homogeneous delays used in the **NB** mechanism. Friedman and Towsley propose a modification to this mechanism that makes Y denote the amount of feedback received *should* the inter-host delays be constant and equal to c. The interval estimator given in (2.5) is tested via simulations

under the **NB** mechanism and the modified one, provided that the inter-host delays are beta distributed. Their simulations revealed that $\tilde{N}_{FT}$ is unbiased under the modified mechanism and biased under the **NB** mechanism. The last contribution of [57] is a maximum likelihood estimator for the **NB** mechanism that uses information from multiple polling rounds. We believe that such an estimator will not fully render the dynamics of the membership, as it requires several polling rounds to return an estimate.

Another timer-based feedback scheme is proposed in [79] in which receivers send their randomly delayed reply only to the source which in turn initiates a new round of replies. Each request for replies sent by the source would reset the timers at the receivers. Two versions of the mechanism are proposed depending on whether the estimation is based on the first arrival solely or on all the received responses. The latter version improves the accuracy of the estimator, but in both versions, there is a risk of a feedback implosion. In each version, a maximum likelihood estimator is derived and multiple polling rounds are necessary to return a single estimate. The paper focused on the quality of the estimator rather than on the dynamic nature of multicast sessions.

To conclude this section on related works, we would like to briefly discuss a paper "On the scaling of feedback algorithms for very large multicast groups" [58]. This paper does not deal with the estimation of the membership itself, but rather on its impact on three feedback algorithms, which all are *at-least-one* scenarios. It is concluded that the possible estimations of the group size might be a source for disturbances and that the *exponential feedback raise* algorithm is the algorithm of choice for very large groups. Both the **BTW** and **NB** mechanisms are considered to have an exponential feedback raise.

2.3 Motivation

In order to fully reproduce the evolution of the multicast membership, we aim at developing a moving average estimator like the one given in (2.3). That estimator was shown to be biased[1], while what we are actually looking for is an *unbiased* estimator that would take advantage of previous estimates in an *optimal* way.

We propose a mechanism in which the receivers probabilistically send "heartbeats" to the sender (hereafter called the source) in a periodic way. This mechanism is much simpler to implement at the receivers side than is the timer-based mechanism used in [87]. The feedback implosion problem is addressed via a convenient choice of the reply (or ACK) probability p. The ACK interval S between two consecutive polling instants has to be larger than the largest round-trip time between a receiver and the source. Inter-hosts delays are not required to be homogeneous. That is because of the periodic nature of the ACKs generation process and because the ACK interval S is large enough in order to have all of the ACKs produced in a round reach the source before the (automatic) start of the next round.

[1]It is shown in [57] that the estimator given in (2.2) is biased and we know that $\mathbf{E}[\hat{N}_{n,\alpha}] = \mathbf{E}[\hat{N}_n]$.

A naive approach to the estimation problem would consist of dividing Y_n, the total number of ACKs received at the nth observation step, by the ACK probability p. This ratio is then a point estimator of the size of the multicast group at time nS. It is expected that this estimator will perform very poorly, both because it does not take into account the "history" of the membership process, and because the ratio of the number of ACKs received over the group size will converge to p a.s. only when the group size is large (the strong law of large numbers). (This is why the point estimator in the center of the interval estimator given in (2.4) takes the expectation of Y instead of just taking Y.)

Our experiments reported in Figure 2.1 have confirmed the poor behavior of this naive estimator. Figure 2.1 depicts the evolution of membership size of an audio session recorded in December 1996 (details in Section 2.4.4) and the estimate returned by the naive approach. The ACK interval S is set to 1 second. Two values (0.01 and 0.5) for the ACK probability p were retained. Figure 2.1(a) displays the evolution of the number of group members and its "naive" estimation over time for $p = 0.01$. Obviously, there is too much noise in the estimation. Results for $p = 0.5$ are shown in Figure 2.1(b). Even for this high (undesirable) probability, the naive estimator performs poorly. It is clear that some filtering is necessary in order to reduce the estimation error.

A first approach to filter out the noisy observations consists of using an exponential weighted moving average (EWMA) like the one used in (2.3). We can write

$$\hat{N}_n = \alpha \hat{N}_{n-1} + (1 - \alpha)\frac{Y_n}{p}. \tag{2.6}$$

In steady-state, we have $\mathbf{E}[\hat{N}_n] = \mathbf{E}[Y_n]/p = \mathbf{E}[N_n]$. The estimator $\hat{N}_n$ is then unbiased. It remains to choose the parameter α in an optimal way. We have tested several values of α, namely 0.95, 0.99 and 0.999. The performance of the EWMA algorithm can be visually observed in Figure 2.2. For the particular audio trace therein plotted, it appears that $\alpha = 0.999$ is a good choice.

The main drawback of this approach, is that one can not know what value of α is the best, and one also does not know whether a good value of α for one trace will also be good for another. The correct approach is to compute the optimal α that minimizes the estimation error. Notice that (2.6) is an autoregressive equation of the form

$$\hat{N}_n = A\hat{N}_{n-1} + BY_n.$$

One might wonder if this form is the best one or not. Instead of computing the optimal α for this particular form, we will rely on adaptive filter theory to construct the best estimator, and the equation giving this estimator might well take another form than the one in (2.6).

Remark 2.3.1 *Throughout this chapter, it is assumed that neither the requests for ACKs sent by the source, nor the ACKs sent by the receivers, are lost. The loss of polling requests has a smaller impact on the membership estimation mechanism as the source can repeatedly*

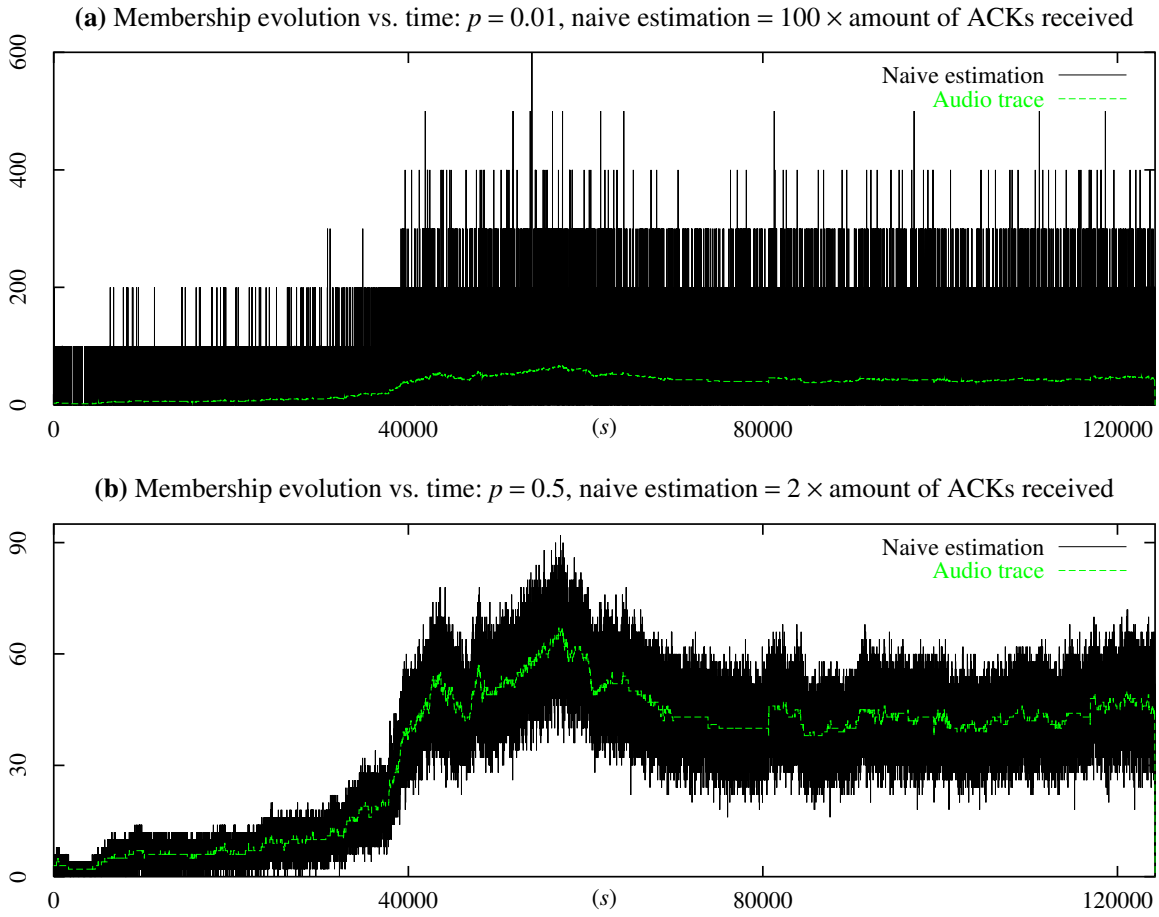


Figure 2.1: Membership evolution of a short audio session and its estimation using a naive approach ($S = 1s$)

send them or send a group of them whenever the parameters S and/or p are to be changed. As for the loss of ACKs, it is possible to incorporate the loss probability in our feedback mechanism. Let p_L denote the probability that an ACK is lost before it reaches the source. It suffices then to require an ACK probability of $p/(1-p_L)$ in order to have, over the session duration, an average of $p\boldsymbol{E}[N]$ ACKs received per round. This is equivalent to requiring an ACK probability of p in a safe environment (i.e. no losses).

2.4 Optimal estimation using a Kalman filter

A precise mathematical formulation of the estimation problem would require the use the theory of nonlinear stochastic filtering, which does not provide us with tractable solutions. We shall instead introduce some simplifying assumptions that will allow us to obtain a good estimation scheme, which, even if not always the optimal, will show good perfor-

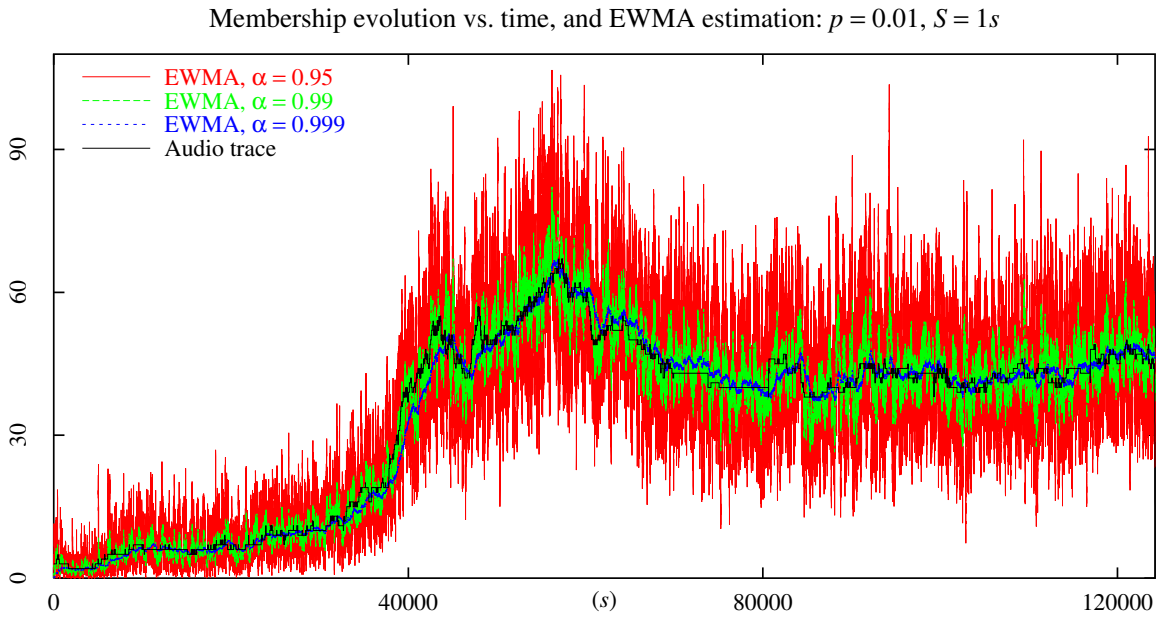


Figure 2.2: Membership evolution of a short audio session and EWMA estimation

mance. To that end we shall consider an exponential distribution for the time during which a receiver stays in the multicast session (referred to as "lifetime" or even "on-time") and make a large group size assumption. This will allow us to obtain a diffusion approximation for the population dynamics. Sampling this process at some regular time intervals will yield a discrete-time linear stochastic difference equation for the population dynamics. We will further derive a linear discrete-time equation for the measurements. The fact that both the population dynamics and the measurements in our approximations are linear, will allow us to use the powerful Kalman filtering theory to design simple dynamic estimation procedures which are optimal for the heavy traffic model (in minimizing the second moment of the error). These schemes thus make the best use of previous estimates in order to update the current estimation optimally. Having proposed a dynamic estimation procedure that is optimal in our simplified mathematical model, we will test it on real traces that do not satisfy the assumptions of that model. We will observe good performance of this procedure in these cases.

We will next present the queueing model for the multicast group (Section 2.4.1). After that, we will derive the membership estimator in Section 2.4.2 and validate our approach in Sections 2.4.3 and 2.4.4.

2.4.1 The model

We consider a multicast group that participants join and leave at random times. Let T_i and $T_i + D_i$ be the join time and the leave time, respectively, of the ith participant. In the following, $D_i > 0$ is called the on-time of the ith participant and $\{D_i\}_i$ is referred to

as the on-time sequence. Let $N(t)$ be the number of participants at time t or, equivalently, the size of the multicast session at time t. Under the enforced assumption that $N(0) = 0$, we have

$$N(t) = \sum_i \mathbf{1}\{T_i \leq t < T_i + D_i\} \tag{2.7}$$

where $\mathbf{1}\{E\}$ is equal to 1 if the event E occurs and to 0 otherwise.

The source requests its receivers to send ACKs with probability p every S seconds. It is assumed that neither polling requests nor ACKs can be lost (see Remark 2.3.1). We assume that S is large enough so that all of the ACKs generated in each round reach the source before the start of the subsequent round. The times $t = nS, n = 1, 2, \ldots$, will denote the end of each round. Under the above assumptions, it is seen that, at time nS, the source possesses all of the ACKs sent to it by participants in the nth round (i.e. in the interval of time $](n-1)S, nS]$). Throughout, p and S are held fixed (see Section 2.7 for possible extensions).

Given this scheme, our objective is to devise an algorithm for estimating the session size at times $t = nS$ for $n = 1, 2, \ldots$.

Primarily for mathematical tractability we shall assume from now on that the arrival process is Poisson with rate $\lambda > 0$ and that on-times form a renewal sequence with common exponential distribution with finite mean $1/\mu$, further independent of the arrival process (more general arrival processes can be considered – see Remark 2.4.1). In this setting, the process $\{N(t), t \geq 0\}$ as defined in (2.7) is simply the occupation process in a $M/M/\infty$ queueing system with arrival rate λ and mean service time $1/\mu$ [75]. For such a queue, it is known that the stationary number of busy servers is distributed according to a Poisson random variable (RV) with parameter $\rho := \lambda/\mu$; in particular, the mean number of busy servers in steady-state is equal to ρ.

Unfortunately, little is known about the transient behavior of the $M/M/\infty$ for a fixed traffic intensity ρ. We will instead investigate the $M/M/\infty$ queue in heavy traffic. To this end, let us introduce the scaled process $\{N_T(t), t \geq 0\}$ which is identical to the original process $\{N(t), t \geq 0\}$ except that the arrivals have been speeded up by a factor T, that is, the arrival rate in the $M/M/\infty$ queue is now λT. The mean service rate is kept unchanged and equal to $1/\mu$.

Since $N_T(t) \to \infty$ a.s. as $T \to \infty$, we will instead work with the normalized process $\{Z_T(t), t \geq 0\}$ defined by

$$Z_T(t) = \frac{N_T(t) - \rho T}{\sqrt{T}}, \quad t \geq 0. \tag{2.8}$$

The process $\{Z_T(t),\ t \geq 0\}$ describes the fluctuations of the scaled process $\{N_T(t), t \geq 0\}$ around its limiting trajectory ρT as $T \to \infty$.

A nice feature of the process $\{Z_T(t),\, t \geq 0\}$ is that it converges to a diffusion process as $T \to \infty$. More precisely, as $T \to \infty$ the process $\{Z_T(t),\, t \geq 0\}$ converges in distribution to the Ornstein-Ühlenbeck process $\{X(t),\, t \geq 0\}$ given by [103, Theorem 6.14, page 155]

$$X(t) = e^{-\mu t}\, X(0) + \sqrt{2\lambda} \int_0^t e^{-\mu(t-u)}\, \mathrm{d}B(u), \tag{2.9}$$

where $\{B(t),\, t \geq 0\}$ is the standard Brownian motion (see also [24, Theorem 1, page 172]). The Ornstein-Ühlenbeck process defined in (2.9) is an ergodic Markov process and its invariant distribution is a normal distribution with mean zero and variance ρ [70, page 358].

In the next section, we shall devise optimal estimators for the elements of the sequence $\{X(nS), n = 1, 2, \ldots\}$ based on Kalman filter theory.

A word on the notation in use: $N(m, v)$ will denote a normal distribution with mean m and variance v and $X \sim N(m, v)$ will denote a RV with distribution $N(m, v)$.

Remark 2.4.1 . *The convergence of the process $\{Z_T(t),\, t \geq 0\}$ to a diffusion process (but with different coefficients than that in (2.9)) still takes place if the arrival process is replaced by a process slightly more general than a Poisson process [24, Theorem 1, pages 172-173]. On the other hand, if the on-times are generally distributed and the arrival process is Poisson, then one only knows that the process $\{Z_T(t),\, t \geq 0\}$ converges to a Gaussian process [106].*

2.4.2 Kalman filter

In order to achieve an optimal estimation for the heavy traffic model, we shall use a Kalman filter that computes the state estimator out of two linear equations: the system dynamic equation and the measurement equation. These two equations are introduced in the next two sections. Throughout these sections we shall assume that the process $\{X(t),\, t \geq 0\}$ is in equilibrium at time $t = 0$, namely, $X(0) \sim N(0, \rho)$.

System dynamics

From (2.9) we obtain

$$X(t) = e^{-\mu(t-s)}\, X(s) + \sqrt{2\lambda} \int_s^t e^{-\mu(t-u)}\, \mathrm{d}B(u)$$

for $0 \leq s \leq t$, from which it follows that

$$\xi_{n+1} = \gamma\, \xi_n + w_n, \quad n = 0, 1, \ldots \tag{2.10}$$

where

$$\xi_n := X(nS),$$
$$\gamma := e^{-\mu S},$$
$$w_n := \sqrt{2\lambda} \int_{nS}^{(n+1)S} e^{-\mu((n+1)S-u)}\, \mathrm{d}B(u).$$

The RVs $\{w_n,\, n = 0, 1, \ldots\}$ are i.i.d. with

$$w_n \sim N(0, Q), \quad n = 0, 1, \ldots \tag{2.11}$$

(see e.g. [24, page 17]) where Q is given by

$$\begin{aligned} Q &= 2\lambda\, \mathbf{E}\left[\int_{nS}^{(n+1)S} e^{-\mu((n+1)S-u)}\, \mathrm{d}B(u)\right]^2 \\ &= 2\lambda \int_{nS}^{(n+1)S} e^{-2\mu((n+1)S-u)}\, \mathrm{d}u \\ &= \rho\,(1-\gamma^2). \end{aligned}$$

Notice that

$$\xi_n \sim N(0, \rho), \quad n = 0, 1, \ldots \tag{2.12}$$

under the assumption that $X(0) = \xi_0 \sim N(0, \rho)$, which implies from (2.10) that

$$\mathrm{Cov}(\xi_n, \xi_{n+1}) = \gamma\rho, \quad n = 0, 1, \ldots. \tag{2.13}$$

Equation (2.10) establishes a simple one-order recursive expression relating the state of the limiting process $\{X(t),\, t \geq 0\}$ between two consecutive polling instants. We shall next derive the corresponding measurement discrete-time equation, which will allow us to use standard optimal estimation techniques.

Measurement equation

Let ζ_n^i be the indicator function that receiver $i = 1, 2, \ldots, N_T(nS)$ has sent an ACK in the nth polling round, with $\zeta_n^i = 1$ if an ACK was sent by receiver i and $\zeta_n^i = 0$ otherwise. From the definition of the model it is seen that, conditioned on $N_T(nS)$, $\zeta_n^1, \ldots, \zeta_n^{N_T(nS)}$ are i.i.d. Bernoulli RVs with $\mathbf{E}[\zeta_n^i] = p$. The conditional expectation and variance of the number of ACKs $Y_n := \sum_{i=1}^{N_T(nS)} \zeta_n^i$ received by the source at time nS are then respectively given by

$$\mathbf{E}\left[Y_n \,|\, N_T(nS)\right] = N_T(nS)\, p \tag{2.14}$$
$$\mathrm{Var}\left[Y_n \,|\, N_T(nS)\right] = N_T(nS)\, p(1-p). \tag{2.15}$$

We define our normalized measurement equation as

$$M_T(nS) = \frac{Y_n - p\rho T}{\sqrt{T}}, \quad n = 0, 1, \ldots. \tag{2.16}$$

which, with the help of (2.8), can be rewritten as

$$M_T(nS) = p\, Z_T(nS) + V_T(nS), \tag{2.17}$$

where

$$V_T(nS) := \frac{Y_n - N_T(nS)p}{\sqrt{T}}. \tag{2.18}$$

The next step is to let $T \to \infty$ in (2.17).

Proposition 2.4.1 *There exist i.i.d. RVs* $\{v_n,\, n = 0, 1, \ldots\}$ *with*

$$v_n \sim N(0, R), \quad n = 0, 1, \ldots \tag{2.19}$$

where $R := \rho\, p\, (1 - p)$*, independent of* $\{w_n, n = 0, 1, \ldots\}$*, such that* $\{v_k, k = n, n + 1, \ldots\}$ *is independent of* $\{\xi_k, k = 0, 1, \ldots, n\}$ *for* $n = 0, 1, \ldots$*, and such that* $(Z_T(nS), V_T(nS))$ *converges weakly to* (ξ_n, v_n) *as* $T \to \infty$*.* ♦

Proof. Define

$$Z(m, n) := \frac{\sum_{i=1}^{m} \zeta_n^i - mp}{\sqrt{m}}.$$

Observe that

$$V_T(nS) = Z(N_T(nS), n) \sqrt{\frac{N_T(nS)}{T}} \tag{2.20}$$

where $V_T(nS)$ is defined in (2.18).

The RV $Z(m, n)$ converges weakly as $T \to \infty$ to a normal RV ℓ_n with mean zero and variance $p\,(1 - p)$. Equivalently, for any bounded continuous function f,

$$\lim_{m \to \infty} \mathbf{E}\,[f(Z(m, n))] = \mathbf{E}[f(\ell_n)].$$

Since $N_T(nS)/T$ converges P a.s. to ρ as $T \to \infty$ [103, Theorem 6.13, pages 153-154], it follows that

$$\lim_{T \to \infty} \mathbf{E}\,[f(V_T(nS)) \,|\, N_T(nS)] = \mathbf{E}[f(v_n)] \tag{2.21}$$

where $v_n = \rho\,\ell_n$ is a normal RV with mean zero and variance $\rho p\,(1-p)$.

Now let f and g be two arbitrary bounded continuous functions. Then

$$
\begin{aligned}
&\lim_{T\to\infty} \mathbf{E}\Big[f(V_T(nS))g(Z_T(nS))\Big] \\
&= \lim_{T\to\infty} \mathbf{E}\Big[\mathbf{E}\Big[f(V_T(nS))g(Z_T(nS))|N_T(nS)\Big]\Big] \\
&= \lim_{T\to\infty} \mathbf{E}\Big[\mathbf{E}\Big[f(V_T(nS))|N_T(nS)\Big]g(Z_T(nS))\Big] \\
&= \lim_{T\to\infty} \mathbf{E}\Big[\Big\{\mathbf{E}[f(V_T(nS))|N_T(nS)] - \mathbf{E}[f(v_n)] + \mathbf{E}[f(v_n)]\Big\}\Big\{g(Z_T(nS)) - g(\xi_n) + g(\xi_n)\Big\}\Big] \\
&= \lim_{T\to\infty} \mathbf{E}\Big[\Big\{\mathbf{E}[f(V_T(nS))|N_T(nS)] - \mathbf{E}[f(v_n)]\Big\}\Big\{g(Z_T(nS)) - g(\xi_n)\Big\}\Big] \qquad (2.22) \\
&+ \lim_{T\to\infty} \mathbf{E}\Big[g(\xi_n)\{\mathbf{E}[f(V_T(nS))|N_T(nS)] - \mathbf{E}[f(v_n)]\}\Big] \qquad (2.23) \\
&+ \lim_{T\to\infty} \mathbf{E}\Big[\mathbf{E}[f(v_n)]\{g(Z_T(nS)) - g(\xi_n)\}\Big] \qquad (2.24) \\
&+ \lim_{T\to\infty} \mathbf{E}\Big[\mathbf{E}[f(v_n)]\,g(\xi_n)\Big]. \qquad (2.25)
\end{aligned}
$$

The limit which we are searching for is the sum of four distinct limits. It appears that:

- the limit (2.22) reduces to 0 due to the bounded convergence theorem,
- the limit (2.23) reduces to 0 thanks to Equation (2.21),
- the limit (2.24) reduces to 0 because of the weak convergence of $Z_T(nS)$ to ξ_n as $T \to \infty$,
- the limit (2.25) is equal to $\mathbf{E}[f(v_n)]\mathbf{E}[g(\xi_n)]$

We therefore have

$$\lim_{T\to\infty} \mathbf{E}\Big[f(V_T(nS))\,g(Z_T(nS))\Big] = \mathbf{E}[f(v_n)]\,\mathbf{E}[g(\xi_n)]. \qquad (2.26)$$

On the probability space that carries the RVs $\{\xi_n, w_n\}_n$ one can always construct the RVs v_n so that they are i.i.d. with a normal distribution with mean zero and variance $\rho p\,(1-p)$, further independent of $\{w_n,\, n \geq 0\}$ and such that, for every $n \geq 0$, the $\{v_k,\, k \geq n\}$ are independent of $\{\xi_0, \xi_1, \ldots, \xi_n\}$. Under this construction, we deduce from (2.26) that

$$\lim_{T\to\infty} \mathbf{E}\Big[f(V_T(nS))\,g(Z_T(nS))\Big] = \mathbf{E}[f(v_n)\,g(\xi_n)]$$

or, equivalently, that $(Z_T(nS), V_T(nS))$ converges weakly to (ξ_n, v_n) as $T \to \infty$ (Note: choose $f(x) = \exp(it_1x)$ and $g(x) = \exp(it_2x)$ with t_1 and t_2 real numbers), which concludes the proof. ■

We deduce from Proposition 2.4.1 that $M_T(nS)$ defined in (2.16) converges weakly as $T \to \infty$ to a RV m_n such that

$$m_n = p\xi_n + v_n, \quad n = 0, 1, \ldots. \tag{2.27}$$

The properties enjoyed by the RVs v_n together with (2.12), (2.13) and (2.27) readily imply that

$$m_n \sim N(0, \rho\, p) \tag{2.28}$$

$$\text{cov } (m_n, m_{n+1}) = \gamma \rho\, p^2, \quad n = 0, 1, \ldots. \tag{2.29}$$

Deriving the filter parameters

Equations (2.10) and (2.27) represent the equations of a discrete time linear filter, for which we can compute the optimal estimator. Throughout we shall assume that the Gaussian initial condition ξ_0, the signal noise sequence $\{w_n\}_n$ and the observation noise sequence $\{v_n\}_n$ are all mutually independent.

Let $\hat{\xi}_n$ be an estimator of ξ_n, and denote by $\epsilon_n = \xi_n - \hat{\xi}_n$ the estimation error. The estimator that minimizes the mean square of the estimation error is given by the following Kalman filter (see e.g. [112, page 347]), which has a simple recursive structure:

$$P_n = \Big(\left(\gamma^2 P_{n-1} + Q\right)^{-1} + p^2/R \Big)^{-1} \tag{2.30}$$

$$K_n = P_n p/R \tag{2.31}$$

$$\hat{\xi}_n = \gamma \hat{\xi}_{n-1} + K_n \Big(m_n - p \left(\gamma \hat{\xi}_{n-1} \right) \Big) \tag{2.32}$$

for $n = 1, 2, \ldots$, with $\hat{\xi}_0 = \mathbf{E}[\xi_0] = 0$ and where the constants γ, R and Q have been defined earlier in the section. Equation (2.30) is called the Riccati equation and P_n gives the variance of the estimation error ϵ_n. In (2.31) K_n is called the filter gain. Equation (2.32) is the state estimate equation, and it is the sum of an extrapolation term and an update term.

The above filter minimizes the sum of mean square estimation errors until time nS. One can also (and will from now on) use the stationary version of the Kalman filter, which minimizes the time average mean square estimation error, namely,

$$P = \Big(\left(\gamma^2\, P + Q\right)^{-1} + p^2/R \Big)^{-1}$$

$$K = Pp/R \tag{2.33}$$

$$\hat{\xi}_n = \gamma \hat{\xi}_{n-1} + K \left(m_n - p \left(\gamma \hat{\xi}_{n-1} \right) \right). \tag{2.34}$$

P now gives the steady-state variance of the estimation error. It is obtained as the unique positive solution of the algebraic Riccati equation

$$p^2\gamma^2 P^2 + (Qp^2 + R\left(1-\gamma^2\right))P - RQ = 0.$$

We find

$$P = -\frac{Qp^2 + R\,(1-\gamma^2) - \sqrt{(Qp^2 + R\,(1-\gamma^2))^2 + 4p^2\gamma^2 RQ}}{2p^2\gamma^2}. \tag{2.35}$$

We may replace $Q = \rho(1-\gamma^2)$ and $R = \rho p(1-p)$ in (2.35). Substituting the resulting expression into (2.33), we can express the filter gain in terms of p and $\gamma = \exp(-\mu S)$

$$K = \frac{-(1-\gamma^2) + \sqrt{(1-\gamma^2)(1-\gamma^2(1-2p)^2)}}{2\gamma^2 p(1-p)}. \tag{2.36}$$

For every n, the error ϵ_n is a normal RV with mean zero and variance P, further independent of the observation m_n [123, page 240].

Membership size estimation

We now return to our original estimation problem, namely, the derivation of an estimate – called $\hat{N}_n$ – for the number of participants $N_T(nS)$ at time nS.

Recall that the process $\{N_T(t),\, t \geq 0\}$ describes the number of busy servers in an $M/M/\infty$ queue with arrival rate λT and service rate μ. If $N_T(0) = 0$, namely the system is initially empty, then we know by Takács [114, Theorem 1, pages 160-161] that $\mathbf{E}[N_T(t)] = \rho T(1 - e^{-\mu t})$ for any time t. In particular, $\mathbf{E}[N_T(t)] = \rho T$ in steady-state (i.e. as $t \to \infty$).

Motivated by (2.8), we define $\hat{N}_n$ as follows:

$$\hat{N}_n = \hat{\xi}_n\sqrt{T} + \rho T \tag{2.37}$$

with $\hat{\xi}_n$ given in (2.34). We go back to the latter equation. Replacing m_n with $M_T(nS)$ as provided in (2.16) and using (2.37), we derive the following estimate for $\hat{N}_n$

$$\hat{N}_n = \gamma(1-Kp)\hat{N}_{n-1} + K\,Y_n + \rho T(1-\gamma)(1-Kp). \tag{2.38}$$

Here Y_n denotes the amount of ACKs collected at time nS.

Starting from $\mathbf{E}[\hat{\xi}_0] = 0$ it is seen from (2.34) and (2.27) that $\mathbf{E}[\hat{\xi}_n] = 0$ for $n = 0, 1, \ldots$, which in turn implies from (2.37) that $\mathbf{E}[\hat{N}_n] = \rho T$. The estimator $\hat{N}_n$ is asymptotically unbiased in the sense that

$$\left|\mathbf{E}\left[\hat{N}_n - N_T(nS)\right]\right| = e^{-\mu n S} \underset{n}{\to} 0.$$

Regarding the variance of the error $e_n := N_T(nS) - \hat{N}_n$, it is a function of the parameter T. We have (use (2.8)) $e_n = \sqrt{T}\left(Z_T(nS) - \hat{\xi}_n\right)$, so that

$$\mathrm{Var}\left(\frac{e_n}{\sqrt{T}}\right) = \mathrm{Var}\left(Z_T(nS) - \hat{\xi}_n\right).$$

Although we have not been able to prove this result, we believe that $\mathrm{Var}\left(e_n/\sqrt{T}\right) \to P$ as $T \to \infty$.

We conclude this section by summarizing the estimation algorithm (ρ and T are known/estimated beforehand)

Initialization: Set p and S to the desired values, and $\hat{N}_0$ to ρT ($\hat{\xi}_0 = 0$). Compute $\gamma = \exp(-\mu S)$ and the filter gain K according to (2.36).

nth observation step: Collect the ACKs received in the interval of time $((n-1)S, nS]$ and compute $\hat{N}_n$ as in (2.38).

Remark 2.4.2 *The autoregressive equation in (2.38) does not exhibit the same form as the one in (2.6) as it further has a constant term $\rho T(1-\gamma)(1-Kp)$. In other words, if we had computed the optimal α in (2.6), we would not have obtained the optimal estimator, at least under the assumptions considered in Section 2.4.1.*

2.4.3 Simulations

Our estimator has been derived under a set of various statistical assumptions (Poisson arrivals, exponential on-times, heavy traffic regime) that may be violated in practice. In this section, we investigate the robustness of our estimator and try to identify situations where it works well/poorly. To do so, we have conducted various simulations (using a C program) where some or all of the assumptions needed for the derivation of the estimator are violated.

Four types of simulations have been performed. For each simulation, the parameters λ and T are taken to be equal to 1 and $1/185.9s^{-1}$, respectively, and the run time is $124240s$. These values have been measured on a real trace (see Section 2.4.4 for details on the traces we have used). The ACK probability p and the ACK interval S have been set to 0.01 and $1.0s$, respectively. (see Section 2.7 for details on how these parameters can be set).

Two figures are associated with each simulation depending on the load (defined as ρT) of the system: $\rho T = 34.1$ referred to as "light load" (Figure 2.3) and $\rho T = 200$ referred to as "heavy-load" (Figure 2.4). Each graph displays three curves: the simulated data, the estimated data and the mean load ρT. For each simulation, the performance of the estimator

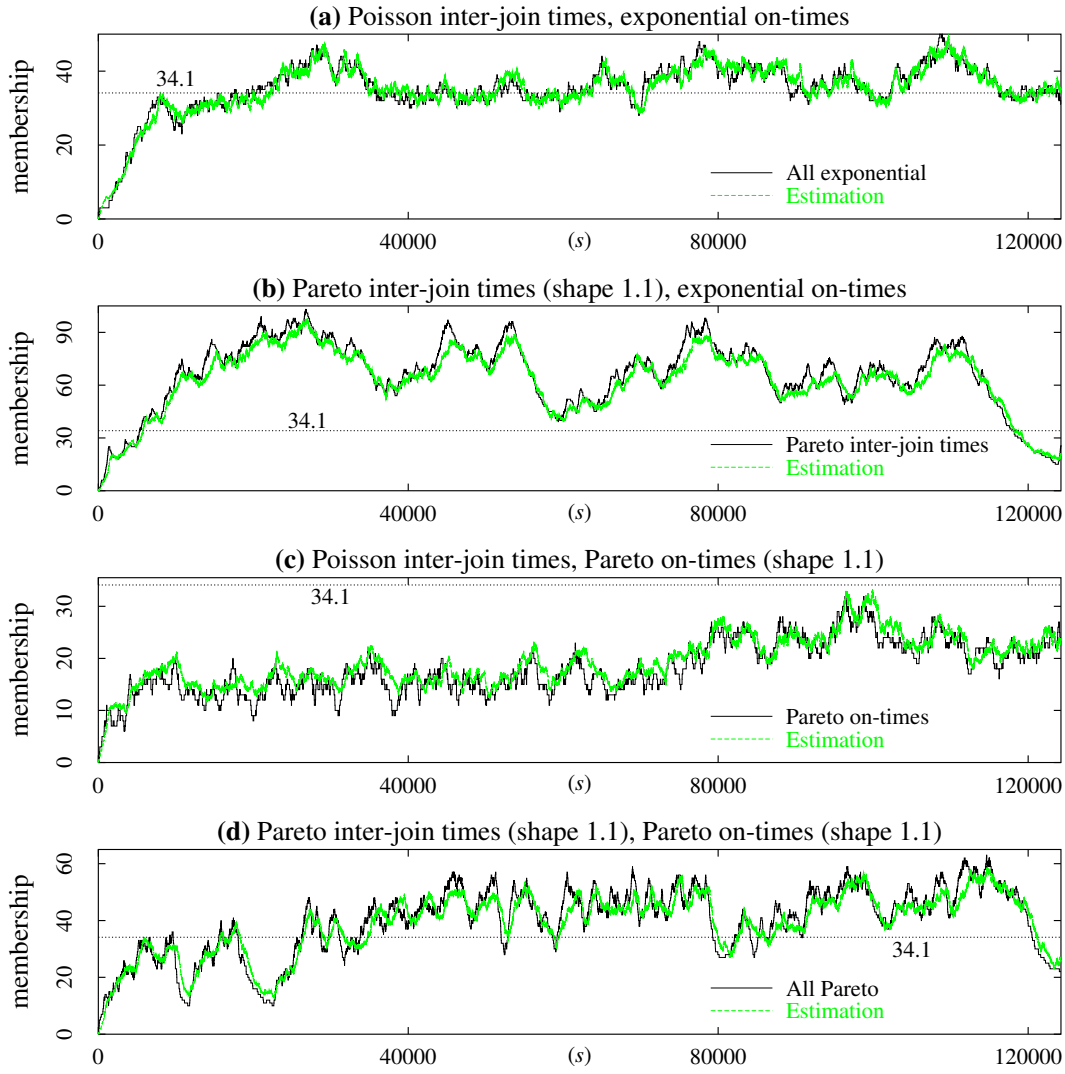


Figure 2.3: Estimation of the multicast membership over time ($p = 0.01, S = 1s$): the light load case $\lambda T = 1/185.9s^{-1}, \mu = 1/6342s^{-1}$

is collected in Table 2.1. The second column gives the sample mean of the relative error expressed in percentage (i.e. $100 \times \frac{|N_T(nS) - \hat{N}_n|}{N_T(nS)}$), column 3 gives the 25th percentile, column 4 reports the median value, etc.

In the first simulation the users join the multicast group according to a Poisson process and their on-times are exponentially distributed. The Poisson assumption for the joining process is fairly realistic, as mentioned in [7]. The validity of the exponential assumption for the on-times has been observed for short sessions. The obtained results are reported in Figures 2.3(a) and 2.4(a). Both for light and heavy loads the estimated value appears to be very close to the true value; in particular, 95% of the time the relative error is less than 13.1% (resp. less than 4.7) when $\rho T = 34.1$ (resp. $\rho T = 200$) (see rows 3–4 in Table 2.1 for

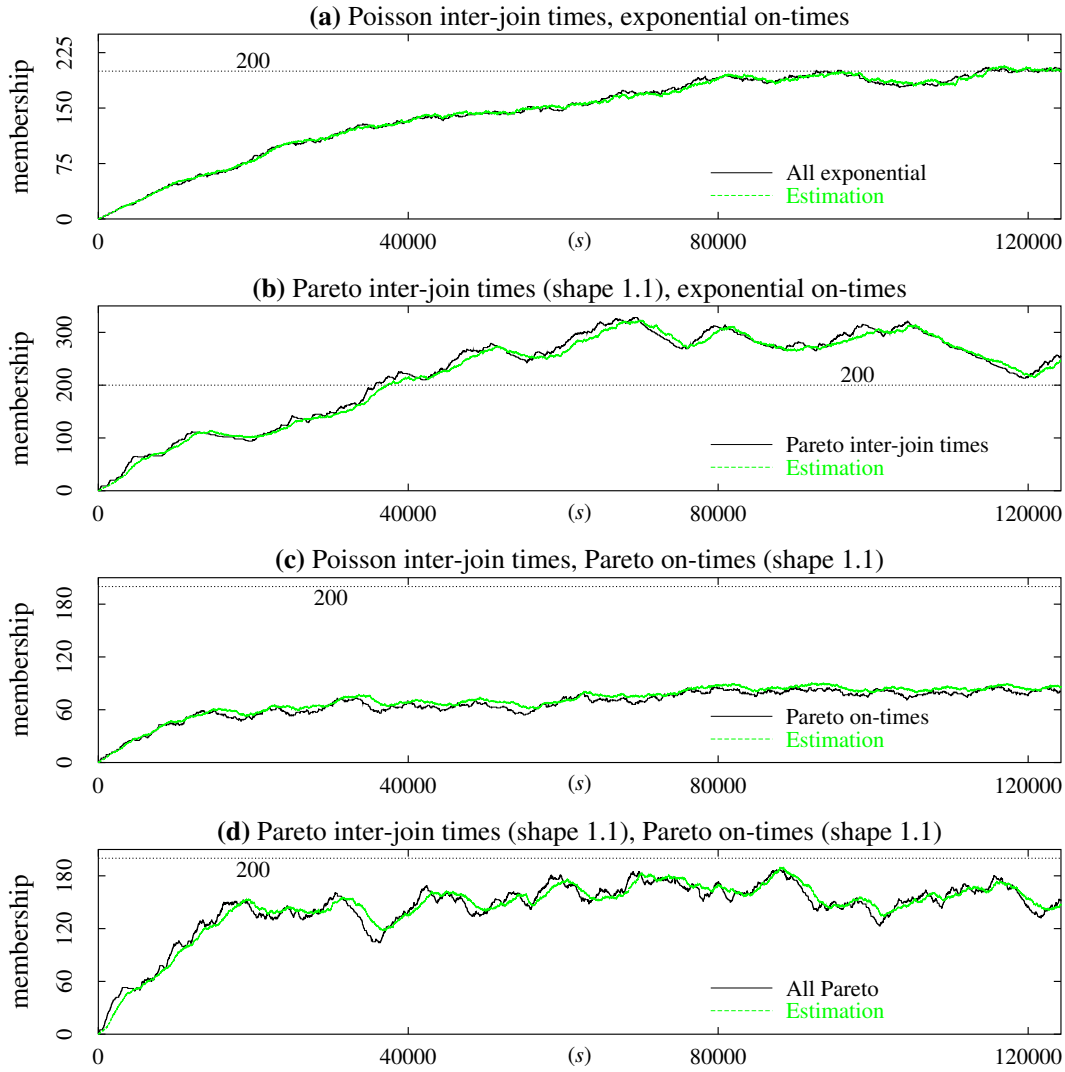


Figure 2.4: Estimation of the multicast membership over time ($p = 0.01, S = 1s$): the heavy load case $\lambda T = 1/185.9s^{-1}, \mu = 1/37180s^{-1}$

details).

In the second simulation the on-times are still exponentially distributed but now the inter-arrivals are Pareto distributed with shape parameter equal to 1.1, leading to an infinite variance of the inter-arrival times. The results are displayed in Figures 2.3(b) and 2.4(b). We first observe that both the estimator and the real values are far away from the "limiting trajectory" when the load is light. This is due to the infinite variance of the inter-arrival times which prevents the stationary regime to be reached rapidly. Nevertheless, the accuracy of the estimator is still remarkable both at light and heavy loads (see rows 6–7 in Table 2.1).

In the third simulation the arrival process is Poisson and the on-times are Pareto distributed with shape parameter equal to 1.1. Both the estimator and the real values are far away from the limiting trajectory for both loads. When the load is light, the accuracy

Table 2.1: Sample mean and percentiles of the relative error expressed in percentage

Simulation	Mean	25	50	75	90	95
All exponential						
Light load $\rho T = 34.1$	5.6	1.9	4.1	7.3	10.6	13.1
Heavy-load $\rho T = 200$	2.3	0.8	1.7	2.7	3.8	4.7
Pareto inter-arrivals						
Light load $\rho T = 34.1$	6.6	2.6	5.3	9.2	12.9	15.7
Heavy-load $\rho T = 200$	4.4	1.5	3.3	5.3	7.2	9.8
Pareto on-times						
Light load $\rho T = 34.1$	12.2	4.0	9.2	16.5	26.7	35.7
Heavy-load $\rho T = 200$	6.8	3.2	6.1	8.9	12.4	15.1
All Pareto						
Light load $\rho T = 34.1$	9.7	3.9	7.8	12.5	18.5	25.6
Heavy-load $\rho T = 200$	5.2	1.7	3.5	6.1	9.9	14.5

Table 2.2: Mean and variance of the error $e_n = N_T(nS) - \hat{N}_n$

Simulation	Mean	Variance
All exponential		
$\rho T = 34.1, TP = 5.514$	0.07	4.7
$\rho T = 200, TP = 14.067$	-0.006	9.95
Pareto inter-arrivals		
$\rho T = 34.1, TP = 5.514$	2.8	19.7
$\rho T = 200, TP = 14.067$	3.6	67.3
Pareto on-times		
$\rho T = 34.1, TP = 5.514$	-1.2	4.1
$\rho T = 200, TP = 14.067$	-3.8	10.1
All Pareto		
$\rho T = 34.1, TP = 5.514$	0.6	16.9
$\rho T = 200, TP = 14.067$	-0.4	57.9

of the estimator is not as good as in the previous simulations but it is still fair. This lack of accuracy is more a consequence of the light measured load (18.1) than of the Pareto on-time assumption. The accuracy of the estimator dramatically increases as the load increases. See Figures 2.3(c) and 2.4(c) and rows 9–10 in Table 2.1 for details. In order to enhance the performance of the estimator for small multicast groups, one should increase the ACK probability p. Notice that the average amount of ACKs generated when the light is load is equal to $18.1 \times 0.01 = 0.18$ whereas it equals $67.05 \times 0.01 = 0.67$ in the heavy-load simulation. Refer to Section 2.7 for details on how to set p and S in order to have a predefined quality of estimation, while avoiding feedback implosion.

In the fourth and last simulation, all assumptions are simultaneously violated: both the inter-arrival times and the on-times are Pareto distributed with shape parameter equal to 1.1. The overall performance of the estimator is better than in the third simulation.

See Figures 2.3(d) and 2.4(d) and rows 12–13 in Table 2.1 for details. Table 2.2 contains the mean (column 2) and variance (column 3) of the error e_n. The expected mean is 0 and the (conjectured) expected variance is TP (see previous Section). "Mean" denotes the measured average and "Var" denotes the measured variance. Looking at Table 2.2, we can easily observe that

- There is a small bias when the distributions of the inter-arrival times and of the on-times are different, that is, when one of them is Pareto and the other one is exponential (see rows 6–7 and 9–10 in Table 2.2). The negative bias -3.8 in the heavy-load case can be observed in Figure 2.4(c) where the estimation is clearly above the simulated group size;
- The variances measured when the arrival process is Poisson are very close to each other for both values of the workload (see rows 3–4 and 9–10 in Table 2.2) and they are not too far from the expected variances;
- The variances measured when the inter-arrival times are Pareto distributed are both far from the expected variances but are relatively close to each other (see rows 6–7 and 12–13 in Table 2.2).

2.4.4 Validation with real traces

An extensive study of the characterization of MBone sessions dynamics is due to Almeroth and Ammar [7, 6]. They have developed a tool called *Mlisten* [8] that collects the join/leave times for multicast group members in MBone sessions. We have applied our algorithm to some of these traces collected in 1996[2]. As stated in [6] (see Remark 2.4.3 page 88), the joining process is reasonably close to a Poisson process. As to the on-times, two cases have to be distinguished depending on the duration of a session. For long sessions some people will join for very long periods while others will join only for a few minutes. In this case, the Zipf distribution fits well in the collected data. In the case where sessions are short, the maximum membership duration is much shorter than for long sessions, thereby eliminating long on-times.

We have run our algorithm on two different traces, one collected from a short audio session that started on 9th of December 96 and lasted for 1 day 10 hours 30 minutes and 40 seconds, i.e. 124240 seconds; the other one results from a long audio session that lasted from 18th of November 96 to 10th of December 96 (21 days 12 hours 37 minutes and 27 seconds, that is 1859847 seconds). Figures 2.5 and 2.6 plot the actual group size and its estimation for each session.

The values of the arrival rate T (throughout $\lambda = 1$, so that the arrival rate is T) and the expected on-times $1/\mu$ were extracted from the traces. For the short duration

[2]The traces are available at ftp://ftp.cc.gatech.edu/people/kevin/release-data/. We were able to collect more recent traces during Summer 2001. Some of these traces are discussed in Section 2.8.

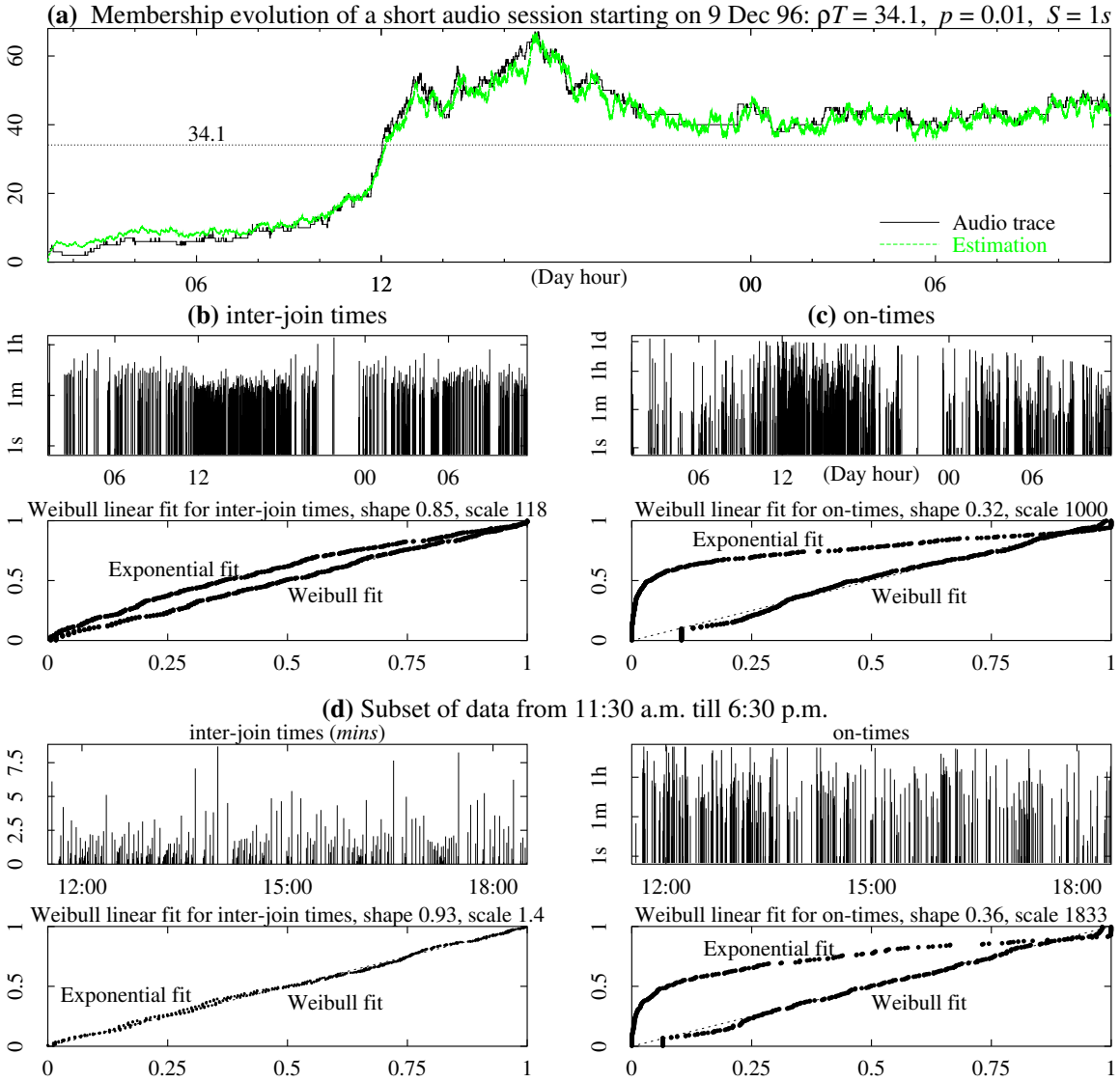


Figure 2.5: Estimation of the multicast group size using the trace of a short audio session and probability plots for the observed data

session, the measured "load" is $\rho T = 34.1$; for the long duration session, the measured load is $\rho T = 63.5$.

We have observed (see Figure 2.5(a)) that our estimator does not work well in case the session gathers a few participants[3]. In this case, it overestimates the size of the group; the absolute error $|N_T(nS) - \hat{N}_n|$ is not significant but the relative error is very high and approaches 100% (see around 2 a.m. in Figure 2.5(a)). This behavior was already reported in Section 2.4.3. The sample mean and some percentiles of the relative error (expressed in percentage) for both sessions are listed in Table 2.3.

The data set corresponding to the short audio session exhibits two very different parts:

[3]Here again, increasing the ACK probability p will definitely enhance the performance of the estimator.

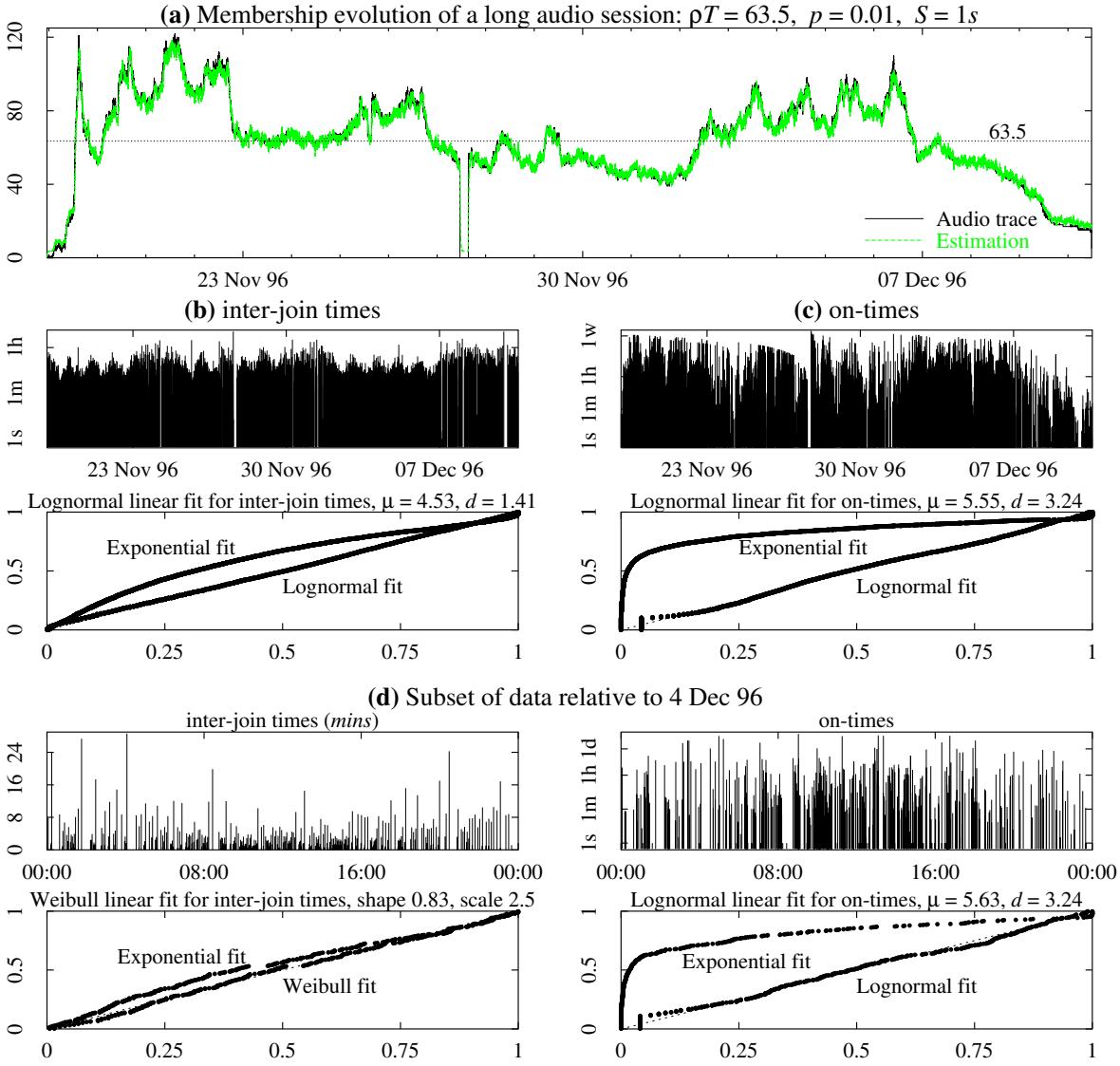


Figure 2.6: Estimation of the multicast group size using the trace of a long audio session and probability plots for the observed data

very few users are connected during the first quarter of the data set, thereby suggesting that this part was recorded before the start of the transmission. The remaining part of the data set records participants activity during the transmission.

Our algorithm has first been run on the entire trace. As expected, it performs poorly on the first quarter of the trace; its performance improves drastically on the remaining part of the trace.

The first quarter of the data set has then been removed and our algorithm has been run on the resulting trace (we found $T = 1/157.7s^{-1}$, $\mu = 1/5994.6s^{-1}$ and $\rho T = 38$). Most of the time, the relative error is less than 15.9% (see row 4 in Table 2.3), which is a satisfactory result.

Table 2.4 reports the mean (column 2) and variance (column 3) of the error e_n. As

Table 2.3: Sample mean and percentiles of the relative error expressed in percentage

Real audio trace	Mean	25	50	75	90	95
Short trace						
Entire trace $\rho T = 34.1$	15.0	2.3	5.2	11.8	44.9	61.2
Last 3/4 of trace $\rho T = 38.0$	6.2	1.8	4.2	7.3	11.4	15.9
Long trace						
$\rho T = 63.5$	4.7	1.3	2.7	4.7	7.7	13.1

Table 2.4: Mean and variance of the error $e_n = N_T(nS) - \hat{N}_n$

Real audio trace	Mean	Variance	TP
Short trace			
Entire trace $\rho T = 34.1$	0.219	6.070	5.514
Last 3/4 of trace $\rho T = 38.0$	0.338	6.339	6.303
Long trace			
$\rho T = 63.5$	0.017	8.083	6.723

before, "Mean" is the measured average error, "Var" is the measured variance and "TP" is the (conjectured) expected variance. We can see from this table that, as expected, the measured average error is close to 0. Also notice that the measured variance and the expected variance are close to each other (see columns 3–4 in Table 2.4).

Distribution fits

The distribution fits for the inter-arrival time process and the on-time process for both the short and long duration sessions are presented in Figures 2.5(b) and (c) and 2.6(b) and (c) and summarized in Table 2.5 (see rows 2–3). We have found that for both types of sessions, the joining process is not Poisson and on-times are not exponentially distributed.

Table 2.5: Distributions that best fitted into the inter-arrivals and on-times sequences

Entire trace	Best fit for inter-join times	Best fit for on-times sequence
Short session	Weibull with shape 0.85, scale 118	Weibull with shape 0.32, scale 1000
Long session	Lognormal with $\mu = 4.53$, $d = 1.41$	Lognormal with $\mu = 5.55$, $d = 3.24$
Subset of trace	Best fit for inter-join times	Best fit for on-times sequence
Short session	Weibull with shape 0.93, scale 1.4	Weibull with shape 0.36, scale 1833
Long session	Weibull with shape 0.83, scale 2.5	Lognormal with $\mu = 5.63$, $d = 3.24$

For the entire short audio session, the inter-arrival time distribution is well represented by a Weibull distribution with shape parameter 0.85 and the on-time distribution fits well a Weibull distribution with shape parameter 0.32 (see Figures 2.5(b) and (c) and row 2 in Table 2.5). Recall that the smaller the shape parameter the heavier the tail (a Weibull

distribution with shape parameter 1 is an exponential distribution).

For the entire long audio session, the inter-arrival time distribution is well represented by a Lognormal distribution with parameters $\mu = 4.53$ and $d = 1.41$ and the on-time distribution fits well a Lognormal distribution with parameters $\mu = 5.55$ and $d = 3.24$ (see Figures 2.6(b) and (c) and row 3 in Table 2.5). Recall that the Lognormal distribution is long-tailed; the higher the parameter d, the longer the tail.

Remark 2.4.3 *Our results are different from the results reported in [6] which is explained by the fact that they have analyzed only the parts of the traces with the highest human activity. We took two subsets of data, one from each trace, and fitted the inter-join times and the on-times sequences into well-known distributions. Our results, plotted in Figures 2.5(d) and 2.6(d) and reported in rows 5–6 in Table 2.5, reveal that the distribution of the inter-arrivals is approximately exponential (even though the best fit is a Weibull distribution) which agrees with the observations of Almeroth and Ammar in [6]. However, the on-times distribution in the subset coming from the short audio session cannot be approximated by an exponential distribution. In fact, we always find that the on-times are subexponentially distributed (either Weibull or Lognormal distribution). In Section 2.8, four video traces are investigated and the distributions that best fitted into the sequences at hand came out to be subexponential also (see Table 2.10 page 107 for details).*

To conclude this section, we would like to point out that our estimator seems to be very robust to changes in the distribution laws. Although the assumptions that the inter-arrival times and the on-times have an exponential distribution is crucial in the theory that we have developed, it is interesting to note that our estimator still performs well for other distributions, including various subexponential distributions (Pareto, Lognormal, Weibull).

2.5 Optimal estimation using a Wiener filter

In Section 2.4, we have proposed a dynamic scheme that tracks the variations of the membership in an optimal way. To derive the optimal estimator we have used a diffusion approximation for the heavy traffic regime. Under the assumptions of Poisson join times and exponentially distributed connection times, the diffusion approximation yields linear dynamics which enabled us to design the optimal filter using Kalman filter theory.

Our purpose now is to develop an estimator under more general assumptions than the ones used in Section 2.4. In this section, we use Wiener filter theory to derive the optimal estimator. The dynamics are not required to be linear as in the case of the Kalman filter. This will allow us to remove the heavy traffic assumption made in Section 2.4. We will first present the model and next derive the optimal filter.

2.5.1 The model

Recall the notation introduced in Section 2.4.1: T_i denotes time that receiver i joins the multicast group; D_i denotes the lifetime (or on-time) of the ith participant, in other words, receiver i leaves the group at time $T_i + D_i$; and $N(t)$ denotes the number of receivers in the multicast group at time t or equivalently the membership at time t. It is seen then that

$$N(t) = \sum_{i \geq 1} \mathbf{1}\{T_i \leq t < T_i + D_i\},$$

where $\mathbf{1}\{E\}$ equals 1 if the event E occurs and 0 otherwise.

We shall assume that the join times form a homogeneous Poisson process (with constant intensity $0 < \lambda = 1/\mathbf{E}[T_{i+1} - T_i]$) and that the on-times form a renewal sequence of random variables (RVs) with common probability distribution $\Psi(x) = P(D_i < x)$, $0 < \mathbf{E}[D_i] < \infty$, further independent of the join times. In the following D will denote a generic RV with probability distribution $\Psi(x)$.

In queueing terminology, $\{N(t), t \geq 0\}$ represents the occupation process (number of busy servers) in an $M/G/\infty$ queue [75].

We still rely on the same polling scheme used before: receivers, with probability p, send ACKs to the source every S seconds. Note that in practice the source will have to regularly multicast the pair (p, S) to ensure that each participant will know these values. We will assume that S is large enough (the order of magnitude is in seconds) so that at times $t = nS$, $n = 0, 1, \ldots$, the source holds all of the ACKs produced in round n. Throughout, p and S are held fixed (see Section 2.7 for possible extensions).

Let Y_n be the number of ACKs received by the source at time nS. Based on the knowledge of $Y_1, \ldots, Y_n$, our objective is to find an optimal estimator (in a sense to be defined below) $\hat{N}_n$ for $N_n := N(nS)$, the size of the multicast group at time nS. In filtering parlance, Y_n is an input signal and we want to generate another signal $\hat{N}_n$ that is as close as possible to an unknown signal N_n (e.g. by minimizing the mean square error).

For later use we briefly review some results on the $M/G/\infty$ queue. In steady-state, the number N of busy servers is a Poisson random variable with parameter $\rho := \lambda\, E[D]$, namely, $P[N = j] = \rho^j \exp(-\rho)/j!$. In particular, both the mean and the variance of the number of busy servers are equal to ρ. The autocovariance function of the stationary version of the process $\{N(t), t \geq 0\}$, also denoted by $\{N(t), t \geq 0\}$, is given by [41, Equation (5.39)]

$$\mathrm{Cov}(N(t), N(t+h)) = \lambda \int_{|h|}^{\infty} P(D > u)\, \mathrm{d}u. \tag{2.39}$$

In the following we will denote by $\mathrm{Cov}_X(\cdot)$ the autocovariance function of any second-order discrete-time stationary process $\{X_n, n = 0, 1, \ldots\}$. With this notation and the defi-

nition of the process $\{N_n\}_n$, we see from (2.39) that

$$\mathrm{Cov}_N(k) = \rho\,\gamma^{|k|}, \quad k = 0, \pm 1, \ldots, \tag{2.40}$$

with $\gamma := \exp(-\mu S)$, when the on-times $\{D_i\}_i$ are exponentially distributed with mean $1/\mu$.

Throughout, we will assume that

$$\sum_{k\geq 0} \mathrm{Cov}_N(k) < \infty. \tag{2.41}$$

In other words, we will exclude the situation where the on-times are heavy-tailed (e.g. Pareto distribution).

2.5.2 Wiener filter

Our objective is to transform a signal Y_n (noisy observation) into another signal $\hat{N}_n$ (estimator) that is the closest to an unknown signal N_n. By closest we mean that the mean error is zero (i.e. $\mathbf{E}[\hat{N}_n] = \mathbf{E}[N_n]$) and that the mean square error is minimized.

Such a transformation can be achieved by the Wiener filter that identifies the optimal linear filter [61]. Applying Wiener theory yields the transfer function of the linear filter, which can be transformed back to the time domain to obtain the impulse response of the filter. From the impulse response of the filter, the expression of $\hat{N}_n$ as a function of Y_n and, possibly, of $\hat{N}_{n-1}, \hat{N}_{n-2}, \ldots$, can be found. We will detail this procedure below.

Since a filter that minimizes the mean square error when the underlying processes are centered (i.e., zero-mean processes) also minimizes the mean square error when the same processes are non-centered, we will derive the Wiener filter for the centered (stationary) versions of processes $\{N_n\}_n$, $\{\hat{N}_n\}_n$ and $\{Y_n\}_n$, denoted by $\{\nu_n\}_n$, $\{\hat{\nu}_n\}_n$ and $\{y_n\}_n$, respectively. We have observed in the previous section that $\mathbf{E}[N_n] = \rho$. On the other hand

$$\mathbf{E}[Y_n] = \mathbf{E}[\mathbf{E}[Y_n \mid N_n]] = \mathbf{E}[p\,N_n] = p\rho. \tag{2.42}$$

Therefore $\nu_n = N_n - \rho$, $\hat{\nu}_n = \hat{N}_n - \rho$ and $y_n = Y_n - p\rho$. We aim at determining the impulse response $\{h_{o,n}\}_n$ of the optimal filter in the mean-square sense. For a discrete-time causal linear filter with impulse response $\{h_n\}_n$, we have

$$\hat{\nu}_n = \sum_{k=0}^{\infty} h_k y_{n-k}.$$

The mean square value of the error signal $N_n - \hat{N}_n = \nu_n - \hat{\nu}_n$ is denoted by $\epsilon := \mathbf{E}[(\nu_n - \hat{\nu}_n)^2]$. It is shown in [61] that ϵ is minimal when the impulse response of the filter satisfies the

Wiener-Hopf equation:

$$\sum_{m=0}^{\infty} h_{o,m}\mathrm{Cov}_y(k-m) = \mathrm{Cov}_{\nu y}(k), \qquad k = 0, 1, \ldots, \tag{2.43}$$

where $\mathrm{Cov}_y(k)$ denotes the autocorrelation of the filter input (the measurements) $\{y_n\}_n$ and $\mathrm{Cov}_{\nu y}(k) = \mathbf{E}[\nu_{n-k}\, y_n]$ denotes the cross-correlation function of processes $\{\nu_n\}_n$ and $\{y_n\}_n$. We can express $\mathrm{Cov}_y(k)$ and $\mathrm{Cov}_{\nu y}(k)$ in terms of $\mathrm{Cov}_\nu(k)$ as follows

$$\mathrm{Cov}_y(k) = p^2\mathrm{Cov}_\nu(k) + \mathbf{1}\{k=0\}\rho p(1-p) \tag{2.44}$$

$$\mathrm{Cov}_{\nu y}(k) = p\,\mathrm{Cov}_\nu(k) \tag{2.45}$$

where we have used the identity $\mathrm{Cov}_\nu(k) = \mathrm{Cov}_N(k)$.

Remark 2.5.1 *Autocorrelation and autocovariance are the same for zero-mean processes; the same remark applies to cross-correlation and covariance.*

One way of solving the Wiener-Hopf equation for the optimal impulse response $\{h_{o,n}\}_n$ is instantiated in the *prewhitening approach* [61, page 81] which makes use of the following basic idea. If the signal $\{y_n\}_n$ were white noise, then $\mathrm{Cov}_y(k-m) = \mathbf{1}\{m=k\}\sigma^2$ where σ^2 is the variance of any noise sample, and the optimal impulse response, denoted $h'_{o,k}$ in this case, will satisfy $h'_{o,k} = \mathbf{1}\{k \geq 0\}\mathrm{Cov}_{\nu y}(k)/\sigma^2$. The structure of the optimal filter consists then of two filters connected in cascade. The first filter, called *whitening* filter, transforms the signal $\{y_n\}_n$ into a white noise $\{\omega_n\}_n$ of variance σ^2, and its transfer function is denoted by $1/G(z)$. The second filter, whose impulse response is $\{h'_{o,n}\}_n$, operates on $\{\omega_n\}_n$ to output $\{\hat{\nu}_n\}_n$, the estimator of $\{\nu_n\}_n$. Its transfer function is denoted by $H'_o(z) := \sum_{k=0}^{\infty} h'_{o,k} z^{-k}$ (throughout, z is a complex number such that $|z| = 1$).

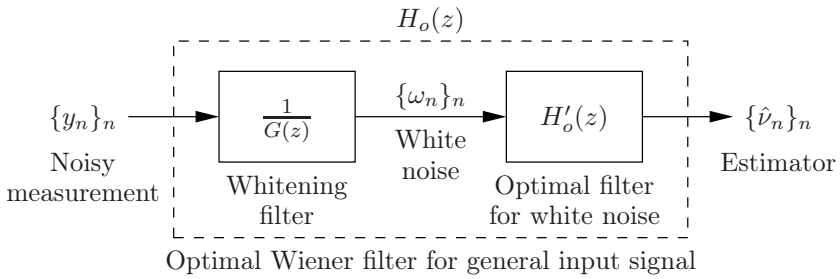


Figure 2.7: The prewhitening approach

The transfer function of the optimal filter which we are looking for is then (see Figure 2.7)

$$H_o(z) := \sum_{k=0}^{\infty} h_{o,k} z^{-k} = \frac{1}{G(z)} \times H'_o(z).$$

For later use, introduce

$$S_y(z) = \sum_{k=-\infty}^{\infty} \mathrm{Cov}_y(k) z^{-k},$$

$$S_{\nu y}(z) = \sum_{k=-\infty}^{\infty} \mathrm{Cov}_{\nu y}(k) z^{-k},$$

the z-transforms of the autocovariance function of $\{y_n\}_n$ and of the cross-correlation sequence $\{\mathrm{Cov}_{\nu y}(k)\}_k$, respectively. For $|z| = 1$, $S_y(z)$ is called the power spectrum of $\{y_n\}_n$ and $S_{\nu y}(z)$ is called the cross-power spectrum between $\{\nu_n\}_n$ and $\{y_n\}_n$. Observe from (2.44) and (2.45) that both $S_y(z)$ and $S_{\nu y}(z)$ are well-defined for $|z| = 1$ under the assumption (2.41).

We are now in position to derive the Wiener filter. First, we write $S_y(z)$ as

$$S_y(z) = \sigma^2 G(z) G(z^{-1}) \tag{2.46}$$

where σ^2 is a constant, the function $G(z)$ is the part of $S_y(z)$ having all of its zeros and poles *inside* the unit circle, therefore $G(z^{-1})$ is the part of $S_y(z)$ having all of its zeros and poles *outside* the unit circle. This operation is called the canonical factorization of the power spectrum of $\{y_n\}_n$, and allows the derivation of the transfer function of the whitening filter, which is nothing but $1/G(z)$ (σ^2 is then the variance of the white noise at the output of the whitening filter).

As for the second filter, which transforms $\{\omega_n\}_n$ into $\{\hat{\nu}_n\}_n$, its transfer function can be rewritten

$$H'_o(z) = \frac{1}{\sigma^2} \sum_{k=0}^{\infty} \mathrm{Cov}_{\nu\omega}(k) z^{-k}.$$

It is shown in [61], that $\sum_{k=-\infty}^{\infty} \mathrm{Cov}_{\nu\omega}(k) z^{-k} = S_{\nu y}(z)/G(z^{-1})$. This ratio is interpreted as the transfer function of a linear filter whose impulse response, $\{\mathrm{Cov}_{\nu\omega}(k)\}_k$, has values at the left and the right of the time origin (non-causal filter). However, what we are actually looking for, is $\sum_{k=0}^{\infty} \mathrm{Cov}_{\nu\omega}(k) z^{-k}$, which is the transfer function of the part of the impulse response at the right of the time origin (causal filter). This can simply be done by expanding the ratio $S_{\nu y}(z)/G(z^{-1})$ into fractions and then taking only the fractions with zeros and poles inside the unit circle. In other words, we transfer the filter from a non-causal filter into a causal one. The transfer function of the causal version of the filter is then

$$H'_o(z) = \frac{1}{\sigma^2} \left[\frac{S_{\nu y}(z)}{G(z^{-1})} \right]_+ .$$

Having expressed the transfer functions $1/G(z)$ and $H'_o(z)$, it remains to invert the transfer function of the optimal filter, $H_o(z) = \frac{H'_o(z)}{G(z)}$, back into the time domain to find the desired recurrence between $\hat{\nu}_n$ and y_n and, subsequently, between the non-centered variables $\hat{N}_n$ and Y_n.

The success of the prewhitening approach rests on the ability to factorize the power spectrum of the original input signal $\{y_n\}_n$ as in (2.46). This procedure is illustrated in Section 2.5.3 for the case where the underlying model is the $M/M/\infty$ queue.

2.5.3 Application to the $M/M/\infty$ model

In light of the results reported in Section 2.5.2, all what we have to do is to find expressions for $S_y(z)$ and $S_{\nu y}(z)$. This can easily be done when the underlying model is the $M/M/\infty$ queueing model, as shown below.

Let us first determine $S_y(z)$. By using (2.44) and (2.40) together with the property $\mathrm{Cov}_N(k) = \mathrm{Cov}_\nu(k)$, we find

$$\mathrm{Cov}_y(k) = \begin{cases} p^2\rho\gamma^{|k|}, & \text{for } k \neq 0 \\ p\rho, & \text{for } k = 0. \end{cases}$$

Since $\gamma = \exp(-\mu S) < 1$ and $|z| = 1$, the z-transform of $\mathrm{Cov}_y(k)$ is

$$\begin{aligned} S_y(z) &= \sum_{k=-\infty}^{-1} p^2\rho\gamma^{-k}z^{-k} + p\rho + \sum_{k=1}^{\infty} p^2\rho\gamma^k z^{-k} \\ &= \frac{p\rho\,[\gamma(p-1)z^2 + [1+\gamma^2(1-2p)]z + \gamma(p-1)]}{z(1-\gamma z)(1-\gamma z^{-1})}. \end{aligned}$$

The second-order polynomial in the variable z in the numerator has two positive real roots given by r and $1/r$, with

$$r = \frac{1+\gamma^2(1-2p) - \sqrt{(1-\gamma^2)[1-\gamma^2(1-2p)^2]}}{2\gamma(1-p)}.$$

Note that $r < 1$. Hence

$$\begin{aligned} S_y(z) &= \frac{\gamma\rho p(1-p)}{r}\left[\frac{(1-rz)(1-rz^{-1})}{(1-\gamma z)(1-\gamma z^{-1})}\right] \\ &= \sigma^2\, G(z)\, G(z^{-1}) \end{aligned}$$

with

$$\sigma^2 := \frac{\gamma\rho p(1-p)}{r}, \qquad \text{and} \qquad G(z) := \frac{1-rz^{-1}}{1-\gamma z^{-1}}.$$

We now compute $S_{\nu y}(z)$. From (2.45) and (2.40) we find

$$\mathrm{Cov}_{\nu y}(k) = p\rho\gamma^{|k|}$$

so that

$$S_{\nu y}(z) = \frac{p\rho(1-\gamma^2)}{(1-\gamma z)(1-\gamma z^{-1})}.$$

The transfer function $H'_o(z)$ is given by

$$H'_o(z) = \frac{1}{\sigma^2}\left[\frac{S_{\nu y}(z)}{G(z^{-1})}\right]_+ = \frac{r(1-\gamma^2)}{\gamma(1-p)(1-\gamma r)(1-\gamma z^{-1})}$$

and the transfer function $H_o(z)$ of the optimal filter takes here the simple form

$$H_o(z) = \frac{r(1-\gamma^2)}{\gamma(1-p)(1-\gamma r)(1-rz^{-1})} = \frac{B}{1-Az^{-1}}$$

where

$$A = r, \qquad B = \frac{r(1-\gamma^2)}{\gamma(1-p)(1-\gamma r)}.$$

The impulse response of this linear filter is given by the first-order recurrence relation [61]

$$\hat{\nu}_n = A\hat{\nu}_{n-1} + By_n$$

with $\hat{\nu}_n$ the estimator of ν_n. We now return to the original processes $\{\hat{N}_n\}_n$ and $\{Y_n\}_n$, to finally obtain the optimal linear filter:

$$\hat{N}_n = A\hat{N}_{n-1} + BY_n + \rho(1-A-pB), \tag{2.47}$$

where

$$A = \frac{1+\gamma^2(1-2p) - \sqrt{\left(1-\gamma^2\right)\left(1-\gamma^2(1-2p)^2\right)}}{2\gamma(1-p)} \tag{2.48}$$

$$B = \frac{-(1-\gamma^2) + \sqrt{\left(1-\gamma^2\right)\left(1-\gamma^2(1-2p)^2\right)}}{2\gamma^2 p(1-p)}. \tag{2.49}$$

It is interesting to compare this filter with the Kalman filter derived in Section 2.4[4]. Recall that the first-order linear filter obtained in Section 2.4 is given by (2.38):

$$\hat{N}_n = \gamma(1-Kp)\hat{N}_{n-1} + KY_n + \rho T(1-\gamma)(1-Kp), \qquad \text{(Kalman)} \tag{2.50}$$

[4]Recall that a Kalman filter is the optimal filter under the condition of linear dynamics and observation, which does not hold in our case. However, the dynamics do converge to a linear diffusion as the traffic load tends to infinity, allowing us in Section 2.4 to obtain a Kalman filter which is optimal for the asymptotic heavy traffic regime.

where the filter gain K is given in (2.36). Looking at (2.49) and (2.36), we can see that they are exactly the same. We therefore have $K = B$. Developing the coefficient of $\hat{N}_{n-1}$ in (2.50), we obtain $\gamma(1 - Kp) = A$. It remains to compare the constant terms in (2.50) and (2.47). Recall that ρT in Section 2.4 denotes the actual average number of receivers which is simply denoted by ρ in the present section. Developing the constant terms in both linear filters we find $(1 - \gamma)(1 - Kp) = 1 - A - pB$. We have therefore shown that the filters returned by both the Kalman theory and the Wiener theory are identical!

This result is not so surprising, since both the Kalman filter and the Wiener filter are optimal (among the class of linear filters) in the sense that they minimize the mean square error. The key point is that the Kalman filter used in Section 2.4 was derived under a heavy traffic assumption, while the Wiener filter computed in the present section holds for any value of the model parameters λ and μ. This partly explains why the estimator behaves well under light or moderate traffic as experimentally observed in Sections 2.4.3 and 2.4.4.

Remark 2.5.2 *In our estimation problem, both the Wiener approach and the Kalman approach returned the same result. But in general, the Kalman filter theory is more powerful than the Wiener filter theory. The latter can be applied in linear time invariant systems, whereas the former can be applied in linear systems that are either time invariant or time variant. Another difference is that a Kalman filter is always a first-order filter whereas the solution to the Wiener filter might use the entire observed data, the Kalman filter is therefore computationally more efficient than the Wiener filter. Last, all of the entities in the Kalman theory are* vectors *allowing the estimation of multiple processes based on multiple observations.*

Note that if we had used the non-stationary version of the Kalman filter in Section 2.4.2 (Equations (2.30)–(2.32)), the resulting estimator would not be identical to the one derived from the Wiener theory which assumes stationarity.

We conclude this section by computing the mean square error $\epsilon_{min} := \mathbf{E}[(N_n - \hat{N}_n)^2]$ of our estimator. It is known that [61]

$$\epsilon_{min} = \sum_{k=1}^{M} \mathrm{Res}[F(z), z_k]$$

with

$$F(z) := \frac{1}{z}\Big(S_\nu(z) - H_o(z)S_{\nu y}\left(z^{-1}\right)\Big)$$

where $z_1, \ldots, z_M$ are the poles (if any) of the function $F(z)$ inside the unit circle. The notation $\mathrm{Res}[F(z), z_k]$ stands for the residue of $F(z)$ at point $z = z_k$, namely, the coefficient of $1/(z - z_k)$ in the Laurent series expansion of $F(z)$ in the vicinity of z_k. The residue is given by

$$\mathrm{Res}[F(z), z_k] = \frac{1}{(m-1)!}\left(\frac{\mathrm{d}^{(m-1)}}{\mathrm{d}z^{m-1}}\,(z - z_k)^m F(z)\right)_{z=z_k},$$

where m denotes the multiplicity of the pole z_k. Specializing $F(z)$ to the values of $S_\nu(z)$, $S_{\nu y}(z)$ and $H_o(z)$ found earlier, yields

$$F(z) = \frac{\rho(1-\gamma^2)((1-Bp)z - A)}{(1-\gamma z)(z-\gamma)(z-A)}.$$

This function has two poles inside the unit circle which are located at $z = A$ and $z = \gamma$; the residues of $F(z)$ at these poles are given by $-\rho pAB(1-\gamma^2)/[(1-\gamma A)(A-\gamma)]$ and $\rho[1 + pB\gamma/(A-\gamma)]$, respectively. Summing up these residues gives

$$\epsilon_{min} = \rho\left(1 - \frac{Bp}{1-\gamma A}\right).$$

By using the expressions of A and B, we finally obtain

$$\epsilon_{min} = \rho\frac{-(1-\gamma^2) + \sqrt{(1-\gamma^2)(1-\gamma^2(1-2p)^2)}}{2\gamma^2 p}. \tag{2.51}$$

This expression for ϵ_{min} can be used to tune the parameters p and γ or equivalently S (see Section 2.7).

2.6 Efficient estimation using an optimal first-order linear filter

The theory reported in Section 2.5.2 applies to any on-time distribution $\Psi(x)$ (with the exception of heavy-tailed distributions). However, it is not easy to identify the function $G(z)$ that appears in the canonical factorization of the spectrum $S_y(z)$ (see (2.46)) and thereby the optimal filter, except when the on-times are exponentially distributed RVs (see Section 2.5.3).

As already pointed out, we would like to develop an estimator under more general assumptions. In Section 2.5, we removed the heavy traffic assumption which was required in Section 2.4. In this section we will try to derive an estimator under the assumption that the lifetime distribution is general (we want to illustrate this approach for the particular case in which the lifetime distribution is not exponential).

2.6.1 The model

The model considered here is identical to the one presented in Section 2.5.1. Mainly, the join times form a homogeneous Poisson process with intensity λ, the on-times form a renewal sequence of RVs with common probability distribution $\Psi(x) = P(D_i < x)$, $0 <$

$\mathbf{E}[D_i] < \infty$, further independent of the join times. Therefore, $\{N(t), t \geq 0\}$ represents the occupation process in an $M/G/\infty$ queue.

Recall that $N(t)$ is a Poisson RV with parameter $\rho := \lambda\, E[D]$. Both the mean and the variance of $N(t)$ are equal to ρ and

$$\mathrm{Cov}(N(t), N(t+h)) = \lambda \int_{|h|}^{\infty} P(D > u)\, \mathrm{d}u.$$

2.6.2 Optimal first-order linear filter

In this section, we will determine the first-order linear filter that minimizes the mean square error. Observe that, unlike the Wiener filter, the proposed approach will not return the optimal filter among all of the linear filters but simply the optimal linear filter among all first-order linear filters. We will illustrate this approach at the end of this section in the case where $\Psi(x)$ is an hyperexponential distribution.

Recall the definition of the centered processes $\{\nu_n\}_n$, $\{\hat{\nu}_n\}_n$ and $\{y_n\}_n$ made at the beginning of Section 2.5.2. Also recall Equations (2.44)–(2.45)

$$\mathrm{Cov}_y(k) = p^2 \mathrm{Cov}_\nu(k) + \mathbf{1}\{k = 0\}\rho p(1-p), \tag{2.52}$$

$$\mathrm{Cov}_{\nu y}(k) = p\, \mathrm{Cov}_\nu(k). \tag{2.53}$$

The methodology is simple: we want to find constants $A \in (0,1)$ and B such that $\epsilon := \mathbf{E}[(\nu_n - \hat{\nu}_n)^2]$ is minimized when the process $\{\hat{\nu}_n\}_n$ satisfies the following first-order recurrence relation

$$\hat{\nu}_n = A\hat{\nu}_{n-1} + By_n. \tag{2.54}$$

In steady-state this implies that

$$\hat{\nu}_n = B \sum_{k=0}^{\infty} A^k y_{n-k}. \tag{2.55}$$

The mean square error ϵ is equal to

$$\epsilon = \mathbf{E}\left[\hat{\nu}_n^2\right] + \mathbf{E}\left[\nu_n^2\right] - 2\mathbf{E}\left[\hat{\nu}_n \nu_n\right].$$

We have $\mathbf{E}[\nu_n^2] = \mathbf{E}[(N_n - \rho)^2] = \rho$. From (2.55) and (2.53) we find

$$\mathbf{E}\left[\hat{\nu}_n \nu_n\right] = pB \sum_{k=0}^{\infty} A^k \mathrm{Cov}_\nu(k) = pBg(A)$$

where

$$g(z) := \sum_{k=0}^{\infty} z^k \mathrm{Cov}_\nu(k). \tag{2.56}$$

The power series $g(z)$ converges for $|z| < 1$ (Note: $k \to \mathrm{Cov}_\nu(k)$ is nonincreasing) and is therefore differentiable for $|z| < 1$. We will denote by $g'(z)$ its derivative.

It remains to express $\mathbf{E}[\hat{\nu}_n^2]$ in terms of the parameters A and B. Squaring both sides of (2.54) and then taking the expectation yields

$$\mathbf{E}\left[\hat{\nu}_n^2\right] = \left(\frac{B}{1-A^2}\right)\left(2A\mathbf{E}[\hat{\nu}_{n-1}y_n] + B\mathbf{E}\left[y_n^2\right]\right).$$

With the identities $\mathbf{E}[y_n^2] = \mathrm{Cov}_y(0) = \rho p$ (see (2.52)) and $\mathbf{E}[\hat{\nu}_{n-1}y_n] = Bp^2\left(g(A) - \rho\right)/A$ (Note: use (2.55), (2.53) and $\mathrm{Cov}_\nu(0) = \rho$), we obtain

$$\mathbf{E}\left[\hat{\nu}_n^2\right] = \left(\frac{pB^2}{1-A^2}\right)\left(2pg(A) + \rho(1-2p)\right).$$

Finally, the mean square error is

$$\epsilon = \rho - 2pBg(A) + \left(\frac{pB^2}{1-A^2}\right)\left(2pg(A) + \rho(1-2p)\right). \tag{2.57}$$

In order to minimize ϵ, $A \in (0,1)$ and B must be the solution of the following system of equations:

$$\begin{cases} \dfrac{\partial \epsilon}{\partial A} = \dfrac{2pB}{1-A^2}\left(AB\left[\dfrac{2pg(A)+\rho(1-2p)}{1-A^2}\right] + g'(A)(pB-(1-A^2))\right) = 0 \\ \dfrac{\partial \epsilon}{\partial B} = 2p\left(B\left[\dfrac{2pg(A)+\rho(1-2p)}{1-A^2}\right] - g(A)\right) = 0. \end{cases}$$

The second equation gives

$$B = \frac{g(A)(1-A^2)}{2pg(A)+\rho(1-2p)}. \tag{2.58}$$

Substituting this value of B into the first equation shows that A must satisfy

$$Ag(A)(2pg(A)+\rho(1-2p)) - g'(A)(1-A^2)(pg(A)+\rho(1-2p)) = 0. \tag{2.59}$$

If this equation has a unique solution $A \in (0,1)$, then substituting this value of A into (2.58) will give the optimal pair (A, B). Proposition 2.6.1 shows that this is indeed the case.

Proposition 2.6.1 (Existence and uniqueness of the solution)

Define

$$f(x) := (2pg(x) + \rho(1-2p))xg(x) - (pg(x) + \rho(1-2p))(1-x^2)g'(x),$$

where $g(x)$ is given in (2.56).

If $g'(x) > 0$ for $x \in [0,1)$, then $f(x)$ has a unique zero in $[0,1)$. ♦

Proof. Write $f(x)$ as $f(x) = f_+(x) - f_-(x)$ where

$$f_+(x) := [2p(g(x) - \rho) + \rho]\, xg(x)$$
$$f_-(x) := [p(g(x) - \rho) + \rho(1-p)]\, (1-x^2)g'(x).$$

The derivative of $f_-(x)$ is given by $f'_-(x) = -\alpha x^2 - \beta x + \alpha$ using

$$\alpha := p\,(g'(x))^2 + [p(g(x) - \rho) + \rho(1-p)]g''(x),$$
$$\beta := 2[p(g(x) - \rho) + \rho(1-p)]g'(x).$$

Since $g'(x) > 0$ and $g(x) > \rho$ (see (2.56)), it is seen that $\alpha > 0$ and $\beta > 0$ which implies that $f''_-(x) = -2\alpha x - \beta < 0$ for $x \in [0,1)$. We further have that $f'_-(x)$ is strictly decreasing in $[0,1)$, with $f'_-(0) = \alpha > 0$ and $f'_-(1) = -\beta < 0$. Thus, the function $f'_-(x)$ has only one zero in $[0,1)$.

Therefore, and under the assumptions of the lemma, it is seen that:

(i) $f_+(x)$ is continuous and strictly increasing in $[0,1)$ with $f_+(0) = 0$;

(ii) $f_-(x)$ is continuous in $[0,1)$ and $f_-(1) = 0$. There exists $x_0 \in (0,1)$ such that $f_-(x)$ is strictly increasing in $[0, x_0)$, strictly decreasing in $(x_0, 1)$ and $f'_-(x_0) = 0$.

We deduce from the above that $f(x)$ has a unique zero in $[0,1)$ if $g'(x) > 0$ in $[0,1)$. This condition will hold as long as $P(D > S) > 0$. In practice, one can always select S such that this condition is true. ■

The reader can check that the filter defined in (2.54) with the optimal pair (A, B) is the same as the Wiener filter found in Section 2.5.3 when the on-times are exponentially distributed.

2.6.3 Application to the $M/H_L/\infty$ model

We now illustrate the approach developed in this section by considering the situation where on-times have an L-stage hyperexponential distribution. More precisely, we assume that

$$\Psi(x) = 1 - \sum_{l=1}^{L} p_l e^{-\mu_l x} \tag{2.60}$$

with $0 < p_l < 1$, $l = 1, 2, \ldots, L$, and $\sum_{l=1}^{L} p_l = 1$. In this setting the underlying queueing model can be seen as L independent $M/M/\infty$ queues in parallel. The arrival rate to queue l is $p_l\lambda$ and the service rate is μ_l. Define $\gamma_l := \exp(-\mu_l S)$, $\rho_l := p_l\lambda/\mu_l$ so that $\rho = \sum_{l=1}^{L} \rho_l$. The autocovariance function of the process $\{\nu_n\}_n$ is equal to

$$\mathrm{Cov}_\nu(k) = \begin{cases} \rho & \text{for } k = 0 \\ \sum_{l=1}^{L} \rho_l \gamma_l^{|k|} & \text{for } k \neq 0 \end{cases}$$

so that

$$g(A) = \sum_{l=1}^{L} \frac{\rho_l}{1 - A\gamma_l}.$$

Numerical example [5]: $L = 2$, $p = 0.0106$ and $S = 2.5s$. Also

$$\begin{array}{lllllll} 1/\mu_1 &=& 3897s, & \rho_1 &=& 19.5, & \gamma_1 = 0.999359 \\ 1/\mu_2 &=& 480061s, & \rho_2 &=& 75.1, & \gamma_2 = 0.999995 \\ 1/\mu &=& 18316s, & \rho &=& 94.7, & \end{array}$$

The optimal first-order filter is

$$\hat{N}_n = 0.99879456\,\hat{N}_{n-1} + 0.10720289\,Y_n + 0.006540864.$$

For comparison, the Wiener filter found in Section 2.5.3 (for exponential on-times) for these values is

$$\hat{N}_n = 0.99828589\,\hat{N}_{n-1} + 0.14885344\,Y_n + 0.012900081.$$

2.7 Guidelines on choosing parameters p and S

A "good" pair (p, S) should (i) limit the feedback implosion while at the same time (ii) achieve a good quality of the estimator. Of course (i) and (ii) are antinomic and therefore

[5] The values of the parameters come from the trace called *video*$_1$ investigated in Section 2.8.

a trade-off must be found. This trade-off will be formalized as follows: we want to select a pair (p, S) so that the mean number of ACKs generated every S seconds (see (2.42)) and the relative error of the variance of the estimator (denoted as η) are bounded from above by given constants, namely

$$\begin{cases} \mathbf{E}[Y_n] = p\rho \leq \alpha \\ \eta = \left| \dfrac{\mathrm{Var}(N_n) - \mathrm{Var}(\hat{N}_n)}{\mathrm{Var}(N_n)} \right| \leq \beta. \end{cases} \tag{2.61}$$

Remark 2.7.1 *Instead of bounding the average number of ACKs generated every S seconds, one may alternatively choose to bound the average number of ACKs generated every I seconds. Unlike S, I will be fixed over all of the multicast sessions. The constraint on the number of ACKs is then $\boldsymbol{E}[Y_n]I/S \leq \alpha$, or equivalently, $p \leq \alpha S/\rho I$.*

When $\hat{N}_n$ is optimal then $\mathrm{Var}(N_n) - \mathrm{Var}(\hat{N}_n) = \mathbf{E}[(N_n - \hat{N}_n)^2]$ and η becomes the "normalized mean square error" [62, page 202]. Optimality was shown for the $M/M/\infty$ queue, therefore $\eta = \epsilon_{min}/\rho$ with ϵ_{min} given in (2.51).

For given constants α and β, it is easy to solve the constrained optimization problem defined in (2.61), provided that η is known. For the $M/M/\infty$ model, where ϵ_{min} is given in (2.51), we find that $p = \alpha/\rho$ and that S, or equivalently γ, is the unique positive solution of the equation $\epsilon_{min} = \rho\beta$.

The problem now is to choose constants α and β so that conditions (i) and (ii) are satisfied. Table 2.6 lists the probability of having 5 ACKs or more, every S seconds, for several values of α and of the average membership ρ. Looking at Table 2.6, we find that α in the range $[0.5, 2]$ is a reasonable choice. We have also found in our experiments that $\beta \leq 0.15$ gives satisfactory results.

Table 2.6: Probability of having 5 ACKs or more every S seconds

Session size	$\alpha = 0.1$	$\alpha = 0.5$	$\alpha = 1$	$\alpha = 2$	$\alpha = 3$	$\alpha = 4$	$\alpha = 5$
$\rho = 10$	$2.4\ 10^{-08}$	0.000064	0.0016	0.0328	0.150	0.36690	0.623
$\rho = 50$	$6.3\ 10^{-08}$	0.000146	0.0032	0.0490	0.179	0.37105	0.569
$\rho = 100$	$7.0\ 10^{-08}$	0.000159	0.0034	0.0508	0.182	0.37114	0.564
$\rho = 150$	$7.2\ 10^{-08}$	0.000163	0.0035	0.0514	0.183	0.37115	0.562

We conclude this section with general remarks on how to adapt the parameters p and S to important variations in the membership. The estimation schemes in Sections 2.4.2, 2.5.3 and 2.6.3 have been obtained under the assumption that parameters p and S are fixed. However, the filters therein constructed can still be used if p and/or S change over time, provided that these modifications do not prevent the system to be most of the time in steady-state. In that setting, a new filter will have to be recomputed after each modification.

Such a modification can be carried out each time the number of ACKs received during a given period of time significantly deviates from the current expectation (i.e. $p\rho$).

2.8 Validation with real video traces

In this section we apply the estimators developed in Sections 2.5.3 and 2.6.3 to four real video traces. Two types of estimators will be used: the estimator – denoted as $\hat{N}_n^E$ – found in (2.47) when the population is modeled as an $M/M/\infty$ queue; the estimator – denoted as $\hat{N}_n^{H_2}$ – derived in Section 2.6.3 in case the join times are Poisson and the on-times have a 2-stage hyperexponential distribution ($M/H_2/\infty$ model).

The objective is twofold: we want to investigate the quality of both estimators when compared to real life conditions, and we want to identify the best one. Notice that the estimator $\hat{N}_n^E$ – as it is identical to the one derived in Section 2.4.2 – has been already validated in Sections 2.4.3 and 2.4.4.

We have collected four MBone traces – denoted $video_i, i = 1, \ldots, 4$ – between August 2001 and September 2001 using the *MListen* tool [8]. Each trace corresponds to a long-lived video session (see duration of each session in Table 2.7, where the superscript "d" stands for "days"). We have run both algorithms (estimators) on each trace.

For each trace, and as detailed in Appendix (page 112), we have identified the parameters of the $M/M/\infty$ model (parameters λ and μ, or equivalently parameters ρ and μ) and of the $M/H_2/\infty$ model (parameters ρ, μ_1, μ_2, p_1 and $p_2 = 1 - p_1$ – see definitions in Section 2.6.3). The values of these parameters are reported in rows 3–8 in Table 2.7.

Table 2.7: Parameter identification

Trace	$video_1$	$video_2$	$video_3$	$video_4$
Session lifetime	$3^d\,13^h\,33^m\,20^s$	$11^d\,1^h\,46^m\,8^s$	$50^d\,22^h\,13^m\,20^s$	$29^d\,16^h\,43^m\,13^s$
ρ	94.7	14.1	8.1	17.9
$1/\mu(s)$	18316	16476	66823	83390
$1/\mu_1(s)$	3897	1	1	1
$1/\mu_2(s)$	480061	226498	900854	473268
p_1	0.97	0.93	0.93	0.82
p_2	0.03	0.07	0.07	0.18
p	0.011	0.034	0.062	0.028
$S(s)$	2.5	3.2	20.0	10.0
α	1.0	0.5	0.5	0.5
β	0.15	0.1	0.1	0.1

Parameters p and S have been chosen by following the guidelines presented in Section 2.7, namely $\alpha \in \{0.5, 1\}$ and $\beta \in \{0.1, 0.15\}$. Values of these parameters are listed in rows

9–10 in Table 2.7. The performance of estimators $\hat{N}_n^E$ and $\hat{N}_n^{H_2}$ are reported in Tables 2.8 and 2.9.

Table 2.8 reports several order statistics (columns 3–7) and the sample mean of the relative error $\frac{|N_n - \hat{N}_n|}{N_n}$ (column 2), where $\hat{N}_n$ is either $\hat{N}_n^E$ or $\hat{N}_n^{H_2}$. All results are expressed in percentages. The first observation is that both estimators perform reasonably well. The sample mean of the relative error is always less than 6.82% and is as low as 3.79%; when averaging over all experiments, this sample mean is less than 4.5% for both $\hat{N}_n^E$ and $\hat{N}_n^{H_2}$ (see last two rows). The last column gives the 95th percentile and reads as follows: the relative error achieved on trace *video*$_3$ by $\hat{N}_n^E$ (resp. $\hat{N}_n^{H_2}$) is 95% of the time less than 11% (resp. 12.6%). The second observation is that no scheme is uniformly better than the other over an entire session but their sample means are very close to each other (see column 2). For instance, $\hat{N}_n^E$ performs better than $\hat{N}_n^{H_2}$ regarding the 90th and the 95th percentiles whereas the result is reversed regarding the 25th percentile. It looks like the relative error on $\hat{N}_n^{H_2}$ is empirically more dispersed around its mean than is the relative error on $\hat{N}_n^E$, and has a longer tail. Across all sessions (see last two rows), 75% of the time $\hat{N}_n^{H_2}$ performs better than $\hat{N}_n^E$. This improvement does not come for free, since it requires the identification of 4 parameters (ρ, μ_1, μ_2 and p_1) instead of 2 (ρ and μ) for $\hat{N}_n^E$.

Table 2.8: Mean and percentiles of the relative error $|N_n - \hat{N}_n|/N_n$

Estimator	Mean	25	50	75	90	95
video$_1$						
$\hat{N}_n^E$	6.82	1.09	2.42	5.25	11.5	19.4
$\hat{N}_n^{H_2}$	6.12	1.08	2.55	6.31	13.5	20.6
video$_2$						
$\hat{N}_n^E$	4.19	1.41	3.08	5.43	8.66	11.9
$\hat{N}_n^{H_2}$	4.12	0.98	2.14	4.41	8.78	12.6
video$_3$						
$\hat{N}_n^E$	4.20	1.55	3.26	5.71	8.71	11.0
$\hat{N}_n^{H_2}$	3.98	1.07	2.36	4.83	9.35	12.6
video$_4$						
$\hat{N}_n^E$	3.79	1.23	2.57	4.51	7.50	11.0
$\hat{N}_n^{H_2}$	4.06	1.02	2.21	4.39	8.98	14.7
Over all traces						
$\hat{N}_n^E$	4.44	1.33	2.88	5.22	8.60	12.0
$\hat{N}_n^{H_2}$	4.34	1.02	2.26	4.73	9.61	14.2

Table 2.9 reports the sample mean and the sample variance of the error $N_n - \hat{N}_n$. In the 4th column, we list the theoretical variance. It is given by ϵ_{min} for $\hat{N}_n^E$ (see (2.51)) and

by ϵ for $\hat{N}_n^{H_2}$ (see (2.57)). The expected average $\mathbf{E}[N_n - \hat{N}_n]$ is zero in both approaches. Both estimators $\hat{N}_n^E$ and $\hat{N}_n^{H_2}$ have almost no bias (see column 2), and their empirical variances closely match the theoretical ones given by ϵ_{min} and ϵ, respectively. It is of interest to point out that for the 4 traces studied, ϵ, the theoretical mean square error provided by $\hat{N}_n^{H_2}$, is smaller than ϵ_{min}, the theoretical mean square error provided by $\hat{N}_n^E$ (however, this result is reversed if we consider the empirical mean square errors). Thus, $\hat{N}_n^{H_2}$ is more efficient[6] than $\hat{N}_n^E$ (again, $\hat{N}_n^E$ is empirically more efficient than $\hat{N}_n^{H_2}$). The last column provides the relative error on $\mathrm{Var}(\hat{N}_n^E)$, called η ($= \epsilon_{min}/\rho$) in Section 2.7. Notice that $\eta < \beta$ (β is given in row 12 in Table 2.7).

Table 2.9: Empirical mean and variance of the error $N_n - \hat{N}_n$

Estimator	Mean	Variance	ϵ_{min}, ϵ	η
$video_1$				
$\hat{N}_n^E$	-0.112	12.664	13.942	0.147
$\hat{N}_n^{H_2}$	-0.047	12.851	12.120	
$video_2$				
$\hat{N}_n^E$	0.006	0.495	1.407	0.099
$\hat{N}_n^{H_2}$	0.019	0.785	0.396	
$video_3$				
$\hat{N}_n^E$	0.037	0.207	0.737	0.091
$\hat{N}_n^{H_2}$	0.019	0.229	0.208	
$video_4$				
$\hat{N}_n^E$	0.052	0.911	1.566	0.087
$\hat{N}_n^{H_2}$	0.065	1.423	0.676	

In Figure 2.8 (resp. 2.9, 2.10 and 2.11) we plot the variations of membership for session $video_1$ (resp. $video_2$, $video_3$ and $video_4$), together with the estimates returned by $\hat{N}_n^E$ and $\hat{N}_n^{H_2}$. Among all 4 sessions, session $video_1$ presents the highest variations in N_n. Figure 2.8(a) (resp. 2.9(a), 2.10(a) and 2.11(a)) displays three curves: the collected video trace, the estimation returned by $\hat{N}_n^E$, labeled "Exponential", and the estimation returned by $\hat{N}_n^{H_2}$, labeled "Hyperexponential". It is clearly visible, especially at the left-hand side of graph 2.8(a), that $\hat{N}_n^E$ tracks better the session dynamics than $\hat{N}_n^{H_2}$. Actually, $\hat{N}_n^E$ follows better N_n during periods of high variations whereas $\hat{N}_n^{H_2}$ is slightly closer to N_n during flat periods. Both estimators $\hat{N}_n^E$ and $\hat{N}_n^{H_2}$ have been derived under some specific and restrictive assumptions: Poisson join times for both of them, exponential (resp. 2-stage hyperexponential) on-times for the first (resp. second) one. It is interesting to know whether or not these assumptions were violated in each session $video_i$, $i = 1, \ldots, 4$. We have therefore

[6] An estimator is said to be more efficient if it has a smaller variance.

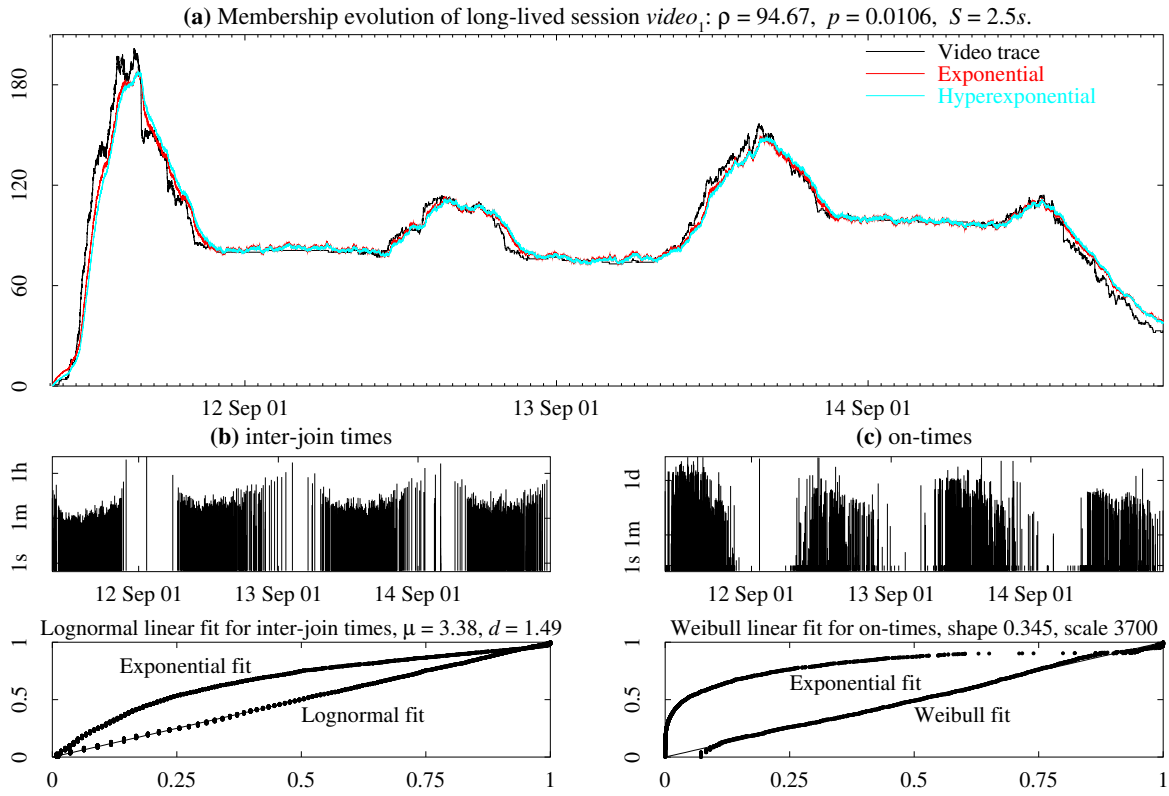


Figure 2.8: Membership estimation of session $video_1$ and corresponding probability plots

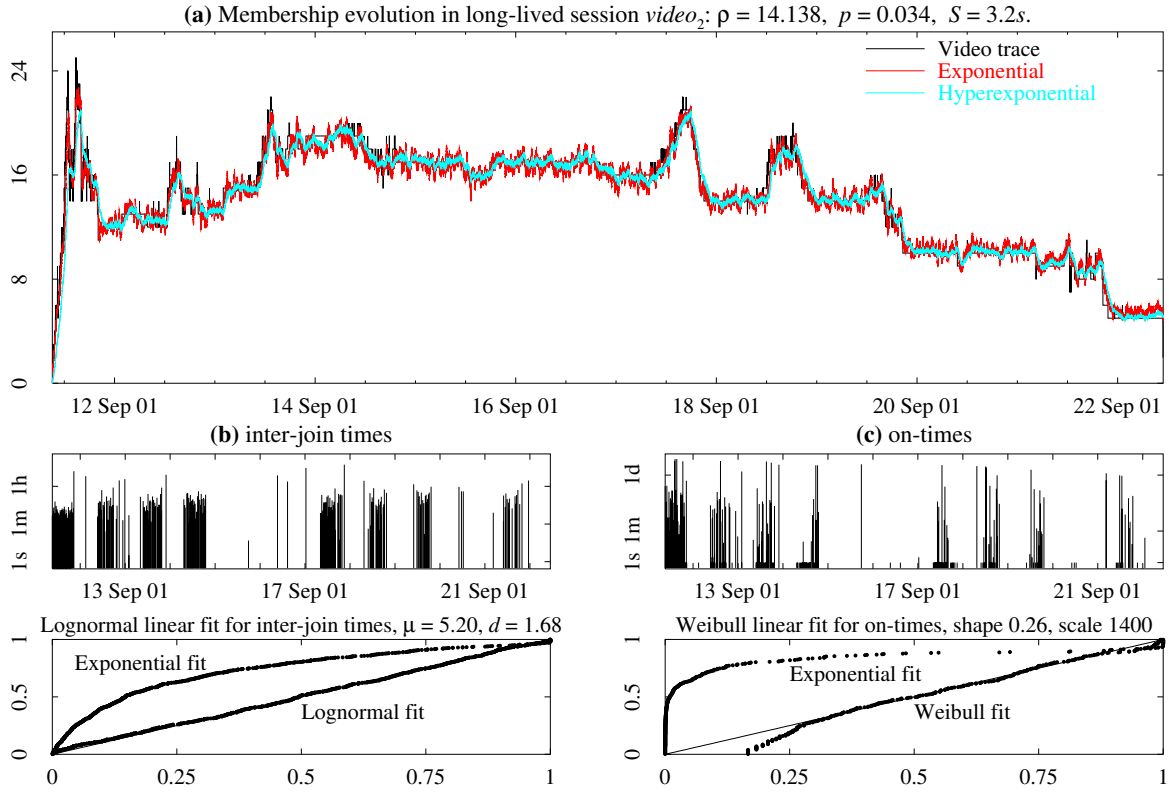


Figure 2.9: Membership estimation of session $video_2$ and corresponding probability plots

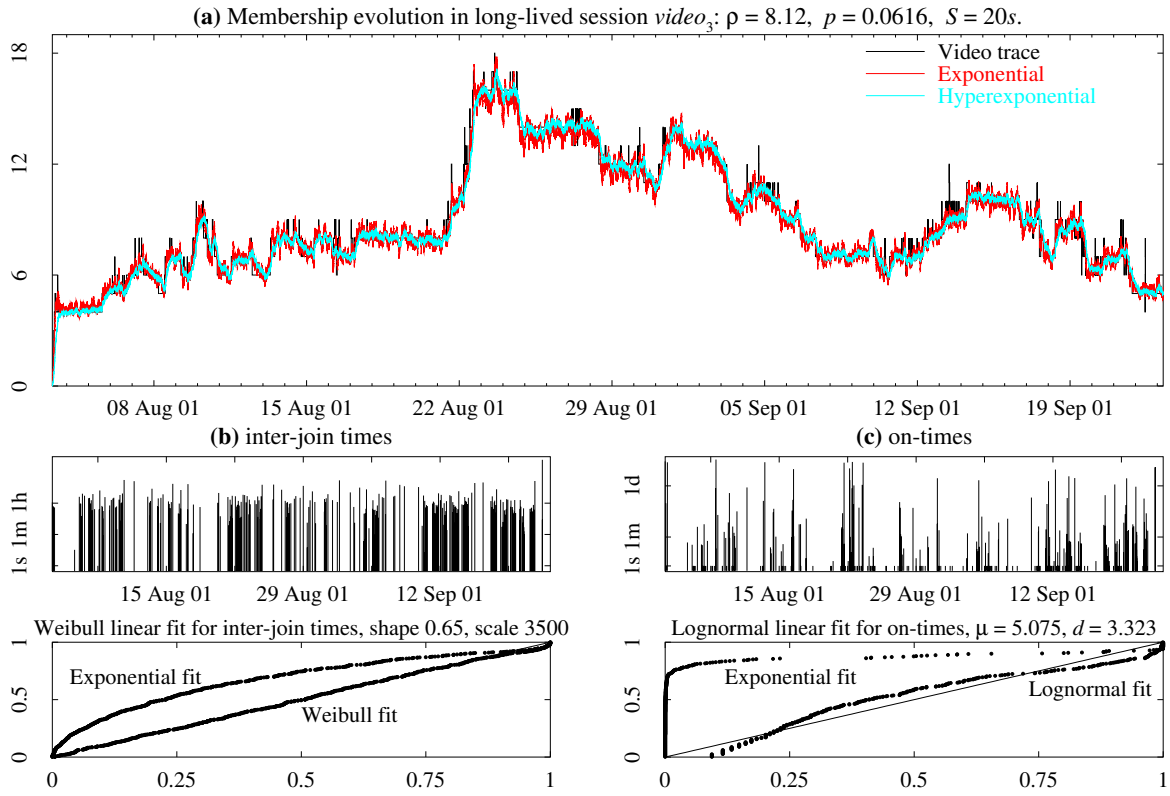


Figure 2.10: Membership estimation of session $video_3$ and corresponding probability plots

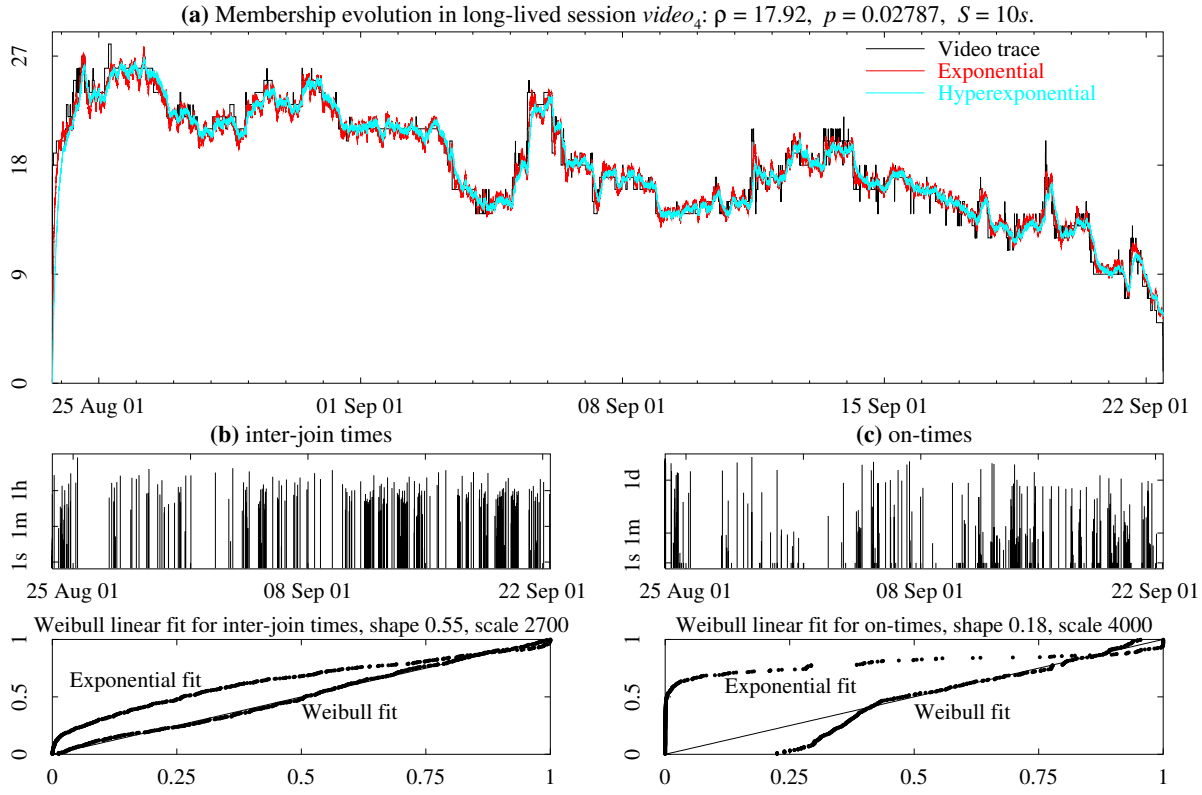


Figure 2.11: Membership estimation of session $video_4$ and corresponding probability plots

carried out a statistical analysis of each trace in order to determine the nature of their join time process and of their on-time sequence.

As shown in Table 2.10 and Figures 2.8, 2.9, 2.10 and 2.11, parts (b) and (c), neither is the join time process Poisson nor are the on-times exponentially distributed (or hyperexponentially distributed) for any of the traces. The inter-join times and the on-times appear to follow subexponential distributions (Lognormal and Weibull distributions), a situation quite different from the assumptions under which the estimators have been obtained. Despite these significant differences, the estimators behave well and therefore show good robustness to assumption violations.

Table 2.10: Distributions that best fitted into the inter-arrivals and on-times sequences

Trace	Best fit for inter-arrivals sequence	Best fit for on-times sequence
$video_1$	Lognormal with $\mu = 3.38$, $d = 1.49$	Weibull with shape 0.35, scale 3700
$video_2$	Lognormal with $\mu = 5.20$, $d = 1.68$	Weibull with shape 0.26, scale 1400
$video_3$	Weibull with shape 0.65, scale 3500	Lognormal with $\mu = 5.08$, $d = 3.32$
$video_4$	Weibull with shape 0.55, scale 2700	Weibull with shape 0.18, scale 4000

In Sections 2.4.3 and 2.4.4, estimator $\hat{N}_n^E$ was shown to behave well in case the distributions of both sequences are subexponential (Pareto, Lognormal, Weibull). We confirm this observation and note that estimator $\hat{N}_n^{H_2}$ performs well for two subexponential distributions (Lognormal, Weibull).

In summary, both estimators perform very well when applied to real video traces and are robust to significant deviations from their (theoretical) domain of validity. Estimator $\hat{N}_n^{H_2}$ returns the best global performance for the relative error criterion, but does not track high fluctuations as well as $\hat{N}_n^E$. Overall, we have found that $\hat{N}_n^E$ is a good estimator, both in terms of its performance and its usability since it only requires the knowledge of two parameters: ρ and μ.

2.9 Estimating parameters ρ and μ

The main pending issue concerns the knowledge of parameters ρ and μ (or equivalently any two parameters among ρ, λ and μ, since $\rho = \lambda/\mu$ in steady-state). When these parameters are not known, the source should estimate them. Again, the source could estimate any two parameters among ρ, λ and μ and infer the third one.

One possible way of estimating λ is to let a newly arrived receiver send a "hello" message to the source with a certain (constant) probability q (q should be small enough to avoid overwhelming the source with hellos). The source would then use the arrival time t_m of the mth hello to estimate λ. The maximum likelihood estimator is $\hat{\lambda} = m/(qt_m)$. This

estimator is unbiased and consistent by the strong law of large numbers ($\lim_{m\to\infty} t_m/m = 1/(q\lambda)$).

In a similar way, the source can estimate μ if receivers probabilistically send a "goodbye" message reporting their lifetime when they leave the session. Let $\tau_{m'}$ be the lifetime indicated in the m'th goodbye message received at the source, then the maximum likelihood estimator of μ is simply $\hat{\mu} = m'/(\sum_{i=1}^{m'} \tau_{m'})$. The estimator $\hat{\mu}$ is unbiased and consistent.

A natural estimator for ρ is $\hat{\rho} = \mathbf{E}[\hat{N}_n]$. As long as there is no estimation of both ρ and μ, it is not possible to compute the filter coefficient A and B. Then only a naive estimator for N_n can be used, defined as the ratio of the number of ACKs received Y_n over the ACK probability p (see Section 2.3). Notice that $\mathbf{E}[Y_n/p] = \rho$.

We have tested the estimator $\hat{N}_n^E$ when λ and ρ are estimated. We have chosen a hello probability $q = 0.1$, which means that, on average, one hello message is sent to the source for every 10 arrivals. The performance of the estimator can visually be observed in Figure 2.12 which displays two graphs. In the upper graph, the ACK probability is $p = 0.011$ yielding an average amount of ACKs equal to $\mathbf{E}[Y_n] = 1.04$; in the lower graph, we have $p = 0.021$ yielding $\mathbf{E}[Y_n] = 1.99$. We have plotted five curves in each graph: (i) the original video trace, (ii) the membership estimation for the case that the parameters are known beforehand, (iii) the membership estimation for the case that estimators $\hat{\lambda} = m/(qt_m)$ and $\hat{\rho} = \mathbf{E}[Y_n]/p$ are used, (iv) the estimation returned by the EWMA algorithm (see (2.6)) for $\alpha = 0.99$ and (v) the estimation returned by the EWMA algorithm for $\alpha = 0.999$. We have represented only a subset of the data which corresponds to the first $17^h\,30^m$, since the beginning of the session is the most challenging for our algorithm. As expected, when the parameters ρ and μ are unknown, the estimator $\hat{N}_n^E$ does not behave as well as when the parameters are known beforehand. Still, its performance is reasonably fair as can be seen in Table 2.11.

Table 2.11 reports the sample mean and some order statistics of the relative error

Table 2.11: Mean and percentiles of the relative error expressed in percentage

Estimator ($p = 0.011$)	Mean	25	50	75	90	95	99
Parameters are known	7.32	1.40	3.00	5.80	12.21	19.34	110.8
Parameters are estimated	6.50	1.81	4.01	7.56	14.39	22.24	42.49
EWMA $\alpha = 0.99$	6.30	2.28	4.91	8.57	12.57	15.55	25.94
EWMA $\alpha = 0.999$	19.20	4.70	10.95	23.42	54.59	61.18	82.89
Estimator ($p = 0.021$)	Mean	25	50	75	90	95	99
Parameters are known	6.03	1.17	2.61	4.95	8.76	14.46	84.91
Parameters are estimated	5.18	1.46	3.16	5.88	10.51	16.39	44.05
EWMA $\alpha = 0.99$	4.60	1.57	3.37	5.99	9.19	11.43	23.70
EWMA $\alpha = 0.999$	6.65	1.31	3.30	7.44	14.49	21.23	59.47

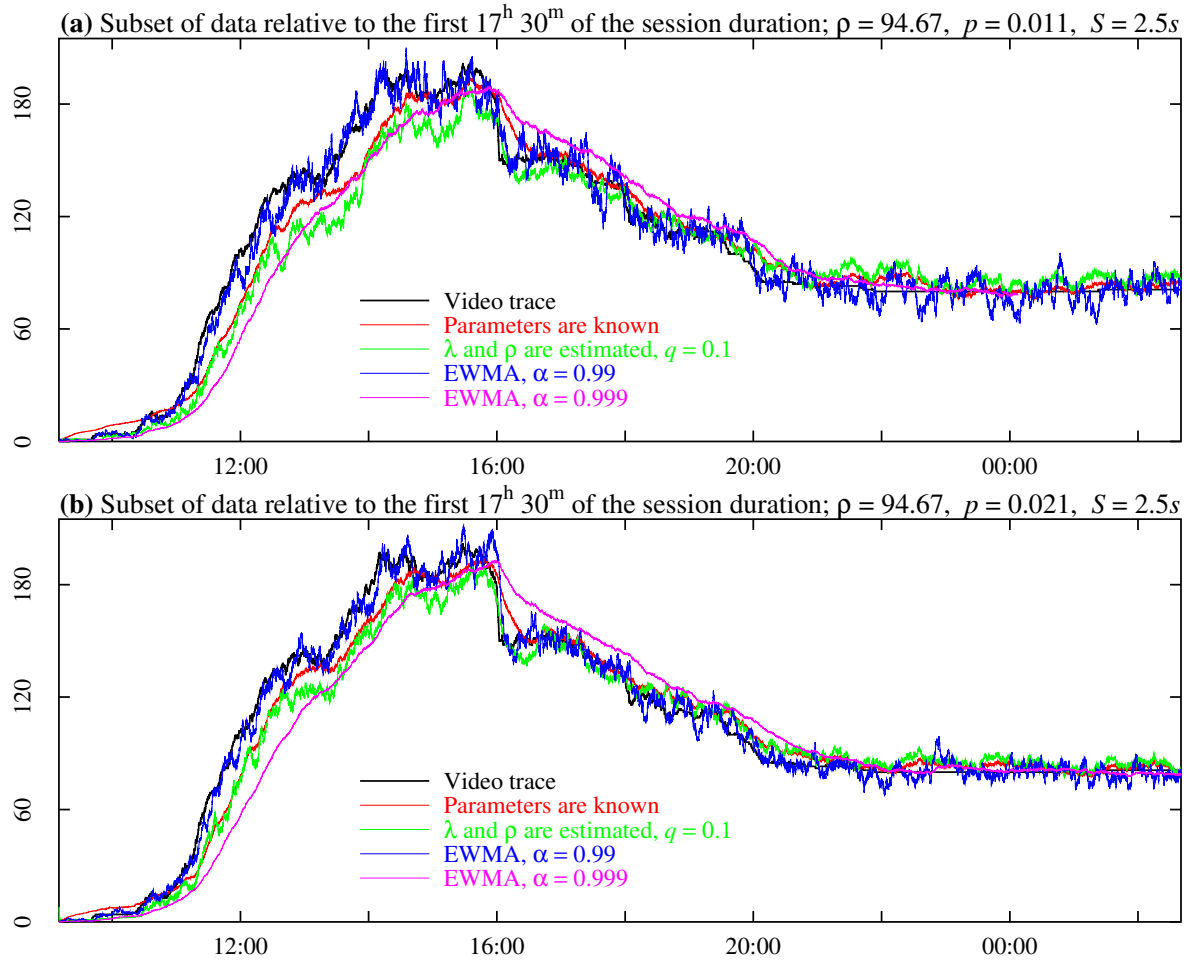


Figure 2.12: Membership estimation of session $video_1$ when (i) parameters are known beforehand, (ii) estimators $\hat{\lambda} = m/(qt_m)$ and $\hat{\rho} = \mathbf{E}[Y_n]/p$ are used $(q = 0.1)$ and (iii) EWMA estimators are used $(\alpha = 0.99, 0.999)$

returned by our scheme and by the EWMA algorithm proposed in (2.6) in both cases. Observe that, when the parameters are estimated and for $p = 0.011$, the relative error is 90% of the time within 14.39% of the true membership which is a fair result (see row 3 in Table 2.11). The performance of the estimator is enhanced when the amount of ACKs received every S seconds is increased. When $p = 0.021$, the average $\mathbf{E}[Y_n]$ is almost doubled (but there is still no feedback implosion), and the relative error on $\hat{N}_n^E$ is 95% of the time within 16.39% of the true membership which is a good result (see row 8 in Table 2.11). As for the EWMA estimator, we observe both in Figure 2.12 and Table 2.11 (rows 4 and 9) that the performance is very good when $\alpha = 0.99$, the corresponding estimator is actually the best one, following the 95th percentile criterion, among all four estimators: $\hat{N}_n^E$ when the parameters are known, $\hat{N}_n^E$ when the parameters λ and ρ are estimated, EWMA with $\alpha = 0.99$ and EWMA with $\alpha = 0.999$ (see column 7 in Table 2.11). However, the EWMA

estimator with $\alpha = 0.999$ does not behave well when $p = 0.011$ as its performance is the worst one (see row 5 in Table 2.11). Last, observe that the sample mean of the relative error returned by $\hat{N}_n^E$ when the parameters are estimated is smaller than the one returned by $\hat{N}_n^E$ when the parameters are known beforehand (see column 2 in Table 2.11, rows 2–3 and 7–8). This is because the 99th percentile of the latter is much greater than the 99th percentile of the former. At the very beginning of the trace, the error returned by $\hat{N}_n^E$, when the parameters are known beforehand, is very large, in contrast to the error returned by the other estimators.

Table 2.12 reports the sample mean (column 2) and the sample variance (column 3) of the error between the true membership and its estimation. Notice how for both considered ACK probabilities, the variance of the EWMA estimator with $\alpha = 0.999$ is high (see rows 5 and 10 in Table 2.12). When increasing the ACK probability p from 0.011 to 0.021, the error variance of $\hat{N}_n^E$ when the parameters λ and ρ are estimated decreases almost by half (see rows 3 and 8, column 3).

Table 2.12: Empirical mean and variance of the estimation error

Estimator ($p = 0.011$)	Mean	Variance
Parameters are known	0.1432	40.3128
Parameters are estimated	0.5606	68.1708
EWMA $\alpha = 0.99$	−0.2732	45.0886
EWMA $\alpha = 0.999$	4.1743	332.966
Estimator ($p = 0.021$)	Mean	Variance
Parameters are known	−0.0871	26.5487
Parameters are estimated	0.2402	37.6369
EWMA $\alpha = 0.99$	0.0006	23.1149
EWMA $\alpha = 0.999$	0.2570	79.6634

Remark 2.9.1 *For the trace video$_1$, the EWMA estimator with $\alpha = 0.99$ behaves very well in contrast to the EWMA estimator with $\alpha = 0.999$. This is exactly the inverse of what we have observed when applying both EWMA estimators on the audio trace shown in Figure 2.2 page 72. There, the EWMA estimator with $\alpha = 0.99$ was still very bursty.*

To conclude this discussion, we believe that using the estimator $\hat{N}_n^E$ and estimating the parameters λ and ρ on-line is appealing in the sense that, even though its performance is not the best one ever, one is sure of having a fair result for a relatively small amount of ACKs. This is not the case of the EWMA estimator as the user will not know in advance what value assign to α.

Current research focuses on identifying the best two parameters to estimate in order to have the fastest convergence and the lowest error while minimizing the overhead bandwidth generated, keeping in mind that moving average estimators for λ and μ are also possible.

Another perspective concerns the design of an efficient estimator provided that the underlying model is either the $M/W/\infty$ queue or the $M/L/\infty$ queue, where W and L stand for Weibull and Lognormal, respectively. Indeed, we have observed that the on-time distribution is either Weibull or Lognormal (refer to Tables 2.5 and 2.10 for details). To compute the filter parameters, one has the choice of adopting the autoregressive equation given in (2.54) and computing the coefficients A and B which minimize the mean-square error, or one can adopt the EWMA scheme given in (2.6) and derive the parameter α which minimize the mean-square error. We believe that the canonical factorization step required in the Wiener theory cannot be achieved when considering either the $M/W/\infty$ queue or the $M/L/\infty$ queue, impairing the use of this theory in our context.

2.10 Conclusion

The major contribution of this work is the design of novel estimators for evaluating the membership in multicast sessions. We have designed estimators capable of efficiently tracking the dynamics of multicast sessions while simultaneously avoiding feedback implosion. We have successively investigated three distinct approaches, each enabling the sender in a multicast application to track on-line the variation of the audience over time.

We have first modeled the multicast group as an $M/M/\infty$ queue and established our results under the assumption that this queue is under heavy traffic. In this regime the backlog process of the $M/M/\infty$ queue is "close" to a diffusion process that can be used to cast our estimation problem into the appealing framework of Kalman filter theory. Using this theory we have derived an estimator that minimizes the variance of the error. We have carried out several simulations to test the robustness of our estimator in the case where the arrivals are not Poisson and/or the on-times are not exponentially distributed. The estimator has also been computed on real audio traces and its performance have been shown to be excellent. It is worthy to point out that it is the first time that a membership estimator is tested on real traces, exhibiting human behavior and correlations between the different processes at hand.

Aiming at generalizing the multicast model, we relied on the appealing Wiener filter theory to compute the optimal linear estimator for session membership when the underlying model is an $M/M/\infty$ queue (the heavy traffic assumption is no longer needed). The optimality refers to the unbiasedness of the estimator and to the fact that the mean square error is minimized. The latter estimator turned out to be identical to the one designed using the Kalman filter theory.

We have also developed the optimal first-order linear filter in case the on-time distribution is arbitrary and have derived the associated estimator in case the on-times have a 2-stage hyperexponential distribution. The last pair of estimators have been validated on real video traces. Their performance have been shown to be excellent, one of them showing a good ability to adapt to highly dynamic multicast sessions.

Appendix: Computing parameters from trace

Each trace records $(T_i, D_i), i \geq 1$. To use the $M/M/\infty$ queue model, we identify $1/\lambda = \mathbf{E}[T_{i+1} - T_i]$ and $1/\mu = \mathbf{E}[D]$, and deduce $\rho = \lambda/\mu$. To use the $M/H_2/\infty$ queue model, λ is computed as before. Identifying μ_1, μ_2, p_1 and p_2 requires the knowledge of the first three moments of D (recall that $p_2 = 1 - p_1$). For a 2-stage hyperexponential distribution, the kth moment is given by

$$\mathbf{E}[D^k] = \sum_{l=1}^{2} \frac{p_l k!}{(\mu_l)^k} = k! \left(\frac{p_1}{(\mu_1)^k} + \frac{p_2}{(\mu_2)^k} \right), \text{ for } k \geq 1.$$

The parameters μ_1, μ_2, p_1 and p_2 are then solution to the following system of four equations, where σ_l stands for $1/\mu_l$ with $l = 1, 2$.

$$p_1 + p_2 = 1 \tag{2.62}$$
$$p_1\sigma_1 + p_2\sigma_2 = \mathbf{E}[D] \tag{2.63}$$
$$p_1\sigma_1^2 + p_2\sigma_2^2 = \mathbf{E}[D^2]/2 \tag{2.64}$$
$$p_1\sigma_1^3 + p_2\sigma_2^3 = \mathbf{E}[D^3]/6. \tag{2.65}$$

Equations (2.62) and (2.63) readily give

$$p_1 = \frac{\sigma_2 - \mathbf{E}[D]}{\sigma_2 - \sigma_1}, \tag{2.66}$$
$$p_2 = \frac{\mathbf{E}[D] - \sigma_1}{\sigma_2 - \sigma_1}. \tag{2.67}$$

Substituting Equations (2.66) and (2.67) for p_1 and p_2 into (2.64) yields

$$\begin{aligned} (\sigma_2 - \sigma_1)\mathbf{E}[D^2]/2 &= (\sigma_2 - \mathbf{E}[D])\sigma_1^2 + (\mathbf{E}[D] - \sigma_1)\sigma_2^2 \\ &= \mathbf{E}[D](\sigma_2 - \sigma_1)(\sigma_2 + \sigma_1) - \sigma_1\sigma_2(\sigma_2 - \sigma_1) \\ \implies \quad \mathbf{E}[D^2]/2 &= \mathbf{E}[D](\sigma_2 + \sigma_1) - \sigma_1\sigma_2. \end{aligned} \tag{2.68}$$

Substituting Equations (2.66) and (2.67) for p_1 and p_2 into (2.65) yields

$$\begin{aligned} (\sigma_2 - \sigma_1)\mathbf{E}[D^3]/6 &= (\sigma_2 - \mathbf{E}[D])\sigma_1^3 + (\mathbf{E}[D] - \sigma_1)\sigma_2^3 \\ &= \mathbf{E}[D](\sigma_2 - \sigma_1)(\sigma_2^2 + \sigma_1\sigma_2 + \sigma_1^2) - \sigma_1\sigma_2(\sigma_2 - \sigma_1)(\sigma_2 + \sigma_1) \\ \implies \quad \mathbf{E}[D^3]/6 &= \mathbf{E}[D]\Big((\sigma_2 + \sigma_1)^2 - \sigma_1\sigma_2\Big) - \sigma_1\sigma_2(\sigma_2 + \sigma_1). \end{aligned} \tag{2.69}$$

Introduce now S_σ and P_σ as the sum and the product of σ_1 and σ_2, respectively. Equations (2.68) and (2.69) become

$$\begin{aligned}
\mathbf{E}[D^2]/2 &= \mathbf{E}[D]S_\sigma - P_\sigma \\
\mathbf{E}[D^3]/6 &= \mathbf{E}[D](S_\sigma^2 - P_\sigma) - P_\sigma S_\sigma \;=\; \Big(\mathbf{E}[D]S_\sigma - P_\sigma\Big)S_\sigma - \mathbf{E}[D]P_\sigma \\
&= \left(\mathbf{E}[D^2]/2\right)S_\sigma - \mathbf{E}[D]P_\sigma
\end{aligned}$$

where the latter identity is obtained when using the first one. It then follows that

$$S_\sigma = \frac{3\mathbf{E}[D]\mathbf{E}[D^2] - \mathbf{E}[D^3]}{3(2\mathbf{E}[D]^2 - \mathbf{E}[D^2])}, \tag{2.70}$$

$$P_\sigma = \frac{3\mathbf{E}[D^2]^2 - 2\mathbf{E}[D]\mathbf{E}[D^3]}{6(2\mathbf{E}[D]^2 - \mathbf{E}[D^2])}, \tag{2.71}$$

and $\sigma_1 = 1/\mu_1$ and $\sigma_2 = 1/\mu_2$ are the (positive) solutions of $x^2 - S_\sigma\, x + P_\sigma = 0$, or equivalently, the solutions of

$$6\Big(2\mathbf{E}[D]^2 - \mathbf{E}[D^2]\Big)x^2 + 2\Big(\mathbf{E}[D^3] - 3\mathbf{E}[D]\mathbf{E}[D^2]\Big)x + 3\mathbf{E}[D^2]^2 - 2\mathbf{E}[D]\mathbf{E}[D^3] = 0.$$

Finally,

$$\sigma_{1,2} = 1/2 \times \Big(S_\sigma \pm \sqrt{(S_\sigma)^2 - 4P_\sigma}\Big).$$

We now can compute p_1 and p_2 as given in Equations (2.66) and (2.67). It is then possible to calculate $\rho_l = p_l\lambda/\mu_l$ and $\gamma_l = \exp(-\mu_l S)$ for $l = 1, 2$. Last $\rho = \rho_1 + \rho_2$.

Chapter 3

Analysis of two agent location mechanisms in a mobile environment

This chapter is devoted to the performance evaluation and comparison of two approaches for locating an agent in a mobile agent environment. The first approach dynamically creates a chain of forwarders to locate a moving agent whereas the second one relies on a centralized server to perform this task. This chapter builds on Markov chain analysis to evaluate the cost of communication in the presence of migration. It undertakes validation of the proposed models by comparing theoretical results first with simulations and then with experimental results. Our research revealed that neither approach is uniformly best than the other as their respective performance depends on the system parameters. The analysis developed in this chapter can be used to select the best scheme based on its average response time. Designers and programmers can benefit from this research by implementing the best mechanism which minimizes communication times between components of the application at hand.

Keywords: mobile code, migration, centralized server, forwarders, Markov chain.

Note: Part of the material presented in this chapter is published in [13, 14, 15].

3.1 Introduction

The Internet has allowed the creation of huge amounts of data located on many different machines. Performing complex operations on some data requires that the data be transferred first to the machine on which the operations are to be executed. This transfer may require a non-negligible amount of bandwidth and may seriously limit performance if it is the bottleneck. However, instead of moving the data to the code, it is possible to move the code to the data, and perform all of the operations locally. This simple idea has led to a new paradigm called *code-mobility* [117]. In this paradigm, a mobile object – sometimes called an agent – is given a list of destinations and a series of operations to perform at each one of them. The mobile object will visit all of the destinations, perform the requested operations and possibly pass the result on to another object. Any machine willing to receive an agent must provide an agent-platform which is a placeholder where the agent is executed.

Code mobility has recently received a lot of attention because of its wide application to fields ranging from e-commerce (e.g. searching for the lowest fare on many different sites) to data mining [33]. Mobility can be implemented as a service provided by an operating system [92]; however this severely limits its usefulness in a heterogeneous environment such as the Internet. Another solution is to use a library which provides an application with all of the necessary features [5, 19, 60, 74, 84, 101, 121].

Any mobility mechanism must first provide a way to migrate code from one host to another. It must also ensure that any communication between objects will not be impaired by migration, namely that two objects should still be able to communicate even if one of them has migrated. Such a mechanism is referred to as a *routing* mechanism or even as a *location* mechanism since it often relies on the knowledge of the location of the mobile objects to ensure communications. Location problems are not exclusive to code mobility. They can be encountered under different forms in many different networking areas, where the object to be located can either be fixed [85] or mobile as in wireless [102] and ad-hoc networks [25, 42, 69], or more recently in peer-to-peer networking [18, 38, 76, 113].

In the more specific setting of code mobility, two location mechanisms are widely used: the first one uses a centralized (location) server which keeps track of the location of agents, whereas the second one relies on special objects – called *forwarders* – whose role is to forward a message to the agents. A more careful description of these approaches will be given later on.

Mobility raises several concerns, among them security [32, 71] (of both the mobile agent and the host sheltering it) and performance issues [20, 55]. In [55] the complexity of using forwarding addresses is extensively studied whereas [20] addresses fault-tolerance properties in forwarder-based mechanisms. In this chapter we will only focus on performance issues and, more specifically, on the cost of communication in the presence of migration. To the best of our knowledge, this is the first time that such an analysis is performed. In [84] the authors only give intuitive criteria on how to select the proper location scheme under certain circumstances. Our analysis can be used to select the best scheme based on its

response time. It also allows us to compute the average number of forwarders, which is useful to study the fault-tolerance of forwarding schemes [20].

In this work we develop Markovian models of the forwarders and of the location server as implemented in *ProActive* [101], a Java library that provides all of the necessary primitives for code mobility. Closed-form expressions for various performance measures are derived, including the time needed to reach an agent and the mean number of forwarders. These expressions are in turn used to evaluate the cost of location under various network conditions and for different applications. For the purpose of validation, we have developed for each mechanism both an event-driven simulator and a benchmark that uses *ProActive*. Simulations and experiments conducted on a LAN and on a MAN have validated both models and have shown that no scheme performs uniformly better than the other, thereby justifying the present research.

The chapter is organized as follows: preliminary definitions and notation are introduced in Section 3.2; the forwarding mechanism is presented and evaluated in Section 3.3 and the centralized server mechanism is investigated in Section 3.4. Simulations and experimental results are reported in Section 3.5 as well as a theoretical comparison between both approaches. Extension of our work to more general cases are discussed in Section 3.6.

3.2 Definitions and Notation

In this section we introduce several random variables (RVs) that we will use to construct our models. Throughout the chapter a mobile object will indifferently be called a mobile agent or simply an agent. The following notation is related ti a given source-agent pair.

The ith message is sent by the source to the agent at time a_i and the communication is over at time $d_i := a_i + \tau_i$. The RV τ_i – referred to as the (ith) *communication time* – is a scheme-dependent quantity that will be defined later for each mechanism (forwarders and centralized server). In the time-interval (d_i, a_{i+1}) no message is generated by the source. Let $w_{i+1} := a_{i+1} - d_i$ be the length of this time-interval and assume that $w_1 := a_1$ and that no message is generated in $[0, a_1)$.

The jth migration of the mobile agent occurs at time $m_j > 0$ and it requires the mobile agent p_j units of times to reach its new location. During a migration period the agent is unreachable. The mobile agent then spends u_{j+1} units of time at its jth location, time during which it can be reached by a message, and then initiates a new migration. We set $u_1 := m_1$ and assume that the mobile agent does not migrate in $[0, m_1)$ (see Figure 3.1).

Both the source and the agent exhibit a two state process: the source alternates between an "idle" state and a "communicating" state, and the agent alternates between an "idle" state and a "migrating" state.

The following assumptions will be enforced throughout the chapter:

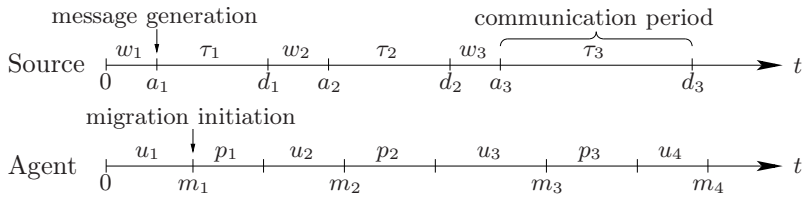


Figure 3.1: A time diagram including all RVs relative to the source and to the agent

A1 The *input* sequences $\{w_i, i \geq 1\}$, $\{p_j, j \geq 1\}$ and $\{u_k, k \geq 1\}$ are assumed to be mutually independent *renewal* sequences of RVs such that w_i, p_j and u_k are exponentially distributed RVs with parameters $\lambda > 0$, $\delta > 0$ and $\nu > 0$, respectively.

3.3 The forwarders

3.3.1 Description

Forwarding techniques were first introduced in distributed operating systems like DEMOS/MP [99] for finding mobile processes. The mechanism is straightforward: on leaving a host (machine), a process leaves a special reference, called a *forwarding reference* which points toward its next location. Upon a later migration, a new forwarding reference, pointing toward the new location, is created, and the previously created forwarding reference is now pointing to the new one. Thus, as the system runs, *chains of forwarders* are built (as illustrated in Figure 3.2, step 1). A consequence of this mechanism is that a caller does not usually know the location of the callee. A special built-in mechanism called short-cutting allows the update of the address as soon as a communication takes place. When a forwarded message reaches a mobile agent, the latter communicates its new location to the caller. As a result, all subsequent requests issued by this caller will not go through the existing forwarders – which are shortcut.

An illustration of the short-cutting feature is given in Figure 3.2: a message is sent by the source to the last known location of the agent (Host B). Since the agent is no longer at this location, the message is then forwarded to the host that was next visited by the agent (Host C). Again, the agent has already moved when the message reaches Host C and the message is forwarded to the next visited host (Host D) where the agent is finally located. A location message is then sent by the agent (located at host D) to the source and the next message will be sent by the source to Host D.

In order to maintain the same semantic as that of a static program (i.e. with no mobile agent) one has to introduce constraints. In particular, communications through a chain of forwarders should be synchronous, i.e. the caller remains blocked during the communication phase. With these assumptions we can now describe the protocol in use:

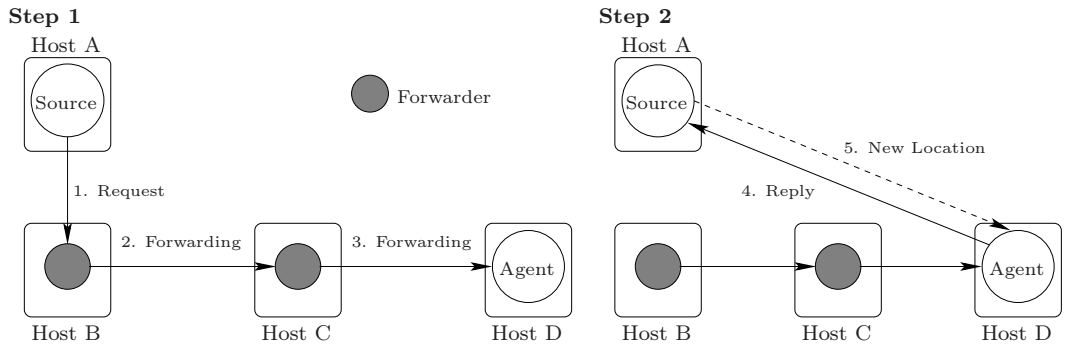


Figure 3.2: The short-cutting feature in the forwarding mechanism

- Upon migration, a mobile agent leaves a forwarder on the current site;
- This forwarder is linked to the mobile agent on the remote host;
- No communication can occur while the mobile agent is migrating;
- Every forwarder is exclusive to the agent that had created it;
- When receiving a message, the forwarder sends it to the next hop (possibly the mobile agent);
- Any successful communication places the mobile agent one hop away from the caller (e.g. after Step 2 in Figure 3.2 the agent is located one hop away from the source).

The above protocol is implemented in various Java libraries (*MOA* [84], *ProActive* [101] and *Voyager* [121]) except for the short-cutting feature that is triggered by the agent at the end of the message processing.

3.3.2 A Markovian analysis of the forwarders

In this section we will evaluate the performance (response time and number of forwarders in Sections 3.3.2.1 and 3.3.2.2 respectively) of the mechanism introduced in Section 3.3.1 through a Markovian analysis. We will assume that a mobile agent does not return to a previously visited site where there is still an active forwarder, i.e. it does not migrate twice (or more) to a particular site between two consecutive epochs d_i (see Section 3.2). Hence, there can be no loops within a chain. It is clear that under these conditions the length of a chain can extend to infinity if the number of hosts is infinite. Observe that a forwarder is used only once by a given source, because of the short-cutting that takes place at the end of a successful communication.

A forwarding mechanism is well represented by the chains of forwarders that it produces. In an application a chain of forwarders connects a single source to a single agent

and its dynamics is not affected by other objects; there will be as many chains as there are source-agent pairs. It suffices then to study the behavior of one chain to evaluate the performance of the forwarders approach. To study the dynamics of a chain, one should take into account the state of a chain and the states of the source and the agent at its endpoints. Notice that this does not place any assumption on the number of objects (source or agent) in the application.

From the description given in Section 3.3.1, it can be seen that at any time the system is in one of the following states (see Figure 3.3):

- States $(i, 0, 0)$, $i \geq 1$, indicate that the agent is available (i.e. not migrating) and located i hops away from the source, and that no message is traveling;
- State $(1, 0, 1^*)$ indicates that the agent is available and located one hop away from a message, and the latter has never been through any forwarder;
- State $(1, 0, 1)$ indicates that the agent is available and located one hop away from a message, the latter having already gone through at least one forwarder;
- States $(i, 0, 1)$, $i \geq 2$, indicate that a message is traveling, the agent is available and located i hops away from the message;
- States $(i, 1, k)$, $i \geq 1$, indicate that the agent is migrating and that before the initiation of its migration it was located i hops away from the source if no message is traveling $(k = 0)$ or it was located i hops away from the message if a message is traveling $(k = 1)$;
- State $(0, 1, 1)$ indicates that a message that has gone through at least one forwarder and the agent are at the same location but that the agent is migrating (i.e. the agent has initiated a migration just before the arrival of the message);
- State $(0, 0, 1)$ indicates that the message has reached the agent after having traveled through at least one forwarder and that the agent is currently communicating its new position to the source.

State $(1, 0, 1^*)$ has been introduced to take into account the fact that if a new message reaches the agent after exactly one hop and that the agent has not initiated a migration before the arrival of the message then the cycle is over and the source can transmit a new message; otherwise, if a message reaches the agent after having gone through at least one forwarder then the agent will have to communicate its location to the source once the message has reached it.

Under the enforced assumption

A2 The traveling time of a message from one host (possibly the source) to the next one (possibly the agent) is an exponential RV with parameter $\gamma > 0$. The successive

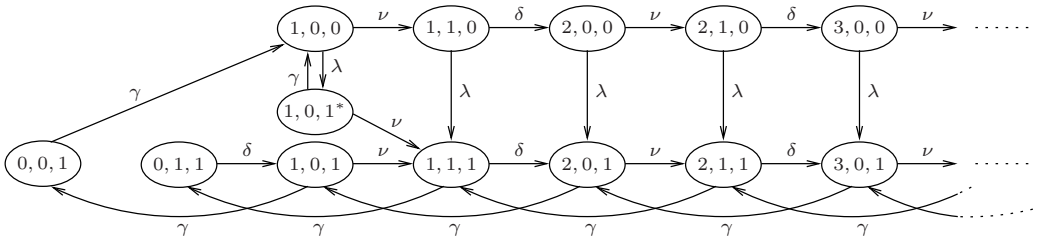


Figure 3.3: System states and transition rates in the forwarding mechanism

traveling times are assumed to be mutually independent and independent of the input sequences $\{w_i\}_i$, $\{p_i\}_i$ and $\{u_i\}_i$ introduced in Section 3.2,

and assumption **A1**[1], it is easily seen that the sojourn time in each state is exponentially distributed and that any state can be reached from any other state in a finite number of steps. In other words, the process defined above is an irreducible Markov process on the state-space $\mathcal{E} := \{(0,0,1),(0,1,1),(1,0,1^*),(i,j,k), i \geq 1, j,k = 0,1\}$.

The transition rates are indicated in Figure 3.3: from state $(1,0,0)$ the process may jump to state $(1,0,1^*)$ with rate λ (new message generated) or to state $(1,1,0)$ with rate ν (migration of the agent); from state $(1,0,1^*)$ the process may move to state $(1,1,1)$ with rate ν (migration) or to state $(1,0,0)$ with rate γ (message has reached the agent, cycle is over); from state $(i,0,0)$ with $i \geq 2$ the process may jump to state $(i,1,0)$ with rate ν (migration) or to state $(i,0,1)$ with rate λ (new message generated); from state $(i,1,1)$ with $i \geq 1$ the process can move to state $(i+1,0,1)$ with rate δ (end of migration) or to state $(i-1,1,1)$ with rate γ (message has reached next host on its route); from state $(i,1,0)$ with $i \geq 1$ the process can jump to state $(i+1,0,0)$ with rate δ (end of migration period) or to state $(i+1,0,1)$ with rate λ (new message generated); from state $(i,0,1)$ with $i \geq 1$ the process can move to state $(i-1,0,1)$ with rate γ (message has reached next host) or to state $(i,1,1)$ with rate ν (migration); from state $(0,1,1)$ the process may only move to state $(1,0,1)$ with rate δ (end of migration); finally, from state $(0,0,1)$ the process may only move to state $(1,0,0)$ with rate γ (location message sent by agent has reached the source; cycle over).

Let $p_{i,j,k}$ be the stationary probability that the process is in state $(i,j,k) \in \mathcal{E}$. If the Markov process is ergodic then the stationary probabilities $\{p_{i,j,k}, (i,j,k) \in \mathcal{E}\}$ are the

[1]Assumptions **A1** and **A2** are mainly made for sake of mathematical tractability. We have however observed in our experiments that our models are fairly robust to deviations from these assumptions – see Section 3.5.

unique strictly positive solution of the Chapman–Kolmogorov (C-K) equations [75]

$$
\begin{aligned}
(\lambda+\nu)\,p_{1,0,0} &= \gamma\,(p_{0,0,1}+p_{1,0,1^*}) && && (3.1)\\
(\lambda+\nu)\,p_{i,0,0} &= \delta\,p_{i-1,1,0} && i=2,3,\ldots && (3.2)\\
(\lambda+\delta)\,p_{i,1,0} &= \nu\,p_{i,0,0} && i=1,2,\ldots && (3.3)\\
\delta\,p_{0,1,1} &= \gamma\,p_{1,1,1} && && (3.4)\\
(\delta+\gamma)\,p_{1,1,1} &= \nu\,(p_{1,0,1}+p_{1,0,1^*})+\lambda\,p_{1,1,0}+\gamma\,p_{2,1,1} && && (3.5)\\
(\delta+\gamma)\,p_{i,1,1} &= \nu\,p_{i,0,1}+\lambda\,p_{i,1,0}+\gamma\,p_{i+1,1,1} && i=2,3,\ldots && (3.6)\\
(\nu+\gamma)\,p_{1,0,1^*} &= \lambda\,p_{1,0,0} && && (3.7)\\
p_{0,0,1} &= p_{1,0,1} && && (3.8)\\
(\nu+\gamma)\,p_{1,0,1} &= \delta\,p_{0,1,1}+\gamma\,p_{2,0,1} && && (3.9)\\
(\nu+\gamma)\,p_{i,0,1} &= \delta\,p_{i-1,1,1}+\lambda\,p_{i,0,0}+\gamma\,p_{i+1,0,1} && i=2,3,\ldots && (3.10)
\end{aligned}
$$

such that $\sum_{(i,j,k)\in\mathcal{E}} p_{i,j,k}=1$.

We will not try to solve Equations (3.1)–(3.10) explicitly. Instead, we will use the standard z-transform approach to characterize the ergodicity condition and the invariant measure of the Markov process. To this end, define for $|z|\le 1$

$$
f(z):=\sum_{i=1}^{\infty} z^i p_{i,0,0},\qquad g(z):=\sum_{i=1}^{\infty} z^i p_{i,1,0},\qquad h(z):=\sum_{i=0}^{\infty} z^i p_{i,0,1},\qquad k(z):=\sum_{i=0}^{\infty} z^i p_{i,1,1},
$$

the z-transform of the stationary probabilities $\{p_{i,0,1}\}_{i\ge 0}$, $\{p_{i,1,0}\}_{i\ge 1}$, $\{p_{i,1,1}\}_{i\ge 0}$ and $\{p_{i,0,0}\}_{i\ge 1}$, respectively. Last, introduce for $|x|\le 1, |y|\le 1$ and $|z|\le 1$

$$
\begin{aligned}
F(x,y,z) &:= \sum_{(i,j,k)\in\mathcal{E}} z^i x^j y^k p_{i,j,k}\\
&= \sum_{i=1}^{\infty} z^i p_{i,0,0}+x\sum_{i=1}^{\infty} z^i p_{i,1,0}+y\sum_{i=0}^{\infty} z^i p_{i,0,1}+xy\sum_{i=0}^{\infty} z^i p_{i,1,1}+zy^2\,p_{1,0,1^*}\\
&= f(z)+x\,g(z)+y\,h(z)+xy\,k(z)+zy^2\,p_{1,0,1^*}
\end{aligned}
$$

the z-transform of the stationary probabilities $\{p_{i,j,k},\,(i,j,k)\in\mathcal{E}\}$. $F(x,y,z)$ is determined in the following proposition:

Proposition 3.3.1 (z-transform of the Markov process)

The Markov process depicted in Figure 3.3 is ergodic if and only if

$$
\frac{1}{\gamma}<\frac{1}{\nu}+\frac{1}{\delta}. \tag{3.11}
$$

In steady-state, the z-transform $F(x,y,z)$ is given by

$$f(z) = \frac{z\,\gamma(\lambda+\delta)(\lambda+\nu)(\gamma+\nu)}{\nu(\nu+\lambda+\gamma)\,a(z)}\,p_{1,0.1} \tag{3.12}$$

$$g(z) = \frac{z\,\gamma(\lambda+\nu)(\gamma+\nu)}{(\nu+\lambda+\gamma)\,a(z)}\,p_{1,0,1} \tag{3.13}$$

$$h(z) = -\frac{z^2\,\delta\gamma}{b(z)}\,p_{0,1,1} + \left[\frac{\delta(\nu-\gamma)z^2 - \gamma(\nu+\delta)z + \gamma^2}{b(z)} - \frac{z^3\delta\gamma[\lambda\,a(z) + (\gamma+\nu)(\lambda\gamma + (\lambda+\delta)(\lambda+\nu))]}{(\nu+\lambda+\gamma)\,a(z)\,b(z)}\right]p_{1,0,1} \tag{3.14}$$

$$k(z) = \frac{\gamma\,(\gamma - z\,(\gamma+\nu))}{b(z)}\,p_{0,1,1} - \frac{z^2\gamma(\lambda+\nu)(\gamma+\nu)(\lambda\gamma + (\lambda+\delta)(\lambda+\nu))}{(\nu+\lambda+\gamma)\,a(z)\,b(z)}\,p_{1,0,1} \tag{3.15}$$

$$p_{1,0,1^*} = \frac{\lambda\gamma}{\nu\,(\nu+\lambda+\gamma)}\,p_{1,0,1} \tag{3.16}$$

$$p_{0,1,1} = \frac{1-\delta\nu/(\gamma(\delta+\nu))}{1 + [1-\delta\nu/(\gamma(\delta+\nu)) + (\lambda+\nu)(\gamma+\nu)(\gamma+\lambda)/(\lambda\nu(\nu+\lambda+\gamma))]\,c(z_0)} \tag{3.17}$$

$$p_{1,0,1} = \frac{[1-\delta\nu/(\gamma(\delta+\nu))]c(z_0)}{1 + [1-\delta\nu/(\gamma(\delta+\nu)) + (\lambda+\nu)(\gamma+\nu)(\gamma+\lambda)/(\lambda\nu(\nu+\lambda+\gamma))]\,c(z_0)} \tag{3.18}$$

for $|z| \le 1$, with

$$\begin{aligned}
a(z) &:= -\delta\nu z + (\lambda+\delta)(\lambda+\nu),\\
b(z) &:= \delta\nu z^2 - \gamma(\gamma+\nu+\delta)z + \gamma^2,\\
c(z) &:= \frac{(\gamma - z\,(\gamma+\nu))\,(\nu+\lambda+\gamma)\,a(z)}{((\lambda+\nu)(\gamma+\nu)(\lambda\gamma + (\lambda+\delta)(\lambda+\nu))\,z^2)},\\
z_0 &= \gamma\left(\gamma+\nu+\delta - \sqrt{(\gamma+\nu+\delta)^2 - 4\nu\delta}\right)/(2\nu\delta).
\end{aligned}$$

♦

Proof. We first derive (3.12) and (3.13). From (3.2) and (3.3) we obtain

$$p_{i,0,0} = \left(\frac{\delta\nu}{(\lambda+\delta)(\lambda+\nu)}\right)^{i-1} p_{1,0,0}, \qquad p_{i,1,0} = \left(\frac{\delta\nu}{(\lambda+\delta)(\lambda+\nu)}\right)^{i-1} \frac{\nu}{\lambda+\delta}\,p_{1,0,0}$$

for $i \ge 1$, so that

$$f(z) = \left[\frac{z\,(\lambda+\delta)(\lambda+\nu)}{(\lambda+\delta)(\lambda+\nu) - z\,\delta\nu}\right]p_{1,0,0}, \qquad g(z) = \left[\frac{z\,\nu(\lambda+\nu)}{(\lambda+\delta)(\lambda+\nu) - z\,\delta\nu}\right]p_{1,0,0}. \tag{3.19}$$

From (3.1) and (3.7)–(3.8) we find

$$p_{1,0,0} = \frac{\gamma(\gamma+\nu)}{\nu(\nu+\lambda+\gamma)}\, p_{1,0,1}. \tag{3.20}$$

Introducing (3.20) into (3.19) gives (3.12) and (3.13). On the other hand, we find (3.16) from (3.20) and (3.7).

We now address the computation of $h(z)$ and $k(z)$. From (3.4)–(3.6) and (3.8)–(3.10) we find

$$[z(\gamma+\nu)-\gamma]\, h(z) = z\nu[p_{1,0,1} - z p_{1,0,1^*}] - \gamma[p_{1,0,1} + z^2 p_{1,0,1^*}] + z\lambda f(z) + z^2\delta k(z) \tag{3.21}$$

$$[z(\gamma+\delta)-\gamma]\, k(z) = \gamma(z-1)p_{0,1,1} + z\lambda g(z) + z\nu h(z) - z\nu[p_{1,0,1} - z p_{1,0,1^*}]. \tag{3.22}$$

Substituting $f(z)$, $g(z)$ and $p_{1,0,1^*}$ into (3.21)–(3.22) for their values found in (3.12), (3.13) and (3.16), respectively, defines a linear system of two equations in the unknowns $h(z)$ and $k(z)$. Solving formally for this system of equations yields (3.14) and (3.15). Observe from (3.14) and (3.15) that $h(z)$ and $k(z)$ are well-defined (i.e. analytic) for $|z| \leq 1$ as long as $b(z)$ does not vanish for $|z| \leq 1$. We will come back in a short while on this analyticity issue.

For the time being, let us consider the normalizing condition $F(1,1,1) = 1 = f(1) + g(1) + h(1) + k(1) + p_{1,0,1^*}$. With the help of equations (3.16) and (3.12)–(3.15) we see that the normalizing condition will be satisfied iff.

$$p_{0,1,1} + \frac{(\lambda+\nu)(\gamma+\nu)(\gamma+\lambda)}{\lambda\nu(\nu+\lambda+\gamma)}\, p_{1,0,1} = \frac{\delta\nu}{\delta+\nu}\left(\frac{1}{\nu}+\frac{1}{\delta}-\frac{1}{\gamma}\right)(1-p_{1,0,1}). \tag{3.23}$$

We conclude from (3.23) that the condition

$$\frac{1}{\gamma} < \frac{1}{\nu}+\frac{1}{\delta}, \text{ or equivalently } 1 - \frac{\delta\nu}{\gamma(\delta+\nu)} > 0,$$

is necessary for the Markov process to be stable. Indeed, if $\frac{1}{\gamma} = \frac{1}{\nu}+\frac{1}{\delta}$ then (3.23) only holds if $p_{0,1,1} = p_{1,0,1} = 0$ and the Markov process is not ergodic on the state-space $\mathcal{E}$; if $\frac{1}{\gamma} > \frac{1}{\nu}+\frac{1}{\delta}$ then (3.23) cannot hold if $p_{0,1,1}, p_{1,0,1} > 0$ and again the Markov process is not ergodic on $\mathcal{E}$.

We now come back to the analyticity of $h(z)$ and $k(z)$ in the unit disk. It is easily seen that under condition (3.11) the polynomial $b(z)$ has exactly one zero $z = z_0$ in $|z| \leq 1$ given by

$$z_0 := \gamma\, \frac{\gamma+\nu+\delta - \sqrt{(\gamma+\nu+\delta)^2 - 4\nu\delta}}{2\nu\delta}$$

so that $h(z)$ (resp. $k(z)$) will be well-defined (i.e. analytic) at $z = z_0$ if the coefficient of $1/b(z)$ in (3.14) (resp. in (3.15)) vanishes at this point. This gives rise to two relations

between $p_{0,1,1}$ and $p_{1,0,1}$ that are actually identical and given by (using (3.15))

$$p_{1,0,1} = c(z_0)\, p_{0,1,1}, \quad \text{with } c(z) = \frac{(\gamma - z\,(\gamma+\nu))\,(\nu+\lambda+\gamma)\,a(z)}{(\lambda+\nu)(\gamma+\nu)(\lambda\gamma+(\lambda+\delta)(\lambda+\nu))\,z^2}. \tag{3.24}$$

Solving for the system of linear equations (3.23) and (3.24) in the unknowns $p_{0,1,1}$ and $p_{1,0,1}$ yields (3.17) and (3.18), which completes the calculation of $F(x,y,z)$. Notice that $p_{0,1,1}$ and $p_{1,0,1}$ are strictly positive due to (3.11) and to the fact that $b(\gamma/(\gamma+\nu)) < 0$ which in turn implies that $z_0 < \gamma/(\gamma+\nu)$.

In summary, we have shown that $F(x,y,z)$, as given in Proposition 3.3.1, is analytic in $|z| < 1$ for any value of x and y, continuous in $|z| \leq 1$ for any value of x and y and satisfies the condition $F(1,1,1) = 1$ if (3.11) holds. We can actually find $\epsilon > 0$ such that $F(x,y,z)$ is analytic in $|z| < 1+\epsilon$ for any value of x and y (if z_1 and z_2 denote the zeros of $b(z)$ and $a(z)$, respectively, in $|z| > 1$, then $\epsilon = \min(|z_1|, |z_2|) - 1$). Therefore, we may invoke a classical result on z-transform [81] to conclude that (3.11) is the stability condition and that $F(x,y,z)$ is the z-transform of the stationary probabilities. Note that $\frac{1}{\gamma} < \frac{1}{\nu} + \frac{1}{\delta}$ is not only a necessary condition for stability, but also a sufficient condition since, when it holds, one can find a unique strictly positive and normalized solution to the C-K equations, that is given by the coefficients of the z-transform $F(x,y,z)$. ■

3.3.2.1 The expected communication time

In this section we determine the expected communication time. The communication time is the time needed to a message to reach the mobile agent, to which we must add the time it takes to the mobile agent to send its new position to the source in case the message has to go through at least one forwarder. If the message reaches the mobile agent after exactly one hop then there is no need for the mobile agent to send its position to the source since the source knows it; in this case, the communication terminates once the message has reached the mobile agent, thereby justifying the definition of the communication time given above.

At the end of a communication the agent is not migrating, the source idles and it is one hop away from the agent. This corresponds to state $(1,0,0)$. At this time, the source stays idle for an exponentially distributed duration with mean $1/\lambda$. After this idling period a new communication is initiated and the system can be in any one of the following states: $(1,0,1^*)$, $(i,0,1)$ for $i \geq 2$, or $(i,1,1)$ for $i \geq 1$. Define $T_{i,j,k}$ with $i \geq 1$, $j = 0,1$, $k = 1$ or with $(i,j,k) = (1,0,1^*)$, as the expected communication time given that a message was generated when the system was in state (i,j,k) just after the generation of the message.

The expected communication time T_F (the subscript F refers to "Forwarders") is given

by

$$T_F = q_F(1,0,1^*)\,T_{(1,0,1^*)} + \sum_{i=2}^{\infty} q_F(i,0,1)\,T_{(i,0,1)} + \sum_{i=1}^{\infty} q_F(i,1,1)\,T_{(i,1,1)}$$

where $q_F(i,j,k)$ denotes the probability of reaching state (i,j,k) given that the process initiated in state $(1,0,0)$. It actually represents the probability that a communication starts when the system moves from state $(i,j,0)$ to state (i,j,k). With the help of Figure 3.3 we find that

$$q_F(1,0,1^*) = \frac{\lambda}{\lambda+\nu} \tag{3.25}$$

$$q_F(i,0,1) = \frac{\lambda(\lambda+\delta)}{\delta\nu}\,r^i \quad i = 2,3,\ldots \tag{3.26}$$

$$q_F(i,1,1) = \frac{\lambda}{\delta}\,r^i \quad i = 1,2,\ldots \tag{3.27}$$

where $r := \dfrac{\delta\nu}{(\lambda+\delta)(\lambda+\nu)} < 1$. By using (3.25)–(3.27) we may rewrite T_F as follows

$$T_F = \frac{\lambda}{\lambda+\nu}\,T_{1,0,1^*} + \frac{\lambda(\lambda+\delta)}{\delta\nu}\,\sum_{i=2}^{\infty} r^i\,T_{i,0,1} + \frac{\lambda}{\delta}\,\sum_{i=1}^{\infty} r^i\,T_{i,1,1}. \tag{3.28}$$

With the definitions $G(z) := \sum_{i=0}^{\infty} z^i\,T_{i,0,1}$ and $H(z) := \sum_{i=0}^{\infty} z^i\,T_{i,1,1}$, (3.28) becomes

$$T_F = \frac{\lambda}{\lambda+\nu}T_{1,0,1^*} + \frac{\lambda(\lambda+\delta)}{\delta\nu}\,\left(G(r) - r\,T_{1,0,1} - T_{0,0,1}\right) + \frac{\lambda}{\delta}\,\left(H(r) - T_{0,1,1}\right).$$

We now need to determine the generating functions $G(z)$ and $H(z)$ at $z = r$. To this end we will use the following recursive equations that follow from the Markovian description of the protocol displayed in Figure 3.3:

$$T_{1,0,1^*} = \frac{1}{\nu+\gamma} + \frac{\nu}{\nu+\gamma}\,T_{1,1,1}$$

$$T_{0,0,1} = \frac{1}{\gamma}$$

$$T_{i,0,1} = \frac{1}{\nu+\gamma} + \frac{\nu}{\nu+\gamma}\,T_{i,1,1} + \frac{\gamma}{\nu+\gamma}\,T_{i-1,0,1} \qquad i = 1,2,\ldots$$

$$T_{0,1,1} = \frac{1}{\delta} + T_{1,0,1}$$

$$T_{i,1,1} = \frac{1}{\delta+\gamma} + \frac{\delta}{\delta+\gamma}\,T_{i+1,0,1} + \frac{\gamma}{\delta+\gamma}\,T_{i-1,1,1} \quad i = 1,2,\ldots$$

which yields

$$(\nu + \gamma\,(1-z))\,G(z) - \nu\,H(z) = \frac{1}{1-z} + \frac{\nu}{\gamma} - \nu T_{0,1,1}$$
$$-\delta\,G(z) + (\delta + \gamma\,(1-z))\,zH(z) = \frac{z}{1-z} - \frac{\delta}{\gamma} + \gamma z\,T_{0,1,1}.$$

Solving for $G(z)$ and $H(z)$ gives

$$G(z) = \frac{1}{D(z)}\left(\frac{(\delta+\nu)\,z}{1-z} + \frac{\nu}{\gamma}(1-z)(\gamma z - \delta) + \gamma z + \nu(\gamma z - \delta)\,zT_{0,1,1}\right) \tag{3.29}$$
$$H(z) = \frac{1}{D(z)}\left(\frac{(\delta+\nu)\,z}{1-z} + (\gamma+\delta)\,z - (\gamma^2\,z^2 - \gamma(\gamma+\nu)\,z + \delta\nu)T_{0,1,1}\right) \tag{3.30}$$

where we have defined

$$D(z) := (z-1)\left(\gamma^2\,z^2 - \gamma(\gamma+\nu+\delta)\,z + \delta\nu\right). \tag{3.31}$$

Using (3.29)–(3.31) and after some algebraic manipulations, we find (use $T_{1,0,1^*} = T_{0,1,1} - 1/\delta - 1/(\gamma+\nu)$)

$$\begin{aligned} T_F \;=\; & \frac{1}{\alpha(\lambda)}\Bigg[((\lambda+\delta)(\lambda+\nu) - \gamma\nu)\,T_{0,1,1} \\ & -\frac{(\lambda+\delta)(\lambda+\nu+\delta)}{\delta} - \frac{(\lambda+\delta)(\lambda+\nu)(\lambda(\lambda+\nu)+\nu(\gamma+\nu)) + \delta\gamma\nu\lambda}{\lambda\,(\lambda+\nu)(\gamma+\nu)}\Bigg] \end{aligned} \tag{3.32}$$

where $\alpha(\lambda) := (\lambda+\delta)(\lambda+\nu) - \gamma(\lambda+\nu+\delta)$. It remains to identify the constant $T_{0,1,1}$ in (3.32). This can be done by noticing that $\alpha(\lambda)$ has a single zero $\lambda = \lambda_0$ in $[0,\infty)$ given by

$$\lambda_0 = \frac{\gamma - \nu - \delta + \sqrt{(\gamma+\nu+\delta)^2 - 4\delta\nu}}{2}.$$

In order for T_F to be well-defined for all non-negative values of λ, the coefficient of $1/\alpha(\lambda)$ in (3.32) must vanish when $\lambda = \lambda_0$, which gives us an extra relation from which we can determine $T_{0,1,1}$. Using the identity $(\lambda_0+\delta)(\lambda_0+\nu) = \gamma(\lambda_0+\nu+\delta)$ and setting the coefficient of $1/\alpha(\lambda)$ in (3.32) to 0 when $\lambda = \lambda_0$, gives

$$T_{0,1,1} = \frac{\gamma(\lambda_0+\nu+\delta) + \delta\,\lambda_0}{\gamma\delta\lambda_0}.$$

Finally

$$T_F = \begin{cases} \dfrac{1}{\alpha(\lambda)}\Bigg[((\lambda+\delta)(\lambda+\nu)-\gamma\nu)\,T_{0,1,1} - \dfrac{(\lambda+\delta)(\lambda+\nu+\delta)}{\delta} \\ \qquad\qquad - \dfrac{(\lambda+\delta)(\lambda+\nu)(\lambda(\lambda+\nu)+\nu(\gamma+\nu))+\delta\gamma\nu\lambda}{\lambda(\lambda+\nu)(\gamma+\nu)}\Bigg] & \text{for } \lambda \neq \lambda_0 \\ \dfrac{(2\lambda+\nu+\delta)(2\gamma(\gamma+\nu)(\lambda+\nu+\delta)+\delta\lambda(\lambda+\nu)((\gamma+\nu)\,T_{0,1,1}-1))}{\lambda\delta(2\lambda+\nu+\delta-\gamma)(\lambda+\nu)(\gamma+\nu)} & \\ \qquad - \dfrac{(2\lambda+\nu)(\lambda+\delta)(\lambda+\nu-\gamma(\gamma+\nu)\,T_{0,1,1})+\gamma\nu\delta}{\lambda(2\lambda+\nu+\delta-\gamma)(\lambda+\nu)(\gamma+\nu)} & \text{for } \lambda = \lambda_0 \end{cases} \tag{3.33}$$

where the latter relation is obtained after a routine application of l'Hôpital's rule.

3.3.2.2 The expected number of forwarders

In this section we compute the expected number of forwarders in steady-state between the agent and the source. Let $q(i)$ be the probability that the agent is located $i \geq 1$ hops away from the source, which corresponds to a situation where there are exactly $i-1$ forwarders in the system. Clearly,

$$q(i) = p_{i,0,0} + p_{i,1,0} + p_{i,0,1} + p_{i,1,1} + \mathbf{1}\{i=1\}p_{1,0,1^*}, \qquad i = 1, 2, \ldots$$

and the expected number N_s of forwarders (the subscript s refers to "source", which is the starting point of the count of the number of forwarders) is given by

$$\begin{aligned} N_s &= \sum_{i=1}^{\infty}(i-1)\,q(i) = \sum_{i=1}^{\infty} i\,q(i) - \sum_{i=1}^{\infty} q(i) \\ &= f'(1) + g'(1) + h'(1) + k'(1) + p_{1,0,1^*} - (1 - p_{0,0,1} - p_{0,1,1}) \end{aligned} \tag{3.34}$$

where $\phi'(1)$ denotes the derivative of $\phi(z)$ at point $z = 1$. From (3.12) and (3.13) we find

$$f'(1) + g'(1) = \frac{\gamma(\lambda+\delta)(\lambda+\nu)^2(\gamma+\nu)}{\lambda^2\nu(\nu+\lambda+\gamma)(\lambda+\delta+\nu)}\,p_{1,0,1} \tag{3.35}$$

whereas the relation

$$\begin{aligned} h'(1) + k'(1) &= \frac{\delta\nu(\delta\gamma+\nu^2) - (\nu^2\delta(\nu-\gamma) + \gamma(\nu+\delta)(\delta\gamma-\nu^2))p_{1,0,1}}{\nu(\delta+\nu)(\gamma(\delta+\nu)-\delta\nu)} \\ &\quad + \frac{\delta\gamma(\gamma+\nu)(\delta\lambda(\nu-\gamma)+\nu(\lambda+\nu)(\lambda+\delta))}{\lambda^2(\nu+\lambda+\gamma)(\lambda+\nu+\delta)(\gamma(\delta+\nu)-\delta\nu)}p_{1,0,1} \end{aligned} \tag{3.36}$$

follows from (3.14) and (3.15). Combining now (3.34), (3.35) and (3.36) with the expressions found for $p_{0,0,1}$, $p_{1,0,1^*}$ and $p_{0,1,1}$ (in (3.8), (3.16) and (3.23), respectively), we find

$$N_s = \frac{\delta^2(\gamma^2 - \gamma\nu + \nu^2)(1 - p_{1.0.1})}{\gamma(\delta+\nu)(\gamma(\delta+\nu) - \delta\nu)} + \left(\frac{\nu\gamma(\gamma+\lambda)(\gamma+\nu)}{\lambda^2(\nu+\lambda+\gamma)} - \frac{\gamma^2-\nu^2}{\nu}\right)\frac{\delta\, p_{1,0,1}}{\gamma(\delta+\nu)-\delta\nu} \quad (3.37)$$

where $p_{1,0,1}$ is given in (3.18). As shown in [20] N_s can be used to evaluate the fault-tolerance of the protocol. This issue is not addressed here.

N_s represents the expected number of forwarders through which a message would go on its way to reach the agent *if* the agent *does not* migrate in the meanwhile. We will compute now the expected number of forwarders, denoted by N, through which a message has to go in order to reach the agent. A similar analysis to the one conducted in Section 3.3.2.1 has to be done. Note that we should have $N > N_s$ as the number of forwarders between the message and the agent may increase over time.

When a communication starts, a message is generated and the system can be in any one of the following states: $(1, 0, 1^*)$, $(i, 0, 1)$ for $i \geq 2$, or $(i, 1, 1)$ for $i \geq 1$. If the number of forwarders is n at the beginning of a communication, this number decreases as the message travels towards the agent but it can increase if the agent migrates in the meanwhile. Define $N_{i,j,k}$ with $i \geq 1$, $j = 0, 1$, $k = 1$ or with $(i, j, k) = (1, 0, 1^*)$, as the expected number of forwarders given that a message was generated when the system was in state (i, j, k) just after the generation of the message.

The expected number of forwarders N is given by

$$N = q_F(1,0,1^*)\, N_{(1,0,1^*)} + \sum_{i=2}^{\infty} q_F(i,0,1)\, N_{(i,0,1)} + \sum_{i=1}^{\infty} q_F(i,1,1)\, N_{(i,1,1)}$$

where $q_F(i, j, k)$ denotes the probability of reaching state (i, j, k) given that the process was initially in state $(1, 0, 0)$ (probability introduced in Section 3.3.2.1). It actually represents the probability that a communication starts when the system moves from state $(i, j, 0)$ to state (i, j, k). The expressions of $q_F(i, j, k)$ are given in (3.25)–(3.27). Using them, we may rewrite N as follows

$$N = \frac{\lambda}{\lambda+\nu}\, N_{1,0,1^*} + \frac{\lambda(\lambda+\delta)}{\delta\nu} \sum_{i=2}^{\infty} r^i\, N_{i,0,1} + \frac{\lambda}{\delta} \sum_{i=1}^{\infty} r^i\, N_{i,1,1}. \quad (3.38)$$

With the definitions $N_0(z) := \sum_{i=0}^{\infty} z^i\, N_{i,0,1}$ and $N_1(z) := \sum_{i=0}^{\infty} z^i\, N_{i,1,1}$, (3.38) becomes

$$N = \frac{\lambda}{\lambda+\nu} N_{1,0,1^*} + \frac{\lambda(\lambda+\delta)}{\delta\nu}\, (N_0(r) - r\, N_{1,0,1} - N_{0,0,1}) + \frac{\lambda}{\delta}\, (N_1(r) - N_{0,1,1})\,. \quad (3.39)$$

It remains to determine the generating functions $N_0(z)$ and $N_1(z)$ at $z = r$. To this end, we will use the following recursive equations that follow from the Markovian description of the

protocol displayed in Figure 3.3:

$$N_{1,0,1^*} = \frac{\nu}{\nu+\gamma}\, N_{1,1,1} \tag{3.40}$$

$$N_{0,0,1} = 0 \tag{3.41}$$

$$N_{1,0,1} = \frac{\nu}{\nu+\gamma}\, N_{1,1,1} + \frac{\gamma}{\nu+\gamma}\, N_{0,0,1} \tag{3.42}$$

$$N_{i,0,1} = \frac{\nu}{\nu+\gamma}\, N_{i,1,1} + \frac{\gamma}{\nu+\gamma}\,(1 + N_{i-1,0,1}) \qquad i = 2, 3, \ldots \tag{3.43}$$

$$N_{0,1,1} = N_{1,0,1} \tag{3.44}$$

$$N_{i,1,1} = \frac{\delta}{\delta+\gamma}\, N_{i+1,0,1} + \frac{\gamma}{\delta+\gamma}\,(1 + N_{i-1,1,1}) \quad i = 1, 2, \ldots. \tag{3.45}$$

Notice from Equations (3.40)–(3.42) and (3.44) that

$$N_{1,0,1^*} = N_{1,0,1} = N_{0,1,1}. \tag{3.46}$$

After some algebraic manipulations on Equations (3.40)–(3.45), we obtain the following linear system of equations in the unknowns $N_0(z)$ and $N_1(z)$

$$\begin{aligned}
&(\nu + \gamma\,(1-z))\, N_0(z) - \nu\, N_1(z) = \frac{\gamma z^2}{1-z} - \nu N_{1,0,1} \\
&-\delta\, N_0(z) + (\delta + \gamma\,(1-z))\, z N_1(z) = \frac{\gamma z^2}{1-z} + \gamma z\, N_{1,0,1}.
\end{aligned}$$

Solving for $N_0(z)$ and $N_1(z)$ gives

$$N_0(z) = \frac{1}{D(z)}\left(\frac{\gamma\, z^2}{z-1}\Big(\gamma z^2 - (\gamma+\delta)z - \nu\Big) + \nu(\gamma z - \delta)\, z N_{1,0,1}\right) \tag{3.47}$$

$$N_1(z) = \frac{1}{D(z)}\left(\frac{\gamma\, z^2}{z-1}\Big(\gamma z - (\gamma+\delta+\nu)\Big) - (\gamma^2\, z^2 - \gamma(\gamma+\nu)\, z + \delta\nu) N_{1,0,1}\right) \tag{3.48}$$

where $D(z)$ is given in (3.31). Using (3.47)–(3.48) together with (3.46) and (3.31), (3.39) can be rewritten as follows:

$$N = \frac{((\lambda+\delta)(\lambda+\nu) - \gamma\nu) N_{1,0,1} - \nu\gamma(\lambda+\delta)/\lambda}{(\lambda+\delta)(\lambda+\nu) - \gamma(\lambda+\nu+\delta)} \tag{3.49}$$

where the denominator is nothing but the polynomial $\alpha(\lambda)$ introduced in the previous section. To identify the constant $N_{1,0,1}$ in (3.49), we proceed as in Section 3.3.2.1. In order for N to be well-defined for all non-negative values of λ, the numerator in (3.49) must vanish when $\lambda = \lambda_0 = 1/2 \times \left(\gamma - \nu - \delta + \sqrt{(\gamma+\nu+\delta)^2 - 4\delta\nu}\right)$, which is the only positive zero

of the denominator $\alpha(\lambda)$ (see Section 3.3.2.1). It is readily seen that $N_{1,0,1} = \nu/\lambda_0$ (Note: use the relation $(\lambda_0 + \delta)(\lambda_0 + \nu) = \gamma(\lambda_0 + \nu + \delta)$). Finally

$$N = \begin{cases} \dfrac{((\lambda+\delta)(\lambda+\nu) - \gamma\nu)\nu/\lambda_0 - \nu\gamma(\lambda+\delta)/\lambda}{(\lambda+\delta)(\lambda+\nu) - \gamma(\lambda+\nu+\delta)} & \text{for } \lambda \neq \lambda_0, \\ \dfrac{\nu(2\lambda_0^2 + (\nu+\delta)\lambda_0 + \gamma\delta)}{\lambda_0^2(2\lambda_0 + \nu + \delta - \gamma)} & \text{for } \lambda = \lambda_0, \end{cases} \tag{3.50}$$

where the latter relation is obtained after a routine application of l'Hôpital's rule.

Both expected numbers computed in this section (i.e. N_s and N) are expected to be increasing functions of δ and ν and decreasing functions of λ and γ which is illustrated in Figure 3.4. Looking at the upper graphs in Figure 3.4 in which the evolution of both N and N_s are plotted against λ (upper graph at the left) and ν (upper graph at the right), we can verify that $N > N_s$. The crosses in each graph indicate the value of N (the expected number of forwarders that a message encounters) corresponding to the same values of the model parameters: $\lambda = 1, \nu = 10, \delta = 20, \gamma = 50$. In Figure 3.4, the plots for $\delta = 30s^{-1}, \delta = 40s^{-1}$ (lower graph at the left) and $\gamma = 9s^{-1}$ (lower graph at the right) depict the behavior of our model when the ergodicity condition is close to being violated. In that situation, the expected number of forwarders grows to infinity.

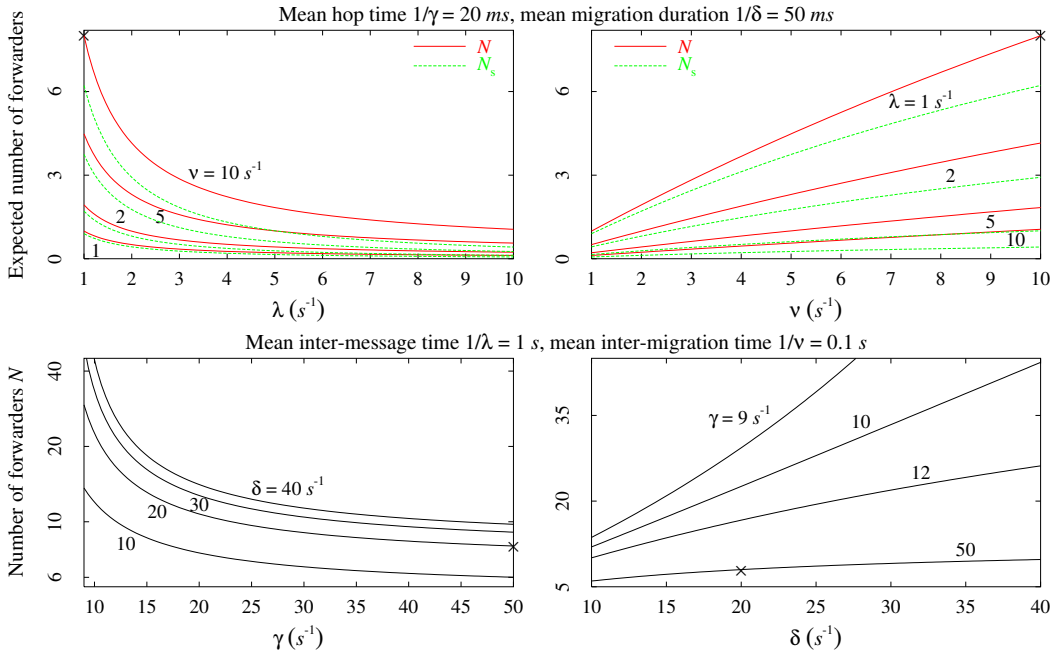


Figure 3.4: The expected number of forwarders

3.4 Centralized Server

3.4.1 Description

An alternative to the forwarders approach for locating a mobile agent is to use a *location server*. Such a server keeps track of the location of mobile agents in a database. Servers like this are widely used in the Internet. For instance, the Domain Name Server [85] uses a hierarchically organized servers to associate a location (IP address) to a symbolic name. For sake of simplicity, we will consider here a single *centralized* server, although many different schemes can be conceived to improve speed and reliability. We will also assume that each object (source or mobile agent) knows the location of this server.

The idea behind the location server is simple: each time a mobile agent migrates, it informs the server of its new location. Whenever the source wants to reach the mobile agent, it sends a message to the last known location of the mobile agent; if this communication fails, then the source sends a `location request` to the server. This solution is often referred to as a lazy solution since the references to a mobile agent are only updated when needed. We now give a careful description of the protocol used by the source and the mobile agent to communicate with the server:

- The Mobile Agent

  Step 1: Performs the migration;

  Step 2: Sends its new location to the server.

- The Source

  Step 1: Issues a message to the mobile agent with the recorded location. Upon failure goes to Step 2;

  Step 2: Queries the server to have the current location of the mobile agent;

  Step 3: Issues a message to the mobile agent with the location provided by the server. Upon failure, returns to Step 2.

The above protocol is implemented in various Java libraries (*MOA* [84] and *ProActive* [101]). An illustration of this approach is given in Figure 3.5. In Figures 3.5(b)-(c) the chronological order of the events is indicated by the numbers $1, 2, \ldots$. In Figure 3.5(b) a migration (events no. 1 & 2) takes place before a message has been sent by the source. When the source finally sends a message (event no. 3) to the agent at Host B (last known position of the agent), it receives a communication error from Host B (event no. 4). The source then asks the home site to give it the location of the agent (event no. 5). Once the source knows the new location of the agent (event no. 6), it sends a copy (event no. 7) of the original message to Host C (i.e. to the agent). The communication is successful and the source becomes idle.

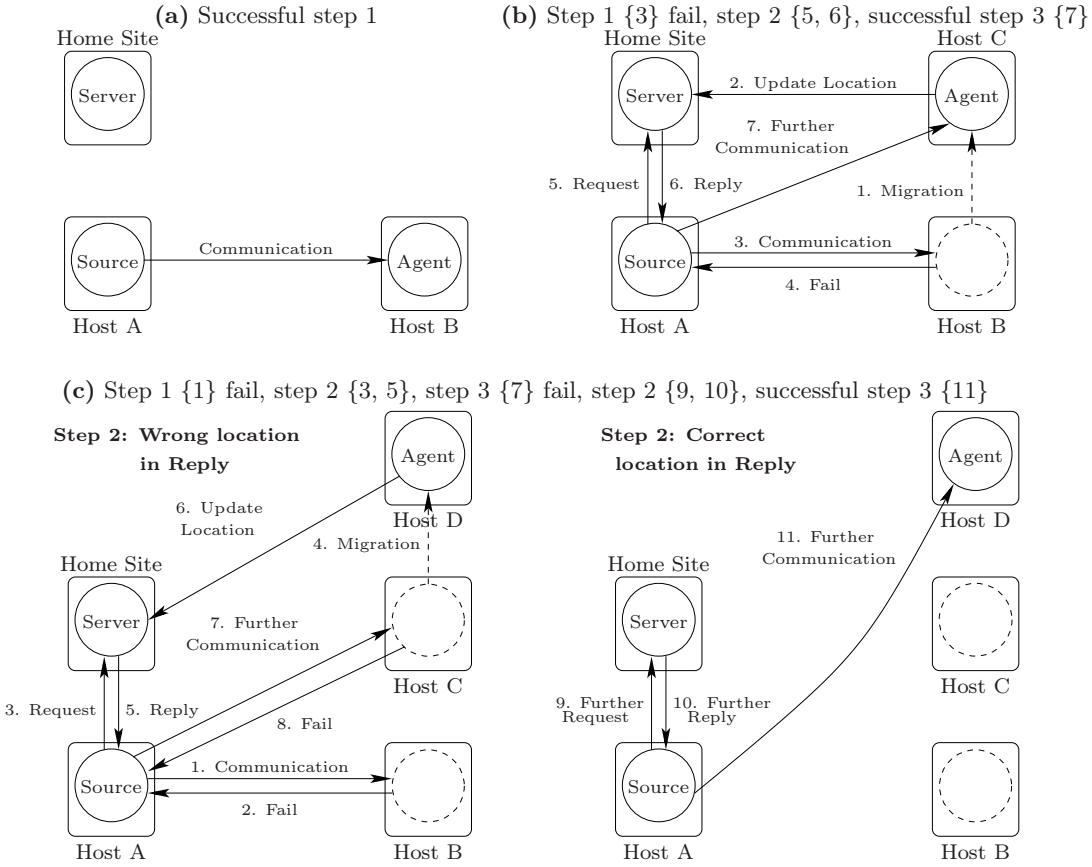


Figure 3.5: Some possible scenarios in the centralized approach from the source point of view

Figure 3.5(c) displays a more complicated situation. In this case, the server does not give the correct location of the agent to the source since the agent has initiated a migration in the meantime. As a result, the source will have to send a second location request to the server (event no. 9) before finally being able to reach the agent (event no. 11).

Regarding the service policy, there exist several different schemes that can be implemented. In *ProActive* [101], the performance of the server has been optimized so that it does not act as a bottleneck. The buffer at the server is completely partitioned in the sense that each source-agent pair possesses its own queue. The server cyclically polls these different queues, serving one request per queue in each cycle. Within each queue, there is a priority scheduling mechanism that gives priority to `update requests` over `location requests`. If a queue contains several `update requests` sent by the same mobile agent then only the newest is processed and the others are destroyed. Note that the service policy is non-preemptive since *ProActive* is a Java library and that the execution of a Java method cannot be stopped. As a consequence an `update request` under processing cannot be preempted by a more recent one, which may harm the performance of the protocol[2]. For-

[2]Notice that only the queue that is attended by the server may have two `update requests`, all other

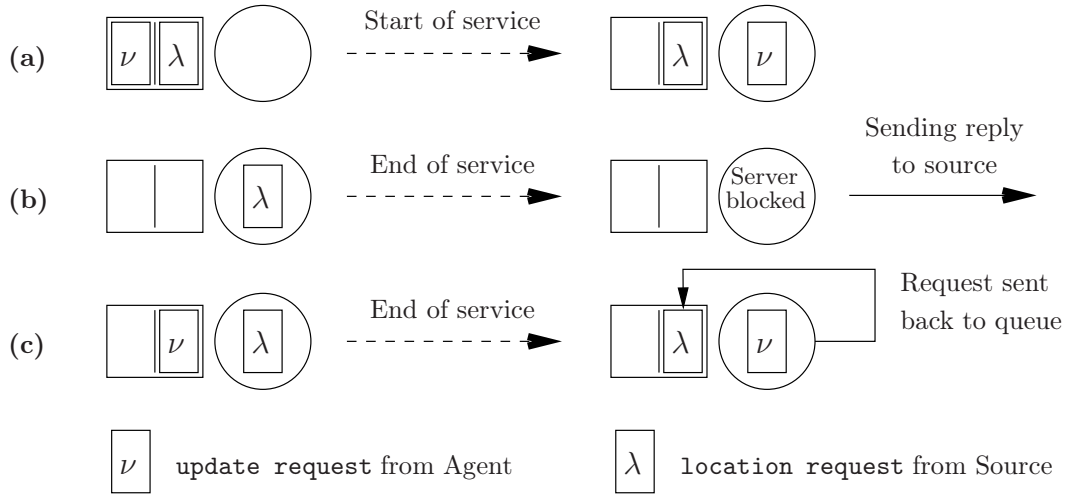


Figure 3.6: Details on the service policy at the server

tunately, it has been observed that this event is very rare in practice. Another consequence of this scheme is that upon serving a `location request`, the server has to check whether the corresponding queue contains an `update request` coming from the mobile agent; if this is the case, then the `location request` is sent back to the queue; otherwise, it sends the position of this mobile agent (as found in the database) to the source. Finally, communications in *ProActive* between the server and the different sources are synchronous. Therefore, a source that has sent a `location request` cannot perform any other task before it gets a reply from the server, which implies there is at most one pending `location request` per source at the server.

Figure 3.6 illustrates some scenarios at the server. In Figure 3.6(a), two requests are queued in the buffer. The `update request` is served with priority even though the `location request` is older. In Figure 3.6(b), the server is blocked after serving a `location request`, until the reply reaches the source. Figure 3.6(c) illustrates one inconvenient of using a Java library. Since the service policy is non-preemptive, a `location request` that is already served, is sent back to the queue to be served again whenever an `update request` is waiting in the buffer.

Observe that the server delivers a customized service to each type of queries: for `update requests` it just updates the agent location in a database, whereas for `location requests` it scans the database for the current agent location and further scans the queue for any `update request`.

3.4.2 A Markovian analysis of the server

An exact modeling of the centralized approach would consist of modeling the system as

ones having at most one single pending `update request`.

a nonexhaustive cyclic server with vacations and finite buffers at the queues. In this section, we develop an approximate model that considers a single queue – thereby a single source communicating with a single agent – where the processing speed of the server is reduced to account for the contention due to the potential presence of several sources and/or agents. This queue may have at most a single `location request` and two `update requests`. By arguing that it is highly unlikely that an arriving `update request` finds an `update request` (from the same agent) being processed, we will assume that there can be at most one pending `update request` at the queue or, in other words, that an `update request` under processing is preempted by a more recent one (cf. Section 3.4.1). The latter restriction can easily been removed at the expanse of enlarging the state-space (which would yield a 29.6% increase in the number of states).

We assume that the set of assumptions **A1** holds (cf. Section 3.2). Note, however, that in this context p_j will represent the sum of the jth traveling time of the agent to its new host and of the travel time of the associated `update request` to the server site. We further assume that the traveling times between the source and the presumed location of the mobile agent (resp. between the source and the location server) are i.i.d. exponentially distributed RVs with parameter $\gamma_1 > 0$ (resp $\gamma_2 > 0$). Finally, we assume that the service times – regardless of the query type – are i.i.d. exponentially distributed RVs with parameter $\mu > 0$ (notice that if there is only one source-agent pair in an application, μ would be the server processing speed). All of these RVs are assumed to be mutually independent.

We model the system behavior by a finite-state Markov process whose transition diagram and rates are given in Figure 3.7. A state has the representation $(\mathbf{i}, j, k)$ with $\mathbf{i} \in \{A, B, \ldots, G\}$, $j \in \{0, 0^*, 1, 1^*\}$ and $k \in \{0, 1, 1^*, 2\}$, where the 2-dimensional vectors $A, B, \ldots, G$ are defined in Figure 3.7.

More precisely, the vector $\mathbf{i} = (i_1, i_2)$ represents the state of a queue at the server, namely, the type of the message (`update request` or `location request`) in the queue, with $i_l \in \{0, \lambda, \nu\}$ the type of the message that occupies the lth position in the queue ($l = 1, 2$). By convention, $i_l = 0$ (resp. $i_l = \lambda$, $i_l = \nu$) indicates that the lth position is not occupied (resp. the lth position is occupied by a `location request`, the lth position is occupied by an `update request`). Recall that `update requests` have non-preemptive priority over a `location request` and that an arriving `update request` that finds one `update request` will destroy it at once.

The component $j \in \{0, 0^*, 1, 1^*\}$ in the state description $(\mathbf{i}, j, k)$ represents the state of the mobile agent: $j = 1$ (resp. $j = 1^*$) will indicate that the mobile agent is migrating and that the source knows (resp. does not know) the location of the host that the mobile agent is leaving; similarly, $j = 0$ (resp. $j = 0^*$) will indicate that the mobile agent is not migrating and that the source knows (resp. does not know) its location.

Finally, the component $k \in \{0, 1, 1^*, 2\}$ in the state description $(\mathbf{i}, j, k)$ represents the state of the source: $k = 0$ if the source has no activity, $k = 1$ if it has sent a message to the

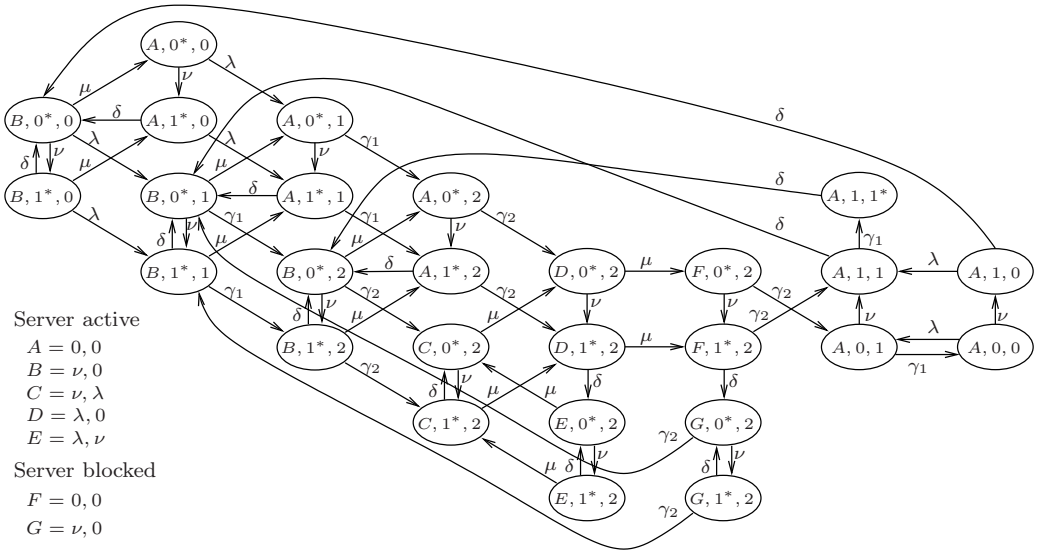


Figure 3.7: System states and transition rates in the centralized approach

mobile agent, $k = 1^*$ if the message sent by the source has reached a site that the mobile agent is about to leave, and $k = 2$ if it has sent a `location request` to the server. The latter only occurs if the presumed location of the mobile agent is no longer valid.

The system enters state $(A, 0, 0)$ just after the end of a communication (defined as the instant when a message has reached the agent). It remains in that state for an exponentially distributed duration with mean $1/\lambda$, and then a new message is generated by the source. The time that elapses between the generation of a new message by the source and the next visit to state $(A, 0, 0)$ is the *communication time* (i.e. quantities τ_i's introduced in Section 3.2). In other words, the successive communication times $\{\tau_i\}_i$ are initialized when k goes from 0 to 1, and are stopped when k goes from 1 to 0. Each time a communication fails, a request is issued to the server and k switches from 1 to 2. As soon as the server replies, k switches back to 1 and the message is re-issued by the source to the location of the mobile agent returned by the server (which may or may not be the current location).

Under the above description/assumptions, the process depicted in Figure 3.7 is an irreducible finite-state Markov process on the state-space $\mathcal{F}$, where $\mathcal{F}$ is the set of all states indicated in Figure 3.7 ($\mathcal{F}$ contains 27 elements). Let $\mathbf{p} = \{p_{\mathbf{i},j,k},\ (\mathbf{i}, j, k) \in \mathcal{F}\}$ be the stationary probability of this Markov process ($\mathbf{p}$ exists since an irreducible finite-state Markov process is ergodic). If $\mathbf{Q}$ denotes the infinitesimal generator of this Markov process (whose elements can easily be identified from Figure 3.7), then we know (see e.g. [75]) that $\mathbf{p}$ is the unique solution of the system of linear equations $\mathbf{p} \cdot \mathbf{Q} = 0$, $\mathbf{p} \cdot \mathbf{1} = 1$, which can be solved by using a routine numerical procedure.

3.4.2.1 The expected communication time

We are interested in finding the expected communication time (see Section 3.4.2), denoted as T_S (the subscript S refers to "Server"). Recall that a communication begins when the source – after an idling period with mean $1/\lambda$ – sends a message to the mobile agent and terminates when the latter is reached. It is seen from Figure 3.7 that a message may only be generated when the system is in 6 distinct states – the states where $k = 0$. As soon as $k = 1$ a message is generated. Hence, a communication may start only when the system is in one of the following states: $(A, 0, 1)$, $(A, 1, 1)$, $(A, 0^*, 1)$, $(A, 1^*, 1)$, $(B, 0^*, 1)$ or $(B, 1^*, 1)$.

Let $T_{\mathbf{i},j,k}$ denote the expected time to hit state $(A, 0, 0)$ starting from state $(\mathbf{i}, j, k)$. The expected response time T_S of the system is given by

$$T_S = \sum_{j=0,1} q_S(A, j, 1)\, T_{A,j,1} + q_S(A, j^*, 1)\, T_{A,j^*,1} + q_S(B, j^*, 1)\, T_{B,j^*,1} \tag{3.51}$$

where $q_S(\mathbf{i}, j, 1)$ denotes the probability that the communication is initiated when the system is in state $(\mathbf{i}, j, 1)$. With Figure 3.7 it is easily seen that

$$q_S(A, 0, 1) = \frac{\lambda}{\lambda + \nu} \tag{3.52}$$

$$q_S(A, 1, 1) = \frac{\nu\lambda}{(\lambda + \delta)(\lambda + \nu)} \tag{3.53}$$

We will next compute $q_S(v)$, the probability that a transmission is initiated in state $v \in \mathcal{V}$, where

$$\mathcal{V} := \{(A, 0^*, 1), (A, 1^*, 1), (B, 0^*, 1), (B, 1^*, 1)\}.$$

Let $p(w; v)$ be the probability that a communication is initiated in state $v \in \mathcal{V}$ given that the system is in state $w \in \mathcal{V} \cup \{(A, 0^*, 0), (A, 1^*, 0), (B, 0^*, 0), (B, 1^*, 0)\}$. We see from Figure 3.7 that

$$q_S(v) = \frac{\delta\nu}{(\lambda + \delta)(\lambda + \nu)}\, P((B, 0^*, 0); v) \tag{3.54}$$

for $v \in \mathcal{V}$. We now need to compute $P((B, 0^*, 0); v)$ for $v \in \mathcal{V}$. The following linear relations readily derive from Figure 3.7:

$$\begin{aligned} p((A, 0^*, 0); v) &= \frac{\lambda}{\lambda + \nu}\, p((A, 0^*, 1); v) + \frac{\nu}{\lambda + \nu}\, p((A, 1^*, 0); v) \\ p((A, 1^*, 0); v) &= \frac{\lambda}{\lambda + \delta}\, p((A, 1^*, 1); v) + \frac{\delta}{\lambda + \delta}\, p((B, 0^*, 0); v) \end{aligned}$$

$$
\begin{aligned}
p((B,0^*,0);v) &= \frac{\lambda}{\lambda+\nu+\mu}\,p((B,0^*,1);v) + \frac{\nu}{\lambda+\nu+\mu}\,p((B,1^*,0);v) \\
&\quad + \frac{\mu}{\lambda+\nu+\mu}\,p((A,0^*,0);v) \\
p((B,1^*,0);v) &= \frac{\lambda}{\lambda+\delta+\mu}\,p((B,1^*,1);v) + \frac{\delta}{\lambda+\delta+\mu}\,p((B,0^*,0);v) \\
&\quad + \frac{\mu}{\lambda+\delta+\mu}\,p((A,1^*,0);v)
\end{aligned}
$$

for all $v \in \mathcal{V}$. Rearranging these equations give, in matrix form,

$$
\begin{bmatrix}
\lambda+\nu & -\nu & 0 & 0 \\
0 & \lambda+\delta & -\delta & 0 \\
-\mu & 0 & \lambda+\nu+\mu & -\nu \\
0 & -\mu & -\delta & \lambda+\delta+\mu
\end{bmatrix}
\begin{bmatrix}
p((A,0^*,0);v) \\ p((A,1^*,0);v) \\ p((B,0^*,0);v) \\ p((B,1^*,0);v)
\end{bmatrix}
= \lambda
\begin{bmatrix}
p((A,0^*,1);v) \\ p((A,1^*,1);v) \\ p((B,0^*,1);v) \\ p((B,1^*,1);v)
\end{bmatrix}
\tag{3.55}
$$

for all $v \in \mathcal{V}$. All terms in the r.h.s. of (3.55) are either 0 or 1, as shown in Table 3.1 (rows 2–5). From rows 2–5 in Table 3.1 and (3.55), we can compute $p((B,0^*,0),v)$ for all $v \in \mathcal{V}$ (see last row of Table 3.1).

Table 3.1: Values of the probabilities, with $K := (\lambda+\nu+\mu)(\lambda+\delta+\nu)$

v	$(A,0^*,1)$	$(A,1^*,1)$	$(B,0^*,1)$	$(B,1^*,1)$
$p((A,0^*,1);v)$	1	0	0	0
$p((A,1^*,1);v)$	0	1	0	0
$p((B,0^*,1);v)$	0	0	1	0
$p((B,1^*,1);v)$	0	0	0	1
$p((B,0^*,0);v)$	$\frac{(\lambda+\delta)\mu}{K}$	$\frac{(2\lambda+\delta+\mu+\nu)\nu\mu}{(\mu+\lambda+\delta)K}$	$\frac{(\lambda+\nu)(\lambda+\delta)}{K}$	$\frac{(\lambda+\nu)(\lambda+\delta)\nu}{(\mu+\lambda+\delta)K}$

Substituting the resulting values of $p((B,0^*,0);v)$ into (3.54) yields

$$
q_S(A,0^*,1) = \frac{\nu\mu\delta}{(\lambda+\nu)(\lambda+\nu+\mu)(\lambda+\delta+\nu)} \tag{3.56}
$$

$$
q_S(A,1^*,1) = \frac{\nu^2\mu\delta(2\lambda+\delta+\mu+\nu)}{(\lambda+\delta)(\lambda+\nu)(\lambda+\nu+\mu)(\lambda+\delta+\nu)(\mu+\lambda+\delta)} \tag{3.57}
$$

$$
q_S(B,0^*,1) = \frac{\nu\delta}{(\lambda+\nu+\mu)(\lambda+\delta+\nu)} \tag{3.58}
$$

$$
q_S(B,1^*,1) = \frac{\nu^2\delta}{(\lambda+\nu+\mu)(\lambda+\delta+\nu)(\mu+\lambda+\delta)}. \tag{3.59}
$$

It remains to compute the hitting times $T_{\mathbf{i},j,k}$. They are easily identified from the infinitesimal generator matrix $\mathbf{Q}$ as follows (see Theorem 3.3.3 pages 113-114 in [89])

$$T_{A,0,0} = 0, \qquad \sum_{\mathbf{i},j,k} q_{(\mathbf{i}',j',k'),(\mathbf{i},j,k)} T_{\mathbf{i},j,k} = -1 \quad \text{for } (\mathbf{i}',j',k') \neq (A,0,0) \tag{3.60}$$

where $q_{(\mathbf{i}',j',k'),(\mathbf{i},j,k)}$ are the elements of the generator matrix $\mathbf{Q}$. In order to simplify (3.60), let us introduce $\mathbf{M}_{A,0,0}$ as the minor of the matrix $\mathbf{Q}$ obtained by removing the row and the column corresponding to state $(A,0,0)$. Let $\mathbf{T} = \{T_{\mathbf{i},j,k},\ (\mathbf{i},j,k) \in \mathcal{F} - \{(A,0,0)\}\}$ be the vector of hitting times, except $T_{A,0,0}$ (which is null). Equation (3.60) then reads

$$\mathbf{M}_{A,0,0} \cdot \mathbf{T} = -\mathbf{1}. \tag{3.61}$$

Solving (3.61) and using (3.52),(3.53) and (3.56)–(3.59), we finally find T_S.

3.5 Validation and comparison

3.5.1 Validation through simulations

The theory developed in Sections 3.3 and 3.4 relies on several assumptions. Mainly, idle times for a source and for an agent, migration durations, traveling times of messages in the network and service times (centralized approach only) were all assumed to be exponential RVs and independent from each other.

In this section we undertake validation of the Markovian models presented in Sections 3.3 and 3.4 against results obtained from event-driven simulations of both schemes. In the simulations, we have considered a single agent and a single source. We have run several simulations that explore the effects of violations of one assumption at a time and the violations of all of the assumptions. This way we can observe the robustness of the corresponding model and the impact of each assumption on its performance.

In order to test realistic distributions for the RVs at hand, we have collected measurements of the travel times, the migration duration and the service times on a LAN and a MAN and fitted the resulting data to well-known distributions. Table 3.2 summarizes our findings. Except for the service times which are approximately constant, all other (network-dependent) parameters are well represented by Weibull distributions. The distribution of the communication rate λ (inverse of the average idle time for the source) and the migration rate ν (inverse of the average idle time for the agent) depends only on the application. To test the robustness of the models against each assumption, we have run simulations where all random variables are exponential except one whose distribution is either deterministic (in the case of idle times and service times) or Weibull (in the case of migration durations and travel times). Last, we simulate the systems where all random variables are non-exponential. In these runs, the only assumptions not violated are the ones concerning the independence of the processes at hand.

Table 3.2: Distribution fits for the model parameters

Random variable (ms)	100Mb/s switched LAN	7Mb/s MAN
Forwarders		
migration durations	Weibull shape 4, scale 100	Weibull shape 2.5, scale 392
travel times	Weibull shape 11, scale 23	Weibull shape 6, scale 88
Server		
migration durations	Weibull shape 2.5, scale 75.5	Weibull shape 3, scale 1010
travel times source-agent	Weibull shape 1.8, scale 9.7	Weibull shape 10, scale 28.6
travel times source-server	Weibull shape 1.8, scale 14.7	Weibull shape 1.8, scale 93
service times	$\approx$ constant	$\approx$ constant

Table 3.3: Sample mean and percentiles of the relative error provided by the models. Default values: $\lambda = 1, \nu = 10, \delta = 11$ (forwarders), $\delta = 15$ (server), $\gamma = 45, \gamma_1 = 115, \gamma_2 = 75, \mu = 2325$

Simulations	Mean	25	50	75	90	95
Forwarders						
deterministic source idle times$\lambda \in [1, 10]$	7.7	6.5	8.8	9.6	10.3	10.5
deterministic agent idle times$\nu \in [1, 10]$	1.6	0.3	1.4	2.4	3.5	4.2
Weibull migration times $\delta \in [8, 25]$ shape 4	8.3	4.3	8.2	10.4	14.7	16.3
Weibull travel times $\gamma \in [33.8, 69.8]$ shape 11	2.7	1.0	2.2	3.9	6.4	7.1
All together $\lambda, \nu \in \{1, 3, 5, 7, 9\}$	14.1	4.9	10.0	20.6	32.4	38.3
Server						
deterministic source idle times$\lambda \in [1, 10]$	14.9	15.1	16.3	17.1	17.1	17.2
deterministic agent idle times $\nu \in [1, 10]$	2.4	2.2	2.2	2.6	4.1	4.8
Weibull migration times $\delta \in [12, 19]$ shape 2.5	12.3	11.7	12.2	13.5	14.8	15.5
Weibull travel times $\gamma_1 \in [90, 131]$ shape 1.8	1.3	0.6	1.5	1.8	2.1	2.3
Weibull travel times $\gamma_2 \in [56, 94]$ shape 1.8	0.9	0.3	0.6	1.3	2.2	2.7
deterministic service times $\mu \in [500, 2500]$	1.5	0.6	1.5	2.2	2.9	3.5
All together $\lambda, \nu \in \{1, 3, 5, 7, 9\}$	14.1	6.1	10.5	17.0	30.0	40.1

Table 3.3 reports the sample mean and the percentiles of the relative error (expressed in percentage) between simulated results and theoretical expected response times as predicted by both models. The values in Table 3.3 are obtained after 3000 seconds of simulation (the steady-state is reached after 1000 seconds of simulation run, see [64] for more details on the convergence issue). It appears that both models are very robust against Weibull travel times. In 95% of the simulations conducted, the relative error on the communication time stays under 7.1% (resp. 2.3% and 2.7%) in the forwarders mechanism (resp. centralized mechanism) (see row 6 (resp. rows 12 and 13) in Table 3.3).

Neither model is sensitive to the distribution of agent idle times. In 95% of the simulations conducted, the relative error on the communication time stays within 4.2% (resp. 4.8%) of the simulated value in the forwarders mechanism (resp. centralized mechanism) (see

row 4 (resp. row 10) in Table 3.3). The model of the location server is very robust against deterministic service times. The largest error observed during the simulations is 3.7%.

However, the models are more sensitive to the distribution of the migration durations and the source idle times. Nevertheless, when simulating the forwarders scheme the mean relative error is 7.7% (resp. 8.3%) for the case that the source idle times are deterministic (resp. the migration durations are Weibull) (see column 2 in Table 3.3 row 3 (resp. row 5)). When simulating the centralized approach, we observe almost the same relative error when the source idle times are deterministic (resp. the migration durations are Weibull), the largest error being 17.2% (resp. 16%).

When all of the assumptions concerning the distribution of the RVs are violated, the performance of the models is fair. In half of the simulations the relative error on the response time is less than 10.5% and its mean is 14.1% in both models (see rows 7 and 15 in Table 3.3). The robustness of the models against correlated processes will be addressed in the next section.

3.5.2 Validation through experiments

In order to further validate the models, we have conducted extensive experiments on a LAN and a MAN. All benchmarks were written using the *ProActive* library [101] which provided us with all of the necessary mobile primitives. The network was composed of various Pentium II and Pentium III machines running Linux (2.2.18) and Sun SPARC machines running SunOS interconnected with a 100Mb/s switched LAN or a 7Mb/s MAN (four machines were used overall). We used Java 1.2.2 Green threads [59] in all experiments.

For each approach (forwarders and location server), we executed a single source-agent pair on the testbed. Source idle times are exponentially distributed with rate λ (i.e. λ is the communication rate when the source is idling) and inter-migration times of the agent are exponentially distributed with rate ν. Parameters λ and ν can be modified from one experiment to another. All of the other model parameters $(\delta, \gamma, \gamma_1, \gamma_2, \mu)$ are system-dependent and cannot be changed. Hence, among all of the assumptions that we made to construct the models, only 2 assumptions are not violated (the ones concerning the *input* sequences $\{w_i, i \geq 1\}$ and $\{u_k, k \geq 1\}$ (see Section 3.2)). Indeed, migration durations and communications latencies were found to have Weibull distributions both on a LAN and a MAN whereas service times were approximately constant. Furthermore, assumptions on the independence of these processes are not valid under real conditions. Migration durations and travel times are particularly correlated in real life.

Figures 3.8 and 3.9 report both the experimental and theoretical expected response times as well as the expected number of forwarders obtained for a LAN and for a MAN. Graphs on the left display the expected response time (upper graphs) or the expected number of forwarders (lower graph) as a function of the communication rate λ for λ ranging from $1s^{-1}$ to $10s^{-1}$, for three different values of the migration rate ν ($\nu = 1, 5, 10s^{-1}$); similarly,

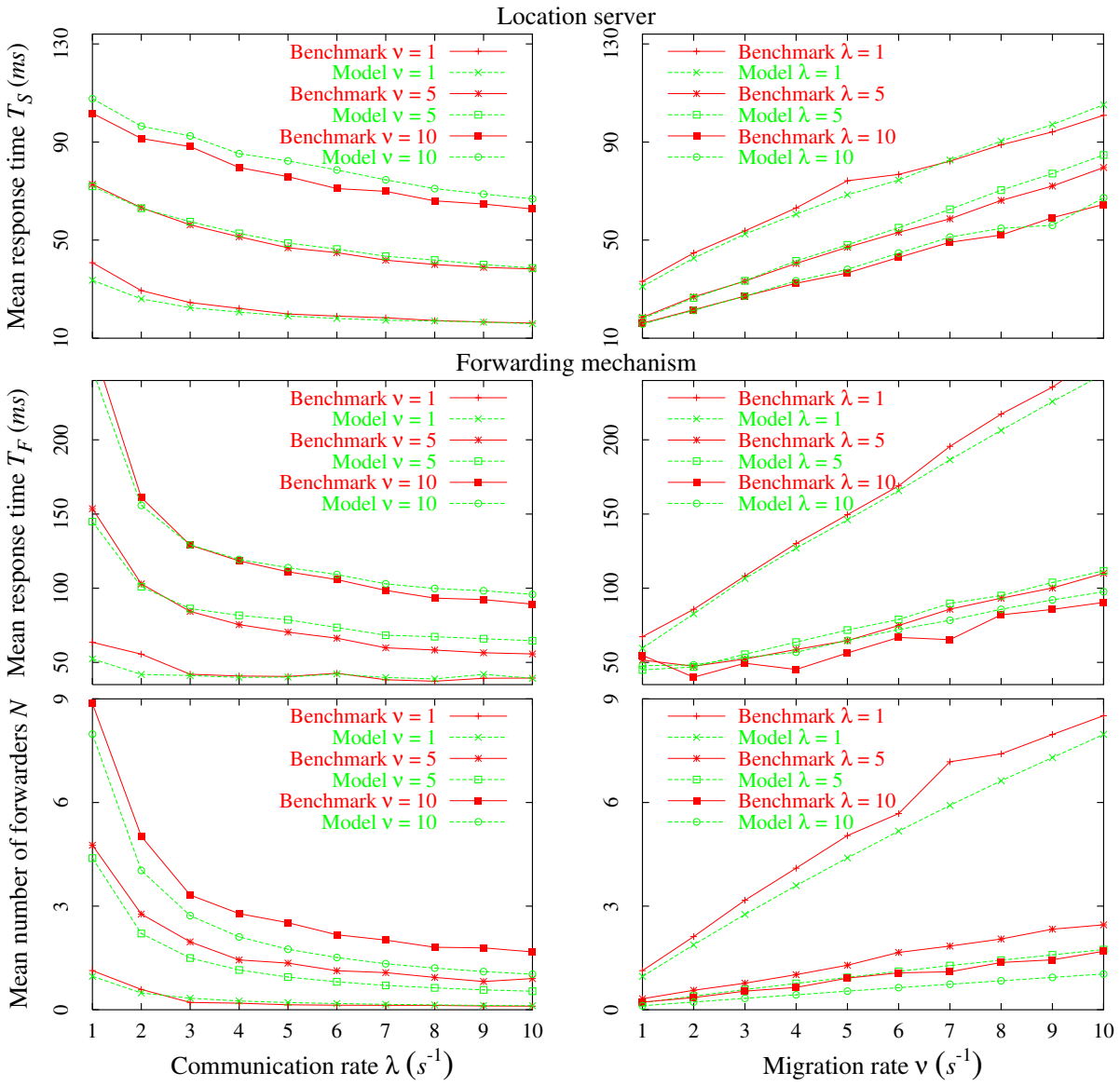


Figure 3.8: Validation with experiments on a 100Mb/s LAN

graphs on the right display the expected response time (upper graphs) or the expected number of forwarders (lower graph) as a function of the migration rate ν for ν ranging from $1s^{-1}$ to $10s^{-1}$, for three different values of the communication rate λ ($\lambda = 1, 5, 10s^{-1}$). For each value of the pair (λ, ν), the empirical values of δ and γ (resp. $\delta, \gamma_1, \gamma_2$ and μ) were substituted into the expressions of T_F and N (resp. T_S) given in (3.33) and (3.50) (resp. (3.51)) when the forwarding mechanism (resp. the centralized mechanism) was used. Observe that analytical and experimental response times are very close to each other across almost all experiments. The average number of forwarders obtained in the experiments is not as well predicted by the model via (3.50), especially on a MAN. It appears that the analytical values underestimate the actual average number of forwarders. The gap between

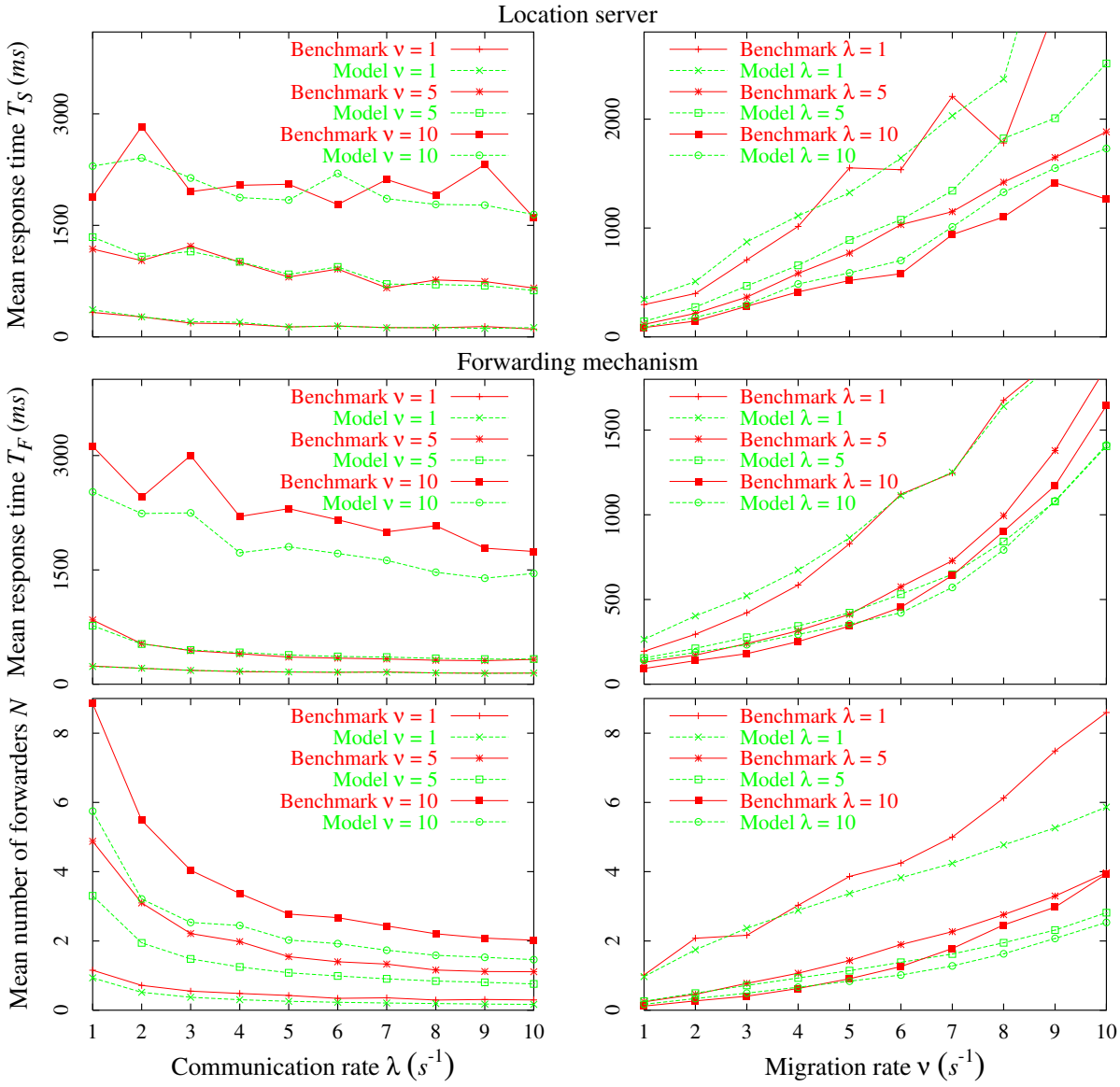


Figure 3.9: Validation with experiments on a 7Mb/s MAN

analytical and experimental values increases when the communication rate λ decreases. For small migration rates $\nu < 5$, there is almost no gap (i.e. almost no absolute error).

For each network configuration (LAN or MAN), the performance of the models are collected in Tables 3.4–3.5. In Table 3.4, we report the sample mean (column 2) and order statistics (25th, median, 75th, 95th and 99th percentiles in columns 3–7) of the *absolute* error (or gap) between analytical N and experimental results. In Table 3.5, we report the sample mean (column 2) and order statistics (25th, median, 75th, 90th and 95th percentiles in columns 3–7) of the relative error between analytical T_F or T_S and experimental results.

We have chosen to report statistics on the absolute error (or gap) $|N_{exp} - N|$ in Table

Table 3.4: Sample mean and percentiles of the absolute error on the average number of forwarders in the forwarding mechanism

Experiments	Mean	25	50	75	95	99
100Mb/s LAN	0.4172	0.1764	0.3914	0.6281	0.8336	1.2336
7Mb/s MAN	0.8360	0.1280	0.4202	1.3881	2.6996	5.2652
Overall	0.6247	0.1467	0.4045	0.7371	1.9469	4.4891

Table 3.5: Sample mean and percentiles of the relative error on the communication time in both mechanisms

Experiments	Mean	25	50	75	90	95
Forwarders						
100Mb/s LAN	7.3	2.1	5.7	10.8	17.0	20.3
7Mb/s MAN	12.1	2.3	7.8	19.0	25.9	35.0
Overall	9.7	2.2	7.1	15.4	22.1	26.3
Server						
100Mb/s LAN	4.6	2.3	4.3	6.2	8.5	10.4
7Mb/s MAN	13.9	6.2	11.3	21.5	28.3	33.0
Overall	9.6	3.7	6.5	13.4	23.4	28.3

3.4, rather than reporting statistics on the relative error $|N_{exp} - N|/N_{exp}$, as we believe that the former are more insightful. For instance, in experiments over a LAN, the plots corresponding to $\nu = 1$ (refer to the lowest graph at the left in Figure 3.8) are almost identical. The maximum gap between the plots is 0.16558 which is an excellent result; however, for the same gap $N_{exp} = 1.13393$ yielding 14.6% of relative error which does not sound very good. In experiments over a LAN, the absolute error between experimental and analytical mean number of forwarders is relatively low, the highest gap being 1.26 (see row 2 in Table 3.4). This is not the case in experiments over a MAN where the maximal gap is 5.38164. Still, the average gap is 0.836 and in 75% of the experiments the gap is under 1.3881 (see row 3 in Table 3.4).

Results in Table 3.5 indicate that the theoretical models behave fairly well. Their overall performance are very close to each other (see rows 5 and 9 in Table 3.5): the sample mean of the relative error is 9.7% (resp. 9.6%) for the model of the forwarders (resp. of the server) (see column 2 in Table 3.5) and in 90% of the experiments using the forwarders (resp. the location server) the relative error is below 22.1% (resp. 23.4%) (see column 6 in Table 3.5). However, both models are more robust when applied on a LAN (see rows 3 and 7 in Table 3.5) than when applied on a MAN (see rows 4 and 8 in Table 3.5).

The model of the server has been validated for one source-agent pair. We still need to perform experiments including multiple sources and/or agents in order to complete the validation of the model. Beside validation, we can use the experimental results to compare the performance of the location mechanisms. For better viewing, the empirical mean response

time returned by both mechanisms is the only variable plotted in Figure 3.10. Graphs on the left display the empirical mean response time as a function of the communication

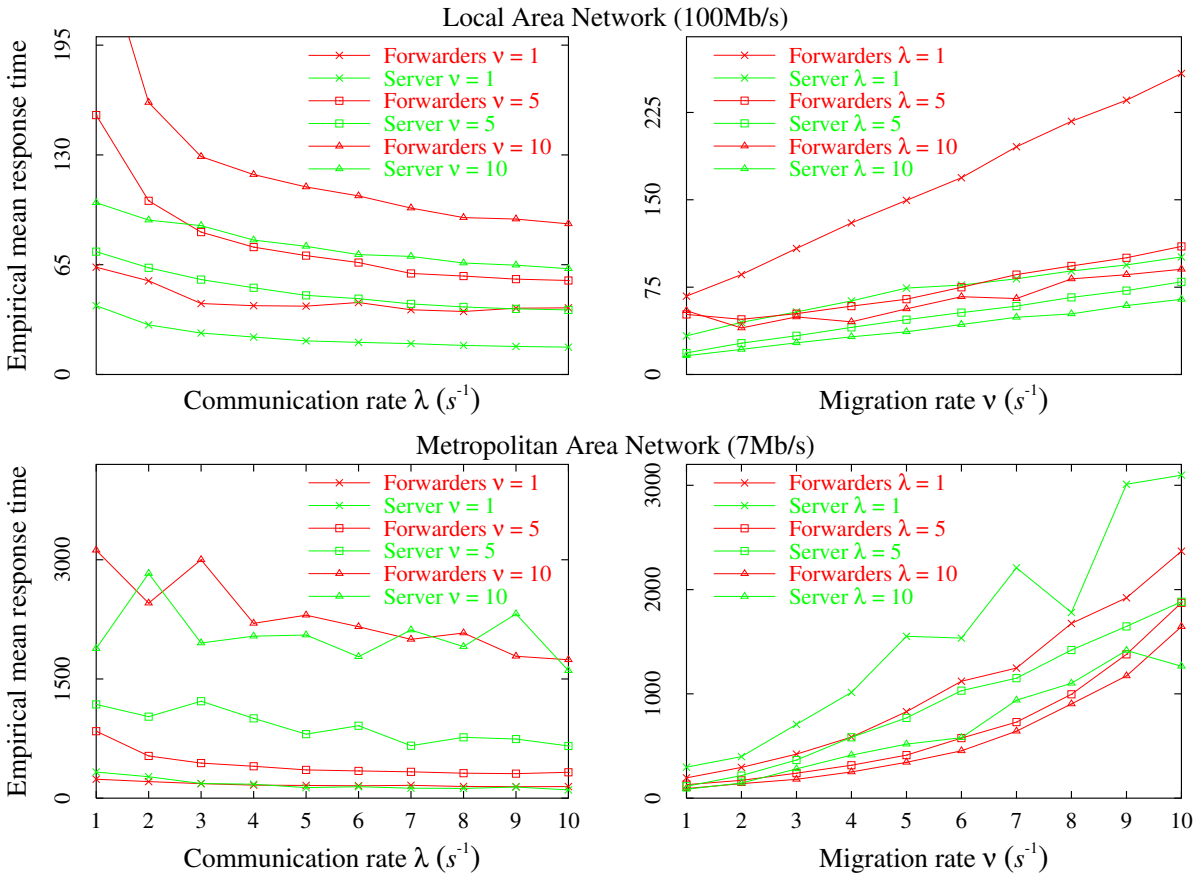


Figure 3.10: Experimental results obtained on a LAN and a MAN

rate λ, for λ ranging from $1s^{-1}$ to $10s^{-1}$, for three different values of the migration rate ν ($\nu = 1, 5, 10$); similarly, graphs on the right display the empirical mean response time as a function of the migration rate ν, for ν ranging from $1s^{-1}$ to $10s^{-1}$, for three different values of the communication rate λ ($\lambda = 1, 5, 10$). Looking at the upper pair of graphs in Figure 3.10 (experiments on a LAN), it appears that the location server scheme has a lower response time when compared to the forwarders scheme. Unexpectedly, we observe the opposite in the lower pair of graphs (experiments on a MAN), where the forwarders scheme achieves the smallest communication time in most of the experiments. It looks like the location server scheme performs the best only within high speed networks. When communication latencies increases (and subsequently the migration durations), the forwarding technique surpasses the centralized technique in performance.

Remark 3.5.1 *We would like to point out that the ergodicity condition was always satisfied in our experiments. Actually, this will always be the case, in practice, due to the fact that the migration of an agent over the network is achieved by sending several messages from*

the old site to the new one. Sending only one of these messages is expected to take $1/\gamma$ *seconds, while the whole migration process accounts for* $1/\delta$ *seconds. Thus, we always have* $1/\delta > 1/\gamma$.

3.5.3 A theoretical comparison of both approaches

As already mentioned, many parameters are network-dependent so that an experimental comparison of both the forwarder approach and the centralized server approach is necessarily limited to a few scenarios. No such limitation occurs when comparing both approaches by using the theoretical results obtained in Sections 3.3 and 3.4. We will focus on the expected response time given by each approach and, more precisely, on their difference ΔT, namely,

$$\Delta T = T_F - T_S \tag{3.62}$$

where T_F and T_S are given in (3.33) and (3.51), respectively. Several cases have been investigated corresponding to different values of the model parameters. Unless otherwise mentioned, the parameters have the values listed in Table 3.6. Each entry in Table 3.6 is the

Table 3.6: Values used for theoretical comparison

Network	$\delta(s^{-1})$	$\gamma(s^{-1})$	$\gamma_1(s^{-1})$	$\gamma_2(s^{-1})$	$\mu(s^{-1})$
100Mb/s LAN	10.9	45.6	115.6	76.3	2325
7Mb/s MAN	1.6	12.3	36.7	12.1	938

average value obtained over all experiments reported in Section 3.5.2. In each case we have identified regions where $\Delta T = 0$, which corresponds to situations where both schemes yield the same expected response (communication) times. When $\Delta T > 0$ (resp. $\Delta T < 0$) the location server (resp. forwarding) scheme gives the best performance. Figure 3.11 displays the sign of ΔT in all of the cases that we have investigated.

We first consider LAN and MAN conditions (refer to rows 2 and 3 in Table 3.6 for self-contained information) and we vary the communication and migration rate from 1 to 50. The sign of ΔT in these cases is shown in Figure 3.11(a). Under LAN conditions, the line $\Delta T = 0$ corresponds to very high migration rates ($\nu > 30$). Above this line, the forwarders mechanism performs better than the centralized mechanism, and below this line, the opposite statement holds. Under MAN conditions, the line $\Delta T = 0$ corresponds to very low migration rates ($\nu < 2.5$) and to high communication rates ($\lambda > 10$). Below this line, the centralized mechanism performs better than the forwarders mechanism, and above it, the opposite statement holds. Observe that in our experiments, we investigated the region $1 \leq \lambda 10$ and $1 \leq \nu 10$ and we have found that, over a LAN, the centralized technique is faster, and over a MAN, the forwarders achieve lower communications time. Our conclusions, drawn from the experiments, agree with the study of the sign of ΔT. However, what we could

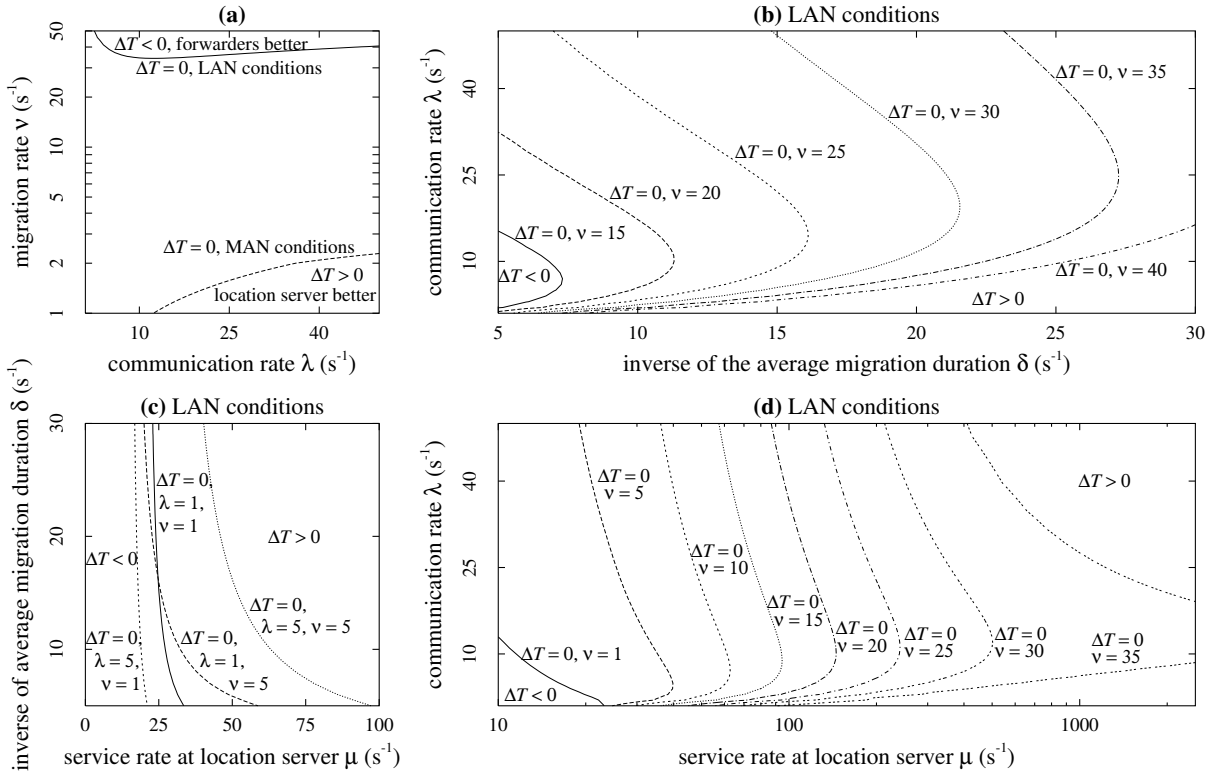


Figure 3.11: Sign of the difference between response times $\Delta T = T_F - T_S$

not foresee in the experiments is that for some migration rates (e.g. $\nu = 35$) the forwarders scheme is better only for intermediate values of the communication rate ($\lambda = 9 \ldots 19$ for instance). This is also observed in Figures 3.11(b) and (d) for fixed values of δ and μ. A closer look at this phenomenon revealed that, for intermediate values of λ, the difference ΔT is slightly negative. In other words, the performance of the mechanisms considered here is almost the same for this choice of parameters. This observation suggests that it would be preferable to draw the lines $|\Delta T| = \epsilon$, where ϵ is small, instead of drawing the line $\Delta T = 0$. Then, for $\Delta T > \epsilon$, the location server is preferable; for $\Delta T < \epsilon$, the forwarders are preferable; and for $|\Delta T| \leq \epsilon$ either mechanism can be chosen.

In Figure 3.11(c), we observe that for extremely low service rate (μ less than $16s^{-1}$), the forwarders technique becomes better than the centralized technique. The frontier between the regions where one approach is better than the other, i.e. the line $\Delta T = 0$, depends on the values of λ and ν. Surprisingly enough, increasing the communication rate while keeping the migration rate unchanged does not have the same effect on this frontier: for $\nu = 1$, an increase in λ from 1 to 5 shifts the line $\Delta T = 0$ to the left (lower service rate) whereas the same increase in λ, but for $\nu = 5$, shifts the frontier to the right (higher service rate). (A shift to the right enlarges the region where forwarders are better and a shift to the left enlarges the region where the server is better.) Notice that for applications generating a single source-agent pair, the operational conditions are far away from these

frontiers ($\mu = 2325$ in our experiments over a LAN). However, when there are multiple sources and/or agents, the buffer at the server is completely partitioned between them, so that each pair would have only a fraction of the server processing speed (which is simply $\mu = 2325$). We have seen from our simulations (reported in Section 3.6), that the service rate per queue depends on parameters λ and ν. For instance, we have seen that for $\lambda = 1$ and $\nu = 1$ there should be around 100 source-agent pairs in order to have a service rate per queue as low as 30, whereas 60 pairs are enough to achieve the same service rate per queue when $\lambda = 10$ and $\nu = 1$. It is only for a large number of source-agent pairs that the performance of the server degrades till the point where forwarders may perform better.

Figure 3.11(d) displays the frontiers $\Delta T = 0$ for several values of the migration rate ($\nu \in \{1, 5, 10, 15, 20, 25, 30, 35\}$) and for $\mu \in [10, 2500]$ and $\lambda \in [1, 50]$; the travel times and the migration durations correspond to a LAN (values in Table 3.6). When ν increases, the line $\Delta T = 0$ shifts to the right (higher service rate) enlarging the region where forwarders are better. Actually, as the migration rate increases, the performance of both approaches degrades: in the forwarders approach, the forwarding chain gets longer, thereby yielding larger communication times; in the centralized approach, the effect of such an increase is threefold. First, the probability that the recorded location at the source is no longer valid increases. Second, the probability of having a wrong location in the server's reply increases as well. Third, there will be much more `update requests` generated. Since these requests have the highest priority, the service of `location requests` coming from the source will be delayed in time (see Section 3.4.1). Each of these effects will increase the communication time. As a result, when ν increases, the performance of the server degrades more rapidly than the performance of the forwarders.

To conclude this section, we would like to stress the fact that selecting the best location scheme is not as intuitive as it could look in the first place. Our study of the sign of ΔT has pointed out several unexpected effects when the value of some parameter is changed.

3.6 Extension to the case of multiple source-agent pairs

The model developed in Section 3.4.2 is precise in the case of a single source-agent pair. But it is just an approximation in the case of applications with several sources and/or agents. In this section, we will first give some guidelines on a precise modeling of systems with multiple objects, and show next how one can use our model in such systems (approximate solutions only).

In applications with several sources and/or agents, there is a single server attending multiple queues. Each queue can have up to two requests, a single `update request` coming from an agent and a single `location request` from a source trying to communicate with the agent. There are as many queues as there are source-agent pairs. The server switches from queue to queue in a cyclic way, serving a unique request in each queue. Such a server is well modeled by a single-server polling model with cyclic non-exhaustive service. In order

to obtain accurate results, the fact that the server stays blocked while sending the replies to the sources must be taken into account. In such cases, the server is unavailable for work even though customers could be waiting. To model this, there are two possible choices: either adopting a server with vacation [115] as illustrated in Figure 3.12(a), or considering the "block" times as making part of the service times of `location requests`, in which case it would be better to have two different service rates (μ_1 for the `update requests` and μ_2 for the `location requests`) (see illustration in Figure 3.12(b)). Such polling models are not easy to analyze. An alternative (and precise) model is presented next.

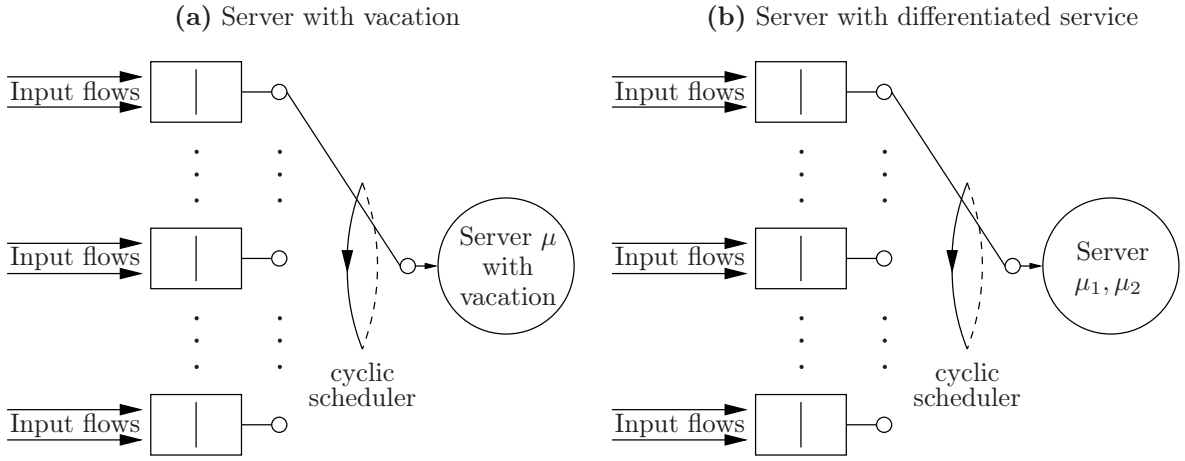


Figure 3.12: Possible choices for modeling the original system

A vacation model describes a single-server queue in which the server can be unavailable for work (away on "vacation") even though customers are waiting [115]. In the polling model illustrated in 3.12(a), the time that the server spends away from any particular queue, serving the other queues or waiting for the end of a "block" period, can be viewed as a vacation from that queue. Adoption of this viewpoint greatly simplifies the analysis of the polling model. The decomposition of the multiple queues system into multiple single queue subsystems is illustrated in Figure 3.13.

Figure 3.13(a) illustrates the real system having a cyclic server with vacation. Figure 3.13(b) illustrates the equivalent system having independent queues, each with one server with vacation. Note that the moments of the vacations are not the same in both systems. In the original system (at the left), the vacations stand for the time in which the server is blocked, whereas in the equivalent system (at the right), the vacations stand for the time the server is away from any particular queue ("block" time + service times of other queues). In both systems, there are two input flows in each queue, one resulting from the `update requests` sent by a mobile agent, and the other resulting from the `location requests` sent by a source. The arrival rates into queue i ($i = 1, \ldots, n$ where n is the total number of queues) are then $\nu_i\delta_i/(\nu_i+\delta_i)$ (`update requests` from agent i) and e_i (`location requests` from source i). The rate e_i and the first two moments of the vacation period can hardly be

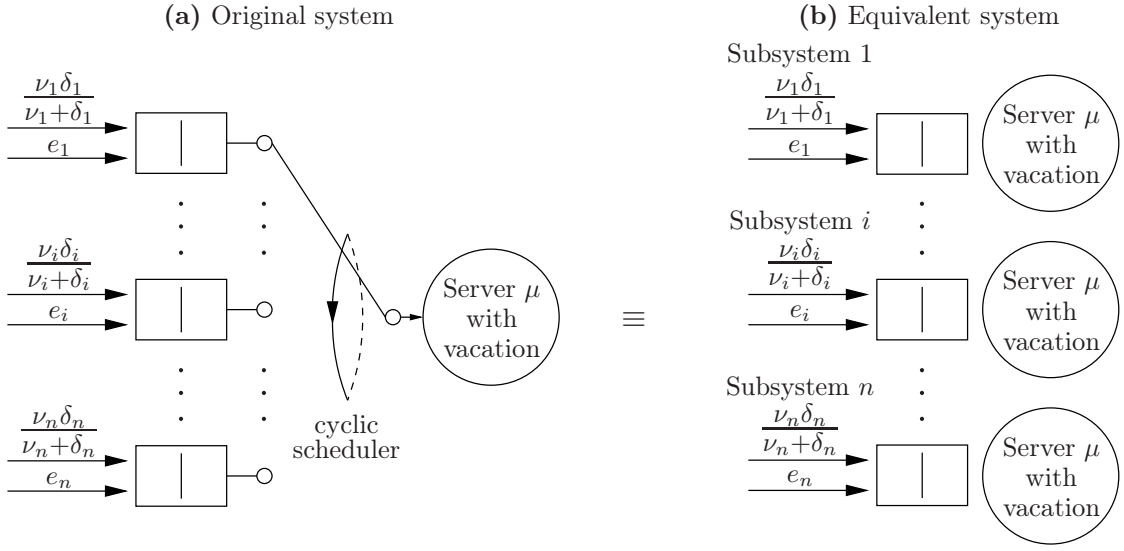


Figure 3.13: Decomposition of the cyclic-service system into n independent single queue subsystems

computed.

The model developed in Section 3.4.2 can be used if one is able to find the equivalence shown in Figure 3.14. At the left of Figure 3.14, the subsystem with one queue and one server with vacation is drawn (the same subsystem as in Figure 3.13(b)). At the right of Figure 3.14, an equivalent system having a server with reduced speed μ_i is illustrated. The

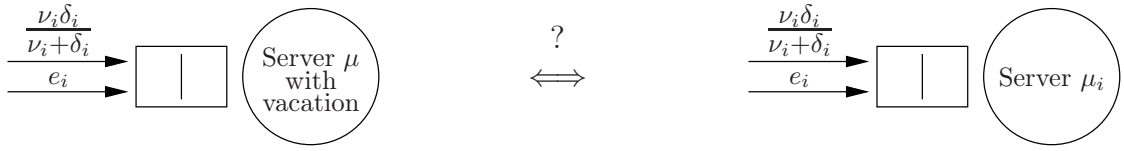


Figure 3.14: Equivalence between two systems, one having a server with vacation (service rate μ) and the other having a server with service rate μ_i

key point of this equivalence resides in the computation of the processing speed μ_i of queue i in the equivalent system. We will necessarily have $\mu_i \leq \mu$ to compensate the vacation periods in the system at the left of Figure 3.14. The equality $\mu_i = \mu$ is obtained when there are a single source-agent pair in the system. Observe that, if there are just a few sources and/or agents, there is likely to be just one active (non-empty) queue most of the time, due to multiplexing. In such cases, it is sufficient to consider $\mu_i \approx \mu$. But what about the general case?

The equivalent systems drawn in Figure 3.14 are assumed to behave identically concerning waiting times and response times. Let $T_{vacation}$ (resp. $T_{reduced}$) denote the response time of the system with vacation (resp. the system with reduced speed). The time $T_{vacation}$

depends on parameters $\nu_i, \delta_i, e_i, \mu$ and the first two moments of the vacation periods. The time $T_{reduced}$ depends on parameters ν_i, δ_i, e_i and μ_i which is to be calculated. Writing $T_{vacation} = T_{reduced}$ yields the computation of μ_i in terms of $\nu_i, \delta_i, e_i, \mu$ and the first two moments of the vacation periods. Unfortunately, it is quite hard to express e_i and the first two moments of the vacation periods, and not much can be done without these parameters.

One may infer the value of μ_i by measuring, at the source, the response time of the server. If the source records the sending time st of a `location request` and the reception time rt of its reply, it can measure the response time as $rt - st - RTT$, where RTT is the round-trip time of a message through the network. The service rate is roughly $1/(rt-st-RTT)$. For systems with a single source-agent pair, we have $1/(rt-st-RTT) \lesssim \mu = \mu_i$. This approximation works only in the special case of high speed networks (LAN for instance). Recall that whenever the server sends a reply to a source, it remains blocked until the reply reaches its destination (synchronous communications between objects). Therefore, when the ratio $\frac{travel\ time}{service\ time}$ increases the approximation gets worse and worse since the block times are considered to be service times.

We already know that over a MAN the forwarders scheme performs better. A single server with multiple queues is definitely slower than a server with a single queue. It is evident that the forwarders will achieve better results in applications with multiple objects. We will therefore concentrate on LAN networks, and determine up to which utilization of the server the approximation holds. Note that it is possible to determine which mechanism achieves the lowest response time by substituting μ_i into ΔT as given by (3.62) (refer to Figure 3.11(d) for more details).

We have carried out four sets of simulations with multiple objects. In each set of simulations, all of the RVs were distributed exponentially. This way, one can precisely measure the goodness of the approximation made in the inference of μ_i. All source-agent pairs had the same migration and communication rates, $\nu_i = \nu$ and $\lambda_i = \lambda$ for $i = 1, \ldots, n$ where n denotes the number of source-agent pairs considered in a particular simulation. In each set of simulations, n was held fixed during a simulation run, and its value ranged from 1 to 100 over all simulations in the same set (the total number of simulations is thus 400). Over all 400 simulations, the travel times, the service times and the migration durations were i.i.d. RVs with rates γ_1, γ_2, μ and δ respectively. The values retained for the latter parameters correspond to a LAN condition (refer to row 2 in Table 3.6 for self-contained information). Finally, each set of simulations is characterized by a single value for the pair (λ, ν). The values retained are (1,1), (1,10), (10,1) and (10,10).

For each simulation, we have computed the expected communication time T_S using (3.51) together with the approximation of μ_i, and measured the utilization of the multiqueue server. The average response time returned by the simulations together with the theoretical T_S are plotted against the number of source-agent pairs n in Figure 3.15. (The analysis curves are not smooth because μ_i is inferred from measurements.) We have also plotted, for each set of simulations, the utilization of the server and the relative error between the simulated result and the analytical one. Figure 3.15 contains eight graphs, two of which

pertain to the same set of simulations. Observe how in all four sets of simulations, the analytical T_S is larger than the simulated result. This is expected since the inferred μ_i (cf. Figure 3.14) is lower than the true one, and the gap between both values increases as the number of queues n increases.

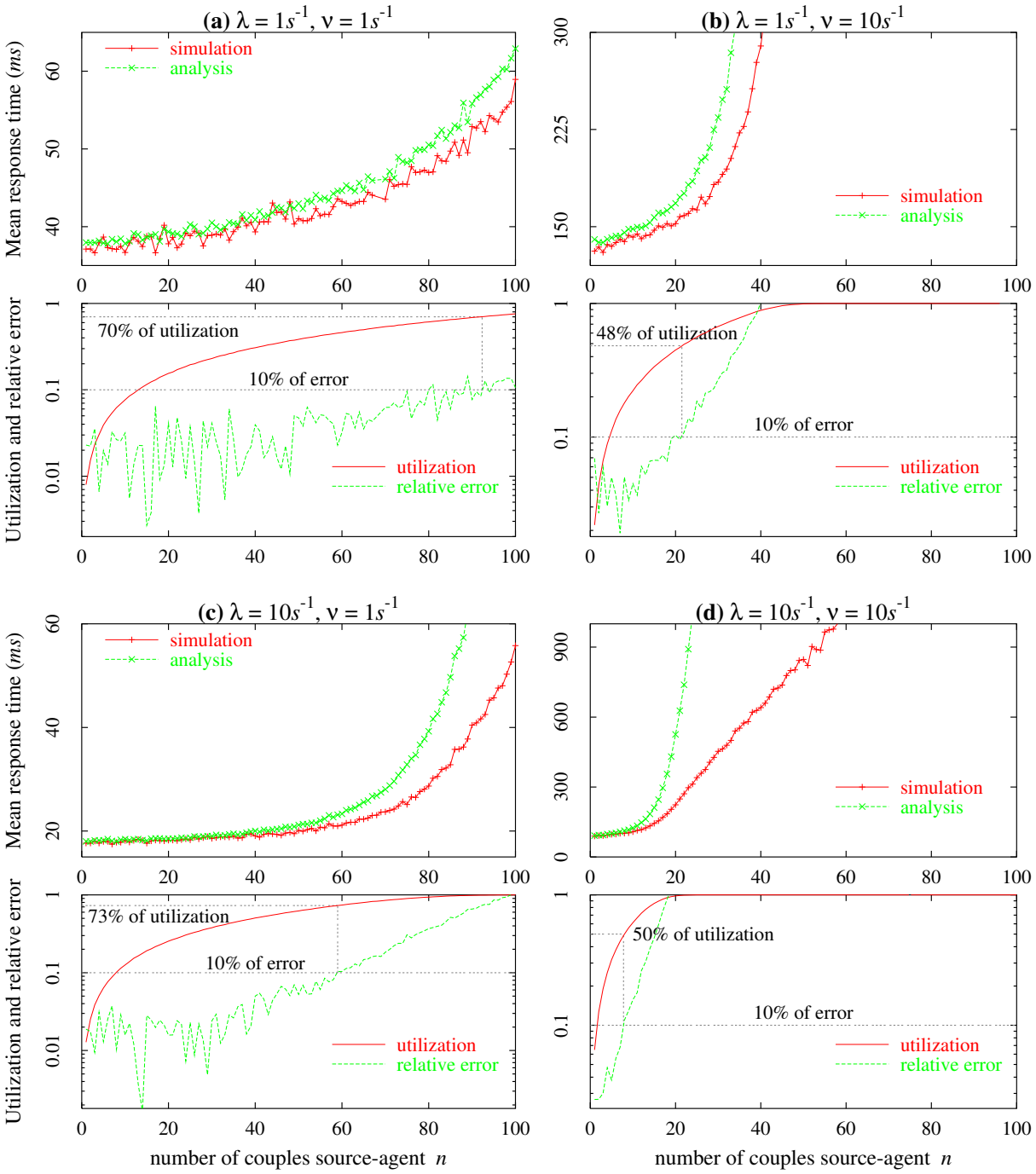


Figure 3.15: Simulated and analytical response times, utilization of the server and relative error between both response times

We have pointed out in Section 3.5.3 that the performance of the location server mechanism degrades rapidly when the migration rate ν increases. This is even more true in the case of multiple objects, especially as the migration rate increases for each agent (all agents have the same migration rate in the simulations). The reader is invited to compare the results shown in Figures 3.15(a) and (b) and the ones in Figures 3.15(c) and (d). The approximation on μ_i yields a smaller error on the response time when the migration rate ν is smaller, for the same utilization of the server and the same communication rate λ. On the other hand, changing the communication rate λ from 1 to 10 while keeping the migration rate ν unchanged does not have a big effect on the quality of the approximation (see Figures 3.15(a) and (c) and Figures. 3.15(b) and (d)). A quantitative comparison of the quality of the approximation in the four sets of simulations is reported in Table 3.7.

Table 3.7 reads as follows: with a tolerance of 10% of error (resp. 15% of error) on the response time, the approximate model can be used as long as the server utilization is below 0.73 (resp. 0.81), which is attained when there are 59 (resp. 66) source-agent pairs, given that $\lambda = 10$ and $\nu = 1$ (see row 5 in Table 3.7). Note that in the set of simulations

Table 3.7: Utilization and number of source-agent pairs yielding 10% and 15% of error

	10% of error		15% of error	
Simulations set	Utilization	Number of pairs	Utilization	Number of pairs
$\lambda = 1, \nu = 1$	0.70	92	N. A. (> 0.77)	N. A. (> 100)
$\lambda = 1, \nu = 10$	0.48	21	0.55	25
$\lambda = 10, \nu = 1$	0.73	59	0.81	66
$\lambda = 10, \nu = 10$	0.50	8	0.61	10

where $\lambda = 1$ and $\nu = 1$, the maximal error obtained was 14.3%. Thus the utilization and the number of pairs which yield 15% of error on the response time are not available, but we know that for such error, the utilization of the server must be higher than 0.77 and that there must be more than 100 source-agent pairs (see row 3 in Table 3.7).

To conclude this section, we can say that in all cases, the approximate model returns fair prediction of the expected communication time as long as the utilization does not exceed 50%.

3.7 Conclusion

We have proposed simple Markovian analytical models for evaluating the performance of two approaches for locating mobile agents. One approach uses forwarders to enable communication between a source and a mobile agent; in the other approach communications are ensured by a centralized server. Our models have been validated through simulations and extensive experiments on a LAN and on a MAN. In all of the experiments that we

have conducted, we have observed that the server yields the best performance on a LAN and the forwarders are more efficient on a MAN. Using the theoretical expected response times of both schemes, we have identified the best location scheme under a wide variety of conditions. Last, note that our contribution in the field of code mobility is twofold. First, we have identified the best location scheme depending on the network conditions. Our conclusions are not that intuitive and they were made possible thanks to our rigorous approach. Second, we have shown that modeling such mechanisms is possible using rather simple techniques, thereby leaving the door open to the performance analysis of similar schemes.

Chapter 4

Conclusion

4.1 Summary

During the course of the last three years, we have been applying a general methodology to different problems encountered in distinct types of applications. This general methodology consists first of proposing mathematical models for the system at hand. Then, metrics of interest or estimators are derived under the assumptions of the proposed models. Finally, the analytical results are compared with simulated and/or experimental results.

In the first chapter of the thesis, we have proposed two single server finite buffer queueing models for a path between two hosts. The single queue is assumed to model the bottleneck link along the path. Under Poisson arrivals and either exponential or deterministic service times, we express the loss probability, the server utilization and the expected response time in terms of the intensity of the background traffic entering the queue and the buffer size at the bottleneck link. Note that it is the first time that the analysis of the $M/D/1/K$ queue has been pushed so far. Conditional probabilities are further derived for the $M+M/M/1/K$ queue model. Taking any pair of these equations and solving for the unknown cross traffic intensity and the unknown buffer size, returns estimates of these unknown parameters, provided that some measurements of the metrics at hand are available. The pairs of estimators – obtained using this procedure over all possible pairs of equations – are then tested through simulations. Pairs of estimators are discarded while others have proved to perform well, which is the case of the pair of estimators relying on measurements of the loss probability and the response time.

In the second chapter of the thesis, we have proposed successively three models for multicast groups. The first model, an $M/M/\infty$ queue under heavy traffic, enables the use of Kalman filtering theory to derive an efficient membership estimator, which is optimal for the $M/M/\infty$ queue under heavy traffic. The second model, an $M/M/\infty$ queue under a general traffic regime, lead to the same estimator, but this time the Wiener filter theory

was used. Note that the later theory can be applied to the $M/G/\infty$ queue as long as the distribution of receivers lifetime is not heavy-tailed. However, the canonical factorization of the power spectrum of the measurements process is possible only for the $M/M/\infty$ queue. The third and last model studied, an $M/H_L/\infty$ queue where H_L denotes an L-stage hyperexponential distribution, allows the derivation of an efficient estimator which is optimal among all estimators satisfying a first-order linear auto-regressive equation. To summarize what precedes, we have proposed three models, have followed three distinct approaches and obtained *two* distinct membership estimators. These estimators have been tested on simulations driven by real traces. Their overall performance are almost identical, however the estimator deriving from the $M/M/\infty$ queue model shows a good ability to track high variations in the membership, the other estimator being more on target in flat periods.

In the third chapter, we have proposed Markovian models for agent location mechanisms in a mobile code environment. To the best of our knowledge, it is the first time that such mechanisms are modeled and their performance are formally evaluated. We derive expressions for the expected response time of each mechanism in each model and further express the expected number of forwarders in the decentralized mechanism. The theoretical results are compared to both simulated and experimental results (experiments conducted over both a LAN and a MAN). It appeared in our experiments that forwarders achieve better performance over a MAN and that the centralized technique is preferable in a LAN. The relatively close match between the latter results validates both Markovian models and allows the use of the theoretical formulas as predictors of the performance of the mechanisms. We have investigated the performance of the mechanisms under a wide variety of conditions using the difference between the expected response times of each mechanism. This theoretical comparison has illustrated several unexpected effects when some parameter is changed and revealed that no mechanism is uniformly better than the other.

4.2 Perspectives

Concerning the first part of the thesis (Chapter 1), we have seen that the scheme $P_L\_U\_R$ does not perform especially well in simultaneously estimating the bottleneck bandwidth μ, the cross traffic intensity at the bottleneck link λ and the buffer size at the bottleneck link K. We should therefore use an estimate of μ (provided by *pathrate*, PBM or ROPP for instance) and then use scheme P_L_R to estimate λ and K. The impact of using $\hat{\mu}$ instead of using μ still needs to be investigated. Beside that, the $M{+}M/M/1/K$ queue model may be extended to account for the propagation delay. The latter could be modeled by a normally distributed random variable or an exponentially distributed random variable.

Concerning the estimation of the membership of multicast groups, we plan to work on more realistic models, such as the $M/W/\infty$ queue or the $M/L/\infty$ queue. The first step will be to compute (2.39) for these queues. After that, one may follow the approach of Section 2.6.2 to find the optimal first-order filter, using either (2.6) or (2.54). We actually do not

know which auto-regressive equation is preferable. Future research will certainly address this issue. Another point which will also need to be addressed concerns the choice of the ACK probability and the ACK interval for the considered model. Instead of limiting the amount of feedback generated each round, we might limit the amount of feedback generated each I seconds, as already suggested in Remark 2.7.1.

Concerning the research on agent location mechanisms, we plan to revisit the model for the centralized mechanism. Instead of the single queue server investigated in Section 3.4.2, we will consider a multiqueue single server with vacation. Another possibility consists of studying a multiqueue server with feedback, in which a query that has already been served can potentially be replaced in the queue. Besides, we are interested in modeling other agent location mechanisms, especially a mixed mechanism recently implemented in the *ProActive* library. In this mixed approach, both forwarders and a location server are used to ensure communications. Forwarders have a fixed lifetime and mobile objects periodically update their locations at the server. When a message finds no object (forwarder or mobile agent) on a given site, the source queries the location server to have the latest known location of the mobile object. The engineering questions which arise in such mechanisms are "what is the ideal lifetime for forwarders?" and "how often should the mobile object inform the location server of its location?" Future research aim at answering these questions.

Glossary

List of abbreviations

ACK	Acknowledgement
a.s.	almost surely
CAC	Call Acceptance Controller
CDF	Cumulative Distribution Function
CCDF	Complementary Cumulative Distribution Function
cf.	confer
C-K	Chapman–Kolmogorov
DiffServ	Differentiated Services
e.g.	example
EWMA	Exponential Weighted Moving Average
FTP	File Transfer Protocol
i.e.	id est
iff.	if and only if
i.i.d.	independent and identically distributed
IP	Internet Protocol
ISP	Internet Service Provider
IVS	INRIA Videoconferencing System
LAN	Local Area Network
l.h.s.	left-hand side
LST	Laplace–Stieltjes Transform
MAN	Metropolitan Area Network
MBone	Multicast Backbone
MINC	Multicast-based Inference of Network-internal Characteristics
MOA	Mobile Objects and Agents
N. A.	Not Available
NAG	Numerical Algorithms Group
`ns`	network simulator
P a.s.	almost surely under the measure P
PASTA	Poisson Arrivals See Time Average
PBM	Packet Bunch Mode
PR	Peak Rate
QoS	Quality of Service
RBPP	Receiver-Based Packet Pair

RED	Random Early Detection
resp.	respectively
r.h.s.	right-hand side
ROPP	Receiver Only Packet Pair
RR	Receiver Report
RTCP	RTP Control Protocol
RTP	Real-time Transport Protocol
SBPP	Sender-Based Packet Pair
RV	Random Variable
SRM	Scalable Reliable Multicast
TCP	Transmission Control Protocol
TV	Television
UDP	User Datagram Protocol
vs.	versus
w.r.t.	with respect to

Bibliography

[1] Differentiated Services (DiffServ). http://www.ietf.org/html.charters/diffserv-charter.html.

[2] Introduction to the internet - using the internet. http://www.sofweb.vic.edu.au/internet/.

[3] MINC: Multicast-based Inference of Network-internal Characteristics. http://www.research.att.com/projects/minc/.

[4] The TCP-friendly website. http://www.icir.org/floyd/tcp_friendly.html.

[5] Aglets Software Development Kit. IBM, 1999. http://www.trl.ibm.com/aglets/.

[6] K. Almeroth and M. Ammar. Collecting and modeling of the join/leave behavior of multicast group members in the MBone. In *Proc. of HPDC '96, Syracuse, New York*, pages 209–216, August 1996.

[7] K. Almeroth and M. Ammar. Multicast group behavior in the Internet's Multicast Backbone (MBone). *IEEE Communications Magazine*, 35:224–229, June 1997.

[8] K. C. Almeroth and M. H. Ammar. *MListen*, 1995. http://www.cc.gatech.edu/computing/Telecomm/mbone/.

[9] S. Alouf, E. Altman, C. Barakat, and P. Nain. Optimal estimation of multicast membership. *submitted to IEEE Trans. on Signal Processing, Special Issue on Signal Processing in Networking.*

[10] S. Alouf, E. Altman, C. Barakat, and P. Nain. Estimating membership in a multicast session, February 2002. INRIA, Research Report RR-4391.

[11] S. Alouf, E. Altman, and P. Nain. Optimal on-line estimation of the size of a dynamic multicast group, November 2001. INRIA, Research Report RR-4329.

[12] S. Alouf, E. Altman, and P. Nain. Optimal on-line estimation of the size of a dynamic multicast group. In *Proc. of IEEE INFOCOM '02, New York, New York*, volume 2, pages 1109–1118, June 2002.

[13] S. Alouf, F. Huet, and P. Nain. Forwarders vs. centralized server: An evaluation of two approaches for locating mobile agents. *Performance Evaluation, (Proc. of Performance '02, Rome, Italy)*, 49(1-4):299–319, September 2002.

[14] S. Alouf, F. Huet, and P. Nain. Forwarders vs. centralized server: An evaluation of two approaches for locating mobile agents, April 2002. INRIA, Research Report RR-4440.

[15] S. Alouf, F. Huet, and P. Nain. Forwarders vs. centralized server: An evaluation of two approaches for locating mobile agents (extended abstract). *Performance Evaluation Review, (Proc. of ACM SIGMETRICS '02, Marina Del Rey, California)*, 30(1):278–279, June 2002.

[16] S. Alouf, P. Nain, and D. Towsley. Inferring network characteristics via moment-based estimators. In *Proc. of IEEE INFOCOM '01, Anchorage, Alaska*, volume 2, pages 1045–1054, April 2001.

[17] S. Asmussen. *Applied Probability and Queues*. John Wiley & Sons, 1987.

[18] A. Bakker, E. Amade, G. Ballintijn, I. Kuz, P. Verkaik, I. van der Wijk, M. van Steen, and A. S. Tanenbaum. The globe distribution network. In *Proc. of USENIX '00 (FREENIX Track), San Diego, California*, pages 141–152, June 2000.

[19] F. Baude, D. Caromel, F. Huet, and J. Vayssière. Objets actifs mobiles et communicants. *Technique et Science Informatique*, 21(6), 2002.

[20] J. Baumann. *Control algorithms for mobile agents*. PhD thesis, University of Stuttgart, Germany, IPVR Department, 1999.

[21] J.-C. Bolot. Characterizing end-to-end packet delay and loss in the Internet. *Journal of High-Speed Networks*, 2(3):305–323, December 1993.

[22] J.-C. Bolot. End-to-end packet delay and loss behavior in the Internet. In *Proc. of ACM SIGCOMM '93, San Francisco, California*, pages 289–298, September 1993.

[23] J.-C. Bolot, T. Turletti, and I. Wakeman. Scalable feedback control for multicast video distribution in the Internet. In *Proc. of ACM SIGCOMM '94, London, UK*, pages 58–67, September 1994.

[24] A. A. Borovkov. *Asymptotic Methods in Queueing Theory*. John Wiley & Sons, 1984.

[25] J. Broch, D. A. Maltz, D. B. Johnson, Y.-C. Hu, and J. Jetcheva. A performance comparison of multi-hop wireless ad hoc network routing protocols. In *Proc. of MobiCom '98, Dallas, Texas*, pages 85–97. ACM Press, October 1998.

[26] R. Cáceres, N. G. Duffield, A. Feldmann, J. Friedmann, A. Greenberg, R. Greer, T. Johnson, C. Kalmanek, B. Krishnamurthy, D. Lavelle, P. Mishra, K. Ramakrishnan, J. Rexford, F. True, and J. van der Merwe. Measurement and analysis of ip network usage and behavior. *IEEE Communications Magazine*, 38(5):144–151, May 2000.

[27] R. Cáceres, N. G. Duffield, J. Horowitz, F. Lo Presti, and D. Towsley. Loss-based inference of multicast network topology. In *Proc. of IEEE CDC '99, Phoenix, Arizona*, volume 3, pages 3065–3070, December 1999.

[28] R. Cáceres, N. G. Duffield, J. Horowitz, and D. Towsley. Multicast-based inference of network-internal loss characteristics. *IEEE Transactions on Information Theory*, 45(7):2462–2480, November 1999.

[29] R. Cáceres, N. G. Duffield, J. Horowitz, D. Towsley, and T. Bu. Multicast-based inference of network-internal characteristics: accuracy of packet loss estimation. In *Proc. of IEEE INFOCOM '99, New York, New York*, volume 1, pages 371–379, March 1999.

[30] R. Cáceres, N. G. Duffield, S. B. Moon, and D. Towsley. Inference of internal loss rates in the MBone. In *Proc. of IEEE/ISOC Global Internet '99, Rio de Janeiro, Brazil*, December 1999.

[31] R. Carter and M. Crovella. Measuring bottleneck link speed in packet-switched networks. *Performance Evaluation*, 27&28:297–318, October 1996.

[32] D. M. Chess. Security issues in mobile code system. In *Lecture Notes in Computer Science 1419*, pages 1–14. Springer-Verlag, 1998.

[33] D. M. Chess, C. G. Harrison, and A. Kershenbaum. Mobile agents: Are they a good idea? Technical report, IBM T.J. Watson Research Division, March 1995.

[34] K. J. Christensen. Tools page for Ken Christensen. http://www.csee.usf.edu/∼christen/tools/toolpage.html.

[35] K. J. Christensen. Reproduction of some key results in Leland et al., May 2002. Available as http://www.csee.usf.edu/∼christen/tools/bellcore.pdf.

[36] Y. Chu, S. Rao, and H. Zhang. A case for end system multicast. In *Proc. of ACM SIGMETRICS '00, Santa Clara, California*, pages 1–12, June 2000.

[37] D. Clark. The design philosophy of the DARPA Internet. In *Proc. of ACM SIGMETRICS '88, Stanford, California*, pages 1–12, August 1988.

[38] I. Clarke, O. Sandberg, B. Wiley, and T. W. Hong. Freenet: A distributed anonymous information storage and retrieval system. In *Lecture Notes in Computer Science 2009*, pages 46–66. Springer-Verlag, 2001.

[39] J. W. Cohen. *The Single Server Queue*. North-Holland publishing company, 1982.

[40] R. M. Corless, G. H. Gonnet, D. E. G. Hare, D. J. Jeffrey, and D. E. Knuth. On the Lambert W function. *Advances in Computational Mathematics*, 5:329–359, 1996.

[41] D. R. Cox and V. Isham. *Point Processes*. Chapman and Hall, New York, 1980.

[42] S. R. Das, C. E. Perkins, and E. M. Royer. Performance comparison of two on-demand routing protocols for ad hoc networks. In *Proc. of INFOCOM '00, Tel-Aviv, Israel*, volume 1, pages 3–12, March 2000.

[43] S. Deering. Host extensions for IP multicasting. RFC 1112, Network Working Group, August 1989.

[44] S. Deering. *Multicast Routing in a Datagram Internetwork*. PhD thesis, Stanford University, December 1991.

[45] C. Diot, B. N. Levine, B. Lyles, H. Kassem, and D. Balensiefen. Deployment issues for the IP multicast service and architecture. *IEEE Network magazine, Special Issue on Multicasting*, 14(1):78–88, January/February 2000.

[46] C. Dovrolis, P. Ramanathan, and D. Moore. What do packet dispersion techniques measure? In *Proc. of IEEE INFOCOM '01, Anchorage, Alaska*, volume 2, pages 905–914, April 2001.

[47] N. G. Duffield, J. Horowitz, and F. Lo Presti. Adaptive multicast topology inference. In *Proc. of IEEE INFOCOM '01, Anchorage, Alaska*, volume 3, pages 1636–1645, April 2001.

[48] N. G. Duffield, J. Horowitz, F. Lo Presti, and D. Towsley. Multicast topology inference from end-to-end measurements. In *Proc. of ITC Specialist Seminar on IP Traffic Measurement, Modeling and Management, Modeling and Management, Monterey, California*, September 2000.

[49] N. G. Duffield, J. Horowitz, F. Lo Presti, and D. Towsley. Network delay tomography from end-to-end unicast measurements. In *Lecture Notes in Computer Science 2170, Proc. of IWDC '01*, pages 576–595. Springer-Verlag, September 2001.

[50] N. G. Duffield and F. Lo Presti. Multicast inference of packet delay variance at interior network links. In *Proc. of IEEE INFOCOM '00, Tel Aviv, Israel*, volume 3, pages 1351–1360, March 2000.

[51] N. G. Duffield, F. Lo Presti, V. Paxson, and D. Towsley. Inferring link loss using striped unicast probes. In *Proc. of IEEE INFOCOM '01, Anchorage, Alaska*, volume 2, pages 915–923, April 2001.

[52] A. Dutta, H. Schulzrinne, and Y. Yemini. MarconiNet - an architecture for Internet radio and TV networks. In *Proc. of NOSSDAV '99, Basking Ridge, New Jersey*, June 1999.

[53] S. Floyd and V. Jacobson. Random early detection gateways for congestion avoidance. *IEEE/ACM Trans. on Networking*, 1(4):397–413, August 1993.

[54] S. Floyd, V. Jacobson, S. McCanne, C. Liu, and L. Zhang. A reliable multicast framework for light-weight sessions and application level framing. In *Proc. of ACM SIGCOMM '95, Cambridge, Massachusetts*, pages 342–356, August 1995.

[55] R. J. Fowler. The complexity of using forwarding addresses for decentralized object finding. In *Proc. of PODC '86, New York, New York*, pages 108–120, August 1986.

[56] P. Francis. Yoid: Extending the internet multicast architecture. Unrefereed report, April 2000. Available at http://www.icir.org/yoid/docs/index.html.

[57] T. Friedman and D. Towsley. Multicast session membership size estimation. In *Proc. of IEEE INFOCOM '99, New York, New York*, volume 2, pages 965–972, March 1999.

[58] T. Fuhrmann and J. Widmer. On the scaling of feedback algorithms for very large multicast groups. *Computer Communications, Special Issue on Integrating multicast into the Internet*, 24(5-6):539–547, March 2001.

[59] J. Gosling, B. Joy, and G. Steele. *The Java Language Specification*. GOTOP Information Inc., 1996.

[60] R. S. Gray. Agent Tcl: A flexible and secure mobile agent system. In *Proc. of the 4th Annual Tcl/Tk Workshop, Monterey, California*, pages 9–23, July 1996.

[61] S. Haykin. *Modern Filters*. Macmillan, New York, 1989.

[62] S. Haykin. *Adaptive Filter Theory*. Prentice Hall, 3rd edition, 1996.

[63] H. W. Holbrook and D. R. Cheriton. IP multicast channels: EXPRESS support for large-scale single-source applications. In *Proc. of ACM SIGCOMM '99, Cambridge, Massachusetts*, pages 65–78, September 1999.

[64] F. Huet. *Objets mobiles: conception d'un middleware et évaluation de la localisation*. PhD thesis, University of Nice - Sophia Antipolis, France, December 2002. in preparation.

[65] INRIA. *IVS*, 1992. http://www-sop.inria.fr/rodeo/ivs.html.

[66] D. S. Isenberg. The rise of the stupid network. *Computer Telephony*, 1997.

[67] V. Jacobson. Congestion avoidance and control. In *Proc. of ACM SIGCOMM '88, Stanford, California*, pages 314–329, August 1988.

[68] J. Jannotti, D. Gifford, K. Johnson, M. Kaashoek, and J. O'Toole. Overcast: Reliable multicasting with an overlay network. In *Proc. of USENIX OSDI '00, San Diego, California*, pages 197–212, October 2000.

[69] P. Johansson, T. Larsson, N. Hedman, B. Mielczarek, and M. Degermark. Scenario-based performance analysis of routing protocols for mobile ad-hoc networks. In *Proc. of MobiCom '99, Seattle, Washington*, pages 195–206. ACM Press, August 1999.

[70] I. Karatzas and S. E. Shreve. *Brownian Motion and Stochastic Calculus*. Springer, 1991.

[71] G. Karjoth, D. B. Lange, and M. Oshima. A security model for Aglets. *IEEE Internet Computing*, 1(4):68–77, July-August 1997.

[72] D. G. Kendall. Stochastic processes occurring in the theory of queues and their analysis by the method of the imbedded markov chain. *Annals of Mathematical Statistics*, 24:338–354, 1953.

[73] S. Keshav. A control-theoretic approach to flow control. In *Proc. of ACM SIGCOMM '91, Zurich, Switzerland*, pages 3–15, September 1991.

[74] J. Kiniry and D. Zimmerman. A hands-on look at Java mobile agents. *IEEE Internet Computing*, 1(4):21–30, July-August 1997.

[75] L. Kleinrock. *Queueing Systems: Theory*, volume 1. John Wiley and Sons, 1975.

[76] J. Kubiatowicz, D. Bindel, Y. Chen, S. Czerwinski, P. Eaton, D. Geels, R. Gummadi, S. Rhea, H. Weatherspoon, W. Weimer, C. Wells, and B. Zhao. Oceanstore: An architecture for global-scale persistent storage. In *Proc. of ASPLOS '00, Boston, Massachusetts*, pages 190–201, November 2000.

[77] K. Lai and M. Baker. Measuring bandwidth. In *Proc. of IEEE INFOCOM '99, New York, New York*, volume 1, pages 235–245, March 1999.

[78] A. M. Lee. *Applied Queueing Theory*. London: Macmillan, New York: St. Martin's Press, 1966.

[79] C. Liu and J. Nonnenmacher. Broadcast audience estimation. In *Proc. of IEEE INFOCOM '00, Tel Aviv, Israel*, volume 2, pages 952–960, March 2000.

[80] J. Mahdavi and S. Floyd. TCP-friendly unicast rate-based flow control, January 1997. Technical note sent to the end2end-interest mailing list, available via http://www.psc.edu/networking/papers/tcp_friendly.html.

[81] V. A. Malyshev. An analytical method in the theory of two-dimensional positive random walks. *Mathematicheskii Zhurnal*, 13(6):1314–1329, 1972.

[82] M. Mathis, J. Semske, J. Mahdavi, and T. Ott. The macroscopic behavior of the TCP congestion avoidance algorithm. *Computer Communication Review*, 27(3):67–82, July 1997. http://www.acm.org/sigcomm/ccr/archive/1997/jul97/ccr-9707-mathis.html.

[83] S. McCanne and S. Floyd. *The LBNL/UCB network simulator*. Lawrence Berkeley National Laboratory, Univ. of California at Berkeley. Software on line at http://www.isi.edu/nsnam/ns/.

[84] D. S. Milojičić, W. LaForge, and D. Chauhan. Mobile Objects and Agents (MOA). In *Proc. of USENIX COOTS '98, Santa Fe, New Mexico*, April 1998.

[85] P. Mockapetris and K. J. Dunlap. Development of the Domain Name System. In *Proc. of SIGCOMM '88, Stanford, California*, pages 123–133. ACM Press, August 1988.

[86] S. B. Moon, P. Skelly, and D. Towsley. Estimation and removal of clock skew from network delay measurements. In *Proc. of IEEE INFOCOM '99, New York, New York*, volume 1, pages 227–234, March 1999.

[87] J. Nonnenmacher. *Reliable multicast transport to large groups*. PhD thesis, Ecole Polytechnique Federale de Lausanne, Switzerland, July 1998.

[88] J. Nonnenmacher and E. Biersack. Scalable feedback for large groups. *IEEE/ACM Trans. on Networking*, 7(3):375–386, June 1999.

[89] J. Norris. *Markov Chains*. Cambridge University Press, 1997.

[90] Numerical Algorithm Group. *NAG C Library Manual, the* `c05tbc` *function*. Available as http://www.nag.co.uk/numeric/cl/manual/pdf/C05/c05tbc.pdf.

[91] Numerical Algorithm Group. *NAG C Library Manual, the* `d01sjc` *function*. Available as http://www.nag.co.uk/numeric/cl/manual/pdf/D01/d01sjc_cl06.pdf.

[92] M. Nuttall. A brief summary of systems providing process or object migration facilities. *Operating Systems Review*, 28(4):64–80, October 1994.

[93] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose. Modeling TCP throughput: A simple model and its empirical validation. In *Proc. of ACM SIGCOMM, Stanford, CA USA*, pages 303–314, August 1988.

[94] V. Paxson. *Measurements and analysis of end-to-end Internet dynamics*. PhD thesis, Lawrence Berkeley National Laboratory, University of California at Berkeley, April 1997. Available via ftp://ftp.ee.lbl.gov/papers/vp-thesis/.

[95] V. Paxson and S. Floyd. Wide-area traffic: the failure of Poisson modeling. In *Proc. of ACM SIGCOMM '94, London, UK*, pages 257–268, September 1994.

[96] D. Pendarakis, S. Shi, D. Verma, and M. Waldvogel. ALMI: An application level multicast infrastructure. In *Proc. of USENIX USITS '01, San Francisco, California*, pages 49–60, March 2001.

[97] J. Postel. User datagram protocol. RFC 768, ISI, August 1980.

[98] J. Postel. Transmission control protocol. RFC 793, DARPA Internet program protocol specification, September 1981.

[99] M. L. Powell and B. P. Miller. Process migration in DEMOS/MP. In *Proc. of SOSP '83, Bretton Woods, New Hampshire*, October 1983. Published as Operating Systems Review, 17(5):110-119.

[100] F. Lo Presti, N. G. Duffield, J. Horowitz, and D. Towsley. Multicast-based inference of network-internal delay distributions. Technical report, University of Massachusetts at Amherst, December 1999.

[101] ProActive. INRIA, 1999. http://www-sop.inria.fr/oasis/ProActive.

[102] S. Ramanathan and M. Steenstrup. A survey of routing techniques for mobile communications networks. *Mobile Networks and Applications*, 1(2):89–104, October 1996.

[103] P. Robert. *Réseaux et Files d'Attente: Méthodes Probabilistes*, volume 35 of *Mathématiques & Applications*. Springer Paris, 2000.

[104] V. Roca and A. El-Sayed. A host-based multicast (HBM) solution for group communications. In *Proc. of ICN '01, Colmar, France*, pages 610–619, July 2001.

[105] J. Rosenberg and H. Schulzrinne. Timer reconsideration for enhanced RTP scalability. In *Proc. of IEEE INFOCOM '98, San Francisco, California*, volume 1, pages 233–241, March/April 1998.

[106] W. A. Rosenkrantz and J. Horowitz. Statistical analysis of variance-time plots used to estimate parameters of a long-range dependent process. Submitted for publication, February 2001.

[107] P. Ruckdeschel. *Robust Kalman Filtering – Optimality of the Kalman Filter*. MD*TECH Method and Data Technologies, June 2002. http://www.xplore-stat.de/tutorials/rkalmframe3.html.

[108] J. H. Saltzer, D. P. Reed, and D. D. Clark. End-to-end arguments in system design. *ACM Trans. in Computer Systems*, 2(4):277–288, November 1984.

[109] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. RTP: a transport protocol for real-time applications. RFC 1889, Network Working Group, January 1996.

[110] V. Sharma and R. Mazumdar. Estimating traffic parameters in queueing systems with local information. *Performance evaluation*, 32(3):217–230, April 1998.

[111] D. Sisalem and H. Schulzrinne. The loss-delay based adjustment algorithm: a TCP-friendly adaptation scheme. In *Proc. of NOSSDAV '98, Cambridge, UK*, pages 215–226, July 1998.

[112] R. F. Stengel. *Stochastic Optimal Control, Theory and Application*. John Wiley & Sons, 1986.

[113] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *Proc. of ACM SIGCOMM '01, San Diego, California*, pages 149–160, August 2001.

[114] L. Takács. *Introduction to the Theory of Queues*. Oxford University Press, Inc., 1962.

[115] H. Takagi. *Queueing Analysis: a Foundation of Performance Evaluation*, volume 1. North-Holland, 1991.

[116] M.S. Taqqu, W. Willinger, and R. Sherman. Proof of a fundamental result in self-similar traffic modeling. *Computer Communication Review*, 27(2):5–23, April 1997.

[117] T. Thorn. Programming languages for mobile code. *ACM Computing Surveys*, 29(3):213–239, September 1997.

[118] E. C. Titchmarsh. *The Theory of Functions*. Oxford University Press, 2nd edition, 1939.

[119] J. van der Merwe, R. Cáceres, Y.-H. Chu, and C. Sreenan. mmdump: a tool for monitoring Internet multimedia traffic. *ACM SIGCOMM Computer Communication Review*, 30(5):48–59, October 2000.

[120] Virtual Laboratories in Probability and Statistics. *Probability Plots*. http://www.math.uah.edu/stat/sample/sample8.html.

[121] Voyager. ObjectSpace, Inc., 1999. http://www.objectspace.com.

[122] D. J. Wetherall. *Service introduction in an active network*. PhD thesis, Dept. of Electrical Engineering and Computer Science, MIT at Cambridge, February 1999. Available via http://www.cs.washington.edu/homes/djw/papers/ants-thesis.ps.gz.

[123] P. Whittle. *Optimal Control. Basics and Beyond*. John Wiley & Sons, 1996.

[124] R. Yavatkar and L. Manoj. Optimistic strategies for large-scale dissemination of multimedia information. In *Proc. of ACM Multimedia '93, Anaheim, California*, pages 1–8, August 1993.

Résumé

Dans cette thèse, nous avons abordé plusieurs problèmes de modélisation dans les réseaux dans le but d'estimer certains paramètres ou d'évaluer les performances de certains mécanismes. Le premier sujet traité concerne les applications point-à-point qui adaptent leur débit à l'état du réseau. Une façon de faire consiste à estimer la capacité de la mémoire et l'intensité du trafic transverse au nœud le plus congestionné de la connexion. À cette fin, nous avons développé deux modèles d'inférence basés sur les files $M/M/1/K$ et $M/D/1/K$. Onze schémas différents, permettant l'estimation des paramètres cités ci-haut, ont été mis en place et leurs performances ont été évaluées et comparées. Nous avons indentifié le meilleur de ces schémas grâce à des simulations réalisées avec `ns-2`. Le deuxième sujet traité concerne l'estimation en ligne de la taille des groupes multipoints. Dans un premier temps, nous avons modélisé le groupe multipoint par une file d'attente $M/M/\infty$ et avons montré qu'en trafic fort le problème d'estimation peut être résolu à l'aide d'un filtre de Kalman. Dans un deuxième temps, nous avons utilisé le même modèle mais sans l'hypothèse d'un trafic fort. L'estimation est obtenue grâce à un filtre de Wiener. Nous avons également construit un estimateur dans le cas d'un modèle $M/H_2/\infty$. Les performances de ces estimateurs ont été évaluées et comparées grâce à des traces réelles. Le troisième sujet étudié concerne des applications à code mobile dans lesquelles un agent en cours d'exécution peut changer de hôte. Différents mécanismes de localisation d'agents mobiles existent. Dans le cadre de cette thèse, nous avons modélisé deux de ces approches à l'aide de chaînes de Markov afin d'évaluer le temps de localisation d'un agent. Ayant validé nos modèles à l'aide de simulations et d'expérimentations autant sur un LAN que sur un MAN, nous avons pu utiliser ces modèles pour comparer formellement les performances des mécanismes étudiés.

Mots-clés — Files d'attente $M/M/1/K$, $M/D/1/K$ et $M/G/\infty$; chaîne de Markov; mesures; estimation en ligne; multipoint; processus de diffusion; filtres de Kalman et de Wiener; code mobile; migration; répéteurs; serveur de localisation; validation; distribution empirique; simulations; experimentations.

Abstract

In this thesis, we have looked at several modeling problems to estimate some parameters or to evaluate the performance of some mechanisms. The first subject studied is about unicast applications performing rate control to adapt to network conditions. To that end, we have proposed to estimate the cross traffic intensity and the buffer size at the bottleneck of a connection. We have developed two inference models based on the $M/M/1/K$ and $M/D/1/K$ queues, and derived eleven schemes for estimating the above-mentioned characteristics. The performance of these schemes have been evaluated and compared using `ns-2` simulations, and the best scheme has been identified. The second subject investigated is about the on-line estimation of the membership of dynamic multicast groups. We first model the multicast group by an $M/M/\infty$ queue and show that under heavy traffic the estimation problem can be solved using a Kalman filter. We next use the same model, but with a general traffic regime, and derive the estimator using Wiener filter theory. We last design an efficient estimator using the $M/H_2/\infty$ queue model. The performance of these estimators have been evaluated over simulations driven by both synthetic and real session traces. The third subject studied is about location mechanisms in a mobile agent environment. There are two widely used mechanisms to ensure communications between components of applications. We have proposed Markovian models to evaluate the cost of these mechanisms in terms of response times. We have validated our models both via simulations and experiments over a LAN and a MAN. We were then able to use the models to formally compare the performance of both mechanisms.

Keywords — Queueing; Markov chain; $M/M/1/K$, $M/D/1/K$ and $M/G/\infty$ queues; measurement; on-line estimation; multicast; diffusion; Kalman and Wiener filters; mobile code; migration; forwarders; location server; validation; distribution fit; simulations; experiments.