MODÉLISATION MARKOVIENNE DU TRAFIC DANS LES RÉSEAUX DE COMMUNICATION

THÈSE N^{O} 1479 (1996)

PRÉSENTÉE À LA SECTION DE SYSTÈMES DE COMMUNICATION

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

POUR L'OBTENTION DU GRADE DE DOCTEUR ÈS SCIENCES

PAR

Stephan ROBERT

Ingénieur électricien diplômé EPF

originaire des Ponts-de-Martel (NE), du Locle (NE) et de la Chaux-du-Milieu (NE)

acceptée sur proposition du jury:

Prof. J.-Y. Le Boudec, directeur de thèse
Dr F. Braun, corapporteur
Prof. R. Dalang, corapporteur
Dr R. Grünenfelder, corapporteur
Prof. J.-P. Hubaux, corapporteur
Prof. J. Walrand, corapporteur

Lausanne, EPFL
1996

Modélisation markovienne du trafic dans les réseaux de communication

A mon épouse Isabelle et à notre fils Josua

A mes parents

J'admire tous les ingénieurs, mais surtout le plus grand d'entre eux: DIEU

Thomas Edison, dans le livre d'or, au pied de la tour Eiffel à Paris

Remerciements

Tout d'abord, j'aimerais remercier le **Professeur Jean-Yves Le Boudec** qui m'a appris ce qu'était la vraie recherche. Il m'a en outre permis de rencontrer de nombreuses personnalités importantes dans le monde scientifique et de participer à des congrès prestigieux pour y présenter nos résultats. Je suis donc tout particulièrement redevable à mon directeur de thèse, un des meilleurs experts mondiaux dans les réseaux ATM, pour l'encadrement intellectuel et humain qu'il m'a offert et pour le temps qu'il a passé avec moi pour discuter de modélisation, de réseaux, mais aussi de sujets non techniques. Ça a été pour moi un privilège énorme que d'avoir pu travailler au sein de son laboratoire. D'autre part, j'aimerais remercier le **Professeur Jean-Pierre Hubaux** qui m'a donné la possibilité de travailler dans un domaine passionnant et pour la liberté qu'il m'a laissée lorsque j'étais au sein du laboratoire de télécommunications. Je tiens par ailleurs à le remercier pour sa participation au jury de cette thèse. Il y a une autre personne, à qui je dois beaucoup car il a passé beaucoup de temps à m'initier au télétrafic et qui m'a aidé au début de mes recherches, il s'agit du **Docteur Reto Grünenfelder**. C'est d'ailleurs en grande partie grâce à lui, et au **Professeur Pierre-Gérard Fontolliet** que je me suis lancé dans le domaine du télétrafic. Au travers de ces quelques lignes, je tiens à le remercier pour avoir accepté de faire partie du jury de cette thèse. Le **Professeur Jean Walrand** de l'université de Californie, Berkeley et Professeur invité à l'EPFL (1995-96), personnalité universellement reconnue dans le monde du télétrafic, détenteur du prestigieux et très convoité Prix Lanchester, m'a toujours acceuilli dans son bureau pour des échanges qui se sont avérés très fructueux. Il a été notre "référence" lorsque nous n'étions pas sûrs de nos calculs ou de nos affirmations. Je le remercie de m'avoir donné la possibilité de m'intégrer dans son équipe de recherche à Berkeley pour deux ans et d'avoir accepté de faire partie du jury de cette thèse. Le soutien financier de cette recherche a été apporté par les **Telecom PTT** suisses et plus particulièrement par le groupe du télétrafic dirigé par le **Docteur Fritz Braun**. Au cours de nombreux rendez-vous, à Berne et à Lausanne, nous avons eu l'occasion d'avoir de fructueux échanges. J'aimerais également le remercier pour sa participation au jury de cette thèse. Merci également au **Professeur Robert Dalang** pour sa participation au jury de cette thèse et pour les

conseils qu'il m'a donnés au cours de nos rares rencontres. **Sam Manthorpe** a été mon collègue de bureau pendant tout mon séjour au laboratoire des télécommunications et au laboratoire des réseaux de communication. Nous avons subi trois déménagements et un changement de laboratoire. Il a été un collègue très agréable et chaleureux.

Le laboratoire des réseaux de communication me laisse une excellente impression. Parmi tous les laboratoires et équipes d'ingénieurs que j'ai visités, c'est certainement celui que j'ai senti le plus soudé et ayant le moins de problèmes au niveau humain. Ça a été pour moi une grande joie de pouvoir en faire partie. Merci donc à **Werner Almesberger, Olivier Crochat, Hans Einsiedler, Eric Gauthier, Silvia Giordano, Sam Manthorpe, Dr Philippe Oechslin**. L'équipe du télétrafic du laboratoire des télécommunications (**César Gallego, Xavier Garcia, Laurent Jaussi, Yves Gachoud** et **Dr Rasti Slosiar**) ainsi que leur système manager **Bruno Dufresne**, alias root, m'ont laissé un très bon souvenir au travers des échanges que nous avons eus.

Au cours de conférences et de rencontres, j'ai eu l'occasion de rencontrer plusieurs personnalités que j'aimerais remercier pour leurs encouragements et leurs précieux conseils: **Prof. Chris Blondia** de l'université de Antwerp, **Prof. Hans Daduna** de l'université de Hambourg, **Dr Patrick Droz** du laboratoire de recherche IBM à Rüschlikon, **Dr Ashok Erramili** de Bellcore, **Prof. Ulrich Erzog** de l'université d'Erlangen, **Dr Annie Gravey** de France Telecom, **Dr Frank Hübner** de Bellcore, **Prof. Hisashi Kobayashi** de l'université de Princeton, **Dr Udo Krieger** de Deutsche Telekom, **Prof. Guy Latouche** de l'université libre de Bruxelles, **Prof San Qi Li** de l'université du Texas à Austin, **Dr David Lucantoni** de AT&T, **Prof. Raymond Marie** de l'université de Rennes, **Prof. Bo Friis Nielsen** de l'université technique du Danemark, **Dr Ilkka Norros** de VTT en Finlande, **Prof. Jean Pellaumail** de l'université de Rennes, **Dr Vaidyanat Ramaswami** de Bellcore, **Dr Jim Roberts** de France Telecom, **Prof. Johann Christoph Strelen** de l'université de Bonn, **Prof. Phuoc Tran-Gia** de l'université de Würzburg, **Dr Jorma Virtamo** de VTT en Finlande, **Prof. Bernard Ycart** de l'université de Grenoble.

Dans le domaine privé, j'aimerais tout d'abord remercier ma chère et tendre épouse **Isabelle**, ma confidente qui m'a beaucoup aidé par son écoute, sa compréhension et

ses encouragements. Elle a d'ailleurs contribué au travail de cette thèse en dactylographiant la grande partie du texte. Je tiens à remercier **Josua**, mon fils, pour sa bonne humeur constante, sa gentillesse et sa joie de vivre. Merci également à **mes parents** qui m'ont permis de choisir le métier qui me plaisait. Par ces quelques lignes, je tiens encore à remercier une personne qui m'a encouragé tout au long de mes études, depuis ma tendre enfance jusqu'à l'école polytechnique et à qui j'aimerais rendre hommage car malheureusement elle n'est plus de notre monde; il s'agit de **Mme Isabelle Fiala**. Enfin j'aimerais remercier mon **oncle Alain (Robert)**, illustre Professeur de mathématiques à l'université de Neuchâtel, qui a toujours été un modèle pour moi.

Version abrégée

La question principale à laquelle ce travail tente de répondre est la suivante: "la modélisation markovienne a-t-elle encore un sens dans l'analyse du trafic des réseaux de communication? " et nous y répondons par l'affirmative. Cette question a été motivée par la remise en question, par une équipe de chercheurs à Bellcore aux USA de la validité des modèles utilisés jusqu'alors dans le télétrafic. Cette remise en question est étayée par l'analyse d'une longue série de mesures effectuées sur leur réseau local. Leurs résultats dont la conclusion principale est que le trafic de données est auto-similaire, ont ébranlé toute la communauté du télétrafic. Après avoir exposé les principales caractéristiques du modèle markovien modulé, nous nous intéressons aux mesures faites sur le réseau de l'EPFL et à Bellcore qui viennent mettre à défaut les modèles du type markovien, tels qu'ils ont été présentés. A partir de là, une étude sur la passerelle Ethernet/DQDB est présentée en admettant que nous ne puissions faire qu'une hypothèse quant au profil du trafic LAN, à savoir que le réseau est complètement chargé. Comme cette hypothèse est drastique, nous avons essayé, par l'analyse spectrale, de comprendre le comportement de la file d'attente afin d'essayer de caractériser de manière plus fine le trafic de données. Au pas suivant, nous avons essayé de reproduire les auto-similarités observées dans le trafic de données à l'aide de chaînes de Markov modulées. Pour cela, la théorie de la décomposabilité de Courtois a été utilisée. Bien que nous ayons pu reproduire du trafic auto-similaire sur un horizon fini à l'aide de chaînes de Markov modulées, le nombre de paramètres à manipuler reste très important, c'est pourquoi nous proposons une technique pour les réduire. Une méthode pour trouver une chaîne de Markov modulée en fonction du paramètre de Hurst local, de l'espérance mathématique et du domaine sur lequel la chaîne de Markov a un comportement de type auto-similaire a été développée. Finalement, nous analysons le problème du multiplexage statistique dans une nouvelle architecture en comparant le multiplexage de sources de classe VBR (à l'aide de la source markovienne produisant du trafic auto-similaire) sur une connexion de classe CBR et sur une connexion de classe VBR.

Abstract

The principal question we address in this work is the following: " Is Markov modelling appropriate to analyse data networks ? ". We will conclude that it is. The motivation for this question comes from the work of a research team at Bellcore who argue that the models used in teletraffic engineering are questionnable. Their argument is based on long measurements performed on their local network. Their results, with the major conclusion that the traffic is self-similar, have shaken the teletraffic community. After having exposed the principal characteristics of the Markovian model, we analyse measurements done on the EPFL network and on the Bellcore network. The results of the analysis lead us to suspect the suitability of Markovian models. Therefore, we study the interworking unit Ethernet/DQDB considering a network totally loaded. No hypothesis is made for the traffic profile, yet the study leads to strong conclusions and we therefore try to understand the queue behavior with the spectral analysis in order to better characterize the data traffic. The next step consists of reproducing the self-similarity observed in the data traffic with modulated Markov chains. Here, Courtois's theory of decomposability is used. We have shown that it is possible to reproduce self-similarity on a finite timescale, but that the number of parameters needed to manipulate in the modulated Markov remains large. In the next step, we propose a technique to reduce them drastically. A method of finding a Markov modulated chain as a function of the expectation, the local Hurst parameter and the domain where the process exhibits self-similarity has been developed. Finally, we analyse the statistical multiplexing in a new ATM architecture by comparing the multiplexing of VBR sources (with a modulated Markovian chains) on a CBR and on a VBR connection, using the above techniques.

Table des matières

Chapitre 1

Introduction

La première question qu'on peut se poser en tentant de faire de la modélisation est de savoir à quoi elle sert. Dans les lignes qui vont suivre, je vais essayer de donner un bref aperçu de son utilité. Tout d'abord, voyons ce qu'est un modèle. Le modèle d'un système est une idéalisation et une simplification de ce dernier. Lorsqu'on observe le système réel, on va négliger à priori certains de ses aspects dont on pense qu'ils sont peu importants quant au comportement qu'on désire étudier. Il faut être conscient que la plupart du temps, cette simplification est intuitive et n'est pas démontrée rigoureusement. Cette étape présente évidemment le risque de négliger d'importants éléments mais donne l'avantage de rendre le système plus facilement analysable. Après l'avoir décrit au moyen d'un nombre limité de paramètres, il devient possible d'étudier son comportement et ses propriétés. En principe, une fois modélisé, le système peut s'étudier de deux façons différentes, via la simulation ou via des calculs numériques ou analogiques. Le modèle doit donc dans la mesure du possible être performant, fiable et facile à utiliser. S'il ne reflète pas la réalité et n'apporte pas une bonne compréhension du système, il sera très probablement inutilisable.

1.1 La modélisation et la simulation en bref

La simulation peut s'effectuer selon deux schémas différents, en temps continu (on observe toutes les variables du modèle au cours de temps) ou en évènements discrets (on

observe les variables chaque fois qu'il se produit un changement). La simulation est un outil très puissant qui permet une modélisation plus fine du système à étudier mais elle présente néanmoins plusieurs inconvénients. La mise en oeuvre d'une simulation nécessite un gros effort de modélisation; il est en outre difficile d'utiliser cet outil efficacement et correctement. D'autre part, le modèle peut devenir extrêmement compliqué à étudier. Notons que la meilleure simulation qu'on puisse effectuer est de soumettre le système lui-même à une série d'expériences pour analyser son comportement. Cette approche est souvent impossible à réaliser car on a souvent besoin de connaître les réactions du système avant de pouvoir le construire et de le déployer efficacement.

Il y a un autre point à mentionner: en ayant modélisé le système au plus près de la réalité possible, une question se pose encore quant à son analyse car il ne faut pas oublier que le nombre de paramètres dont dépend le modèle peut devenir extrêmement grand. Il faut alors choisir lesquels on décidera de faire varier et sur quelle échelle. D'autre part, l'analyse des résultats est également délicate, notamment quand on fait intervenir des grandeurs statistiques (ce qui est la plupart du temps le cas pour l'analyse des réseaux informatiques). Les modèles simulés sont souvent utilisés pour vérifier l'exactitude de modèles plus simples, moins réalistes tels que les modèles analytiques. A l'autre extrême de la modélisation, nous trouvons les modèles analytiques qui nous fournissent une formule mathématique dans laquelle nous pouvons substituer les caractéristiques du système en question. A l'aide de cette technique, nous pouvons évaluer assez rapidement la performance du système. La sophistication réside dans la manière d'obtenir la formule mathématique. A cause de cela, il est indispensable de simplifier le système plus que ce qu'il est nécessaire de le faire avec la simulation. Une fois que la formule mathématique (ou algorithme) est obtenue, il est aisé de l'utiliser. Cette approche a beau être élégante, elle peut devenir trop simplificatrice face au système à étudier.

Nous voyons, pour résumer, que la simulation d'un système est plus crédible. Néanmoins le modèle analytique nous donne une meilleure intuition du comportement du système et nous permet souvent d'explorer des terrains inabordables par la simulation. Ainsi, nous nous trouvons souvent face à un compromis, bien qu'un modèle complexe ne soit pas nécessairement plus réaliste qu'un modèle simple. Nous ne voulons en général pas

dès le départ considérer un modèle très sophistiqué. Non seulement, il est difficile à implanter ou à calculer mais en plus les résultats qu'il va nous fournir sont difficiles à interpréter. Au lieu de cela, nous préférons commencer avec un modèle simplifié qui pourra être perfectionné par la suite. Une fois le modèle implanté, il faudra le soumettre à la vérification ainsi qu'à sa validation. La vérification consiste à contrôler son modèle, à le soustraire de toutes les erreurs qui auraient pu s'y glisser (erreurs fonctionnelles, de programmation, utilisation de données erronées). La validation est l'étape qui va suivre la vérification. On admet que le modèle est libéré de toutes ses erreurs alors la question se pose quant à sa qualité, sa capacité d'approcher la réalité avec suffisamment de précision. A cause des approximations qui ont été faites sur le modèle, ce dernier ne peut se comporter de manière identique au système réel. Il est alors important de pouvoir interpréter les différences entre les deux comportements. Bien que la vérification soit séparée de la validation, ces deux opérations vont s'effectuer de manière itérative puisque lors de la validation, il est possible d'introduire des erreurs qui ne la concernent pas, mais qui concernent la vérification. Cette étape peut entraîner une modification du modèle. Il faut être conscient du sacrifice que cela peut impliquer. Il est parfois plus facile de fermer les yeux plutôt que de remettre l'ouvrage sur le métier. Il faut dire que le pouvoir de séduction d'un modèle peut être très important pour son concepteur qui peut en tomber amoureux à tel point qu'il accorde une priorité absolue à la résolution exacte de son modèle plutôt que de se préoccuper si ce dernier reflète la réalité qu'il est censé approcher. D'autre part, si la complexité du système est trop grande, alors ce processus ne convergera certainement que très lentement, d'autant plus que la validation peut s'avérer très difficile, notamment dans le contexte des réseaux informatiques. Pour la méthode analytique, la simulation sert à la vérification des calculs alors que l'inverse ne peut être vrai. Ainsi, les résultats de la simulation ne vont pas nous renseigner quant à la validation du modèle mais seulement quant à sa vérification. Voilà en résumé le parcours qu'il faudra suivre pour arriver à obtenir de bons résultats. Néanmoins, le chemin est difficile car chacune des étapes est semée d'embûches qu'il faudra essayer d'éviter. Dans la section suivante, nous verrons quelles sont les particularités des réseaux de télécommunication et quelles sont les mesures de performances avec lesquelles on est

capable de les évaluer.

1.2 Les réseaux de communication

Les réseaux de communication, comme les autoroutes, sont chargés de convoyer du trafic. Dans les réseaux de communication, il est constitué d'appels téléphoniques, de données informatiques (conversation entre ordinateurs, requêtes à certaines bases de données, sessions interactives) ou de trafic vidéo (télévision, *vidéo on demand*, transfert d'images statiques, télémédecine). Lorsque le trafic augmente, il faut sans cesse redimensionner le réseau faute de quoi ce dernier risque d'être complètement encombré. Sur les autoroutes, cet encombrement se traduira par un certain nombre de "bouchons" tandis que dans les réseaux de communication l'usager aura souvent ses demandes de connexions refusées (ligne occupée) ou le débit de ses connexions ralenti. Dans un cas comme dans l'autre, les utilisateurs se sentent frustrés et agacés par le comportement du réseau. Pour satisfaire les besoins des utilisateurs du réseau, leurs concepteurs sont jour après jour confrontés à des problèmes qui ne peuvent pas se résoudre au hasard mais qui nécessitent l'appui de la science du télétrafic. Cette science a pour objectif d'aider les concepteurs à prédire le comportement de leur réseau, à comparer différentes topologies ou différents systèmes, à identifier des points d'encombrement, à caractériser la charge de leur réseau, à déterminer le nombre et la taille des composants nécessaires (planification) et même à prédire la performance de leur réseau sous des charges futures. Le but ultime est de pouvoir optimiser le réseau au niveau de son architecture de manière à satisfaire les standards de qualité de service requis en empruntant la voie la plus économique possible. Pour résumer, la science du télétrafic est largement concernée par la relation qui existe entre le volume de trafic, la capacité du réseau et la performance réalisée [1]. Dans la section suivante, nous allons voir que la science du télétrafic a beaucoup contribué au développement des réseaux de communication mais que dans certains cas, elle n'a pas su proposer de solution fiable.

1.3 Exemples de succès et d'échecs du passé

Dès les débuts de la commutation, les concepteurs de réseaux ont été confrontés à des problèmes de dimensionnement. Il est intéressant de savoir que la théorie des files d'attente est née dans ce contexte. Erlang a été incontestablement le pionnier en ce qui concerne les problèmes de dimensionnement de réseaux; ses formules sont universellement connues. En 1909, il a publié "The Theory of Probabilities and Telephone Conversations" [2] alors que la théorie des probabilités n'avait pas encore été bien développée. Cette théorie était alors très peu populaire. On peut dire que Erlang, alors employé par la compagnie des téléphones danois à Copenhague, a eu une intuition remarquable. Au cours de ses études ultérieures (1917) [3], il a observé que les systèmes téléphoniques étaient caractérisés par des entrées poissoniennes. Erlang a en outre développé les premières équations d'équilibre (*balance-of-state*) de la théorie des files d'attente. Le travail s'est poursuivi notamment avec les travaux de Molina [4] qui a publié "Application of the Theory of Probability to Telephone Trunking Problems" et qui a poursuivi les travaux d'Erlang. La science du télétrafic a joué un rôle fondamental dans le développement du réseau téléphonique. Elle peut même être considérée comme un des grands succès de l'application de la modélisation mathématique. Il y a beaucoup de raisons à ce succès mais les principales sont que la théorie du télétrafic a pu produire des modèles simples qui ont pu prédire les principales mesures de performance intéressantes tout en ne requiérant que peu d'informations (débit des arrivées, temps de services) qui sont faciles à estimer en pratique. Dans les années 1930 [5, 6], Pollaczek a calculé un modèle ayant des entrées poissoniennes et des temps de service arbitraires pour un ou plusieurs serveurs. Durant la même période, Feller [7] a introduit le concept de processus de naissance et de mort qui a permis à la théorie des files d'attente de faire son entrée dans le monde des mathématiques. Plus tard, d'autres personnes ont beaucoup contribué au développement de cette théorie: Kolmogorov et Kintchine en Russie, Crommelin en France, Palm en Suède. D'autre part, de grands progrès ont été effectués durant la deuxième guerre mondiale. Malgré les avancées notoires au point de vue théorique de cette nouvelle discipline, elle a stagné quelques temps avant d'avoir été littéralement relancée avec l'apparition

des réseaux informatiques. La nature du trafic rencontré sur ce genre de réseaux diffère radicalement par rapport au trafic téléphonique. Les modèles utilisés alors sont inapplicables et pourtant les progrès qui ont été faits dans ce domaine nous ont permis de mieux comprendre le comportement de files d'attentes ou de structures de files d'attente sous certaines conditions particulières de trafic, de développer de nouveaux modèles de source, d'analyser certaines architectures. Ces études, en général, ont conduit à considérer des modèles d'une grande sophistication, ce qui n'était pas le cas pour la téléphonie. Mais ces avances de la théorie du télétrafic n'ont que peu contribué au déploiement des réseaux informatiques en comparaison des efforts fournis. Ceci est principalement dû à la complexité inhérente au trafic des réseaux de données.

A ce sujet, Boggs a écrit un article qui est devenu célèbre [8] bien qu'il ne traite que du réseau de données Ethernet. Notons tout de même que c'est la technologie qui certainement a eu le plus de succès. A l'époque, beaucoup d'études de performances ont été faites sur Ethernet [8, 9, 10, 11, 12] et il en a résulté une grande confusion concernant la capacité réelle du réseau. Comme nous l'avons décrit au cours de la section 1.1, l'étape de la validation permet de confronter les résultats obtenus par la modélisation avec ceux qui sont mesurés. C'est dans cette étape qu'un certain nombre de problèmes sont soulevés par Boggs. Ce qu'il dénonce principalement dans son article, ce sont les mythes qui se sont créés à propos des performances d'Ethernet suite à une mauvaise interprétation des analyses ou suite à des modèles trop réducteurs. Le mythe le plus connu est certainement celui de la saturation d'Ethernet à un trafic offert de 37%. C'est en effet ce qui se passe sous des hypothèses très pessimistes qui ne se rencontrent presque jamais en réalité. Boggs a montré qu'Ethernet était capable de fonctionner sous un régime beaucoup plus important en reproduisant des conditions réalistes d'expérimentation. Shoch et Hupp [13] ont également fait des mesures sur un réseau Ethernet expérimental de 3 Mb/s et ont également observé qu'il était possible d'utiliser le réseau à près de 100% (pour 64 stations) à condition de ne transmettre sur le réseau que des paquets de grande taille (512 octets) alors que pour de petits paquets, l'utilisation approchait $1/e$ pour un grand nombre de stations. Le deuxième exemple, que nous développerons à la section suivante, est celui des mesures qui ont été effectuées à Bellcore de 1989 jusqu'à nos jours (1996)[14].

1.4 Mesures de Bellcore

Les articles [15, 16, 14] de l'équipe de Bellcore ont eu un effet détonant dans la communauté du télétrafic. Ils mettent en accusation les modèles classiques utilisés pour l'étude des réseaux informatiques. C'est en comparant une série de mesures très précises effectuées sur un réseau Ethernet avec plusieurs modèles utilisés pour l'analyse des réseaux qu'ils fondent leurs accusations. Dans leur premier article [15], ils commencent par décrire comment les mesures ont été effectuées. Wilson a construit une carte avec laquelle il est capable de mesurer le temps d'arrivée d'un paquet avec une précision de $20\mu s$ [14]. Une partie de leurs mesures est mise à disposition du public (à l'aide d'un transfert *ftp*, avec un nom d'utilisateur *anonymous* à l'adresse suivante: `bellcore.flash.com` dans le répertoire `pub/lan`) : les fichiers (`pAug.TL, pOct.TL, OctExt.TL, OctExt4.TL`) contiennent chronologiquement le temps d'arrivée ainsi que la taille de chaque paquet observé sur le réseau. Après avoir fait de longues mesures sur le réseau et analysé les arrivées de paquets, ils ont remarqué que le trafic était extrêmement fluctuant au cours du temps, et ceci pour des échelles de temps s'étendant jusqu'à 6 ordres de grandeur. Ils ont comparé ce comportement avec celui des modèles de trafic conventionnels en observant leurs granularités (*burstiness*) respectives. Pour cela, ils ont mesuré le rapport entre la largeur de bande maximale sur la largeur de bande moyenne, le coefficient de variation et l'indice de dispersion. L'observation est la suivante: les modèles conventionnels tels que les processus de Poisson, *batch* Poisson, hyperexponentiels, markoviens modulés ont très peu de variabilité à long terme. Par contre, de façon très contrastante, le trafic observé circule en rafales (*bursty*) sur plusieurs échelles de temps s'étendant de l'ordre de la milliseconde à plusieurs heures. Dans un deuxième article [16], ils montrent que les modèles conventionnels ne sont pas à même de reproduire le trafic observé sur un *Local Area Network* (LAN), ce qui a un sérieux impact sur l'analyse de la congestion de trafic. A ce titre les auteurs ont écrit un programme pour effectuer la simulation d'un modèle ayant un réseau de LAN connectés entre eux par des commutateurs *Asynchronous Transfer Mode* (ATM) et fournissant un service sans connexion. Chaque LAN représente une source et utilise les données mesurées pour reproduire du trafic réaliste. La simulation

montre que les périodes de congestion peuvent être longues. Pendant ces périodes de congestion, les pertes peuvent être considérables. Les auteurs indiquent également qu'un sous-dimensionnement des tampons (*buffers*) peut avoir de fâcheuses conséquences quant au taux de perte. L'augmentation à outrance des tampons n'est pas une solution viable non plus car ils ne fournissent pas une protection assez efficace contre les périodes de congestion. D'autre part, les retards deviennent très importants. Les auteurs concluent que les réseaux futurs devraient être capables d'accroître leur capacité pour pouvoir répondre à une demande plus ou moins imprévisible de l'usager. Dans un troisième article [14], célèbre également, les membres de l'équipe de Bellcore proposent une analyse statistique plus approfondie que celle effectuée dans le premier article. Ils ont trouvé que le comportement du réseau avait des propriétés statistiques similaires, quelle que soit l'échelle de temps à laquelle on l'observe. Autrement dit, ils arrivent à la conclusion que le trafic est auto-similaire sur plusieurs échelles de temps. D'après les auteurs, aucun des modèles classiques ne peuvent reproduire du trafic auto-similaire. Ils rejettent la modélisation markovienne en argumentant que même si elle était possible (en approximant la décroissance hyperbolique de la fonction d'autocorrélation par une somme d'exponentielles), elle serait beaucoup trop compliquée car le nombre de paramètres ($O(n^2+n)$) devient énorme dès que le nombre d'états n de la chaîne de Markov devient grand. D'autre part la signification physique à donner à tous ces paramètres devient de plus en plus difficile. Alors ils proposent deux nouvelles méthodes pour modéliser le comportement auto-similaire du trafic LAN. Une méthode est basée sur des processus stochastiques auto-similaires, les mouvements Browniens fractionnaires (fBm, *fractional Brownian motion*) et les *fractional autoregressive integrated moving average*, (ARIMA) et l'autre sur des *deterministic nonlinear chaotic maps*. L'avantage de ces modèles est qu'ils ne dépendent que d'un nombre très restreint de paramètres. Les implications de la nature auto-similaire du trafic sur l'évaluation de la performance de réseaux à hauts débits sont très importantes. Les modèles de source doivent refléter une très grande variabilité, ce qui n'est pas le cas pour les modèles de trafic conventionnels. D'autre part, ils proposent également d'évaluer la granularité du trafic à l'aide du paramètre de Hurst.

1.5 Mouvance actuelle

Nous voyons que les conséquences sont graves pour toute la science du télétrafic car tous les modèles utilisés jusqu'alors (basés sur des chaînes de Markov principalement) pourraient se révéler être invalides. A ce titre, Partridge, éditeur de *IEEE Network Magazine*, écrit une note sur les découvertes de l'équipe de Bellcore sous le titre "The End of Simple Traffic Models" en septembre 1993 [17]. En commentant l'article des chercheurs de Bellcore, il constate que l'implication majeure de leurs résultats est que les modèles de télétrafic qu'on a utilisé dans le passé sont terriblement faux et qu'il faut arrêter de penser en terme de "processus de Poisson" mais qu'il faut sérieusement remettre l'ouvrage sur le métier et considérer d'autres modèles, plus complexes et d'essayer de comprendre comment se comporte le trafic de données car aujourd'hui, on doit prendre des décisions pour la configuration des réseaux de demain et il serait préjudiciable de les baser sur des modèles inadéquats. De nombreux chercheurs ont déjà investigué cette nouvelle voie, comme par exemple Norros [18] qui a étudié le comportement d'une file d'attente étant alimentée par du trafic fBm. Il a trouvé que la distribution de l'occupation du tampon (*buffer*) pouvait être approximée par une distribution de Weibull. Droz [19] présente un générateur de trafic basé sur le fBm qui peut facilement être parallélisé en vue de générer un très grand nombre d'échantillons rapidement. Erramilli [20], chez Bellcore, investigue la voie des *chaotic maps*. Veitch [21] a présenté une nouvelle classe de modèles à la conférence *Globecom '93* qui sont capables de générer du trafic auto-similaire. Avec Roberts [22], Brichet et Simonian, il a présenté des résultats au meeting *COST 242* en considérant le comportement d'une file d'attente sous un certain nombre de sources ON-OFF identiques, à variance infinie et à moyenne non nulle. Dernièrement, Paxson [23] a également proposé une méthode basée sur la transformée de Fourier pour synthétiser du trafic auto-similaire. Bien que la génération de trafic auto-similaire soit une chose relativement aisée, il n'en va pas de même pour son étude ultérieure car les calculs peuvent se révéler très compliqués.

1.6 Problème étudié

Dans le contexte très animé que nous avons évoqué dans le précédent paragraphe, nous allons essayer de répondre à la question suivante: "la modélisation markovienne a-t-elle encore un sens dans l'analyse du trafic des réseaux informatiques? ". Nous verrons dans les chapitres qui suivent que nous pouvons répondre par l'affirmative. Le modèle markovien qui a été trouvé ne dépend que de trois paramètres et permet de représenter du trafic ayant les caractéristiques du trafic mesuré par les chercheurs de Bellcore. Le modèle trouvé a le grand avantage d'être simple et facilement manipulable. Il est possible de réutiliser les théories des files d'attente qui ont été développées dans le passé pour évaluer la performance des réseaux de communication.

1.7 Structure du travail de thèse

Après avoir brièvement introduit les réseaux informatiques (chapitre 2) qui ont fait l'objet de nos recherches, nous exposons le modèle markovien (chapitre 3) que nous utilisons ainsi que ses propriétés. Ensuite (chapitre 4), des mesures effectuées à l'EPFL et à Bellcore viennent mettre à défaut les modèles du type markovien, tels qu'ils ont été présentés. A partir de là (chapitre 5), une étude sur la passerelle Ethernet/*Distributed Queue Dual Bus* (DQDB) est présentée en admettant que le réseau soit complètement chargé; ainsi, aucune hypothèse n'est faite quant au profil du trafic LAN. Comme cette approche est drastique, nous avons essayé, par l'analyse spectrale (chapitre 6), de comprendre le comportement de la file d'attente. Au pas suivant (chapitre 7), nous avons essayé de reproduire les auto-similarités observées dans le trafic de données à l'aide de chaînes de Markov modulées. Bien que nous ayions pu reproduire du trafic auto-similaire sur un horizon fini à l'aide de chaînes de Markov modulées, le nombre de paramètres à manipuler reste très important, c'est pourquoi nous proposons une technique pour les réduire notablement (chapitre 8). De plus, une méthode pour trouver une chaîne de Markov modulée en fonction du paramètre de Hurst local, de l'espérance mathématique et du domaine sur lequel la chaîne de Markov a un comportement de type auto-similaire a été développée. Finalement, le problème du multiplexage statistique est analysé à l'aide

de sources markoviennes produisant du trafic auto-similaire.

1.8 Cadre du travail de la thèse

Cette thèse a pu être réalisée grâce au financement des **Telecom PTT** suisses. La collaboration avec l'équipe du Dr Braun aux PTT s'est faite au travers de deux projets de deux ans chacun (F+E 240 et RD 303). Le premier projet s'intitule "Interconnexion de réseaux à large bande: analyse des performances de trafic et conception de mécanismes de contrôle" et le second "Analyse des performances de trafic dans les réseaux à large bande".

Chapitre 2

Trafic issu de réseaux informatiques

2.1 Introduction

Dans ce chapitre, nous allons voir les principes des réseaux qui vont directement nous concerner par la suite, ATM et DQDB. Le but n'est pas d'examiner leur comportement en détail mais plutôt de les approcher sous l'angle de la modélisation. Dans l'introduction, nous avons mentionné le problème que cette étape soulevait dès qu'on désire simplifier le système. On va décrire ici les comportements des différents réseaux qui nous semblent importants ou plutôt qui ont semblé importants aux personnes qui ont créé des modèles permettant de les étudier.

2.2 ATM

ATM est une technologie de multiplexage et de commutation choisie par l'*International Telecommunication Union* (ITU) pour le futur réseau à large bande à intégration de services, *Broadband Integrated Service Digital Network* (B-ISDN). Avant d'expliquer le fonctionnement d'ATM, commençons par exposer les raisons qui ont motivé son développement car ATM a été défini pour répondre à un besoin des télécommunications. Le premier mode de transfert qui a été utilisé dans le monde des télécommunications a été le mode de transfert par paquets avec la télégraphie. Le message (ou paquet) transportait une adresse de source et de destination ainsi que le contenu du message. L'achemine-

ment du message à la bonne destination nécessitait l'apport de l'intelligence humaine puisque des opérateurs étaient chargés de la commutation. A la télégraphie a succédé la téléphonie (c'est en 1876 qu'Abraham Graham Bell a déposé un brevet qui a marqué le début du téléphone). La commutation est faite manuellement par des opératrices et la liaison est maintenue durant la totalité de la conversation (commutation de circuits). Mais en 1891, Almon Strowger invente un sélecteur automatique télécommandé par le poste de l'abonné. A l'aide de différents milieux de transmission, le réseau téléphonique est l'unique réseau qui permet d'acheminer des informations d'un bout à l'autre de la planète, celui-ci se développant rapidement. Mais autour des années 1960, le besoin de connecter différents ordinateurs entre eux se fait ressentir. Au début, on a utilisé le réseau téléphonique mais on s'est vite rendu compte qu'il existait de meilleures solutions, mieux adaptées au trafic de données qui circule en rafales (*bursty*). Ainsi sont nés de nouveaux réseaux tels que *Advanded Research Projects Agency of the US Department of Defense* (ARPANET), X.25, et plus tard: Ethernet, *token ring, token bus* (la liste est très longue, seulement quelques exemples sont donnés ici [24]). D'un autre côté, la distribution de la télévision se fait en partie par un réseau en arbre appelé réseau *Community Antenna TV* (CATV). Ainsi, plusieurs réseaux publics et privés coexistent et ont le grand désavantage d'être très spécialisés, attachés à un type de service uniquement. Par exemple, le transfert de la voix sur X.25 poserait de gros problèmes à cause des retards introduits de bout-en-bout du réseau. De plus, chaque réseau est optimisé pour son propre service; le dimensionnement de chaque réseau est basé sur des conditions de trafic considérant l'heure la plus chargée. Or, pour le service téléphonique, l'heure la plus chargée se situera entre 8h00 et 17h00 alors que pour le réseau CATV, elle se situera entre 18h00 et 23h00. Comme les réseaux sont indépendants, un partage de ressources n'est pas possible. Un autre inconvénient de cette architecture est son manque de flexibilité; un réseau spécialisé a beaucoup de peine à satisfaire aux exigences d'un nouveau service plus évolué. Il faut donc que le nouveau réseau à large bande soit indépendant des services, fiable, flexible, efficace dans la gestion des ressources et bon marché. C'est le défi qui est lancé au futur réseau à large bande qui devra intégrer tous les différents types de services existants et futurs. L'ITU a choisi ATM comme technologie de multiplexage et de commutation

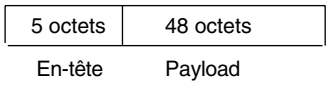

Figure 2.1: Structure de la cellule ATM

pour le futur réseau à large bande avec intégration de services B-ISDN [25]. La technique ATM est basée sur une transmission de l'information par paquets de longueur fixe qu'on appelle **cellules**. Sur un lien ATM, on a une succession de créneaux qui sont soit vides, soit pleins (ils contiennent une cellule dans ce cas). Une cellule, montrée à la figure 2.1, est composée de deux champs, d'un en-tête et d'un champ d'information (*Payload*). Pour garantir un traitement rapide au travers du réseau, l'en-tête a un nombre restreint de fonctionnalités. Toutes les cellules appartenant à une même communication sont acheminées de noeud en noeud à travers le réseau selon le même itinéraire, mis en place lors de l'établissement de la connexion et maintenu par un lien logique pendant toute sa durée et libéré à la fin de la communication. Le choix de la grandeur de la cellule (53 octets) est le résultat d'un grand débat au sein des organismes de standardisation (ITU) [26]. En fait 48 octets d'information utile est le résultat d'un compromis entre 32 octets (position européenne qui tendait à favoriser le trafic de la voix) et 64 octets (position américaine qui préférait favoriser le transfert de données). L'en-tête est un label et non une adresse explicite; une adresse explicite n'est pas envisageable, à cause de la taille de la cellule. Il y a une différence fondamentale entre cette technologie et la téléphonie classique au point de vue du télétrafic car si la téléphonie classique considère l'appel comme entité de base, il n'en va pas de même avec la technologie ATM. Avec la technologie ATM, on va partager la bande passante entre tous les utilisateurs. La capacité de transmission sera attribuée aux usagers que lorsque ces derniers en auront réellement besoin. ATM est une technologie basée sur la commutation de paquets de longueur fixe qui sont expédiés dès que le canal est libre. Dans chaque noeud de commutation, un paquet peut être stocké plus ou moins longtemps. Ainsi, le retard que subira une cellule peut varier, entre autres, en fonction de la charge du réseau. L'ATM permet ainsi de mélanger sur un même canal des services dont le taux d'activité (exprimé en cellules par seconde) peut être très différent de l'un à l'autre.

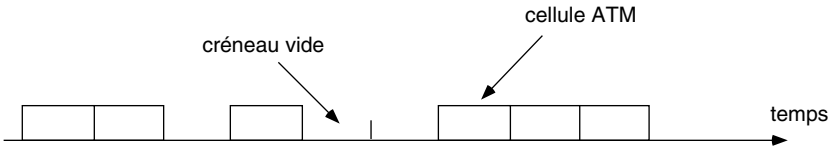

Figure 2.2: Suite de cellules ATM

ATM doit fournir aux usagers une certaine qualité de service (QoS, *Quality of Service*) pour différentes classes de services. La qualité de service est généralement exprimée en termes de retards et de pertes de cellules. Le transfert de données sera en général plus sensible aux pertes de cellules qu'aux retards de ces dernières alors qu'un service supportant de la voix sera plus sensible aux retards des cellules qu'à leurs pertes par exemple. Pour préserver la qualité de service au sein du réseau, il est nécessaire de le protéger. Un contrat de trafic est négocié entre l'usager et le réseau ATM à chaque établissement de connexion. Suivant les paramètres déclarés (largeur de bande moyenne requise, largeur de bande maximale requise), une fonction de contrôle d'acceptation d'appel (CAC, *Connection Acceptance Control*) décide si l'appel est accepté ou non suivant les paramètres déclarés. La décision est prise sur la base des caractéristiques du trafic et les exigences du QoS du nouvel appel. Plusieurs fonctions CAC ont été proposées [27], [28]. Une fois que l'appel a été accepté, pour que l'usager respecte les termes du contrat passé avec le réseau, ITU a prévu une fonction de lissage (UPC, *Usage Parameter Control*) [29]. Cette fonction a pour mission de protéger le réseau. L'action à entreprendre lorsqu'une violation de contrat est constatée est d'éliminer certaines cellules appartenant à la connexion violant le contrat.

Ainsi, en mode ATM (comme sur le réseau DQDB d'ailleurs), l'occupation des créneaux successifs (figure 2.2) pourra être décrite par une variable aléatoire X binaire. Si à l'instant t ($[t-1, t)$), $X_t = 0$, le créneau sera vide. Par contre si $X_t = 1$ alors le créneau contiendra une cellule. Nous voyons que le trafic du réseau ATM au niveau des cellules se prête bien à la modélisation en temps discret. Une autre représentation du trafic peut être celle d'une variable aléatoire qui représente les temps d'interarrivées entre cellules.

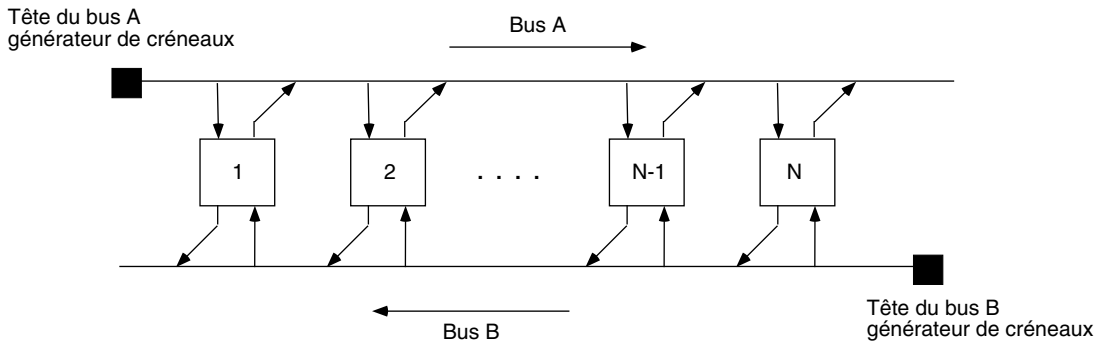

Figure 2.3: Réseau DQDB

2.3 DQDB

Le principe de base de l'accès DQDB est montré à la figure 2.3. Nous allons ici résumer les caractéristiques des opérations qui sont faites pour accéder au réseau DQDB. Pour plus de détails, il est possible de consulter [30, 31, 32]. Le réseau DQDB est constitué de 2 bus unidirectionnels. Les informations qui transitent sur ces bus le font dans deux directions opposées. Cette paire de bus comprenant un bus A (sur lequel le trafic se déplace dans la direction des noeuds croissants) et un bus B (sur lequel le trafic se déplace dans la direction opposée) opère au niveau *Medium Access Control* (MAC). Pour chaque bus, on a une tête de bus. Les deux bus convoient des créneaux de longueur fixe, 53 octets, pour le trafic synchrone et asynchrone. Les créneaux pour le trafic synchrone sont réservés dès le départ par la tête du bus A. Chaque station i (i= $1, \cdots$,N), connectée aux deux bus, peut être passive (en observant les données qui transitent sur ces derniers) ou active (en se soumettant à une discipline de file d'attente pour accéder au réseau). Pour le trafic asynchrone, il faut savoir que l'accès au réseau n'est pas instantané dû au fait que les autres stations ont également des messages à envoyer. L'accès au réseau se fait en deux temps. Premièrement, la station doit faire une demande au réseau via le bus B. Deuxièmement, lorsque la station a pu faire sa demande, il pourra déposer ses données sur le bus A. Le mécanisme d'accès est identique pour les deux bus, c'est pourquoi, pour la modélisation, nous nous concentrerons sur une direction uniquement. Pour pouvoir accéder suffisamment rapidement au réseau, les messages de chaque station doivent être fragmentés en cellules de petite dimension (53 octets, format compatible avec

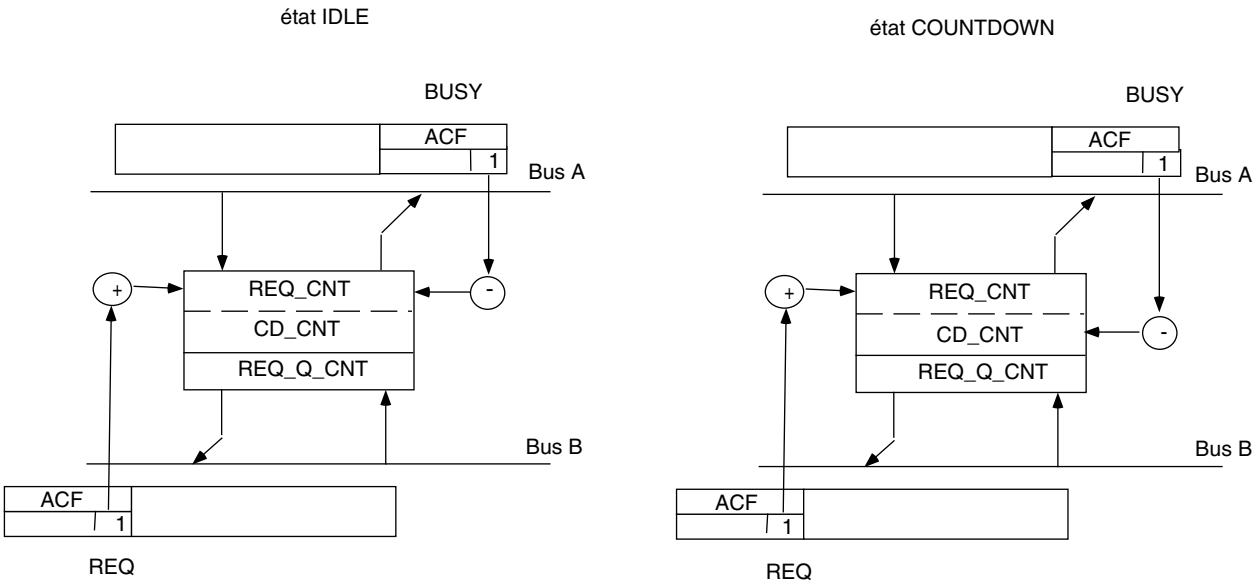

Figure 2.4: Etats logiques d'une station DQDB

le format de la cellule ATM). Jusqu'à présent, nous n'avons parlé que du principe d'accès pour un trafic de type asynchrone. Pour les services de type synchrone, l'accès se fait différemment, le générateur de créneaux (tête du bus A) réserve d'office des créneaux pour les noeuds ayant du trafic synchrone à envoyer (typiquement des services tels que voix ou vidéo) en les marquant par BUSY=1 (1 bit dans le champ *Access Control Field* (ACF)) comme montré à la figure 2.4. Ainsi, ces services n'auront pas à se soumettre à une discipline de file d'attente comme c'est le cas pour le transfert de données. Puisque le sujet de notre étude est relatif au transfert de données, nous allons nous concentrer un peu plus sur son mode d'accès.

Le noeud i (i= $1, \cdots$,N) qui désire envoyer des données sur le bus A doit premièrement en faire la demande au bus B en tenant compte que les noeuds en aval (i+1, i+2, .., N) auront également pu faire des demandes qui se traduiront par des créneaux occupés. Il faudra donc qu'il attende un créneau vide pour pouvoir émettre sa requête. Considérons un transfert de données sur le bus A avec un niveau de priorité. Une station peut se trouver dans deux états, IDLE et COUNTDOWN. Dans l'état IDLE, le noeud n'a rien à émettre alors que dans l'état COUNTDOWN, le noeud a des données à transmettre [33]. Dans l'état IDLE, le compteur de requêtes (REQ_CNT, *Request counter*) est incrémenté

de 1 lorsqu'une requête est émise depuis un noeud en aval (i+1, i+2, .., N) et décrémenté de 1 lorsque la requête est satisfaite pour une des stations en aval. Le noeud i sait qu'une requête est satisfaite pour un noeud en aval en observant le premier bit du champ ACF (BUSY) de chaque créneau passant sur le bus A. Si ce dernier est égal à 0, ça signifie qu'il sera utilisé pour un noeud en aval. Notons qu'une cellule réservée l'est jusqu'au dernier noeud, même si le trafic est interne au réseau DQDB. Dans l'état COUNTDOWN, le compte à rebours (CD_CNT, *Countdown*) est décrémenté de 1 chaque fois qu'une requête est satisfaite pour un noeud en aval. Le compteur de requêtes continue à incrémenter. Pour passer de l'état IDLE à l'état COUNTDOWN, il faut suivre la procédure suivante: la station envoie une requête à sa file d'attente locale et REQ_Q_CNT est incrémenté de 1 (il sera décrémenté de 1 lorsque la requête aura passé sur le bus B). Quand la requête est satisfaite et peut passer sur le bus B, le contenu du compteur de requêtes (REQ_CNT) est transféré au compte à rebours et le compteur de requêtes est mis à zéro. Prenons un exemple, admettons que nous voulions transmettre des données alors que

REQ_CNT=3

CD_CNT=0

REQ_Q_CNT=0

Notre requête va être mise dans la file d'attente locale

REQ_CNT=3

CD_CNT=0

REQ_Q_CNT=1

Admettons qu'une requête d'un noeud inférieur soit émise avant la notre

REQ_CNT=4

CD_CNT=0

REQ_Q_CNT=1

Ensuite, notre requête passe sur le bus B

REQ_CNT=0

CD_CNT=4

REQ_Q_CNT=0

2 requêtes viennent des noeuds inférieurs, en même temps que les 4 cellules vides passent

sur le bus A

REQ_CNT=1

CD_CNT=3

REQ_Q_CNT=0

REQ_CNT=2

CD_CNT=2

REQ_Q_CNT=0

REQ_CNT=2

CD_CNT=1

REQ_Q_CNT=0

REQ_CNT=2

CD_CNT=0

REQ_Q_CNT=0

Le prochain créneau vide nous est réservé, donc les données pourront être transférées sur le bus, mais pendant ce temps, 2 requêtes des noeuds inférieurs sont arrivées. Donc si nous voulons à nouveau transférer des données, il faudra attendre que ces dernières soient satisfaites. Le débit du réseau DQDB est de 34 Mb/s à 155 Mb/s.

Chapitre 3

Modèle markovien

Dans ce chapitre, nous allons décrire le modèle markovien que nous avons utilisé. Dans nos précédentes publications [34, 35, 36, 37, 38, 39], ce modèle porte le nom de *Special Semi-Markov Process* (SSMP). Dans un autre domaine, la reconnaissance de la parole, un modèle équivalent a été introduit au début des années 1970, le *Hidden Markov Model* (HMM) [40, 41]. Ce modèle peut aussi être vu comme un modèle *Discrete Batch Markovian Arrival Process* (D-BMAP)[42].

3.1 Introduction

Dans cette section, nous allons brièvement rappeler ce qu'est une variable aléatoire et un processus stochastique pour la clarté du texte qui suit. Soit Ω l'espace des épreuves et *prob* une mesure de probabilité sur Ω.

Définition 3.1 *Une variable aléatoire X prenant des valeurs dans un ensemble E est une fonction qui assigne une valeur $X(\xi)$ dans E à chaque réalisation ξ (ξ réalise l'évènement $A_i \in \mathcal{A}$ si $\xi = A_i$) dans Ω, l'espace des épreuves.*

Dans notre cas, E est l'ensemble des entiers non négatifs $\mathbb{N} = 0, 1, 2, \cdots$ et est infini mais dénombrable dans le cas général. Le modèle mathématique de toute expérience aléatoire est donc un triplet $(\Omega, \mathcal{A}, prob)$ appelé **espace probabilisé**. $\mathcal{A}$ est le catalogue des évènements qui sont des sous-ensembles de Ω. $\mathcal{A}$ est une $\sigma-$algèbre booléenne aussi appelée **tribu**.

Définition 3.2 *Un processus stochastique dans E est une collection $\{X_t,\ t \in T\}$ de variables aléatoires X_t définies sur le même Ω et prenant des valeurs dans E. Si l'ensemble T est rationel, (ce qui est le cas avec notre processus stochastique) alors le processus est dit discret sinon il est appelé continu.*

Une classe importante des processus stochastiques est celle des processus markoviens. Son importance est certainement dûe aux nombreux domaines auxquels ces processus s'appliquent.

Définition 3.3 *Le processus stochastique $\{Y = Y_t,\ t \in T = I\!N\}$ est appelé chaîne de Markov si $prob(Y_{t+1} = i|Y_0, Y_1, \cdots, Y_t) = prob(Y_{t+1} = i|Y_t)$ pour tout $i \in E = \{1, \cdots, n\}$.*

Une chaîne de Markov est donc une séquence de variables aléatoires tel que pour chaque t, Y_{t+1} est conditionnellement indépendant de $Y_0, Y_1, \cdots, Y_{t-1}$ étant donné Y_t. Ainsi, le prochain état Y_{t+1} du processus ne dépend des états passés $Y_0, Y_1, \cdots, Y_{t-1}$ qu'à travers l'état présent Y_t. On appelle $prob(Y_{t+1} = j|Y_t = i) = a_{ij}(t+1, t)$ avec $i, j \in E$ les probabilités de transition de la chaîne de Markov. Si les probabilités de transition sont indépendantes du temps alors la chaîne de Markov est **homogène**. Il est de coutume de former une matrice carrée avec les a_{ij} et de la nommer **matrice de transition** de la chaîne de Markov. Soit $\vec{\pi_t}$ la distribution de probabilité sur E et supposons que $\vec{\pi_t} = (\pi_{1_t}$ $\pi_{2_t} \cdots \pi_{n_t}) = (prob(Y_t = 1)\ prob(Y_t = 2) \cdots prob(Y_t = n))$. Soit une chaîne de Markov avec un espace d'états E et une matrice de transition $\mathbf{A}$.

Définition 3.4 *Un état j est dit absorbant si $a_{jj} = 1$.*

Définition 3.5 *Une chaîne de Markov est dite irréductible ssi tous ses états peuvent être atteints à partir de chaque autre état.*

Définition 3.6 *Une chaîne de Markov modulée, de type SSMP ou HMM est un processus X_t tel que la loi de X_t est une fonction de Y_t. X_t est conditionnellement indépendant de $(Y_{t\pm1}, Y_{t\pm2}, Y_{t\pm2}, \cdots)$ étant donné Y_t.*

Définition 3.7 *Une chaîne de Markov modulée est un processus dit de type Markov Modulated Poisson Process (MMPP) si la chaîne de Markov est de type SSMP et que les lois associées sont poissonniennes.*

Soit X_t le nombre de cellules générées dans un intervalle $[t-1, t)$. Soit $\mathbf{A}$ la matrice de transition du modulateur et $\vec{\pi_t}$ le vecteur des probabilités d'état du modulateur. La probabilité que la chaîne de Markov modulée génère un *batch* de k cellules alors que le modulateur se trouve dans l'état i est ϕ_{ik}. Ce type de chaîne de Markov est dit caché car on ne peut pas directement savoir dans quel état se trouve la chaîne de Markov à un instant t, même si on connait X_t. Notons que la grandeur du *batch* généré ne dépend que de l'état du modulateur et de la loi de probabilité associée à cet état et pas des états précédents.

3.2 Moments

Soit Ω l'espace des épreuves, *prob* une mesure de probabilité et X_t un processus stochastique discret défini sur Ω. X_t prend des valeurs $x_1, x_2, x_3, \cdots \in \mathbb{N}$.

Définition 3.8 *Le k^e moment du processus stochastique discret X_t prenant des valeurs dans l'ensemble $\mathbb{N}$ est*

$$E[X_t^k] = \sum_{x_l \geq 0} x_l^k prob(X_t = x_l) \tag{3.1}$$

si la série $\sum_{x_l \geq 0} x_l^k prob(X_t = x_l)$ est absolument convergente.

Dans le cas d'une chaîne de Markov modulée, de type SSMP,

$$prob(X_t = x_l) = \sum_{i=1}^{n} prob(X_t = x_l | Y_t = i) prob(Y_t = i) \tag{3.2}$$

par le théorème des probabilités totales. Ainsi le k^e moment s'écrit

$$E[X_t^k] = \sum_{x_l \geq 0} \sum_{i=1}^{n} x_l^k prob(X_t = x_l | Y_t = i) prob(Y_t = i) \tag{3.3}$$

$$= \sum_{i=1}^{n} E[X_t^k | Y_t = i] prob(Y_t = i) \tag{3.4}$$

Cette expression se met aisément sous forme matricielle

$$E[X_t^k] = \vec{\pi_t} \mathbf{\Lambda}_t^{(k)} \vec{e} \tag{3.5}$$

avec

$$\vec{e}^{\,t} = (\ 1 \quad 1 \quad \cdots \quad 1\ ) \tag{3.6}$$

où $\vec{e}^{\,t}$ est le vecteur transposé de $\vec{e}$

$$\mathbf{\Lambda_t}^{(k)} = diag(\ E(X_t^k|Y_t=1) \quad \cdots \quad E(X_t^k|Y_t=n)\ ) \tag{3.7}$$

$$\vec{\pi_t} = (\ prob(Y_t=1) \quad \cdots \quad prob(Y_t=n)\ ) \tag{3.8}$$

Soit T_{visite} le temps qu'il faut pour visiter la première fois l'état j, alors

Définition 3.9 *Récurrence:*

- *Un état j est dit récurrent si $prob(Y_{T_{visite}<\infty}=j)=1$. Si $prob(Y_{T_{visite}=\infty}=j)>0$, alors j est dit transitoire.*
- *Un état récurrent j est dit récurrent nul si $E[Y_{T_{visite}}=j]=\infty$, sinon il est récurrent positif.*
- *Un état récurrent j est dit périodique avec une période δ si $\delta \geq 2$ est le plus grand diviseur commun de $\{k : prob(Y_k=j)>0\}$, sinon il est dit apériodique.*

3.3 Autocovariance

Soit Ω l'espace des épreuves, *prob* une mesure de probabilité et X_t un processus stochastique discret défini sur Ω. X_t prend des valeurs $x_1, x_2, x_3, \cdots \in I\!N = 0, 1, 2, \cdots$.

Définition 3.10 *L'autocovariance du processus stochastique discret, réel, X_t prenant des valeurs dans l'ensemble $I\!N$ est donné par*

$$C(X_{t+\tau}^s, X_t^k) = E[X_{t+\tau}^s X_t^k] - E[X_{t+\tau}^s]E[X_t^k] \tag{3.9}$$

Pour commencer, calculons $E[X_{t+\tau}^s X_t^k]$ pour une chaîne de Markov modulée

$$E[X_{t+\tau}^s X_t^k] = \sum_{x_1,x_2\geq 0} x_1^s x_2^k prob(X_{t+\tau}=x_2 \cap X_t = x_1) \tag{3.10}$$

en appliquant de nouveau le théorème des probabilités totales, $prob(X_{t+\tau}=x_2 \cap X_t=x_1)$ peut s'écrire

$$\begin{aligned} prob(X_{t+\tau}=x_2 \cap X_t=x_1) = \ & \textstyle\sum_{i=1}^n \sum_{j=1}^n prob(X_{t+\tau}=x_2 \cap \\ & X_t=x_1|Y_{t+\tau}=j \cap Y_t=i) \\ & prob(Y_{t+\tau}=j \cap Y_t=i) \end{aligned} \tag{3.11}$$

mais comme l'opération $\cap$ est distributive, on peut écrire cette expression sous la forme suivante

$$\begin{aligned} prob(X_{t+\tau} = x_2 \cap X_t = x_1) = \quad & \sum_{i=1}^{n} \sum_{j=1}^{n} prob((X_{t+\tau} = x_2 \cap X_t = x_1 | Y_{t+\tau} = j) \cap \\ & (X_{t+\tau} = x_2 \cap X_t = x_1 | Y_t = i)) prob(Y_{t+\tau} = j \cap Y_t = i) \end{aligned} \tag{3.12}$$

$$= \sum_{i=1}^{n} \sum_{j=1}^{n} prob((X_{t+\tau} = x_2 | Y_{t+\tau} = j) \cap (X_t = x_1 | Y_t = i)) prob(Y_{t+\tau} = j \cap Y_t = i) \tag{3.13}$$

$$= \sum_{i=1}^{n} \sum_{j=1}^{n} prob(X_{t+\tau} = x_2 | Y_{t+\tau} = j) prob(X_t = x_1 | Y_t = i) prob(Y_{t+\tau} = j \cap Y_t = i) \tag{3.14}$$

Maintenant si on reprend l'équation (3.10), on peut écrire que

$$\begin{aligned} E[X_{t+\tau}^s X_t^k] = \quad & \sum_{x_1, x_2 \geq 0} \sum_{i=1}^{n} \sum_{j=1}^{n} x_1^s x_2^k prob(X_{t+\tau} = x_2 | Y_{t+\tau} = j) \\ & prob(X_t = x_1 | Y_t = i) prob(Y_{t+\tau} = j \cap Y_t = i) \end{aligned} \tag{3.15}$$

$$= \sum_{i=1}^{n} \sum_{j=1}^{n} E[X_t^k | Y_t = i] E[X_{t+\tau}^s | Y_{t+\tau} = j] prob(Y_{t+\tau} = j \cap Y_t = i) \tag{3.16}$$

$$= \sum_{i=1}^{n} \sum_{j=1}^{n} E[X_t^k | Y_t = i] prob(Y_t = i) E[X_{t+\tau}^s | Y_{t+\tau} = j] prob(Y_{t+\tau} = j | Y_t = i) \tag{3.17}$$

En reprenant les notations matricielles, on voit que

$$E[X_{t+\tau}^s X_t^k] - E[X_{t+\tau}^s] E[X_t^k] = \vec{\pi_t} \mathbf{\Lambda}_t^{(k)} (\mathbf{A}(t+\tau, t) - \vec{e} \vec{\pi}_{t+\tau}) \mathbf{\Lambda}_{t+\tau}^{(s)} \vec{e} \tag{3.18}$$

avec

$$\vec{e}^{\,t} = \begin{pmatrix} 1 & 1 & \cdots & 1 \end{pmatrix}$$

$$\mathbf{\Lambda_t}^{(k)} = diag\begin{pmatrix} E(X_t^k | Y_t = 1) & \cdots & E(X_t^k | Y_t = n) \end{pmatrix}$$

$$\vec{\pi_t} = \begin{pmatrix} prob(Y_t = 1) & \cdots & prob(Y_t = n) \end{pmatrix}$$

Si le processus stochastique discret est stationnaire au sens large, alors on peut écrire

$$C(X_{t+\tau}^s, X_t^k) = C(X_\tau^s, X_0^k) \tag{3.19}$$

$$\mathbf{A}(t+1, t) = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix} \tag{3.20}$$

de plus, si $k = s = 1$, on a

$$C(X_\tau^s, X_0^k) = C(X_\tau, X_0) = C(\tau) = \vec{\pi}\mathbf{\Lambda}(\mathbf{A}^\tau - \vec{e}\vec{\pi})\mathbf{\Lambda}\vec{e}$$

Par la suite, on considérera des processus stochastiques discrets stationnaires au sens large avec $k = s = 1$.

3.4 Décomposition spectrale

Dans cette section, nous allons voir comment décomposer $\mathbf{A}^\tau$.

Théorème 3.1 *Supposons que* $\mathbf{A}$ *soit une matrice (opérateur linéaire) diagonalisable. Soient* $\lambda_1, \lambda_2, \cdots, \lambda_n$ *les valeurs propres distinctes de* $\mathbf{A}$*. Alors il existe des projections orthogonales* $\mathbf{P}_1, \mathbf{P}_2, \cdots, \mathbf{P}_n$ *avec* $\mathbf{P}_i^2 = \mathbf{P}_i$ *sur les sous-espaces propres* $S(\lambda_1), S(\lambda_2), \cdots, S(\lambda_n)$ *tels que*

- $\mathbf{G}^{-1}\mathbf{A}\mathbf{G} = \lambda_1\mathbf{P}_1 + \lambda_2\mathbf{P}_2 + \cdots + \lambda_n\mathbf{P}_n$
- $\mathbf{P}_1 + \mathbf{P}_2 + \cdots + \mathbf{P}_n = \mathbf{I}$
- $\mathbf{P}_i\mathbf{P}_j = \mathbf{0}$ *pour tout* $i \neq j$

Avec $\mathbf{P}_i^2 = \mathbf{P}_i$ (définition du projecteur), $\mathbf{G}$ étant la matrice de passage et les valeurs propres λ_i, $i = 1, 2, \cdots, n$ étant distinctes, on vérifie facilement que $(\mathbf{P}_i)_{km} = 1$ ($(\mathbf{P}_i)_{km}$ étant l'élément km de la matrice $\mathbf{P}_i$) seulement si $k = m = i$ sinon $(\mathbf{P}_i)_{km} = 0$. Ainsi, $C(\tau)$, pour un processus stationnaire au sens large, peut s'écrire sous la forme

$$C(\tau) = \vec{\pi}\mathbf{\Lambda}\mathbf{G}^{-1}(\lambda_1^\tau\mathbf{P}_1 + \lambda_2^\tau\mathbf{P}_2 + \cdots + \lambda_n^\tau\mathbf{P}_n - \vec{e}\vec{\pi})\mathbf{G}\mathbf{\Lambda}\vec{e} \tag{3.21}$$

mais $\lambda_1 = 1$ (théorème 3.3), donc on peut écrire que

$$C(\tau) = \sum_{j=2}^{n} \lambda_j^\tau \vec{\pi}\mathbf{\Lambda}\mathbf{G}^{-1}\mathbf{P}_j\mathbf{G}\mathbf{\Lambda}\vec{e} \tag{3.22}$$

L'autocovariance $C(\tau)$ d'une chaîne de Markov modulée est ainsi constituée d'une somme d'exponentielles complexes car les termes $\vec{\pi}\mathbf{\Lambda}\mathbf{G}^{-1}\mathbf{P}_j\mathbf{G}\mathbf{\Lambda}\vec{e}$, $j = 2, \cdots, n$, sont constants.

3.5 Transformée de Fourier

Grâce au théorème de Wiener-Kintchine [43], on peut calculer la densité spectrale de la puissance d'un processus stochastique à l'aide de sa fonction d'autocovariance. Le processus stochastique est supposé stationnaire au sens large. La transformée de la fonction d'autocovariance s'écrit

$$\mathcal{F}(C(\tau)) = \Phi(\omega) = \sum_{\tau=-\infty}^{\infty} C(\tau)e^{-i\omega\tau} \tag{3.23}$$

On peut écrire cette expression sous la forme suivante

$$\Phi(\omega) = C(0) + \sum_{\tau=-\infty}^{-1} C(\tau)e^{-i\omega\tau} + \sum_{\tau=1}^{\infty} C(\tau)e^{-i\omega\tau} \tag{3.24}$$

en sachant que $C(\tau) = C(-\tau)$ par l'équation (3.19)

$$\Phi(\omega) = C(0) + \sum_{\tau=1}^{\infty} C(\tau)(e^{i\omega\tau} + e^{-i\omega\tau}) \tag{3.25}$$

mais on a vu que $C(\tau)$ pouvait s'écrire sous la forme donnée par l'équation (3.22), donc

$$\Phi(\omega) = C(0) + \sum_{\tau=1}^{\infty}\sum_{j=2}^{n} \lambda^{\tau}\vec{\pi}\mathbf{\Lambda}\mathbf{G}^{-1}\mathbf{P}_j\mathbf{G}\mathbf{\Lambda}\vec{e}(e^{i\omega\tau} + e^{-i\omega\tau}) \tag{3.26}$$

remplaçons $\vec{\pi}\mathbf{\Lambda}\mathbf{G}^{-1}\mathbf{P}_j\mathbf{G}\mathbf{\Lambda}\vec{e}$ par $\beta_j e^{-\vartheta_j}$ on obtient

$$\Phi(\omega) = C(0) + \sum_{\tau=1}^{\infty}\sum_{j=2}^{n} \lambda^{\tau}\beta_j e^{(-\vartheta_j)}(e^{i\omega\tau} + e^{-i\omega\tau}) \tag{3.27}$$

$$\Phi(\omega) = C(0) + \sum_{j=2}^{n}(\frac{\beta_j}{1 - |\lambda_j|e^{i(\omega-\vartheta_j)}} + \frac{\beta_j}{1 - |\lambda_j|e^{-i(\omega+\vartheta_j)}} - 2\beta_j) \tag{3.28}$$

$$\Phi(\omega) = C(0) + \sum_{j=2}^{n} 2\beta_j(\frac{1 - |\lambda_j|e^{-i(\vartheta_j)}cos(\omega)}{1 + |\lambda_j|^2e^{-2i(\vartheta_j)} - 2|\lambda_j|e^{-i(\vartheta_j)}cos(\omega)} - 1) \tag{3.29}$$

3.6 Particularités de la matrice de transition

La matrice de transition est carrée $n \times n$ dont chacun de ses éléments est réel et compris entre 0 et 1. D'autre part la matrice est stochastique (i.e. $\sum_j a_{ij} = 1$). Nous verrons que les valeurs propres de la matrice de transition ont une grande influence sur la fonction d'autocovariance. Il est donc intéressant de savoir où ces valeurs propres sont placées dans le champ complexe étant donné les restrictions inhérentes à la matrice de transition $\mathbf{A}$. Pour commencer, énonçons un théorème.

Théorème 3.2 *Soit une matrice complexe, carrée, $n \times n$. Pour $i = 1, 2, \cdots, n$, définissons:*

$$r_i = \rho_i(\mathbf{A}) - |a_{ii}| \tag{3.30}$$

avec, pour $i = 1, 2, \cdots, n$:

$$\rho_i(\mathbf{A}) = \sum_{j=1}^{n} |a_{ij}| \tag{3.31}$$

et soit C_i un disque de centre a_{ii}, de rayon r_i. Alors chaque valeur propre de $\mathbf{A}$ est comprise dans un cercle C_i.

Corollaire 3.1 *Soit λ une valeur propre de $\mathbf{A}$, alors $|\lambda| \leq \rho(\mathbf{A})$*

avec $\rho(\mathbf{A}) = max\{\rho_i(\mathbf{A}) : 1 \leq i \leq n\}$

Corollaire 3.2 *Si λ est une valeur propre de la matrice de transition $\mathbf{A}$, alors $|\lambda| \leq 1$.*

Théorème 3.3 *Chaque matrice de transition a 1 comme valeur propre*

L'énoncé de ces deux théorèmes, ainsi que leurs preuves se trouvent dans [44]. Enonçons un autre théorème [45] qui concerne les valeurs propres de la matrice de transition $\mathbf{A}$ dans le cercle unité. On montre que plus le nombre d'états devient important dans la chaîne de Markov, plus la zone des valeurs propres possibles devient importante jusqu'à recouvrir complètement le cercle unité.

Théorème 3.4 [1] *Soit $\mathbf{A}$ une matrice stochastique. Soit Θ_n la région des valeurs propres possibles. Alors:*

La région Θ_n est symétrique par rapport à l'axe réel. Elle est contenue dans un cercle $|z| \leq 1$ et il intercepte les bords uniquement aux points $e^{2\pi i a'/b'}$ où a et b sont des entiers qui satisfont à $0 \leq a' < b' \leq n$. La frontière de Θ_n est constituée de points et d'arcs curvilignes. Pour $n \geq 3$, chacun de ces arcs est donné par les équations paramétriques suivantes:

$$\lambda^{v'}(\lambda^{w'} - t^*)^{r'} = (1 - t^*)^{r'}(\lambda^{b'} - t^*)^{d'} = (1 - t^*)^{d'}\lambda^{q'}$$

t^* est un paramètre réel compris entre 0 et 1. b', d', r', v', w' sont des nombres naturels définis ainsi:

[1] de Karpelevič

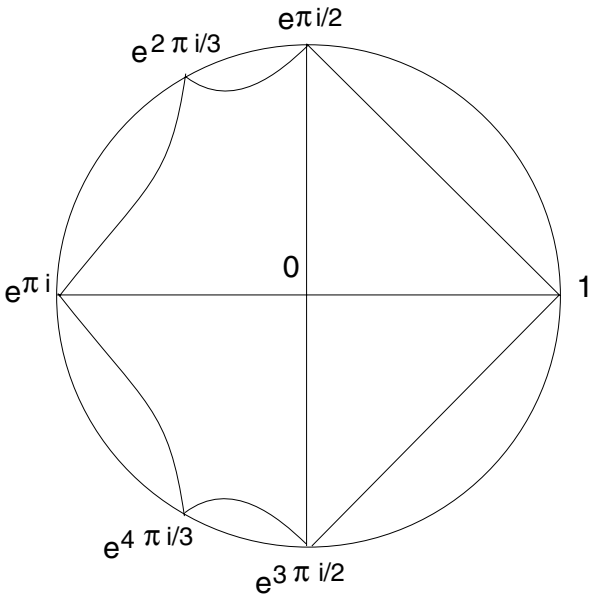

Figure 3.1: Région des valeurs propres des chaînes de Markov à 4 états

En considérant un arc connectant des points consécutifs de Θ_n sur le cercle unité. Soient ces points $e^{2\pi i a''/b''}$ et $e^{2\pi i a'''/b'''}$ dans le sens contraire des aiguilles de la montre, alors:

$$b'''.int\left(\frac{n}{b'''}\right) \geq b''.int\left(\frac{n}{b''}\right) \tag{3.32}$$

ou:

$$b'''.int\left(\frac{n}{b'''}\right) \leq b''.int\left(\frac{n}{b''}\right) \tag{3.33}$$

avec $int(x)$ =partie entière(x). Si (3.32) représente un arc, alors (3.33) représente le complexe conjugué. A cause de la symétrie, il suffit de considérer uniquement (3.32). La démonstration de ce théorème est très longue [45]. La figure 3.1 [46] montre les frontières pour Θ_4. Θ_3 est compris dans un triangle reliant 1, $e^{2\pi i/3}$, $e^{-2\pi i/3}$ alors que Θ_2 est situé sur l'axe réel. Il y a donc une restriction: une matrice de transition ne peut pas avoir n'importe quelle valeur propre dans le cercle unité. Le lieu des valeurs propres possibles est limité par le nombre d'états que comprend la chaîne de Markov. Plus le nombre d'états devient élevé, plus la zone des valeurs propres possibles devient importante, jusqu'à recouvrir totalement le cercle unité. A titre d'exemple illustratif,

considérons une chaîne de Markov à 5 états ayant la matrice de transition suivante

$$\mathbf{A} = \begin{pmatrix} 0.2 & 0.8 & 0 & 0 & 0 \\ 0 & 0.1 & 0.45 & 0 & 0.45 \\ 0 & 0.05 & 0 & 0.95 & 0 \\ 0 & 0.05 & 0.1 & 0 & 0.85 \\ 0.95 & 0.02 & 0.03 & 0 & 0 \end{pmatrix} \tag{3.34}$$

et

$$\mathbf{\Lambda} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \tag{3.35}$$

Les valeurs propres de la matrice de transition sont toutes complexes sauf la première qui est par définition égale à 1. L'ensemble des valeurs propres est le suivant {1, -0.537546+0.533363i, -0.537546-0.533363i, 0.187546+0.623019i, 0.187546-0.623019i}. La figure 3.2 montre la fonction d'autocovariance d'une telle chaîne de Markov et la figure 3.3 sa densité spectrale. Cet exemple montre bien l'influence des valeurs propres. La fonction d'autocovariance oscille avec k. Pour certaines valeurs de k, cette fonction est négative. Si nous regardons d'un peu plus près la matrice de transition $\mathbf{A}$, nous remarquons facilement que deux cycles sont privilégiés, le cycle passant par les états 1-2-5 et le cycle passant par les états 1-2-3-4-5. La figure 3.3 montre bien l'existence de ces deux cycles. Remarquons que le nombre de pics qui apparaissent est limité par l'ordre de la chaîne de Markov.

3.7 Superposition

Lorsqu'on superpose deux (ou plus) chaînes de Markov de n états du type décrit précédemment, stationnaires au sens large, on obtient une nouvelle chaîne de Markov étant caractérisée par les matrices suivantes

$$\mathbf{A} = \mathbf{A}_1 \otimes \mathbf{A}_2 \tag{3.36}$$

$$\mathbf{\Lambda} = \mathbf{\Lambda}_1 \oplus \mathbf{\Lambda}_2 = \mathbf{\Lambda}_1 \otimes \mathbf{I}_n + \mathbf{I}_n \otimes \mathbf{\Lambda}_2 \tag{3.37}$$

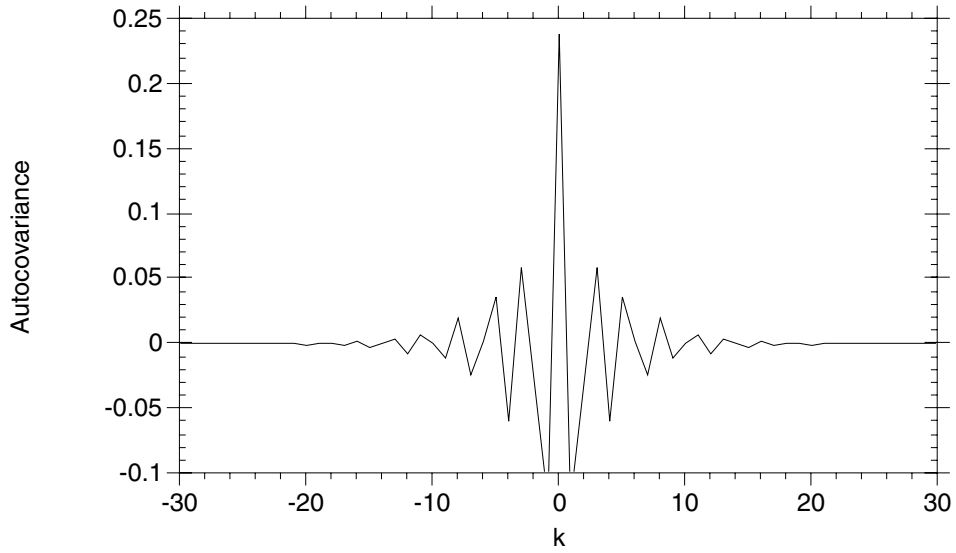

Figure 3.2: Fonction d'autocovariance d'une chaîne de Markov à 5 états dont 4 valeurs propres sont complexes

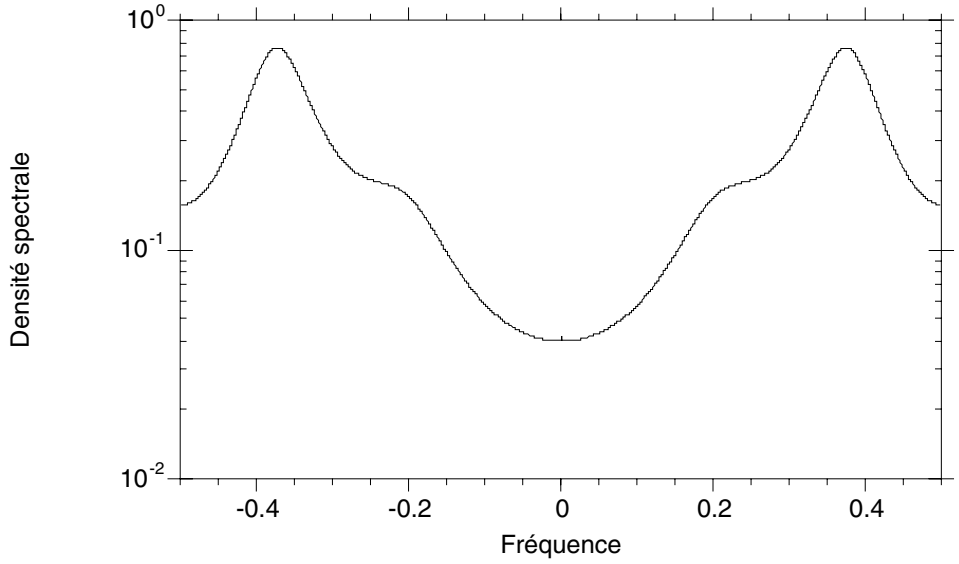

Figure 3.3: Densité spectrale d'une chaîne de Markov à 5 états dont 4 valeurs propres sont complexes

$$\vec{\pi} = \vec{\pi}_1 \otimes \vec{\pi}_2 \tag{3.38}$$

$\mathbf{A}_1$ (resp. $\mathbf{A}_2$) représente la matrice de transition du modulateur 1 (resp. 2); $\vec{\pi_1}$ (resp. $\vec{\pi_2}$) représente le vecteur des probabilités d'état du modulateur 1 (resp. 2); $\mathbf{\Lambda}_1$ (resp. $\mathbf{\Lambda}_2$) représente la matrice diagonale de la chaîne de Markov 1 (resp. 2); $\oplus$ représente la somme de Kronecker et $\otimes$ le produit de Kronecker [47]. La superposition de plusieurs chaînes de Markov est donc très facile à réaliser mais le modulateur devient très vite énorme car la croissance du nombre d'états est exponentielle. Prenons un exemple très simple où on superpose deux chaînes de Markov ayant les caractéristiques suivantes

$$\mathbf{\Lambda}_1 = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} \qquad \mathbf{\Lambda}_2 = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}$$

$$\mathbf{A}_1 = \begin{pmatrix} 1-p_1 & p_1 \\ q_1 & 1-q_1 \end{pmatrix} \qquad \mathbf{A}_2 = \begin{pmatrix} 1-p_2 & p_2 \\ q_2 & 1-q_2 \end{pmatrix}$$

$$\vec{\pi}_1 = \begin{pmatrix} q_1/(p_1+q_1) & p_1/(p_1+q_1) \end{pmatrix} \quad \vec{\pi}_2 = \begin{pmatrix} q_2/(p_2+q_2) & p_2/(p_2+q_2) \end{pmatrix}$$

alors

$$\mathbf{A} = \mathbf{A}_1 \otimes \mathbf{A}_2 = \begin{pmatrix} (\mathbf{A}_1)_{11}\mathbf{A}_2 & (\mathbf{A}_1)_{12}\mathbf{A}_2 \\ (\mathbf{A}_1)_{21}\mathbf{A}_2 & (\mathbf{A}_1)_{22}\mathbf{A}_2 \end{pmatrix}$$

$$\mathbf{A} = \begin{pmatrix} (1-p_1)(1-p_2) & (1-p_1)p_2 & p_1(1-p_2) & p_1p_2 \\ (1-p_1)q_2 & (1-p_1)(1-q_2) & p_1q_2 & p_1(1-q_2) \\ q_1(1-p_2) & q_1p_2 & (1-q_1)(1-p_2) & (1-q_1)p_2 \\ q_1q_2 & q_1(1-q_2) & (1-q_1)q_2 & (1-q_1)(1-q_2) \end{pmatrix}$$

$$\vec{\pi} = \vec{\pi}_1 \otimes \vec{\pi}_2 = \frac{1}{(p_1+q_1)(p_2+q_2)} \begin{pmatrix} q_1q_2 & q_1p_2 & p_1q_2 & p_1p_2 \end{pmatrix}$$

$$\mathbf{\Lambda} = \mathbf{\Lambda}_1 \oplus \mathbf{\Lambda}_2 = \mathbf{\Lambda}_1 \otimes \mathbf{I}_n + \mathbf{I}_n \otimes \mathbf{\Lambda}_2 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 2 \end{pmatrix}$$

3.8 Calcul de la variance du nombre de cellules dans une fenêtre de taille m

L'évaluation de la variance du nombre de cellules dans une fenêtre de temps de taille m va nous être très utile pour la suite; elle va entre autre nous permettre d'évaluer le paramètre de Hurst local. Le processus stochastique considéré ici est supposé stationnaire au sens large. Si N_m est une variable aléatoire représentant le nombre de cellules arrivant dans une fenêtre de temps de taille m, on peut écrire

$$N_m = X_1 + X_2 + \cdots + X_m \tag{3.39}$$

avec X_t, $t = 1, 2, \cdots, m$ étant le nombre de cellules arrivant dans l'intervalle $[t-1, t)$. L'espérance mathématique de N_m s'écrit alors

$$E[N_m] = E[X_1 + X_2 + \cdots + X_m] \tag{3.40}$$

et comme le processus est stationnaire au sens large et que $E[\cdots]$ est un opérateur linéaire

$$E[N_m] = mE[X] \tag{3.41}$$

il faut encore calculer le deuxième moment pour pouvoir ensuite évaluer la variance

$$E[N_m^2] = E[(X_1 + X_2 + \cdots + X_m)^2] \tag{3.42}$$

$$\begin{aligned} E[N_m^2] \quad = \quad & E[X_1^2 + X_2^2 + \cdots + X_m^2 + \\ & 2(X_1X_2 + X_2X_3 + \cdots + X_{m-1}X_m + \\ & 2(X_1X_3 + X_2X_4 + \cdots + X_{m-2}X_m + \cdots] \end{aligned}$$

comme X_t est stationnaire au sens large et que $E[\cdots]$ est un opérateur linéaire alors

$$\begin{aligned} E[N_m^2] &= mE[X^2] + \\ & 2(m-1)\vec{\pi}\mathbf{\Lambda}\mathbf{A}\mathbf{\Lambda}\vec{e} + \\ & 2(m-2)\vec{\pi}\mathbf{\Lambda}\mathbf{A}^2\mathbf{\Lambda}\vec{e} + \\ & + \cdots + \\ & 2\vec{\pi}\mathbf{\Lambda}\mathbf{A}^{m-1}\mathbf{\Lambda}\vec{e} \end{aligned}$$

car $E[X_t X_{t+\tau}] = \vec{\pi}\mathbf{\Lambda}\mathbf{A}^\tau\mathbf{\Lambda}\vec{e}$, donc nous pouvons en déduire la variance

$$Var[N_m] = E[N_m^2] - E^2[N_m] \tag{3.43}$$

$$Var[N_m] = mE[X^2] - m^2E^2[X] + 2\sum_{i=1}^{m-1}(m-i)(\vec{\pi}\mathbf{\Lambda}\mathbf{A}^i\mathbf{\Lambda}\vec{e}) \tag{3.44}$$

avec $E[X] = \vec{\pi}\mathbf{\Lambda}\vec{e}$ et $E[X^2] = \vec{\pi}\mathbf{\Lambda}^{(2)}\vec{e}$.

3.9 Indice de dispersion

L'indice de dispersion (IDC, *index of dispersion for counts*) [48] est défini comme étant le rapport de la variance du nombre d'arrivées dans un intervalle de longueur m divisé par l'espérance mathématique du nombre d'arrivées dans le même intervalle.

$$IDC(m) = \frac{Var[N_m]}{E[N_m]} \tag{3.45}$$

$Var[N_m]$ ainsi que $E[N_m] = mE[X]$ sont connus. Pour un processus de Poisson, on vérifie facilement que $IDC(m) = 1$, $\forall m = 1, 2, ...$

3.10 Coefficient de variation

Le coefficient de variation (CV, *coefficient of variation*) est défini comme étant le rapport de la racine carrée de la variance du nombre d'arrivées dans un intervalle de longueur m divisé par l'espérance mathématique du nombre d'arrivées dans le même intervalle.

$$CV(m) = \frac{\sqrt{Var[N_m]}}{E[N_m]} \tag{3.46}$$

3.11 Problème de la file d'attente SSMP/G/1/c

Avant d'attaquer le problème de la file d'attente, mentionnons que nous avons utilisé la notation de Kendall pour caractériser le système. A/B/ns/c/Z est la notation conventionnelle. A représente le processus des arrivées, B le processus des temps de service, ns le nombre de serveurs, c le nombre de places dans le système et Z représente la discipline de la file d'attente (Z=*First Come First Serve*, FCFS). En pratique, on n'indique pas les valeurs infinies. Si la discipline de la file d'attente est FCFS, on ne l'indique pas non plus.

Dans ce chapitre, nous allons décrire la manière dont on peut résoudre la file d'attente limitée de type SSMP/G/1/c. Il s'agit ici de trouver le vecteur de probabilités stationnaires de la chaîne de Markov représentant le système de la file d'attente. En calculant ce vecteur, nous obtenons immédiatement la probabilité d'occupation de la file d'attente. A partir de là, il sera assez facile d'en déduire la probabilité de perte. La méthode présentée ici a été initiée par Neuts et Hashida [49, 50], elle ne s'applique que dans des systèmes de files d'attentes limitées car le nombre d'états de la chaîne de Markov doit rester fini. Le processus d'arrivées est décrit dans la section 3.1. Il s'agit d'une chaîne de Markov en temps discret pouvant produire des arrivées en *batch*. La stratégie de service de la file d'attente est de type FCFS. La file d'attente ne comprend qu'un serveur, sans priorités et possède c places (file d'attente et serveur). L'arrivée des cellules est considérée au début d'un créneau. Chaque départ du système est considéré à la fin du créneau. N_t représente le nombre de cellules dans le système au temps t, R_t le temps résiduel de service au temps t. Si $N_t = 0$ alors R_t est défini égal à 0. Rappelons que Y_t représente l'état du modulateur. Le processus formé des trois variables $\{N_t, R_t, Y_t\}$ est une chaîne

de Markov ayant une matrice de transition $\mathbf{Q}$ formée de blocs $n \times n$, de la forme

$$\mathbf{Q} = \begin{pmatrix} \mathbf{B}_0 & \mathbf{B}_1 & \mathbf{B}_2 & \cdots & \cdots & \mathbf{B}_{c-1} & \sum_{l \geq c} \mathbf{B}_l \\ \mathbf{H}_0 & \mathbf{H}_1 & \mathbf{H}_2 & \cdots & \cdots & \mathbf{H}_{c-1} & \sum_{l \geq c} \mathbf{H}_l \\ \mathbf{0} & \mathbf{H}_0 & \mathbf{H}_1 & \cdots & \cdots & \mathbf{H}_{c-2} & \sum_{l \geq c-1} \mathbf{H}_l \\ \mathbf{0} & \mathbf{0} & \mathbf{H}_0 & \cdots & \cdots & \mathbf{H}_{c-3} & \sum_{l \geq c-2} \mathbf{H}_l \\ \vdots & & & & \cdots & & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & & \mathbf{H}_0 & \sum_{l \geq 1} \mathbf{H}_l \end{pmatrix} \tag{3.47}$$

Le vecteur des probabilités stationnaires est

$$\vec{P} = (\vec{P_0}\ \vec{P_1}\ \cdots\ \vec{P_c}) \tag{3.48}$$

avec $\vec{P_i}$, $i = 1, 2, \cdots, c$ étant un vecteur horizontal de n éléments

$$\vec{P} = (P_{01}\ P_{02}\ \cdots\ P_{0n}\ P_{11}\ P_{12}\ \cdots P_{1n}\ \cdots\ P_{c1}\ P_{c2}\ \cdots\ P_{cn}) \tag{3.49}$$

On admet ici que la chaîne de Markov est irréductible et apériodique avec un nombre fini d'états, alors $\vec{P} = \vec{P}\mathbf{Q}$ a une solution unique. La solution $\vec{P}$ est strictement positive. La matrice $\mathbf{Q}$ est du type *upper-block Hessenberg*. Cette structure va nous permettre de trouver le vecteur $\vec{P}$ efficacement à l'aide d'un algorithme développé par Le Boudec [51], l'algorithme *Matrix Block Hessenberg* (MBH). Le vecteur $\vec{P}$ est en fait un vecteur horizontal de taille $min(d, c)$ ayant pour éléments des vecteurs horizontaux de taille n. Il est possible de connaître la probabilité que j cellules soient dans le système alors que le modulateur est dans la phase i, $prob(N_t = j | Y_t = i)$. Si nous nous préoccupons uniquement du nombre de cellules dans le système sans se soucier de leur origine alors la probabilité d'avoir j cellules dans le système est donnée par la somme $\sum_{j=1}^{n} prob(N_t = j | Y_t = i)$. Rappelons brièvement la notation adoptée à la section 3.1, $a_{ij} = prob(Y_t = j | Y_t = i)$, Y_t est l'état du modulateur à l'instant $[t-1, t)$, $\phi_{ij} = prob(X_t = j | Y_t = i)$ avec X_t étant le nombre de cellules arrivant à l'instant $[t-1, t)$, $\gamma_r = prob(R_t = r)$. Alors

$$\begin{aligned} (\mathbf{B}_k)_{ij} = \phi_{ik} a_{ik} \qquad & 0 \leq k \leq c-1 \\ & 1 \leq i, j \leq n \end{aligned} \tag{3.50}$$

$$(\textstyle\sum_{l \geq c} \mathbf{B}_l)_{ij} = a_{ij} - \sum_{k=0}^{c-1} (\mathbf{B}_k)_{ij} \qquad 1 \leq i, j \leq n \tag{3.51}$$

$$(\mathbf{H}_k)_{ij} = a_{ij}(\gamma_1\phi_{ik} + (1-\gamma_1)\phi_{i,k-1}) \qquad \begin{matrix} 0 \leq k \leq c-1 \\ 1 \leq i,j \leq n \end{matrix} \tag{3.52}$$

$$(\sum_{l\geq c}\mathbf{H}_l)_{ij} = a_{ij} - \sum_{k=0}^{c-1}(\mathbf{H}_k)_{ij} \qquad 1 \leq i,j \leq n \tag{3.53}$$

La démonstration de ces formules se trouve en annexe (section 11.1). Maintenant que $\mathbf{Q}$ est complètement déterminée, il suffit de trouver le vecteur $\vec{P}$ (bien que l'algorithme MBH nous permette de nous affranchir de cette opération pour déterminer la probabilité de perte). Un avantage de l'algorithme est qu'il est uniquement basé sur des additions et des multiplications de matrices de dimension $n \times n$. Avant d'appliquer l'algorithme à notre matrice $\mathbf{Q}$, calculons la probabilité de perte dûe au débordement de la file d'attente.

3.12 Probabilité de perte

Soit L la probabilité de perte et $l(i,j)$ le nombre moyen de cellules perdues par créneau, alors que la chaîne de Markov est dans l'état i et que j cellules sont dans la file d'attente.

$$L = \sum_{j=0}^{c}\sum_{i=1}^{n}\pi_i l(i,j) \tag{3.54}$$

$$l(i,j) = \sum_{k=1}^{\infty} max(0,(k+j-1-c))\phi_{ik} \tag{3.55}$$

avec

$$\phi_{ik} = prob(X_t = k | Y_t = i)$$

mais dans l'algorithme, on va mettre cette expression sous forme matricielle

$$L = \vec{P}_c\vec{l}_c \tag{3.56}$$

avec $\vec{l}^{\,t}$=$(\vec{l}_1\ \vec{l}_2\ \cdots\ \vec{l}_c) = (l(1,1)\ l(2,1)\ \cdots\ l(n,1)\ \cdots\ l(c,1)\ l(c,2)\mathrm{l}(c,n))$, $\vec{l}_c$ se calcule de manière itérative à l'aide de la formule suivante

$$\vec{l}_j = \mathbf{R}_j\vec{l}_{j-1} + \vec{f}_j \tag{3.57}$$

avec

$$(\vec{l}_j)_i = l(i,j) \tag{3.58}$$

et

$$\vec{l}_0 = \vec{f}_0 \tag{3.59}$$

3.13 Application de l'algorithme MBH à notre problème

Dans cette section, l'algorithme MBH est donné, il permet de calculer la probabilité de perte efficacement [51]

Algorithme 3.1 Algorithme MBH

Initialisation

entrée de $\mathbf{Q}$

$\vec{S_0} = \vec{e}$

$\mathbf{R}_i = \mathbf{0}$ pour $i \leq 0$

pour i=1 to c **faire**

$\mathbf{C}_i = \mathbf{Q}_{i-1,i-1} + \mathbf{R}_{i-1}(\mathbf{Q}_{i-2,i-1} + \mathbf{R}_{i-2}(\mathbf{Q}_{i-3,i-1} + \cdots + \mathbf{R}_1(\mathbf{Q}_{0,i-1}\cdots))$

$\mathbf{R}_i = \mathbf{Q}_{i,i-1}(\mathbf{I} - \mathbf{C}_i)^{-1}$

$\vec{S_i} = \mathbf{R}_i\vec{S}_{i-1} + \vec{e}$

$\vec{l_i} = \mathbf{R}_i\vec{l}_{i-1} + \vec{f_i}$

$\mathbf{D} = \mathbf{Q}_{c,c} + \mathbf{R}_c(\mathbf{Q}_{c-1,c} + \mathbf{R}_{c-1}(\mathbf{Q}_{c-2,c} + \cdots + \mathbf{R}_1(\mathbf{Q}_{0,c}\cdots))$

résoudre $\vec{X}_c = \vec{X}_{c-1}\mathbf{D}$

$G = \vec{X}_c\vec{S}_c$

$\vec{X}_c = \vec{X}_c/G$

pour i=0 to $c-1$ **faire**

$\vec{X}_{c-i-1} = \vec{X}_{c-i}\mathbf{R}_{c-i}$

$L = \vec{X}_c\vec{L}_c$

fin

Les éléments de la matrice $\mathbf{Q}$ sont connus pour nous, $\mathbf{Q}_{i-j,i} = \mathbf{H}_{j+1}$ pour $j+1 \leq i \leq c-2$ avec $j = -1, \cdots, c-3$, $\mathbf{Q}_{0,i} = \mathbf{B}_i$ pour $0 \leq i \leq c-1$. Remarquons qu'il est conseillé d'utiliser un algorithme de "sur-relaxation" (*over-relaxation*) [52] pour résoudre l'équation $\vec{P_c} = \vec{P}_{c-1}\mathbf{D}$ qui converge plus rapidement que l'algorithme Gauss-Seidel par exemple.

Chapitre 4

Mesures

4.1 Introduction

Très tôt, dès l'avènement des réseaux informatiques, des mesures ont été effectuées. Nous ne mentionnerons que les plus connues, par exemple celles de Metcalfe et Boggs [53] ou Schoch et Hupp [13] qui ont mesuré, en 1980, les performances réelles d'un réseau Ethernet comprenant 120 stations connectées entre elles. Les mesures ont été menées sur un jour. Des programmes ont été développés pour pouvoir faire des mesures sur le réseau. Comme Ethernet est un réseau diffusé, il est relativement aisé de faire des mesures de manière passive. Une station individuelle peut observer ce qui passe sur le réseau en utilisant son interface en un mode de lecture et peut ainsi collecter les paquets passant sur le réseau sans perturber le trafic. Les mesures qu'ils ont faites sur le réseau sont les suivantes: charge du réseau, distribution de la longueur des paquets, distribution des temps d'interarrivées entre paquets, ainsi que des mesures de performances sous forte charge. Plus tard, en 1986, Jain et Routier [54] ont également publié un article célèbre dans lequel ils proposent un modèle (*Packet Trains*) basé sur leurs mesures. La principale mesure qu'ils ont effectuée sur le réseau est la distribution des interarrivées entre paquets (le temps d'interarrivée entre deux paquets est défini comme étant le temps écoulé entre le début des deux paquets en question). En 1991, Gusella [48] propose de caractériser la granularité (*burstiness*) des arrivées et des interarrivées de paquets à l'aide d'indices de dispersion. Les mesures qu'il a effectuées sur le réseau Ethernet sont

les temps d'interarrivées et d'arrivées des paquets. Les mesures effectuées par Bellcore se différencient des autres mesures effectuées jusqu'alors: ils ont construit un mesureur *hardware* qui leur permet d'obtenir les temps d'arrivées avec une grande précision (de l'ordre de $20\mu s$) et très peu de pertes. Notons tout de même que les mesures de Gusella atteignent une précision d'environ $100\mu s$, ce qui est déjà remarquable. La longueur d'un paquet Ethernet varie entre 64 et 1500 octets soit entre 51 μs et 1.2 ms sur un réseau Ethernet fonctionnant à 10 Mb/s. D'autre part la campagne de mesures menée par Bellcore s'étend sur plusieurs années alors que les études précédentes se faisaient sur un jour de mesures au maximum. Pour ce faire, l'équipe de Bellcore n'a pas mesuré tout le trafic comme on avait coutume de faire mais l'a filtré de manière à n'en garder qu'un certain type en considérant les adresses de sources et de destination, par exemple le trafic externe à Bellcore. Sinon, les fichiers de mesures deviennent vite énormes, ce qui limite l'analyse. Le trafic mesuré est un trafic aggrégé qui pourrait par exemple sortir d'un LAN et être offert à un *Wide Area Network* (WAN) fournissant un service d'interconnexions de LANs.

4.2 Mesures effectuées à l'EPFL

Dans la section précédente, il a été vu qu'il n'est pas très difficile de collecter les paquets circulant sur Ethernet à partir d'une station de travail, il suffit de faire fonctionner son interface en mode lecture, par contre les problèmes arrivent plus tard, lorsqu'il s'agit de traiter les données. Deux approches sont possibles, la première est appelée **à la volée** (*on-the-fly*) et la seconde **analyse rétrospective** (les données sont sauvées sur le disque pour l'analyse). Avant de commenter ces deux possibilités, considérons un ordinateur effectuant les mesures. Ça peut être une station de travail ou une carte *hardware* dédiée à cette tâche. Si l'ordinateur fait d'autres choses pendant les mesures, il va perdre tous les paquets qui passent à cause de son autre activité (notons que mémoriser des paquets dans l'interface d'entrée/sortie n'est pas une bonne solution parce que l'horodatage (*timestamping*) ne peut pas être effectué avec précision). C'est principalement pourquoi nous ne pouvons pas faire de mesures sans pertes. Quelle que soit la configuration utilisée, le

mesureur a des tâches a effectuer. Considérons l'approche à la volée. Le traitement des données est effectué en même temps que la mesure; ça requiert de longs calculs. Cette approche n'est envisageable que pour un *hardware* muni de plusieurs processeurs si nous voulons peu de pertes. L'analyse rétrospective est mieux adaptée pour un système ne comprenant qu'un seul processeur car aucun calcul n'est effectué pendant les mesures. Cependant, lors de longues mesures, les mémoires doivent être vidées périodiquement et sauvées sur le disque. Cette opération peut causer beaucoup de pertes, d'autant plus que les opérations sur disque sont en général assez lentes.

4.3 Minimisation des pertes

L'analyse à la volée (*on-the-fly*) ne donne de bons résultats que si on utilise un *hardware* à plusieurs microprocesseurs partageant la même mémoire. Dans ce cas, un processeur peut être dédié pour la mesure alors que les autres calculent les statistiques et mettent les données sur le disque. Les calculs doivent se faire assez rapidement pour que le tampon (*buffer*) puisse être vidé assez vite, avant de devenir complètement saturé. Cette solution est faisable et certainement assez précise mais elle est coûteuse et techniquement difficilement réalisable.

Une autre solution consiste à utiliser une station de travail connectée au réseau tout en étant pilotée de l'extérieur. Aucun *hardware* supplémentaire n'est nécessaire. L'analyse rétrospective est la seule possible étant donné le nombre de données à manipuler. Néanmoins un problème subsiste car il faut sauver les données en faisant les mesures de telle sorte qu'elles soient suffisamment précises. Une solution à ce problème est d'utiliser la technique du tampon (*buffer*) dédoublé [48]. Un grand tampon (*buffer*) met en mémoire les paquets étiquettés et quand il est plein, un autre tampon est rempli pendant qu'on vide le premier et qu'on met ses données sur le disque avec une priorité basse. Avec cette technique, les temps d'arrivée des paquets sont estimés avec la précision que le permet la station de travail. La précision de la station Sun utilisée est de 500 μs.

4.4 *Tcpdump*

Le laboratoire *Lawrence Berkeley Laboratory* (LBL) à Berkeley a développé un outil pour faire des mesures sur le réseau qui s'appelle *Tcpdump*. Il est basé sur *etherfind* qui a été développé au laboratoire LBL pour investiguer et améliorer la performance de TCP. *Tcpdump* extrait les en-têtes des paquets passant sur l'interface du réseau Ethernet. Il est ainsi très facile de filtrer un trafic spécifique (par exemple le trafic sortant d'un port spécifique ou arrivant dans une station de travail particulière). La distribution de *Tcpdump* est disponible via un transfert *ftp* à l'adresse suivante: `ftp.ee.lbl.gov` et contient des utilitaires pour le traitement des données. La première chose à faire dans ce contexte est de tester la performance de ce logiciel. Dans ce but, un petit programme client-serveur a été écrit. Le but de ce programme était de générer du trafic connu où chaque paquet était numéroté de manière à pouvoir détecter les paquets perdus soit par le réseau, soit par *Tcpdump*. *Tcpdump* s'est montré très fiable.

4.5 Configuration du réseau sur lequel les mesures ont été effectuées

La première mesure que nous avons effectuée est une mesure de 24 heures de tout le trafic externe de trois laboratoires de l'EPFL (52 stations de travail): le laboratoire de traitements de signaux (LTS) avec 24 stations de travail (16 Sun, 3 Silicon Graphics, 4 Dec pmax et 1 NeXT), le laboratoire des télécommunications (TCOM) avec 24 stations de travail (20 Hewlett Packard (HP) et 4 Sun) et la chaire de circuits et systèmes (CIRC) avec 5 stations de travail (2 HP, 2 Apollo et 1 NeXT). Ces mesures ont été effectuées du 30 août 1993 (8h15) au 31 août 1993 (8h15). D'autres mesures ont été effectuées, par exemple sur la boucle FDDI du laboratoire TCOM pour faire des mesures sur du trafic multimédia mais nous ne considérerons que les premières mesures par la suite.

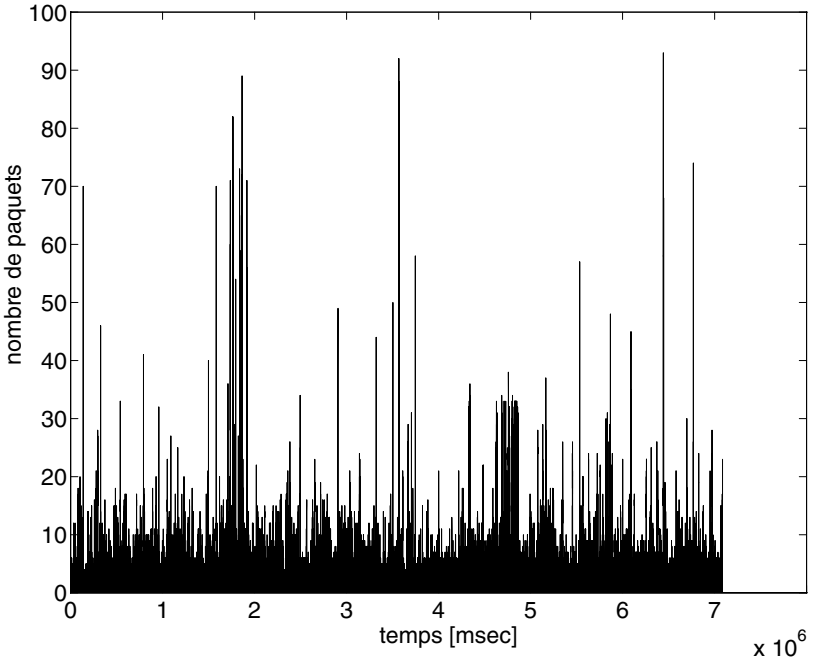

Figure 4.1: Distribution des arrivées

4.6 Analyse des données

Dans cette section, nous allons montrer les résultats obtenus pour la distribution des arrivées, des interarrivées et de la longueur des paquets.

4.6.1 Distribution des arrivées

La métrique la plus immédiate est certainement la distribution des arrivées en fonction du temps. Il est possible de calculer cette métrique sur plusieurs échelles de temps. Dans la figure 4.1, nous voyons la distribution des arrivées en fonction du temps sur une période de 2 heures environ alors que la figure 4.2 montre un jour complet où l'intervalle de temps de mesure est de 1 heure. Il apparaît très nettement un pic situé entre 18 et 19 heures. Ceci est certainement dû à des sauvegardes automatiques des fichiers lorsque les gens quittent leur bureau.

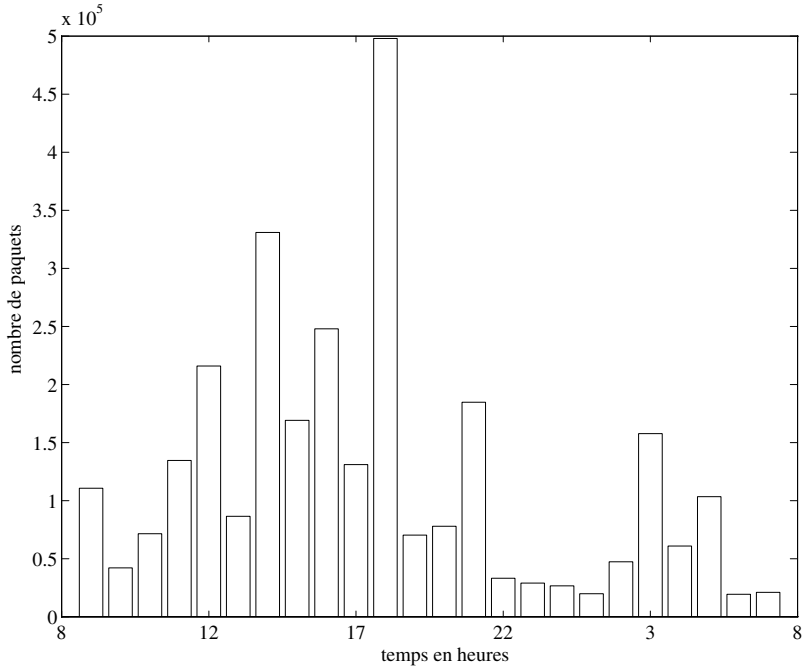

Figure 4.2: Distribution des arrivées

4.6.2 Distribution des interarrivées

Une importante métrique est également la distribution des interarrivées en fonction du temps. On peut voir assez facilement qu'une grande majorité de paquets ont de très petits temps d'interarrivées, ça confirme que le trafic sur le réseau circule en rafales (*bursty*). La figure 4.3 montre la distribution des interarrivées calculée sur une période de 10 minutes (entre 10h25 et 10h35) avec une résolution de 500 μs. Le grand pic observé entre 1 et 1.5 ms correspond aux interarrivées des grands paquets (qui sont une composante importante du trafic circulant sur le réseau). Un paquet de 1500 octets dure approximativement 1.2 ms sur un réseau à 10 Mb/s.

4.6.3 Distribution de la longueur des paquets

La statistique suivante est le calcul de la distribution de la longueur des paquets. La figure 4.4 montre la distribution de la longueur des paquets calculée sur la période des 24 heures. Nous remarquons que beaucoup de petits paquets circulent sur le réseau

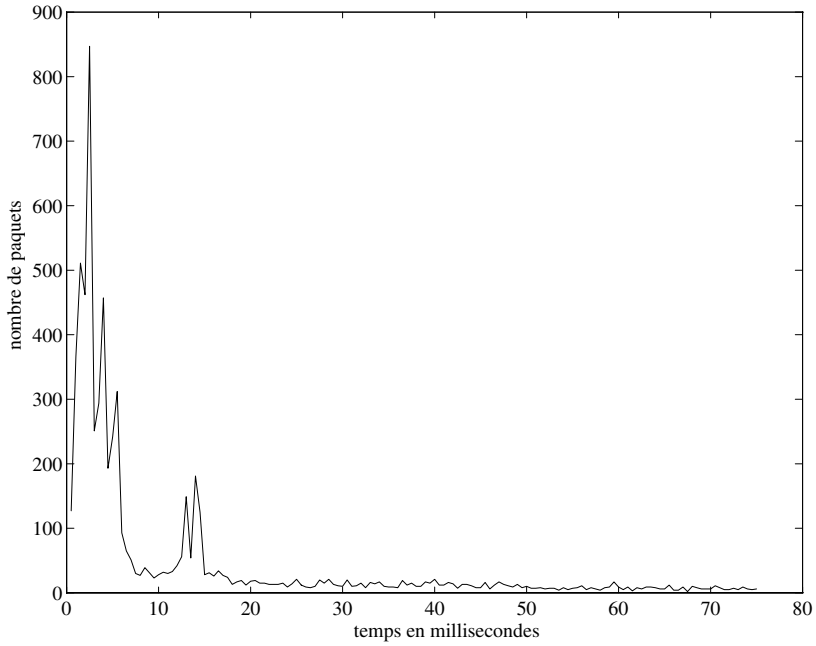

Figure 4.3: Distribution des interarrivées

(acquittements, ouvertures de sessions) mais aussi beaucoup de grands paquets qui correspondent à des transferts de fichiers.

4.7 Mesures effectuées à Bellcore

Dans l'introduction, nous avons déjà décrit brièvement les fichiers de mesures effectuées par Bellcore. Ici, ce sont les caractéristiques de ces fichiers qui sont exposées. Les fichiers disponibles sont les suivants:

- *pAug.TL*
  contient le premier million des arrivées (environ 3142.82 sec.) d'une mesure qui a duré 1 jour et qui a débuté le 29 août 1989 à 11h25.

- *pOct.TL*
  contient le premier million des arrivées (environ 1759.62 sec.) d'une mesure qui a duré 1 jour et qui a débuté le 5 octobre 1989 à 11h00.

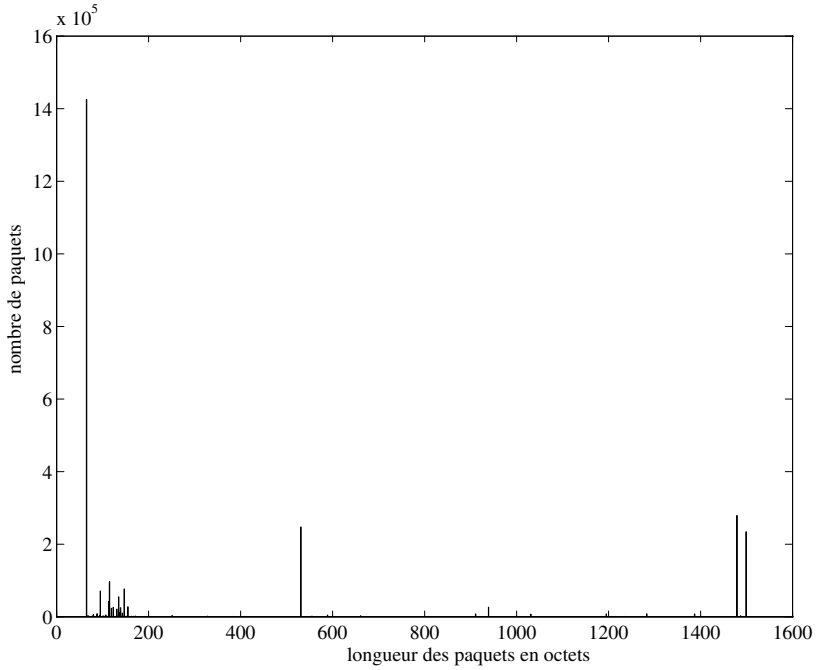

Figure 4.4: Distribution de la longueur des paquets

- *OctExt.TL*
  contient le premier million des arrivées externes (environ 122797.83 sec.) d'une mesure qui a duré 35 heures et qui a débuté le 3 octobre 1989 à 23h46.

- *OctExt4.TL*
  contient un million d'arrivées externes (environ 75943.08 sec.) d'une mesure qui a duré 307 heures et qui a débuté le 10 octobre 1989 à 14h37. Le million d'arrivées commence à partir de la 215^{e} heure.

Notons que ces mesures contiennent le temps d'arrivée de tous les paquets qui ont circulé sur le réseau mais pas les collisions ou les fragments. Les fichiers qui sont disponibles sont extraits de mesures qui ont répertorié environ 300 millions de paquets.

4.8 Introduction aux processus auto-similaires

Le terme "auto-similaire" (*self-similar*) a été introduit par Kolmogorov [55] en 1941 mais les statisticiens ne semblaient pas porter un grand intérêt à ces processus avant que

Mandelbrot et van Ness n'y travaillent [56]. Dans le cadre des processus stochastiques, l'auto-similarité se défini comme suit [57]

Définition 4.1 *Soit X_t un processus stochastique dépendant du temps. X_t est dit auto-similaire avec le paramètre d'auto-similarité H, si pour un facteur positif g, le processus calibré sur gt, $g^{-H}X_{gt}$ a la même distribution que le processus original X_t.*

Nous voyons donc, sur une série de points $t_1, t_2, t_3, \cdots, t_k$, que la distribution de $(X_{t_1}, X_{t_2}, \cdots, X_{t_k})$ est identique à la distribution de $(g^{-H}X_{gt_1}, g^{-H}X_{gt_2}, \cdots, g^{-H}X_{gt_k})$. A la différence des processus auto-similaires déterministes, ceux-ci ne sont pas la réplique exacte du modèle sur plusieurs échelles de temps. L'auto-similarité exprime en fait une invariance du processus lorsqu'on multiplie le temps par une constante. Construisons une nouvelle série de blocs conjoints de taille $m = 1, 2, \cdots$, $X_k^{(m)} = 1/m(X_{km-m+1} + \cdots + X_{km})$, $k \geq 1$. $X_k^{(m)}$ est la moyenne de X_t sur l'intervalle m. La variance de cette nouvelle série vaut $var(X_k^{(m)}) = \sigma^2 m^{-\beta}$, $H = 1 - \beta/2$ si le processus X_t est exactement auto-similaire au second ordre avec le paramètre d'auto-similarité H [14]. Il y a d'autres aspects qui se manifestent avec ce genre de processus, suivant la définition ci-dessous [58]

Définition 4.2 *Un processus à dépendance à long-terme est caractérisé par*

- $\sum_{\tau=-\infty}^{\infty} C(X_t, X_{t+\tau})$ *est divergent.*
- *le spectre du processus en* 0 *est infini,* $\Phi(0) = \infty$.
- *la variance du processus aggrégé* $var(X_k^{(m)})$ *est de la forme* $\sigma^2 m^{-\beta}$ *pour de grandes valeurs de* m.

4.9 Estimation de l'auto-similarité par des méthodes heuristiques

Dans cette section, nous allons exposer quelques méthodes souvent utilisées pour estimer le degré d'auto-similarité d'un processus.

4.9.1 Méthode de la variance

Comme nous l'avons vu dans la section 4.8, une caractéristique importante des processus auto-similaires est que la variance de la moyenne du processus X_t décroit plus lentement qu'en $1/m$. Il est donc possible de calculer la moyenne de la série $X_k^{(m)}$ par la formule classique de l'estimation de la moyenne sur un nombre suffisant d'échantillons, n_{suf}.

$$\overline{X^{(m)}} = 1/n_{suf} \sum_{k=1}^{n_{suf}} X_k^{(m)} \tag{4.1}$$

Pour chaque m, calculons la variance des moyennes $X_k^{(m)}$ $(k = 1, 2, 3, \cdots, n_{suf})$

$$\widehat{\sigma^2}(X^{(m)}) = \frac{1}{n_{suf} - 1} \sum_{k=1}^{n_{suf}} (X_k^{(m)} - \overline{X^{(m)}})^2 \tag{4.2}$$

Ensuite, il suffit de reporter $log(\widehat{\sigma^2}(X^{(m)}))$ en fonction de $log(m)$. Pour des grandes valeurs de m, les points de ce graphe vont se situer autour d'une droite de pente négative $-\beta$ avec $\beta = 2 - 2 \times H$. Dans le cas où le processus n'aurait pas de dépendances à long terme, $\beta = 1$ et $H = 0.5$. Un inconvénient de cette méthode est que l'estimation correcte de la variance se fait lentement car il faut un grand nombre d'échantillons [59, 57]. Les figures 4.5 et 4.6 montrent les graphes que nous appelerons "temps-variances" pour les mesures effectuées à l'EPFL et à Bellcore. Sur un grand nombre d'échelles de temps, l'estimation du paramètre de Hurst montre que $H \neq 0.5$. Les mesures n'ont donc pas des dépendances à court terme mais des dépendances à long terme.

4.9.2 Indice de dispersion

Une autre façon de caractériser la variabilité du trafic est de calculer la granularité (*burstiness*) de ce dernier. Pour cela, plusieurs chercheurs [48, 13, 60] proposent d'évaluer l'indice de dispersion et le coefficient de variation. Dès le début des études de performances sur les réseaux informatiques, on a remarqué que le trafic circulait en rafales (*bursty*). C'est alors qu'on a commencé à vouloir le caractériser plus formellement car on avait vu que cette caractéristique avait une influence sur le comportement des files d'attente. Pour cela, l'indice de dispersion et le coefficient de variation ont été trouvés adéquats. L'indice de dispersion est depuis longtemps utilisé par les statisticiens comme

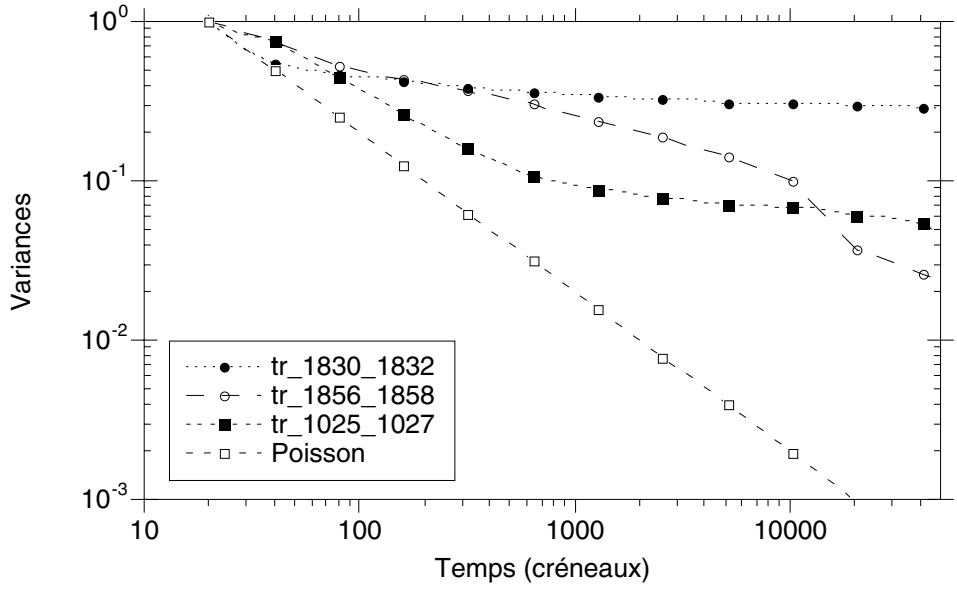

Figure 4.5: Paramètre de Hurst de plusieurs mesures faites sur le réseau de l'EPFL

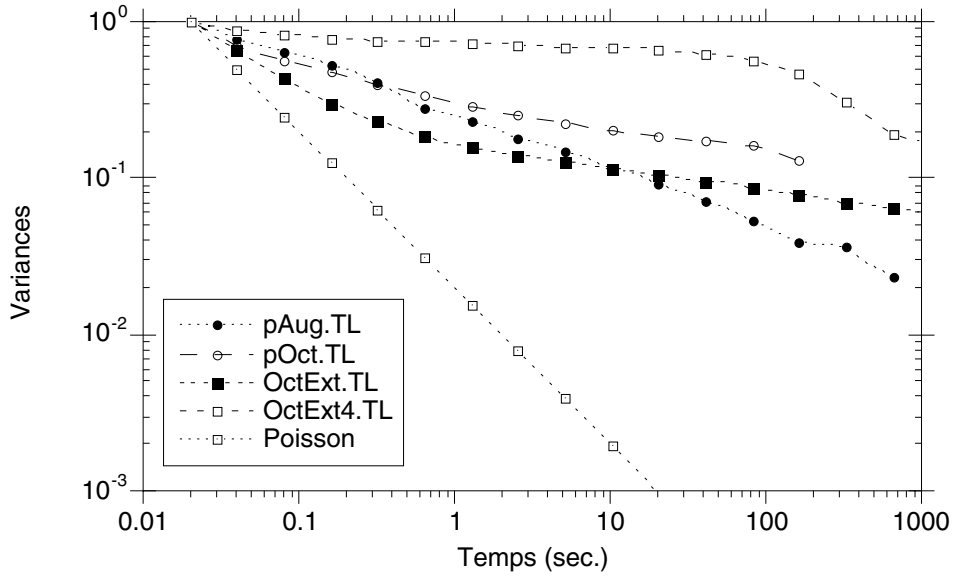

Figure 4.6: Paramètre de Hurst de plusieurs mesures faites sur le réseau de Bellcore

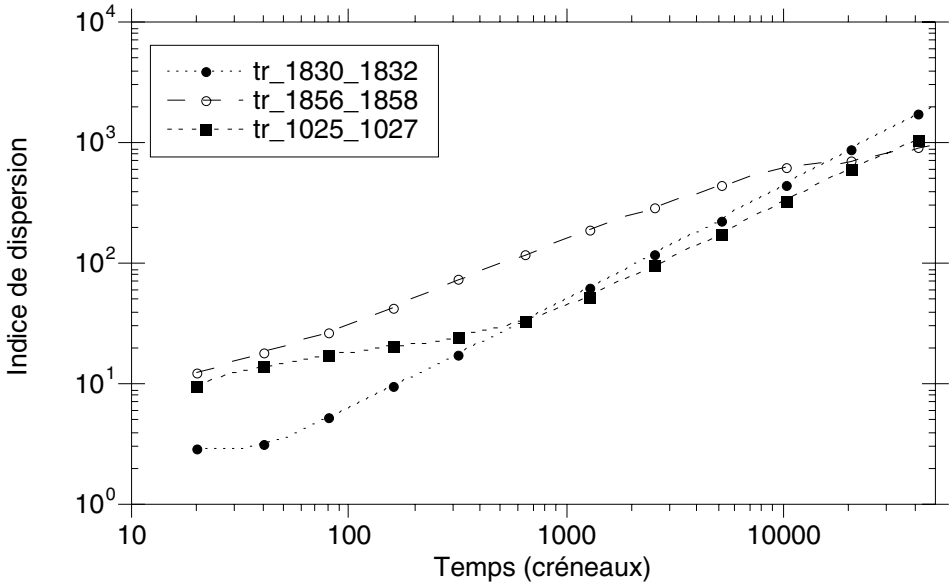

Figure 4.7: Indice de dispersion de plusieurs mesures faites sur le réseau de l'EPFL

étant un bon outil pour analyser les propriétés du second ordre de certains processus [48, 58]. Heffes et Lucantoni l'ont calculé pour les processus MMPP [60].

Comme auparavant (section 4.9.1), nous définissons une fenêtre de longueur m dans laquelle nous comptons le nombre des arrivées qui sera égal à $mX_k^{(m)}$ selon la notation que nous avons adoptée. L'indice de dispersion est donné par

$$IDC(m) = \frac{\widehat{\sigma^2}(mX^{(m)})}{\overline{(mX_k^{(m)})}} \tag{4.3}$$

qui est égal à la variance du nombre d'arrivées dans une fenêtre divisé par la moyenne du nombre d'arrivées dans cette même fenêtre. Les figure 4.7 et 4.8 nous montrent les indices de dispersion des mesures de l'EPFL et de Bellcore. Dans chaque cas, nous voyons que l'indice de dispersion augmente sur plusieurs échelles de temps. les processus ayant des dépendances à court terme n'ont pas un tel comportement; ils se stabilisent très rapidement. Ici, nous avons encore une confirmation que les mesures ont des dépendances à long terme.

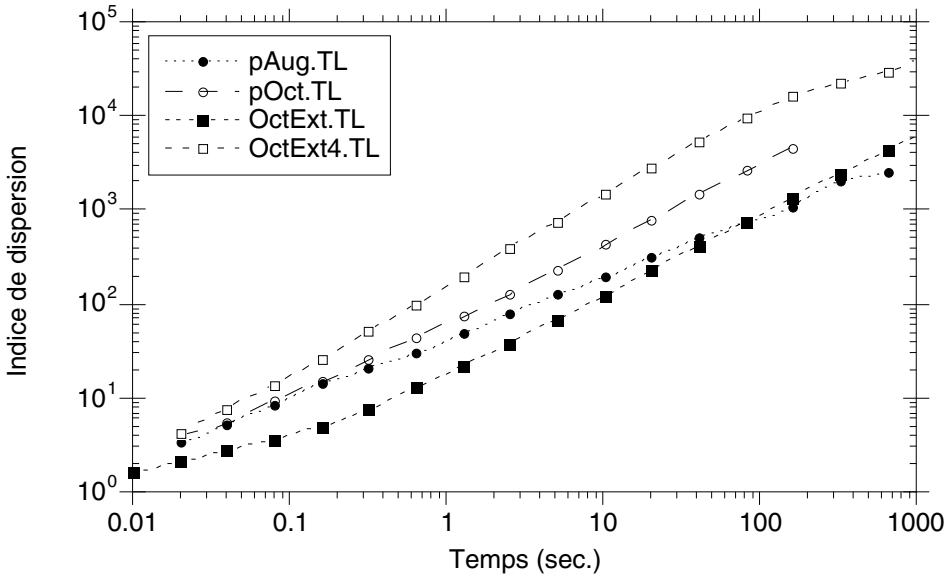

Figure 4.8: Indice de dispersion de plusieurs mesures faites sur le réseau de Bellcore

4.9.3 Coefficient de variation

Le coefficient de variation est donné par

$$CV(m) = \frac{\sqrt{\widehat{\sigma^2}(mX^{(m)})}}{\overline{(mX_k^{(m)})}} \tag{4.4}$$

qui est égal à l'écart type du nombre d'arrivées dans une fenêtre divisé par la moyenne du nombre d'arrivées dans cette même fenêtre. Les figures 4.9 et 4.10 nous montrent les coefficients de variation des mesures de l'EPFL et de Bellcore.

4.9.4 Test visuel

La définition 4.1 nous montre qu'un processus auto-similaire a la même distribution dans les petites échelles que dans les grandes si nous ajustons le facteur d'échelle par $g^{-H}X_{gt}$. Ainsi, nous relevons $mX_k^{(m)}$ (nombre de paquets arrivant dans une fenêtre de taille m) est donné en fonction du temps, (m est appelé **unité de temps**). Si nous traçons ce graphe pour plusieurs m différents, nous pouvons observer le comportement du processus à travers les échelles de temps depuis 1 ms jusqu'à 1 jour ou plus. Le facteur entre les différentes unités de temps est choisi arbitrairement égal à 10. Un exemple est montré à la figure 7.5.

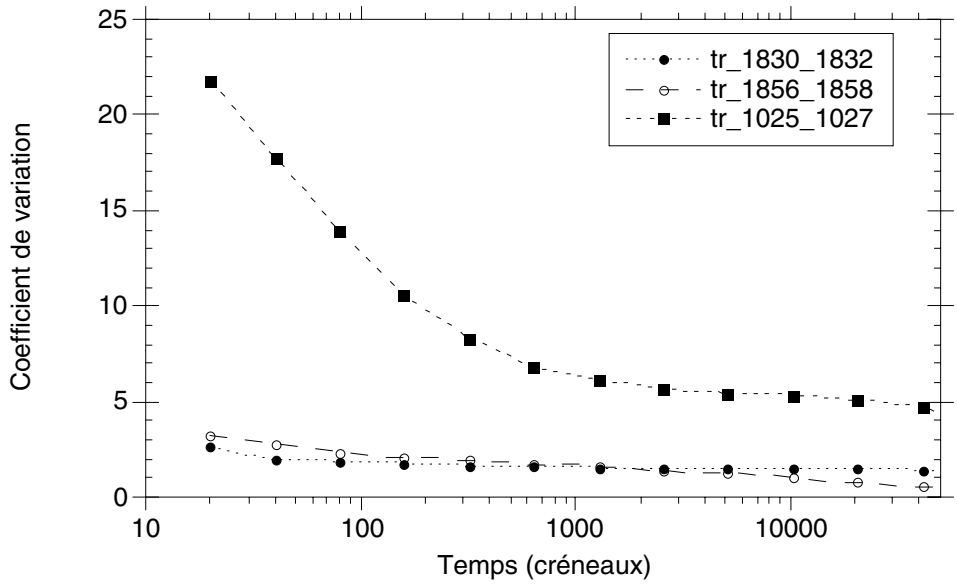

Figure 4.9: Coefficient de variation de plusieurs mesures faites sur le réseau de l'EPFL

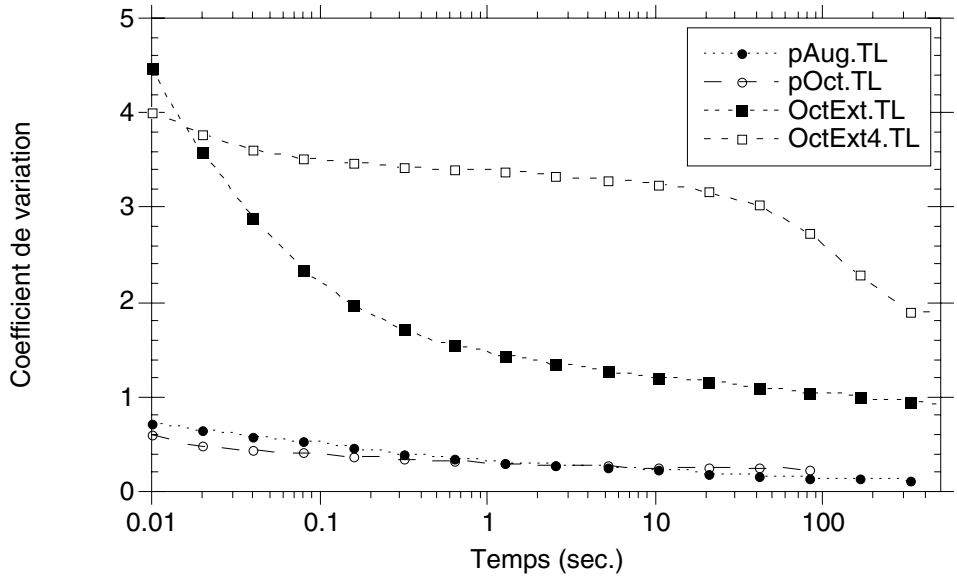

Figure 4.10: Coefficient de variation de plusieurs mesures faites sur le réseau de Bellcore

Chapitre 5

Etude d'une passerelle Ethernet/DQDB

5.1 Introduction

Les mesures sur le réseau Ethernet nous montrent que les modèles usuels ne suffisent pas pour construire une source de trafic pour un réseau LAN. Il est en fait très difficile de mettre sur pied un descripteur de trafic pour du trafic de données. C'est pourquoi, dans un premier temps, nous avons décidé de ne faire aucune hypothèse quant à la forme du trafic. Nous avons admis le pire des cas au point de vue trafic [61]. Ça signifie que nous faisons l'hypothèse que Ethernet transporte des trames de 1500 octets en permanence au débit de 10 Mb/s. Les trames sont séparées entre elles par des intervalles de 9,6 μs. Pour le trafic DQDB, nous ne faisons aucune hypothèse quant au processus d'arrivée des autres noeuds ni quant à la longueur maximale d'un paquet DQDB. Ainsi, les temps d'interarrivées entre les créneaux vides (c'est ce que voient les trames Ethernet depuis le noeud DQDB) sont identiquement et indépendamment distribués. Les paramètres de distribution des temps d'interarrivées sont l'espérance mathématique et la variance. DQDB est basé sur un bus symétrique ayant des flux de trafic qui peuvent se décomposer [33]. Nous suggérons par conséquent de considérer le trafic dans une seule direction car l'analyse dans l'autre direction est identique. Un seul niveau de priorité est envisagé pour la suite.

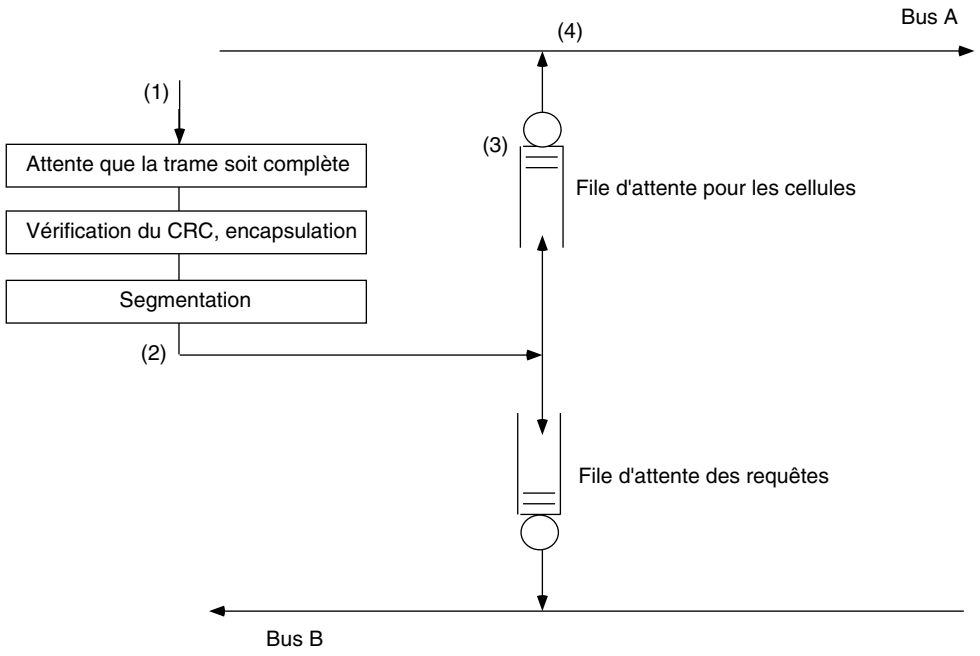

Figure 5.1: Modèle de DQDB

5.2 Interconnexion

L'interconnexion entre Ethernet et DQDB se fait au niveau MAC. La figure 5.1 qui est basée sur la figure 3 de [33] montre comment les données du réseau Ethernet sont transmises sur le réseau DQDB. Premièrement, on attend que la trame Ethernet soit complètement arrivée pour la vérification du *Cyclic Redundancy Check* (CRC). Ensuite, la trame va être encapsulée dans une IM-PDU de DQDB (la taille maximale d'une trame Ethernet est de 1500 octets tandis que celle d'une trame IM-PDU de DQDB est de 9188 octets). La segmentation peut alors commencer. Les informations de la trame Ethernet vont être réparties en plusieurs fragments dans les cellules DQDB. La trame Ethernet est transferée sur le bus A de DQDB alors que les requêtes sont transferées sur le bus B. Comme le montre la figure 5.1, nous devons tenir compte de la chronologie des évènements étant donné le type d'interconnexion entre Ethernet et DQDB. (1)-(2) représente le temps écoulé depuis le commencement et la fin d'une trame Ethernet d'une taille maximale de 1500 octets à un débit de 10 Mb/s, le temps est donc de 1.2 ms. En (2), les cellules sont prêtes pour être transmises en position d'attente pour ensuite pouvoir être transmises sur le bus A, une requête est créée pour être envoyée sur le bus

B. De (2)-(3), un flot de 34 cellules collées les unes aux autres est envoyé sur la file d'attente. Si on considère un débit de 155 Mb/s sur DQDB, le temps de silence entre deux trames Ethernet consécutives sera de 408 créneaux (1 créneau correspondant à 53 octets, donc 2.73 μs sur le réseau considéré). En (3), les cellules sont transmises suivant une discipline de type FCFS. De (3)-(4), la cellule à la tête de la file d'attente est prête pour être transmise mais doit attendre un créneau libre sur le bus A. Le temps de service peut être interprété comme étant l'intervalle de temps entre deux créneaux libres, vu de la station émettrice. Ici, on fait l'hypothèse que les temps de service sont indépendants les uns des autres.

5.3 Calcul de l'espérance mathématique et de la variance du temps d'attente dans le système

Soit T_{k+1} le temps d'interarrivée entre la k^{e} cellule et la k+1^{e} cellule et soit S_k le temps de service de la k^{e} cellule. Le temps d'attente du k^{e} client dans la file d'attente est noté W_k. Alors on peut écrire l'équation de Lindley [62]:

$$W_{k+1} = max(0, W_k + S_k - T_{k+1}) \tag{5.1}$$

Le temps d'attente pour la k^{e} cellule dans le système U_k est égal au temps qu'elle passe dans la file d'attente W_k plus le temps de service $S_k, U_k = W_K + S_k$. A la position (2) de la figure 5.1, le trafic est constitué de rafales (*bursts*), où n_c cellules sont collées les unes aux autres, et de silences. T_b représente le temps qui sépare 2 cellules dans une rafale; normalement $T_b = 1$ créneau. T_s représente le temps (en créneaux) qui sépare la dernière cellule d'une rafale à la première cellule de la rafale suivante; T_s dépend de la différence de débit entre les 2 réseaux; si les 2 réseaux ont le même débit alors $T_s = 1$ créneau (si on néglige l'effet de l'adjonction des en-têtes au niveau du DQDB).

Maintenant, si on regarde ce qui se passe pour la première cellule d'une rafale, nous remarquons qu'elle ne doit jamais attendre sauf s'il reste encore d'autres cellules dans la file d'attente ou dans le serveur qui n'ont pas pu être transmises sur le réseau DQDB. La figure 5.1 nous montre que dès que la première cellule d'une rafale arrive, une requête

va être faite au réseau DQDB. Il y aura également une requête pour chaque cellule de la rafale. Imaginons maintenant qu'un noeud situé entre notre station et le générateur de créneaux soit en train d'occuper tous les créneaux libres qui passent sur le réseau. Alors il faudra attendre que la requête aille jusqu'à cette station pour qu'on puisse avoir un créneau libre (figure 2.3). Au pire, cette station est située près du générateur de créneaux. Ainsi S_0 représente le temps nécessaire à un créneau pour faire l'aller-retour depuis notre station jusqu'au générateur de créneaux. Le temps S_0 doit être inférieur à 1.2 ms (temps nécessaire pour transmettre une trame Ethernet de 1500 octets) pour être sûr que la transmission des cellules suivantes puisse se faire correctement. Ce temps de 1.2 ms impose que la distance entre la station et le générateur de créneaux soit suffisamment petite pour pouvoir transmettre la trame Ethernet sur le réseau DQDB. A titre d'exemple, notons qu'il faudra attendre l'équivalent de 36 créneaux (sur DQDB, débit = 155 Mb/s, 1 créneau = $53 \times 8/155 \times 10^6 = 2.73\mu$s) si on considère une distance de 20 km (temps pour parcourir 20 km à 200000 km/s = 0,1 ms). Ça signifie que la station sera située à 10 km du générateur de cellules. Les temps de service des autres cellules $(S_1, S_2, ...S_{n-1})$ devront être évidemment plus petits car les requêtes sont envoyées juste derrière la première requête. On admet que les temps de service sont identiquement et indépendamment distribués. En fait la stratégie de service dépend des noeuds en aval cette fois et des requêtes qu'ils font au réseau. Nous avons admis que le temps de service était distribué selon une loi de probabilité arbitraire ayant comme paramètre libre l'espérance mathématique $E[S]$ et la variance $Var[S]$. Les cellules dans une rafale sont numérotées de 0 à $n_c - 1$. Idéalement, la cellule $n_c - 1$ doit quitter le système avant que la cellule de la prochaine rafale n'arrive, autrement dit

$$U_{n_c-1} \leq T_s \tag{5.2}$$

Notons d'autre part que le temps de service de la k$^{\text{e}}$ cellule est plus grand ou égal au temps d'interarrivées des cellules dans une rafale, $S_k \geq T_b$ pour $0 \leq k \leq n-2$. Avec cette condition et la formule (5.1), nous déduisons facilement que

$$W_k = \sum_{j=0}^{k-1} S_j - kT_b \tag{5.3}$$

$$W_k = S_0 + \sum_{j=1}^{k-1} S_j - kT_b \tag{5.4}$$

mais on sait que

$$U_k = W_k + S_k \tag{5.5}$$

donc, on peut alors calculer l'espérance mathématique et la variance de la k^e cellule d'une trame Ethernet arbitraire

$$E[U_k] = S_0 + k(E[S] - T_b) \quad 0 \leq k \leq n_c - 1 \tag{5.6}$$

$$Var[U_k] = kT_s^2 \quad 0 \leq k \leq n_c \tag{5.7}$$

On peut encore calculer la moyenne et la variance du temps d'attente dans le système.

$$\mu_u = S_0 + \frac{(n_c - 1)}{2}(E[S] - T_b) \tag{5.8}$$

$$\sigma_u^2 = \frac{(n_c^2 - 1)}{12}(E[S] - T_b)^2 + \frac{(n_c - 1)}{2}Var^2[S] \tag{5.9}$$

La démonstration de ces expressions se trouve en annexe, à la section 11.2. On peut aussi voir dans [63].

5.4 Exemple pratique

Considérons un réseau Ethernet dont les trames ont 1500 octets et fonctionnant à 10 Mb/s. Le réseau DQDB fonctionne à 155 Mb/s. La trame Ethernet sera donc divisée en 34 cellules. Le temps de silence est donné par l'équation (5.10).

$$T_s = n_c(\frac{8 \times 44}{\text{débit Ethernet}} - \frac{8 \times 53}{\text{débit DQDB}}) \tag{5.10}$$

Les figures 5.2 et 5.3 illustrent les formules (5.8) et (5.9) qu'on observe en fonction de la longueur des rafales (pour des paquets Ethernet de différentes tailles) pour les paramètres suivants: $S_0 = 34$ créneaux (0,1 ms), T_b=1 créneau (2.94 μs). Nous remarquons que S_0 influence μ_u mais pas σ_u qui ne dépend que de la loi de probabilité selon laquelle sont distribués les temps de service pendant la phase de la rafale. Donc plus la distance entre la station et le générateur de cellules devient grand plus l'écart-type du temps d'attente dans le système diminue relativement à la moyenne du temps d'attente. Ces résultats

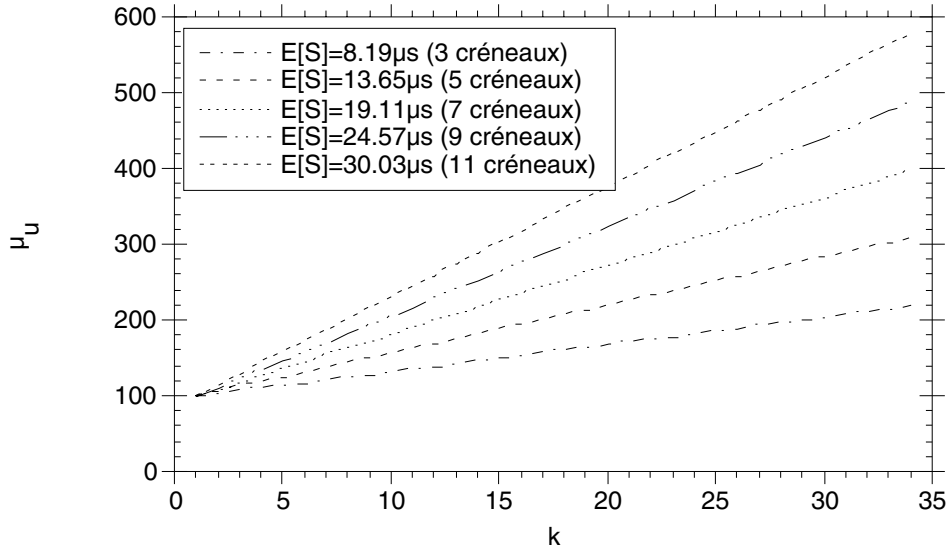

Figure 5.2: Moyenne du temps d'attente des cellules DQDB

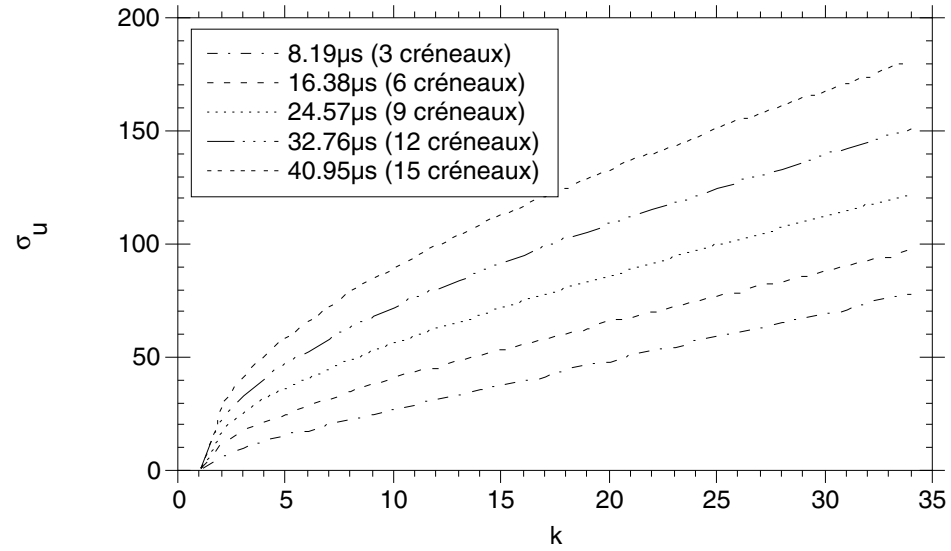

Figure 5.3: Variance du temps d'attente des cellules DQDB, $E[S] = 8$ créneaux

Z	$prob(Y \leq 0)$	$prob(Y \geq 0)$
0	0.5	5×10^{-1}
0.5	0,6914624612	$3,09 \times 10^{-1}$
1	0.8413447460	1.59×10^{-1}
1.5	0.9331927987	6.68×10^{-2}
2	0.9772498680	2.28×10^{-2}
2.5	0.9937903346	6.21×10^{-3}
3	0.9986501019	1.35×10^{-3}
3.5	0.9997673709	2.33×10^{-4}
4	0.9999683288	3.17×10^{-5}
4.5	0.9999966023	3.38×10^{-5}
5	0.9999997133	2.87×10^{-7}

Tableau 5.1: Loi de la distribution normale

ont été vérifiés par simulation [63]. Un avantage est que ces formules sont explicites et peuvent être facilement calculées. Dans les équations (5.8), (5.9), nous avons admis que le système était vide lorsque la première cellule de chaque rafale arrivait dans la file d'attente. Pour que cette condition soit satisfaite, il faut imposer des contraintes à $var[S]$.

Soit

$$Som = S_0 + \sum_{i=1}^{n_c-2} S_j - (n_c - 1)T_b - T_s \tag{5.11}$$

Nous vérifions facilement que pour que le système soit vide lors de l'arrivée de la première cellule de la rafale suivante, $Som \leq 0$. Si on admet n_c suffisamment grand, alors la variable aléatoire Som est distribuée normalement, à cause du théorème central limite [43], c'est-à-dire que

$$prob(Som \leq 0) = prob(Z \leq \frac{(n_c - 1)T_b + T_s - S_0 - (n_c - 2)E[S]}{\sqrt{(n_c - 1)var[S]}}) \tag{5.12}$$

avec Z étant une variable aléatoire réduite étant asymptomatiquement normale.

Le tableau 5.1 est tiré de [64]. Si $E[S]$, n_c et S_0 sont connus alors nous pouvons directement déterminer la variance $var[S]$ pour une probabilité $prob(Som \geq 0)$ voulue.

$$var[S] \leq \left(\frac{(n_c - 1)T_b + T_s - S_0 - (n_c - 2)E[S]}{Z\sqrt{n_c - 1}} \right)^2 \tag{5.13}$$

Les hypothèses que nous avons faites au cours de ce chapitre sont drastiques. Au chapitre 4, nous avons vu que le réseau Ethernet n'était pas constamment chargé mais qu'au contraire, le trafic subit de grandes variations au cours du temps. Dans le chapitre suivant, nous allons regarder quels sont les paramètres dans le trafic qui sont les plus influents du point de vue de l'occupation et des pertes de la file d'attente. A partir de là, nous allons essayer de trouver un modèle markovien qui ait les mêmes paramètres influents que ceux observés sur un réseau Ethernet.

Chapitre 6

Analyse spectrale

6.1 Introduction

Dans ce chapitre nous allons chercher les paramètres du processus d'arrivées de cellules ayant la plus grande influence sur l'occupation et la perte de cellules dans une file d'attente ayant un nombre de places limité. Ensuite, nous allons appliquer le résultat de ces recherches au problème du *fitting*. Nous allons rechercher un modèle markovien basé sur ces observations. Auparavant, nous allons voir quels ont été les travaux précédents effectués dans ce domaine.

Livny et al. [65] ont considéré l'impact de l'autocorrélation sur des files d'attente. Leur étude est basée sur une série de simulations. Pour cela, ils ont généré des arrivées corrélées avec différents types de corrélations. En outre, ils ont remarqué que l'impact de ces différents types de trafic pouvait être catastrophique du point de vue du comportement de la file d'attente . Luoni [27] a essayé de trouver les coefficients d'un modèle *Autoregressive Moving Average* (ARMA) à partir d'un spectre donné. Pour cela, il a remarqué qu'au point de vue de l'occupation de la file d'attente, le spectre autour de 0 Hz avait une grande importance. Le raisonnement est très simple, intuitif. Il considère l'occupation de la file d'attente lorsque cette dernière n'est ni pleine, ni vide. N_t représente l'occupation de la file d'attente à l'instant t et X_t représente le nombre de cellules arrivant entre les instants $t-1$ et t. On admet qu'une cellule est servie à chaque instant t, autrement dit

la file d'attente est du type $G/D/1$. Alors on peut écrire que

$$N_t = max(0, N_{t-1} + X_t - 1) = N_{t-1} + X_t - 1 \tag{6.1}$$

Si nous considérons N_t et X_t en fonction du temps, N_t est l'intégrale de X_t. Cette relation intégrale peut être translatée dans le domaine féquenciel, ainsi nous trouvons facilement que

$$\mathcal{F}(N_t) = \frac{\mathcal{F}(X_t)}{f} \tag{6.2}$$

avec $\mathcal{F}(N_t)$ étant la transformée de Fourier de N_t et f la fréquence. Ça signifie que le spectre autour de la fréquence 0 d'une source a une grande importance sur l'occupation de la file d'attente. Le pionnier de ce domaine est certainement San Qi Li. Dans un de ses premiers articles dans le domaine [66], il analyse le comportement d'une file d'attente de type $G/D/1$ et il a trouvé que l'intensité du trafic ainsi que les basses fréquences jouent un grand rôle sur l'occupation moyenne de la file d'attente. Dans un autre article, présenté à *Infocom'92* [67], il observe la réponse de la file d'attente soumise à différentes fonctions périodiques (sinusoïdales, impulsions rectangulaires, impulsions triangulaires). Là encore, il trouve que les fréquences basses ont un impact dominant sur le comportement de la file d'attente. Il pense très important d'étudier le spectre des processus d'entrée de la file d'attente car il pense possible de pouvoir approcher n'importe quel type de spectre d'un processus stochastique d'entrée stationnaire en combinant plusieurs fonctions de base. En outre, il pense qu'une limitation fondamentale de l'utilisation des chaînes de Markov à deux états est que sa fonction d'autocorrélation décroit selon une loi géométrique, ce qui est certainement insuffisant pour représenter tout le spectre des sources multimédia. Dans un autre article consacré au domaine [68], San Qi Li élargit les observations faites sur le comportement d'une file d'attente. Il considère le bi-spectre et le tri-spectre du processus d'entrée qui est en fait un MMPP. En général, cette démarche est possible mais par contre la démarche inverse est plus difficile (trouver la chaîne de Markov à partir d'un spectre donné en s'aidant de l'analyse de l'influence de chacune des valeurs propres), en partie car on sait que cette manière de procéder implique le *problème inverse des valeurs propres* [45], c'est-à-dire de créer une matrice de transition à partir de valeurs propres données. De tels problèmes sont difficiles en général; les matrices de transition peuvent

être inexistantes pour un certain ensemble de valeurs propres par exemple. Dans ces articles, San Qi Li ne donne pas de réponse à ce problème bien qu'il l'ait mis en évidence. Mais une conclusion très importante qui découle de [68] est que les ordres supérieurs ont un effet minime sur le comportement de la file d'attente et qu'il suffit de considérer uniquement la distribution et la covariance des deux premiers ordres. Zuckermann [69] propose un nouvel algorithme de CAC basé sur l'estimation de 3 paramètres (moyenne, variance et $\Phi(0)$) et en faisant l'hypothèse que le trafic multiplexé ATM se comporte comme un processus gaussien. Courcoubetis [70] affirme que pour un grand nombre de sources stationnaires, la largeur de bande effective est donnée par une formule impliquant l'intensité du trafic, son indice de dispersion et la grandeur de la file d'attente. Mukherjee [71] a remarqué des composantes de basses fréquences dans les retards des paquets. Avant de nous lancer dans nos calculs, sachons que d'autres chercheurs ont déjà essayé de baser leur modèle sur des mesures. Gusella [48] a tenté d'incorporer les mesures de variabilité dans ses modèles analytiques. Il propose une procédure de *fitting* basé sur l'indice de dispersion pour les arrivées pour un processus markovien modulé MMPP. Ding [72] a étudié un modèle de type SSMP, il pense que ce modèle est suffisamment général pour pouvoir faire le *fitting* de données mesurées. Le *fitting* qu'il propose est basé sur la fonction de distribution du SSMP mais notons que cet argument sert uniquement à la justification de l'emploi de ce modèle. Aucun exemple pouvant soutenir son hypothèse n'est présent. Andrade [73] propose de caractériser le trafic LAN à l'aide de 3 paramètres: la moyenne et deux points de l'indice de dispersion. Il compare la performance d'une file d'attente limitée sous deux types de trafic, l'un mesuré et l'autre généré à l'aide d'une superposition de sources ON-OFF.

6.2 Paramètres les plus influents pour la file d'attente

Les résultats sur lesquels nous nous appuions dans cette section sont dûs à Grünenfelder. Il a trouvé [38] que les paramètres qui influencent le plus le comportement de la file d'attente en ce qui concerne son occupation moyenne, son taux de pertes et le délai

moyen d'une cellule la traversant sont

- L'espérance mathématique du processus $U(t)$, $U(t)$ étant le nombre des arrivées moins les départs au temps t. $U(t)$ est supposé être ergodique et stationnaire au sens large.
- La variance du processus $U(t)$.
- La densité spectrale du processus $U(t)$ en zéro, $\Phi_u(0)$.

La densité spectrale en zéro $\Phi_u(0)$ représente la somme sur tous les intervalles de temps de la fonction d'autocovariance du processus $U(t)$. Il est important de remarquer que le comportement de la file d'attente dépend de la globalité de la fonction d'autocovariance car $\Phi_u(\omega) = \sum_{k=-\infty}^{\infty} C(k)e^{-j\omega k}$. Pour les systèmes de type ATM, le temps de service est constant donc le comportement de la file d'attente ne dépend que de la densité spectrale en zéro du processus des arrivées. En plus, si les arrivées sont non corrélées, $\Phi(0)_u = \Phi(0)$ est égale à la variance du processus des arrivées. On voit donc que $\Phi(0)$ comprend un terme relatif à la variance et un terme relatif à la corrélation. L'influence de la corrélation sur le comportement de la file d'attente devient plus évident. Il faut relever que ces paramètres sont seulement des proportionnalités et ne servent pas au calcul de l'occupation de la file d'attente ou de son taux de perte par exemple. Dans la section suivante, nous allons calculer ces paramètres pour une chaîne de Markov à deux états.

6.3 Source markovienne à 2 états

La chaîne de Markov à 2 états comprend un état dans lequel la taille du *batch* est nulle et dans un autre état dans lequel la taille du *batch* est égale à 1. On appelera aussi cette source une source ON-OFF. La matrice de transition du modulateur est donnée par

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \tag{6.3}$$

$$\mathbf{A} = \begin{pmatrix} 1-p & p \\ q & 1-q \end{pmatrix} \tag{6.4}$$

Le processus est admis stationnaire au sens large. La matrice $\mathbf{\Lambda}$ est donnée par

$$\mathbf{\Lambda} = \begin{pmatrix} E[X|Y_t = 0] & 0 \\ 0 & E[X|Y_t = 1] \end{pmatrix} \tag{6.5}$$

$$\mathbf{\Lambda} = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} \tag{6.6}$$

Les principales caractéristiques de la chaîne de Markov à 2 états sont connues. Il suffit d'appliquer les résultats du chapitre 3 pour trouver que le vecteur des probabilités d'état est égal à

$$\vec{\pi} = \begin{pmatrix} \pi_0 & \pi_1 \end{pmatrix} \tag{6.7}$$

$$\vec{\pi} = \frac{1}{p+q} \begin{pmatrix} q & p \end{pmatrix} \tag{6.8}$$

On peut démontrer très facilement que pour $p + q = 1$ [74], les arrivées ne sont pas corrélées. Tous les moments de ce processus stochastique sont égaux, $E[X] = E[X^2] = ... = E[X^k] = \pi_1$ car

$$E[X^k] = \sum_{x=0}^{1} x^k prob(X_t = x) = \tag{6.9}$$

$$\begin{aligned} = \textstyle\sum_{x=0}^{1} \quad & x^k (prob(X_t = x|Y_t = 0) prob(Y_t = 0) \\ & + prob(X_t = x|Y_t = 1) prob(Y_t = 1)) \end{aligned} \tag{6.10}$$

par le théorème des probabilités totales

$$E[X^k] = \sum_{x=0}^{1} x^k prob(X_t = x|Y_t = 1) prob(Y_t = 1) \tag{6.11}$$

$$= 1^k prob(Y_t = 1) = \pi_1 \tag{6.12}$$

Les valeurs propres de la matrice $\mathbf{A}$ sont données par: $\lambda_1 = 1$ et $\lambda_2 = 1 - p - q$ qui sont réelles, nous pouvons donc déduire l'autocovariance de ce processus par (3.22)

$$C(0) = \frac{pq}{(p+q)^2} \tag{6.13}$$

$$C(k) = pq/((p+q)^2)(1-p-q)^{|k|} \quad k = 1, 2, ... \tag{6.14}$$

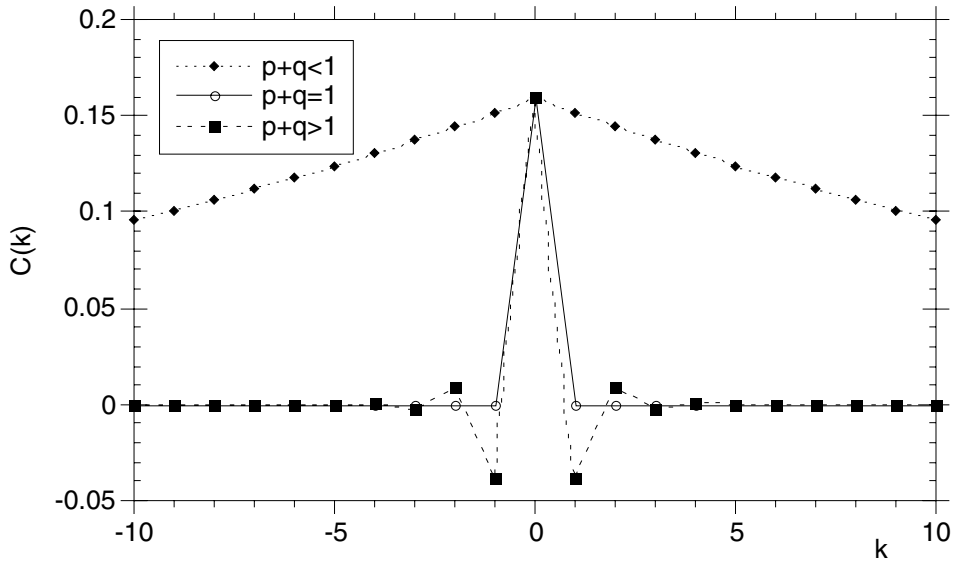

Figure 6.1: Covariance du processus MMPP à deux états

Connaissant l'autocovariance du processus, il est également très facile d'en déduire la transformée de Fourier

$$\Phi(\omega) = C(0) + 2\frac{pq}{(p+q)^2}\left(\frac{1-(1-p-q)cos(\omega)}{1-2(1-p-q)cos(\omega)+(1-p-q)^2} - 1\right) \tag{6.15}$$

$\Phi(\omega)$ est une fonction paire car $C(k)$ est une fonction paire. $\Phi(0)$ est donné par

$$\Phi(0) = C(0) + 2\frac{pq}{(p+q)^2}\left(\frac{1-(1-p-q)}{1-2(1-p-q)+(1-p-q)^2} - 1\right) \tag{6.16}$$

Avec ce processus, il n'y a que deux paramètres libres, p et q. Il est donc possible de trouver exactement ces deux paramètres en fonction de $E[X]$ et $\Phi(0)$ en résolvant un système de deux équations à deux inconnues. Notons ici que la variance est complètement déterminée par $E[X]$ puisque $Var[X] = E[X](1-E[X]) = C(0)$. $E[X]$ est donné par la formule (6.12) et $\Phi(0)$ par la formule (6.15). Ainsi

$$p = \frac{2E[X]}{1+\phi(0)/(E[X](1-E[X]))} \tag{6.17}$$

et

$$q = p\frac{1-E[X]}{E[X]} \tag{6.18}$$

Pour $p+q=1$, le trafic n'est pas corrélé, le spectre est constant pour toutes les fréquences (figure 6.2). Ce type de processus est de type bernoullien, sans mémoire. Pour

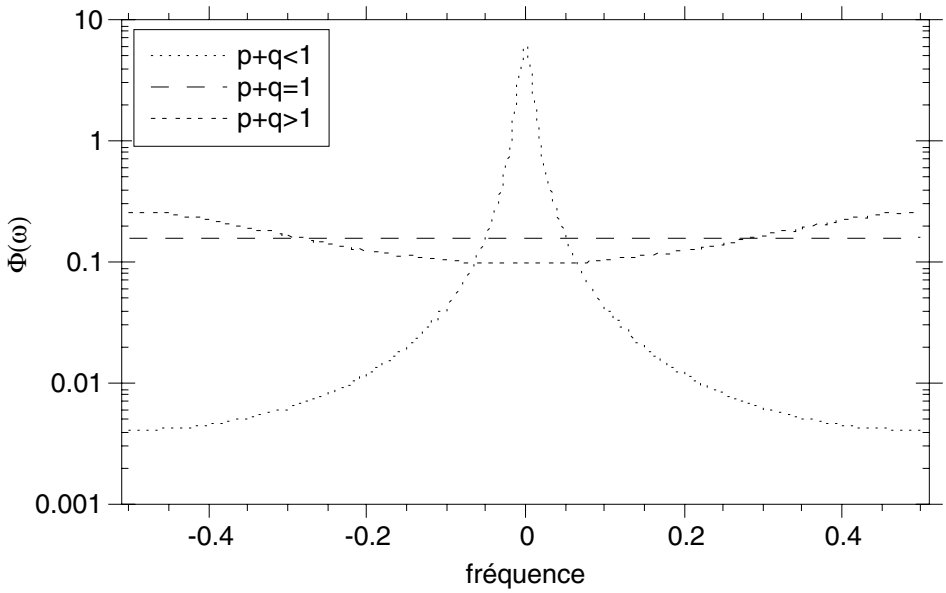

Figure 6.2: Transformée de Fourier du processus MMPP à deux états

$p+q < 1$, l'autocovariance du processus stochastique $C(k)$ décroit exponentiellement avec k (figure 6.1) et est toujours positive alors que pour $p+q > 1$, $C(k)$ est alternativement positive et négative. Dans ce cas, on a également une décroissance exponentielle de $|C(k)|$.

6.4 Domaine de définition

Il est possible que pour un certain ensemble $\{E[X], \Phi(0)\}$, il ne soit pas possible de trouver une chaîne de Markov à deux états qui ait ces valeurs. On sait que $p, q \in [0, 1]$. Si on pose $p = 1$, on trouve que $\Phi_{min}(0) = -2E^3[X] + 3E^2[X] - E[X]$ et si on pose $q = 1$ alors $\Phi_{min}(0) = 2E^3[X] - 3E^2[X] + E[X]$ avec $\Phi_{min}(0)$ étant le plus petit $\Phi(0)$ possible. $E[X]$ varie de 0 à 1. Les deux équations ci-dessus sont l'inverse l'une de l'autre. Pour tous les cas possibles, $\Phi_{min}(0) = |2E^3[X] - 3E^2[X] + E[X]|$. Seulement pour $E[X] = 0.5$, $\Phi(0)$ peut devenir aussi petit qu'on veut.

6.5 Procédure de *fitting*, MMPP(2)

Dans cette section, il sera traité de la procédure de *fitting* avec MMPP(2). La démarche peut être appliquée à d'autres types de distributions que Poisson. Il suffit de calculer les différents moments de la distribution envisagée et de recalculer les équations. Les équations pour les différents moments et $\Phi(0)$ s'écrivent (section 11.3)

$$\begin{aligned} E[X] &= \pi_1\alpha_1 + \pi_2\alpha_2 && (6.19)\\ E[X^2] &= E[X] + \pi_1\alpha_1^2 + \pi_2\alpha_2^2 && (6.20)\\ E[X^3] &= -2E[X] + 3E[X^2] + \pi_1\alpha_1^3 + \pi_2\alpha_2^3 && (6.21)\\ \Phi(0) &= E[X] + \left(\frac{2}{p+q}\right)(var[X] - E[X]) && (6.22)\\ \pi_1 + \pi_2 &= 1 && (6.23) \end{aligned}$$

avec $\vec{\pi_t} = (prob(Y_t = 1)\ prob(Y_t = 2))$. De (6.19) nous tirons α_2 que nous remplaçons dans (6.20). En tenant compte de (6.23), nous obtenons l'équation suivante:

$$(\pi_1^2 + \pi_1(1-\pi_1))\alpha_1^2 - \alpha_1(2\pi_1 E[X]) + (E[X])^2 - \pi_2((E[X^2]) - (E[X])) = 0 \qquad (6.24)$$

A ce stade, nous avons encore deux inconnues dans l'équation: α_1 et π_1. A partir de l'équation (6.24), il est possible d'obtenir α_1:

$$\alpha_1 = E[X] \pm \frac{\sqrt{(var[X] - E[X])\pi_1(1-\pi_1)}}{\pi_1} \qquad (6.25)$$

Mais comme α_2 satisfait à la même équation, il suffit de choisir $\alpha_1 > \alpha_2$ pour déterminer α_1:

$$\alpha_1 = E[X] + \frac{\sqrt{(var[X] - E[X])\pi_1(1-\pi_1)}}{\pi_1} \qquad (6.26)$$

La contrainte supplémentaire pour déterminer π_1 est fournie par l'équation (6.21), en tirant α_2 de (6.19) et π_2 de (6.23):

$$f(\pi_1) = E[X^3] + 2E[X] - 3E[X^2] - \pi_1\alpha_1^3 - (1-\pi_1)\left(\frac{E[X] - \alpha_1\pi_1}{1-\pi_1}\right)^3 = 0 \qquad (6.27)$$

α_1 est donné par (6.26). Deux situations se présentent alors

- $E[X^3]$ et $\Phi(0)$ peuvent être *fittés* , alors π_1 est donné par le zéro de (6.27).

- $E[X^3]$ et $\Phi(0)$ ne peuvent pas être *fittés* de manière optimale. Il faudrait *fitter* premièrement $\Phi(0)$ et ensuite minimiser $|(E[X^3] - \mu^3)|^2$

avec μ^k étant le k^e moment des données observées. Maintenant que les probabilités stationnaires sont trouvées, il s'agit de se concentrer sur la recherche des probabilités de transition de la chaîne de Markov (pour déterminer la fonction d'autocovariance). Il est bien connu que dans le cas d'un processus MMPP(2) $\pi_1 = q/(p+q)$ et $\pi_2 = p/(p+q)$ avec $p = pr(Y_t = 2|Y_{t-1} = 1)$ et $q = pr(Y_t = 1|Y_{t-1} = 2)$ [74]. Y_t est l'état de la chaîne de Markov à l'instant t. De (6.22):

$$p + q = \frac{1}{2}\left(\frac{\widehat{\Phi(0)} - \mu}{\sigma^2 - \mu} + 1\right)^{-1} \tag{6.28}$$

$$q = \frac{\pi_1}{2}\left(\frac{\widehat{\Phi(0)} - \mu}{\sigma^2 - \mu} + 1\right)^{-1} \tag{6.29}$$

$$p = \frac{\pi_2}{2}\left(\frac{\widehat{\Phi(0)} - \mu}{\sigma^2 - \mu} + 1\right)^{-1} \tag{6.30}$$

avec σ^2 étant la variance des données observées et $\widehat{\Phi(0)}$ la densité spectrale des données observées en zéro. Nous remarquons selon les équations (6.20) et (6.22) que les données observées ne peuvent être *fittées* avec précision uniquement si $\sigma^2 \geq \mu$ et si $\widehat{\Phi(0)} \geq \mu$. La première restriction vient de l'hypothèse selon laquelle les arrivées dans chaque état de la chaîne de Markov sont générées selon une loi de Poisson tandis que la seconde est valable pour n'importe quel processus SSMP à deux états. L'équation (6.22) est indépendante de l'hypothèse poissonnienne car la densité spectrale du processus ne dépend que du modulateur et de la moyenne des arrivées de cellules dans chacun de ses états. Nous voyons donc qu'il n'est pas toujours possible de faire un *fitting* des données observées à l'aide d'un processus SSMP(2). Pour montrer l'importance des paramètres sus-mentionnés, nous allons consacrer une section à des exemples illustratifs.

6.6 Exemples

Dans cette section, nous allons montrer trois exemples de *fitting*. Les données observées seront en fait générées par une chaîne de Markov à trois états, MMPP(3), ayant

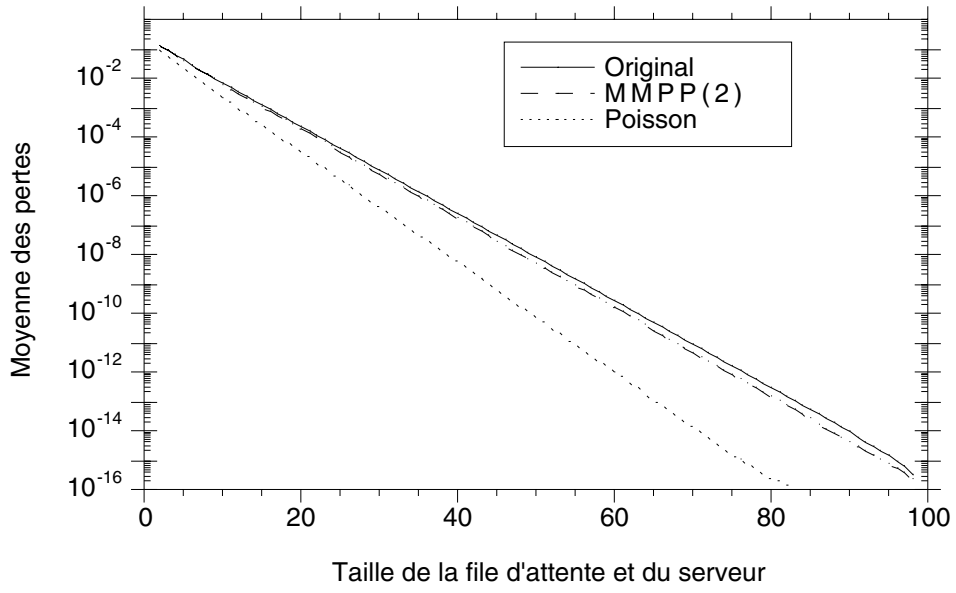

Figure 6.3: Moyenne des pertes dans la file d'attente en fonction de sa longueur

différentes caractéristiques. La première série de données observées est issue d'une chaîne de Markov MMPP(3) ayant une matrice de transition

$$\mathbf{A} = \begin{pmatrix} 0.1 & 0.8 & 0.1 \\ 0.2 & 0.1 & 0.7 \\ 0.8 & 0.1 & 0.1 \end{pmatrix} \tag{6.31}$$

et les paramètres des distributions de Poisson dans les différents états sont $\alpha_1 = 0.1$, $\alpha_2 = 2.0$, $\alpha_3 = 0.25$. On en déduit $\mu = 0.8$, $\sigma^2 = 1.56$, $\mu^3 = 7.88$ et $\widehat{\Phi(0)} = 0.9873$. En appliquant la procédure de *fitting* de ci-dessus, on trouve un processus MMPP(2) ayant une matrice de transition

$$\mathbf{A} = \begin{pmatrix} 0 & 1 \\ 0.604 & 0.396 \end{pmatrix} \tag{6.32}$$

avec $\alpha_1 = 1.92173$, $\alpha_2 = 0.122475$. On en déduit $E[X] = 0.8$, $var[X] = 1.56$, $E[X^3] = 7.673$ et $\Phi(0) = 0.9873$. On peut également trouver un processus MMPP(1) qui est en fait une distribution de Poisson ayant le premier moment identique à celui des données observées $\alpha_1 = 0.8$ ce qui donne $E[X] = 0.8$, $var[X] = 0.8$, $E[X^3] = 3.232$ et $\Phi(0) = 0.8$. Lors de ce *fitting*, nous avons volontairement "mal" choisi les paramètres de manière à ce que nous puissions voir l'influence du troisième moment sur le comportement de la file

d'attente au niveau des pertes (qui sont calculées avec la méthode exposée au chapitre 3). La figure 6.3 montre qu'une petite déviation du troisième moment a une influence relativement faible sur le comportement de la file d'attente. Par contre nous verrons dans le deuxième exemple qu'une petite déviation de la densité spectrale en zéro a une grande influence sur la probabilité de perte (figure 6.4). Lorsque la taille de la file d'attente et du serveur est de 20 cellules, nous avons déjà une différence d'une décade entre les deux moyennes de pertes dans le système alors qu'elle n'est que d'un facteur deux pour une taille de la file d'attente et du serveur de 100 cellules dans le premier exemple. La deuxième série de données observées est également issue d'un processus MMPP(3) ayant les caractéristiques suivantes

$$\mathbf{A} = \begin{pmatrix} 0.1 & 0.8 & 0.1 \\ 0.2 & 0.1 & 0.7 \\ 0.8 & 0.1 & 0.1 \end{pmatrix} \tag{6.33}$$

avec $\alpha_1 = 0.0001$, $\alpha_2 = 0.85$, $\alpha_3 = 0.025$, ce qui donne $\mu = 0.3$, $\sigma = 0.459$, $\mu^3 = 1.277$ et $\widehat{\Phi(0)} = 0.340$. Le *fitting* au processus MMPP(2) nous fournit une matrice de transition

$$\mathbf{A} = \begin{pmatrix} 0 & 1 \\ 0.568 & 0.432 \end{pmatrix} \tag{6.34}$$

avec $\alpha_1 = 0.833$, $\alpha_2 = 0.0$, ce qui donne $E[X] = 0.3$, $var[X] = 0.549$, $E[X^3] = 1.275$et $\Phi(0) = 0.3438$ et au processus MMPP(1) qui est une distribution de Poisson avec $\alpha_1 = 0.3$, ce qui donne $E[X] = 0.3$, $var[X] = 0.3$, $E[X^3] = 0.597$ et $\Phi(0) = 0.3$.

La troisième série de données observées est générée par un processus MMPP(3) avec

$$\mathbf{A} = \begin{pmatrix} 0.05 & 0.95 & 0.0 \\ 0.0001 & 0.99989 & 0.00001 \\ 0.0 & 0.95 & 0.05 \end{pmatrix} \tag{6.35}$$

et $\alpha_1 = 50.0$, $\alpha_2 = 0.50$, $\alpha_3 = 5.0$ ce qui donne $\mu = 0.505$, $\sigma^2 = 0.763$, $\mu^3 = 15.328$ et $\widehat{\Phi(0)} = 0.790$. Le *fitting* au processus MMPP(2) donne

$$\mathbf{A} = \begin{pmatrix} 0.999909 & 0.000091 \\ 0.950017 & 0.049983 \end{pmatrix} \tag{6.36}$$

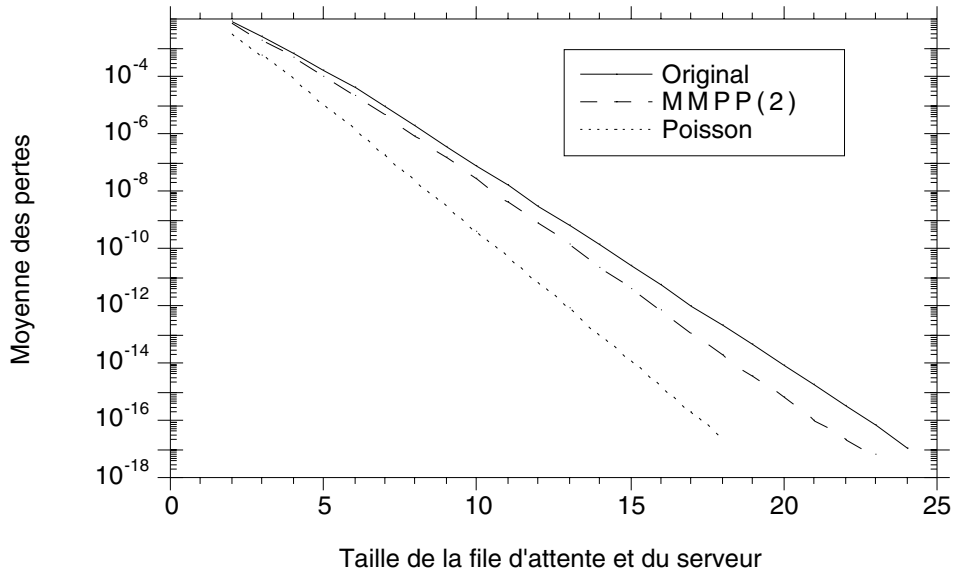

Figure 6.4: Moyenne des pertes dans la file d'attente en fonction de sa longueur

avec $\alpha_1 = 0.500283$, $\alpha_2 = 52.386$ ce qui donne $E[X] = 0.505$, $var[X] = 0.763$, $E[X^3] = 15.953$ et $\Phi(0) = 0.790$ et au processus MMPP(1) qui est une distribution de Poisson avec $\alpha_1 = 0.505$ ce qui donne $E[X] = 0.505$, $var[X] = 0.505$, $E[X^3] = 1.399$, $\Phi(0) = 0.505$

Dans ce troisième exemple, nous voyons que même un processus qui provoque un "coude" dans le graphe de la moyenne des pertes peut être *fitté* par un processus MMPP(2). Nous savons que ce type de comportement est observé lorsque le processus d'entrée de la file d'attente est une superposition d'un grand nombre de sources ON-OFF [75].

6.7 Source markovienne à 5 états

Une grande limitation de la chaîne de Markov à deux états est que ses deux valeurs propres sont nécessairement réelles. Par le théorème 3.3, une valeur propre est égale à 1 alors que l'autre est obligatoirement réelle. La fonction d'autocovariance $|C(k)|$ est ainsi une exponentielle décroissante. Le spectre du processus stochastique est certainement trop pauvre pour pouvoir représenter toutes les composantes spectrales des données mesurées, c'est pourquoi nous suggérons l'emploi d'une chaîne de Markov à cinq états. D'après le théorème 3.4, les valeurs propres de la matrice de transition $\mathbf{A}$ sont soit réelles, soit conjuguées complexes. Autrement dit, avec une chaîne de Markov à cinq états,

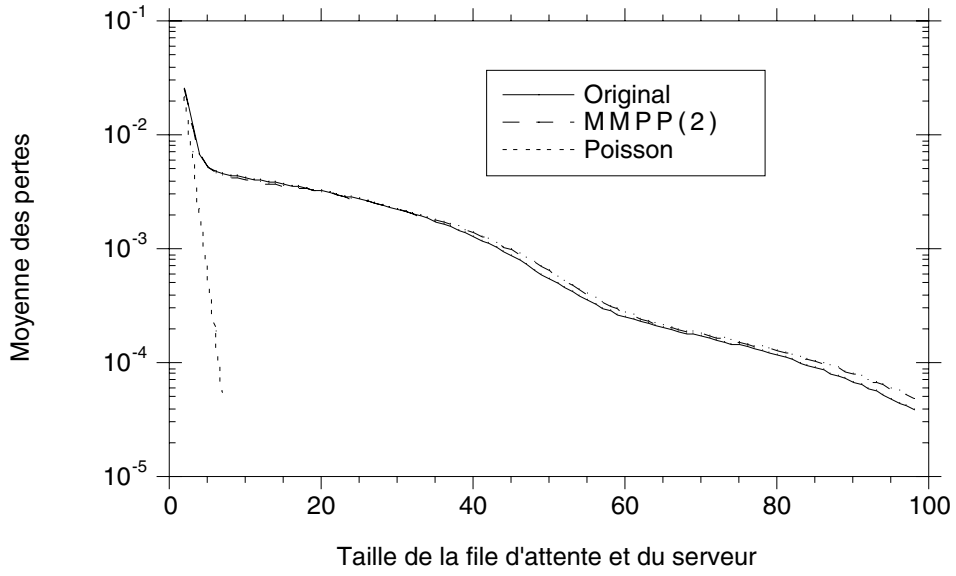

Figure 6.5: Moyenne des pertes dans la file d'attente en fonction de sa longueur

les valeurs propres peuvent être toutes réelles, ou alors trois sont réelles et deux sont conjuguées complexes, ou encore une est réelle ($\lambda_1 = 1$) et 4 sont conjuguées complexes. D'autre part, la couverture du cercle unité est importante (théorème 3.4). Toutes les caractéristiques du processus stochastisque qui vont nous intéresser par la suite sont calculées au chapitre 3. Notons que le processus stochastique est stationnaire au sens large et que $X_t \in \{0, 1\}$. Autrement dit, on considère un lien ATM sur lequel on ne peut avoir qu'une seule cellule par créneau.

6.8 Fonction à minimiser

Dans cette section, le but est de trouver une chaîne de Markov dont la transformée de Fourier va se rapprocher le plus possible du spectre estimé à partir de mesures faites sur le réseau de l'EPFL, spécialement dans les basses fréquences. Il faudrait aussi que l'espérance mathématique du processus stochastique soit proche de la moyenne observée du trafic mesuré. Il a été vu que ces paramètres ont une grande influence sur le comportement de la file d'attente. Les paramètres libres de la chaîne de Markov sont les probabilités de transition de cette dernière ainsi que les probabilités conditionnelles $prob(X_t = i|Y_t = j)$ avec $i = 0, 1$ et $j = 1, 2, .., 5$. Il y a ainsi $n \times n + n = 30$ pa-

ramètres inconnus dont 5 sont des entiers binaires, les autres sont réels et compris entre 0 et 1. Il s'agit maintenant de définir une fonction objectif qui va pouvoir quantifier la valeur du *fitting*. La fonction choisie est la suivante

$$FO = \sum_i PS(i) \times de(SM(i), \Phi(\omega = i)) + \sum_j PD(j) \times de(DM(j), E[X^j]) \qquad (6.37)$$

avec FO étant la fonction objectif, $PS(i)$ étant la fonction de poids à la fréquence i (compris entre -512 et 512 si la fenêtre est de 1024 échantillons par exemple); de représente la distance euclidienne; $SM(i)$ est le spectre mesuré en i; $PD(j)$ est la fonction de poids pour le j^{e} moment; $DM(j)$ est égal au j$^{\text{e}}$ moment mesuré. Les fonctions de poids ont pour but de renforcer le *fitting* dans les basses fréquences et sur les premiers moments (ici tous les moments sont égaux donc la fonction de poids n'a qu'un rôle relatif à jouer). Le but maintenant est de minimiser FO. Il s'agit là d'un problème très classique d'optimisation en recherche opérationelle. Avant de parler de la méthode utilisée, regardons comment le spectre a été évalué pour les mesures faites sur le réseau de l'EPFL.

6.9 Estimation du spectre

Pour obtenir une bonne estimation du spectre, il est nécessaire d'avoir à disposition un grand nombre de données. C'est dans la situation dans laquelle nous nous trouvons. Il circule environ $9,6 \times 10^6$ créneaux sur un réseau DQDB fonctionnant à 34 Mb/s. Nous avons mesuré le trafic sur le réseau Ethernet pour ensuite le convertir en trafic DQDB (comme à la section 5.2). Pour l'estimation du spectre, il y a principalement deux techniques très connues, l'estimation spectrale non-paramétrique et l'estimation spectrale paramétrique. La première technique est appropriée quand le nombre de données est suffisamment grand. Quant aux méthodes non paramétriques, les plus populaires sont 1) l'estimateur spectral du périodogramme. 2) l'estimateur spectral du périodogramme moyenné. 3) l'estimateur à variance minimum. Ces algorithmes ont l'avantage de ne faire aucune hypothèse quant à la nature du signal sinon qu'il doit être stationnaire au sens large. L'inconvénient par contre est le compromis à faire entre la variance et le biais de

l'estimation. En effet, si l'estimateur spectral fournit une bonne estimée en moyenne (peu de biais), alors on peut s'attendre à obtenir une grande variabilité d'une réalisation à l'autre (variance élevée). Par contre si l'estimateur spectral a peu de variabilité, alors la moyenne de ce dernier peut être mauvaise (biais important). La seule façon de résoudre ce dilemme est d'augmenter le nombre de données. Dans notre cas, ce nombre peut être aussi élevé qu'on veut. Ainsi, la méthode choisie a été l'estimateur spectral du periodogramme moyenné [76]. La seconde technique est bien appropriée lorsque le nombre de données à analyser est petit. Un avantage de cette technique est que les modèles obtenus peuvent décrire le processus stochastique. Les modèles les plus connus sont les modèles AR, MA et ARMA.

6.10 Méthode d'optimisation, *Tabu Search*

Le but est de minimiser FO. Afin d'être assez clair, nous allons définir un peu plus précisément quel est le problème à résoudre. Etant donné un ensemble V de solutions $\mathbf{A}$ et FO assignant à chaque $\mathbf{A}$ dans V une valeur réelle $FO(\mathbf{A})$ dont nous voulons trouver une solution A^+ dans V telle que $FO(A^+)$ soit minimum. Dans notre problème $\mathbf{A}$ est une matrice stochastique. La matrice $\mathbf{A}$ ne comprend pas tous les paramètres puisqu'il reste encore $\mathbf{\Lambda}$ à déterminer: les probabilités conditionnelles $prob(X_t = i|Y_t = j)$ avec $i = 0, 1$ et $j = 1, 2, \cdots, n$. Le problème sera résolu séquentiellement. On va chercher une solution $\mathbf{A}^+$ pour chaque configuration possible. Dans la première configuration par exemple $\phi_{i1} = prob(X_t = 1|Y_t = i) = 1$, $\phi_{i0} = 0$ et $\phi_{j1} = 0$, $\phi_{j0} = 1$ $\forall i = 1, \cdots, k$ et $\forall j = k+1, \cdots, n$ avec $k = 1$, pour la seconde $k = 2$ et ainsi de suite jusqu'à $k = n - 1$. Pour une chaîne de Markov à n états, il y aura $n-1$ configurations possibles. Ce problème ne peut pas être résolu de manière analytique, d'un seul coup. Il faut alors utiliser des procédures itératives qui vont permettre à $\mathbf{A}$ d'explorer l'ensemble V. Une procédure itérative permet de passer d'une solution dans V à une autre, $\mathbf{A}'$ par exemple, autant de fois que cela semble nécessaire. Pour garder la procédure relativement simple, il est usuel de restreindre les modifications à une classe de modifications simples. Soit $\mathbf{M}$ une matrice de modification tel que $\mathbf{A}' = \mathbf{A} + \mathbf{M}$. Soit $\mathcal{M}$ l'ensemble des modifications acceptables.

Pour que $\mathbf{A}'$ reste dans l'ensemble V, il faut que la somme des éléments de chaque ligne de $\mathbf{M}$ soit nulle. Il sera ainsi créé un nouvel ensemble appelé $\mathcal{M}_A$ inclu dans $\mathcal{M}$. Nous pouvons maintenant introduire la notion de voisinage $N(\mathbf{A})$ pour chaque solution $\mathbf{A}$

$$N(\mathbf{A}) = \mathbf{A}' | \exists \mathbf{M} \in \mathcal{M}_A : \mathbf{A}' = \mathbf{A} + \mathbf{M} \tag{6.38}$$

Soit ϵ un nombre réel, petit (de l'ordre de 0.01 à 0.05 dans nos calculs), alors nous allons restreindre $\mathcal{M}_A$ aux matrices $\mathbf{M}$ n'ayant que deux éléments non nuls. Autrement dit si $m_{ij} = \epsilon$ alors $m_{ik} = -\epsilon$, $\forall i, j, k = 1, 2, .., n$, les autres éléments sont nuls. Pour résoudre ce problème, il existe un grand nombre de méthodes, entre autres les méthodes de type gradient. Avec ce type de méthode, la stratégie est relativement simple, elle consiste à trouver une solution $\mathbf{A}'$ dans le voisinage $N(\mathbf{A})$ tel que $FO(\mathbf{A}') < FO(\mathbf{A})$, la solution $\mathbf{A}'$ est alors appelée meilleure solution. La procédure continue tant qu'elle peut trouver une meilleure solution mais dès qu'il n'y a plus de meilleure solution alors elle s'arrête. Un inconvénient majeur avec ce type de méthode est que la procédure peut être piégée dans un minimum local. La solution $\mathbf{A}^+$ (minimum global) peut alors se trouver très loin de celle donnée par la procédure. Pour remédier à ce problème, certaines méthodes ont été développées autour des années 1950 comme le recuit simulé (*simulated annealing*). Cet algorithme utilise un générateur de nombres aléatoires pour sa recherche. La matrice $\mathbf{M}$ de modifications est changée à chaque pas d'itération; ϵ n'est pas un paramètre fixe mais est déterminé par une loi de probabilité qui elle-même dépend d'un paramètre appelé **température** qui va diminuer au cours du temps. A chaque température, on attendra d'avoir atteint un équilibre thermique avant de continuer de la descendre. L'équilibre thermique est atteint quand un nombre maximum (prédéterminé) de transformations ont été acceptées ou générées. Quand la température diminue, de moins en moins de nouvelles solutions augmentant FO ne sont acceptées. Les variations de ϵ deviennent de plus en plus faibles au fur et à mesure que la recherche du minimum global avance. Une nouvelle solution $\mathbf{A}'$ est acceptée avec une certaine probabilité qui dépend de la température si $FO(\mathbf{A}') > FO(\mathbf{A})$ mais si $FO(\mathbf{A}') < FO(\mathbf{A})$ alors $\mathbf{A}'$ est toujours acceptée. Il est intéressant de savoir que la littérature concernant le recuit simulé est abondante [77, 78, 79, 80, 81, 82]. L'inconvénient de cet algorithme est qu'il peut accepter des cycles et rester bloqué sur un ensemble restreint de solutions. Pour

remédier à cet inconvénient, un nouvel algorithme a été inventé par Glover [83, 84] et Hansen [85] indépendamment, l'algorithme *Tabu Search*. Cet algorithme recherche le minimum global de façon plus intelligente en appliquant quelques règles simples de l'intelligence artificielle. Dès qu'une solution $\mathbf{A}'$ pire que $\mathbf{A}$ a été évaluée, il y a un risque de retour en arrière (cycle). Pour réduire les cycles, une méthode consiste à garder en mémoire tous les derniers états visités et de ne pas y retourner pendant un certain temps. Ça consiste à tenir à jour une liste T qui comprend tous les derniers états visités. T est appelée la **liste tabu**. Quand on examine le voisinage $N(\mathbf{A})$ d'une solution $\mathbf{A}$, on va vérifier que la prochaine solution $\mathbf{A}'$ n'est pas dans T pour pouvoir l'accepter. Ainsi, on élimine les cycles. Si on veut les éliminer complètement, il faudrait qu'une solution déjà visitée ne puisse plus jamais l'être. Cette contrainte est trop restrictive, c'est pourquoi nous limitons la taille de la liste tabu aux T derniers états visités. Revenons à notre problème, nous n'allons pas mémoriser toutes les dernières matrices $\mathbf{A}$ à cause de la place mémoire que ça prend, ni même toutes les dernières matrices $\mathbf{M}$. A la place de ça, on va introduire un indicateur + et un indicateur -. Dans la table tabu, on va mémoriser les derniers déplacements de la manière suivante: $[i, j, +]$ indique que $m_{ij} = \epsilon$ et $[i, j, -]$ indique que $m_{ij} = -\epsilon$. Nous allons alors maintenir une liste tabu T séparée en 2, T^- et T^+. T^- comprendra toutes les modifications négatives et T^+ les modifications positives. Ces deux parties de listes sont numérotées, on arrive donc à déterminer exactement quelles ont été les directions prises dans le passé. Il s'agit donc à chaque pas d'itération, de regarder dans la liste tabu si le déplacement est autorisé ou non. Les déplacements interdits sont ceux qui nous font revenir sur nos pas et ceux qui vont dans la même direction que nous avons prise précédemment. Pourquoi interdire d'aller dans la même direction? Parce que nous avons adopté la stratégie suivante: lorsque une direction est favorable, on ne va pas se limiter à une avance de ϵ pour le pas suivant mais on va rechercher la grandeur du pas la plus favorable dans la direction trouvée à l'aide d'une loi dichotomique. Ainsi la grandeur du pas passera de ϵ à 2ϵ à 4ϵ et ainsi de suite jusqu'à ce que la distance FO devienne plus grande que la distance FO du pas précédent. Revenons à la taille de la liste tabu; si elle comprend au moins un élément, il sera impossible par exemple d'effectuer un premier déplacement tel que $m_{11} = \epsilon$ et

$m_{13} = -\epsilon$ et au pas suivant le déplacement inverse $m_{11} = -\epsilon$ et $m_{13} = \epsilon$. On comprend donc qu'il faut limiter la liste tabu à T directions, sinon on risque de se trouver bloqué. Le nombre d'éléments stockés dans la liste tabu reste constant et est déterminé au début de la recherche. Les éléments les plus anciens de la liste seront éliminés au profit des nouveaux. On appelera **conditions tabu** tout déplacement interdit par la liste tabu T. Des cycles peuvent apparaître dès lors qu'on limite la taille de la liste tabu (entre 5 et 10 pour nous) mais ils seront au pire de l'ordre de la taille de la liste tabu. Notons pour terminer que le choix de la condition initiale peut influencer la vitesse de convergence de l'algorithme. S'il est possible de choisir une bonne condition initiale, alors autant la choisir pour autant que cet effort reste limité.

6.11 Algorithme

Dans cette section, nous exposons l'algorithme du *Tabu Search* appliqué à notre problème.

Algorithme 6.1 Algorithme TS

Initialisation

Construction d'une matrice dans V

$\mathbf{A}' = \mathbf{A}$

$nb\_iter = 0$

$meilleur\_iter = 0$

nb_tabu

initialisation de la liste tabu

k_{max} = nombre maximum d'itérations entre deux améliorations de $\mathbf{A}'$

répéter

$nb\_iter = nb\_iter + 1$

Examiner toute les possibilités dans le voisinage de $\mathbf{A}$ dont les changements ne font pas partie de la liste tabu

Sélectionner la meilleure direction de changement

Aller le plus loin possible dans cette direction

$\mathbf{A}' = \mathbf{A}$

mettre la liste tabu à jour

si $(FO(\mathbf{A}) < FO(\mathbf{A}^+)$ **alors**

$meilleur\_iter = nb\_iter$

$\mathbf{A}^+ = \mathbf{A}$

jusqu'à ce que $(FO(\mathbf{A}) < FO(\mathbf{A}')$ et $(nb\_iter - meilleur\_iter > k_{max}))$

fin

6.12 Exemple numérique

La figure 6.6 montre le spectre mesuré du trafic Ethernet sur le réseau de l'EPFL le 30 août 1993 de 18h52 à 18h54. Le trafic est relativement dense à cette heure. La densité spectrale des mesures est donnée par le trait plein alors que que celle du modèle est donnée en traits-tillés. La fonction de poids pour le spectre est $PS(i) = 100 \times e^{-|i| \times 4.6/256} + 100 \times \delta(i)$ avec $\delta(0)$ étant l'impulsion de Dirac. La moyenne du trafic converti est de 0.0618 et l'estimation du spectre en zéro est 6.165. Pour faire le *fitting* de notre modèle à 5 états, la méthode décrite à la section 6.10 nous permet de trouver la matrice de transition $\mathbf{A}$ et la matrice $\mathbf{\Lambda}$ en admettant qu'on peut avoir au plus une arrivée par état. Après optimisation, les matrices suivantes ont été trouvées

$$\mathbf{A} = \begin{pmatrix} 0.95106 & 0.00009 & 0.04715 & 0.00072 & 0.00098 \\ 0.00092 & 0.99244 & 0.00089 & 0.00534 & 0.00041 \\ 0.54829 & 0.42370 & 0.00876 & 0.00784 & 0.01141 \\ 0.00002 & 0.89857 & 0.03925 & 0.00371 & 0.05845 \\ 0.00000 & 0.00000 & 1.00000 & 0.00000 & 0.00000 \end{pmatrix} \tag{6.39}$$

et

$$\mathbf{\Lambda} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \tag{6.40}$$

Cette chaîne de Markov a une espérance mathématique de 0.083. Pour valider ces résultats, nous avons comparé la moyenne des pertes du trafic mesuré entrant dans une

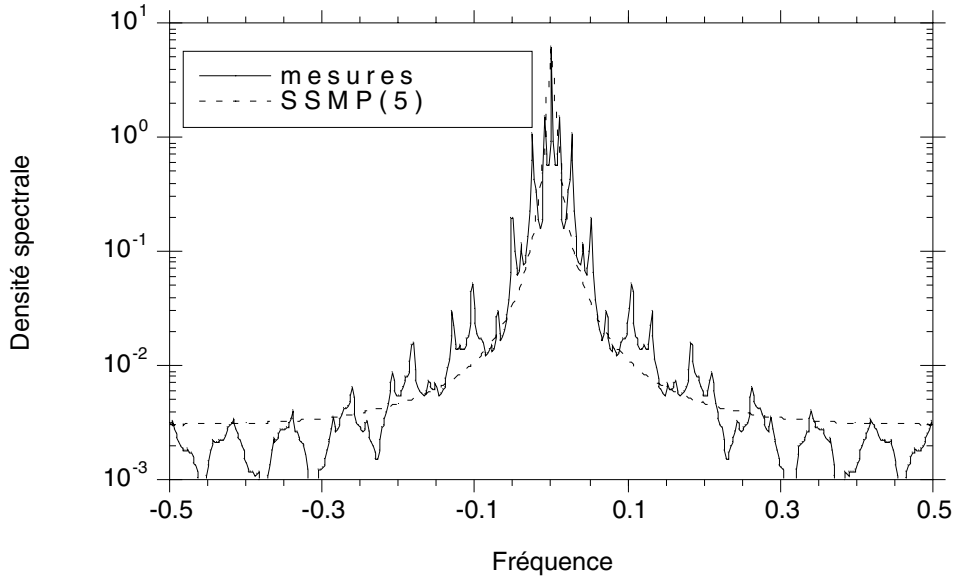

Figure 6.6: Spectre mesuré du trafic Ethernet sur le réseau de l'EPFL et spectre du modèle trouvé par la méthode tabu

file d'attente avec la probabilité de pertes de la file SSMP/G/1/c (c est la taille de la file d'attente). La distribution des temps de service est géométrique. Ainsi, il est possible de varier le temps de service, ou, dans notre cas la probabilité que le temps résiduel de service soit égal à un, $prob(R_t) = 1 = \gamma_1$. La figure 6.7 montre une bonne correspondance entre la moyenne des pertes des mesures dans la file d'attente et la probabilité de perte du modèle dans la même file d'attente. Une comparaison avec le processus de Poisson est montrée lorsqu'on *fitte* uniquement le paramètre de l'espérance mathématique du processus (seul paramètre libre). Les corrélations présentes au sein même du trafic jouent un grand rôle dans la probabilité de perte.

6.13 Réflexion

Les résultats sont encourageants, néanmoins cette façon de faire présente un certain nombre de désavantages. Imaginons que la solution $\mathbf{A}^+$ soit composée d'éléments ayant des valeurs très faibles, alors, l'optimisation prendra beaucoup de temps pour atteindre le minimum global. D'autre part ϵ ne peut pas être aussi petit qu'on veut car nous nous trouvons très vite confronté à des problèmes numériques et le calcul de FO peut alors

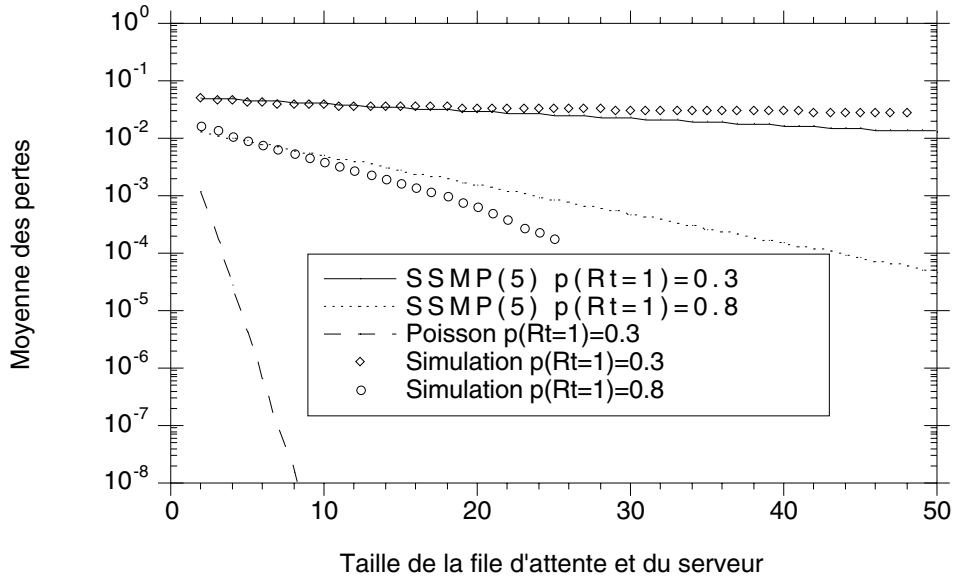

Figure 6.7: Comparaison de la moyenne des pertes du trafic Ethernet mesurée sur le réseau de l'EPFL et de la probabilité de perte du modèle trouvé dans la même file d'attente

indiquer une mauvaise direction, ce qui peut être catastrophique pour l'optimisation. Un autre problème est qu'on ne connaît absolument pas le rôle des paramètres inconnus que nous manipulons. On n'a aucune idée de la signification des éléments de la matrice $\mathbf{A}$. Si par exemple on considère une matrice plus importante (10×10 par exemple), ce ne sont plus 30 paramètres qu'il faut trouver mais 110. Dans le chapitre suivant, nous allons voir comment il est possible d'attaquer le problème différemment, plus intuitivement dans un premier temps. Nous allons essayer de comprendre un peu mieux la signification des probabilités de transition.

Chapitre 7

Premier modèle markovien basé sur la théorie de la *pseudo-décomposabilité*

7.1 Introduction

L'analyse spectrale du chapitre précédent nous apporte de nombreux éclairages quant au comportement de la file d'attente mais elle n'explique pas la forme observée du trafic. Pour répondre à la question "la modélisation markovienne a-t-elle encore un sens dans l'analyse du trafic des réseaux informatiques? ", nous allons essayer de comprendre comment, globalement, les réseaux informatiques fonctionnent. Pour cela, nous allons nous aider de la théorie de la décomposabilité de Courtois [86].

Comme nous l'avons vu dans l'introduction (section 1.1), une bonne compréhension du fonctionnement du système à modéliser est indispensable pour pouvoir espérer obtenir de bons résultats. Or les problèmes liés aux performances de réseaux d'ordinateurs sont difficiles à aborder car la manière dont les ordinateurs communiquent entre eux est complexe. Dans cette introduction, nous allons essayer de comprendre les principes de base du comportement du système que nous voulons étudier. L'analyse du comportement de systèmes d'ordinateurs s'est souvent faite sur un aspect très particulier, en profondeur de telle sorte à ce que les modèles soient bien définis. Ainsi, il est possible d'analyser en détail un aspect très particulier du comportement d'un système d'ordinateurs avec des techniques mathématiques connues. Ce genre d'analyse nous permet de comprendre

comment chaque aspect du système fonctionne mais pas comment il se comporte globalement. Cette manière de faire nous vient d'une coutume adoptée depuis longtemps dans le domaine des sciences où beaucoup de progrès ont pu être effectués grâce à l'isolation artificielle de sous-systèmes. On peut même dire que c'est devenu un réflexe de scientifique que d'essayer d'isoler une partie d'un système lorsque ce dernier est très complexe et dont le comportement est régi par un grand nombre de variables. Néanmoins, lorsque nous avons pour charge de nous occuper du comportement de systèmes complexes dans leur globalité, cette approche est inefficace. En général, un système d'ordinateurs ne possède pas de caractère de "sommabilité" (le comportement du système est régi par la somme de tous les comportements individuels des sous-systèmes), car les ressources (mémoire, temps d'accès au processeur) sont partagées; les systèmes d'exploitation permettent la programmation parallèle et les accès multiples. Les différents processus ont ainsi des dépendances entre eux. Face à cette complexité, l'analyse de petits sous-systèmes isolés ne suffit pas pour nous renseigner quant aux conséquences de certaines options prises lors de la conception d'un réseau informatique. C'est ici que se trouve la réelle difficulté de l'analyse car comme nous l'avons mentionné dans l'introduction (section 1.1), la modélisation mathématique reste limitée à un degré de complexité "raisonnable". Dès qu'on essaie d'analyser froidement le comportement global d'un système complexe, on se heurte à de très grosses difficultés de modélisation d'une part et de résolution d'autre part. Il est toujours possible de faire de la simulation mais le prix à payer est la difficulté de l'interprétation. Le chemin que nous allons prendre pour guider nos recherches est basé sur la théorie de la décomposabilité de Courtois [86].

7.2 La décomposabilité quasi-complète

L'idée de base nous vient des économistes [87] qui se sont penchés sur le problème de l'évolution dynamique de gros systèmes complexes dépendant d'un grand nombre de variables. On va admettre que ces gros systèmes sont constitués d'un petit nombre de groupes de sorte à ce que les groupes individuels puissent être étudiés indépendamment les uns des autres, comme si les autres groupes n'existaient pas, et que l'étude du com-

portement entre les groupes puisse se faire sans qu'il soit nécessaire de savoir ce qui se passe dans chacun des groupes. Cette idée est très générale et peut s'appliquer à de nombreux autres domaines (mécanique statistique, mécanique quantique, sciences sociales). On appelera un **système quasi-complètement décomposable** un système dont les interactions entre les différents groupes sont faibles en comparaison avec les interactions qui existent à l'intérieur de chacun des groupes. Cette définition est dûe à Ando et Fisher [87]. Beaucoup d'exemples, en économie, en physique [87, 88, 89] ont montré que les gros systèmes ont tendance à vérifier cette hypothèse de *pseudo-décomposabilité* plutôt que celle de la *décomposabilité complète* (lorsque les interactions entre groupes sont inexistants). Simon [87] a fait la conjecture suivante: les systèmes complexes sont souvent organisés de façon hiérarchique et ces hiérarchies sont souvent quasi-décomposables car les interactions au même niveau de la hiérarchie sont du même ordre, alors que les interactions entre deux différents niveaux sont beaucoup plus faibles. Regardons maintenant comment considérer ces observations dans le cadre des réseaux informatiques. Dans un premier temps, nous allons nous placer dans le cas du réseau ATM et ensuite analyser le comportement du réseau Ethernet.

7.3 Les réseaux ATM et Ethernet

Il y a longtemps qu'on sait que les applications qui utilisent des transferts de données et d'images génèrent du trafic corrélé dans le réseau ATM. L'impact de cette corrélation sur la performance de l'ATM est importante. Si on regarde d'un peu plus près comment fonctionne l'ATM, nous voyons très vite apparaître plusieurs échelles de temps au niveau ATM (figure 7.1): une communication est délimitée par un établissement et une libération de blocs qui eux-mêmes sont composés de cellules (entité de base de l'ATM). Le modèle de trafic devra prendre en compte le comportement statistique de ces différents niveaux. Si nous considérons le trafic mesuré sur Ethernet comme étant une source, nous devons tenir compte des personnes qui travaillent sur le réseau, en ayant leurs propres horaires. Le comportement humain a une grande influence sur le réseau mais l'inverse est vrai également, la patience de l'être humain n'est pas illimitée.

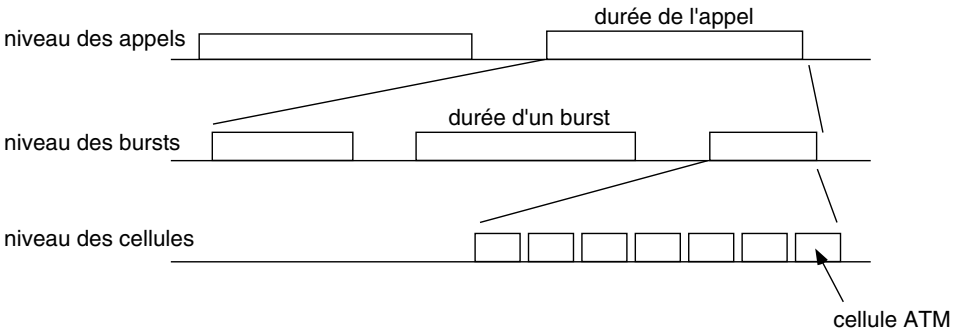

Figure 7.1: Echelles de temps dans l'ATM

L'influence du protocole qui est utilisé est aussi à prendre en considération. Nous allons considérer uniquement le trafic de données dans un premier temps. Le trafic de données est constitué d'une grande gamme de services (transferts de fichiers, communications entre stations de travail, communications entre terminaux, ...). Les données de Bellcore comprennent $99,5\%$ de trafic *Internet Protocol* (IP). Ce dernier peut supporter différents types de protocoles comme par exemple *User Datagram Protocol* (UDP), *Transmission Control Protocol* (TCP), *Internet Control Message Protocol* (ICMP), *Exterior Gateway Protocol* (EGP).

Prenons l'exemple de TCP/IP [90] qui a un comportement très spécifique. Manthorpe [91] a étudié l'influence de la couche transport sur la modélisation de trafic et a trouvé que dans certains cas la couche transport avait une grande influence sur la couche ATM (en admettant que TCP/IP soit supporté par ATM) et donc sur la manière de modéliser le trafic à ce niveau. Le protocole TCP/IP fait l'usage d'une fenêtre glissante pour être sûr que la transmission s'effectue efficacement. Le flux envoyé sur le réseau dépend de la grandeur de la fenêtre mais aussi de la charge sur le réseau et de sa taille. TCP définit la structure des données qui seront envoyées sur le réseau ainsi que les accusés de réception que deux machines s'échangent pour réaliser un transport fiable. Les procédures utilisées par les machines pour garantir la réception correcte des données sont également définies par TCP. Le protocole spécifie également comment les deux machines établissent une connexion et comment elles se mettent d'accord pour la libérer. Il prend également en charge la manière dont les machines qui communiquent entre elles corrigent des erreurs (perte ou duplication de paquets). Un aspect compliqué

et très important de TCP concerne la façon dont sont gérées les temporisations et les retransmissions. Comme d'autres protocoles fiables, TCP va admettre que la destination va lui envoyer un acquittement dès que celle-ci a reçu un certain nombre de données. Dès que la station émettrice envoie un segment de données, elle arme une temporisation et attend un acquittement en retour. Si la temporisation expire avant que les données du segment ne soient acquittées, TCP de la station émettrice va supposer que les données du segment sont perdues ou ont été détruites, donc elle va retransmettre les données. La complexité vient du fait que TCP est en général utilisé dans une interconnexion de réseaux. Un segment de données peut traverser plusieurs unités d'interconnexion avant d'arriver à sa destination ou n'en traverser aucune (dans le cadre d'un réseau local par exemple). Dans le premier cas, le retard des segments de données sera plus important que dans le second si on considère des réseaux de même débit et en ayant le même éloignement physique entre les deux stations qui s'échangent les données. Il est ainsi impossible de connaître à priori la vitesse à laquelle les acquittements seront reçus par la source. De plus, le retard (*delay*) introduit par chaque unité d'interconnexion dépendra de la charge que cette dernière supporte. Le retard de transmission de l'acquittement peut beaucoup varier d'un moment à l'autre. Pour cela TCP va gérer les performances pour chaque connexion et déduire des valeurs de temporisations raisonnables et s'adapter aux changements du réseau. Lorsqu'un encombrement du réseau survient, TCP va réduire son taux de transmission à l'aide de deux techniques: le démarrage lent (*slow start*) et la diminution dichotomique (*multiplicative decrease*). Il adapte la grandeur de la fenêtre. De son côté, le réseau Ethernet, lorsqu'il a quelque chose à transmettre sur le canal attend que ce dernier soit libre. En plus, il est capable de détecter une collision sur le réseau. Si deux stations sont à l'écoute du canal et le savent vide, elles vont commencer à transmettre simultanément mais elles vont vite détecter une collision. Alors les deux stations vont arrêter brusquement de transmettre. Ainsi notre modèle pour Ethernet consistera en une alternance de périodes de transmission et de collisions. Pour résumer, si le trafic est un trafic de données utilisant TCP/IP nous pouvons admettre qu'il peut être analysé à trois niveaux différents d'activité.

- Le niveau de la connexion décrit le comportement humain. La durée de la connexion

est déterminée par le temps qu'il faut pour transmettre le fichier. Le temps entre deux appels sur le réseau Ethernet est typiquement compris entre 10 et 1000 sec.

- Le niveau transport est décrit par TCP/IP. Comme nous l'avons vu précédemment, le trafic envoyé sur le réseau dépend d'un nombre incalculable de paramètres mais l'influence la plus grande vient du comportement du réseau. Le temps de transmission d'un paquet TCP/IP varie typiquement de 0.01 sec. à 10 sec.

- Au niveau du réseau Ethernet, le temps de transmission d'un paquet dépend essentiellement du trafic local qui circule sur le réseau. Le temps typique de transmission, alors que le paquet est en attente, est de 1 à 50 msec.

Pour construire un premier modèle basé sur ces considérations, on va admettre que l'être humain est caractérisé par deux modes distincts d'opération: soit il essaie de transmettre un fichier, soit il est occupé à d'autres tâches. Le changement entre ces deux modes d'opération dépend uniquement du comportement humain, comment il réagit lorsque le réseau est encombré par exemple. Au niveau TCP/IP, le protocole est principalement contrôlé par le comportement du réseau. Ici également, nous admettrons que le protocole se trouve dans deux modes opératoires distincts selon qu'il envoie ou non des paquets de données sur le réseau. L'analyse des passages entre les deux modes opératoires est difficile à cause des dépendances entre le réseau et le protocole. Au niveau le plus bas, Ethernet attend que le canal soit libre pour pouvoir transmettre ses données.

7.4 Construction d'un premier modèle

A partir des considérations de la section précédente, nous avons défini un premier modèle qui respecte les différentes échelles de temps. La chaîne de Markov est modulée et en temps discret. Lorsque le modulateur de la chaîne de Markov se trouve dans l'état 0 ou dans l'état 1, on a l'arrivée d'une cellule sinon on a l'arrivée d'aucune cellule, en d'autres termes $prob(X_t = 1|Y_t = 1) = prob(X_t = 1|Y_t = 2) = 1$ alors que $prob(X_t = 0|Y_t = 3) = prob(X_t = 0|Y_t = 4) = prob(X_t = 0|Y_t = 5) = 1$. Les probabilités de transition entre les différents états, dont les valeurs proviennent de la discussion de la

section précédente, sont données par la matrice $\mathbf{A}$:

$$\mathbf{A} = \begin{pmatrix} 0.99 & 0.00001 & 0.0099889 & 0.000001 & 0.0000001 \\ 0.00001 & 0.5999889 & 0.4 & 0.00001 & 0.0000001 \\ 0.001 & 0.001 & 0.997999 & 0.000001 & 0 \\ 0.000001 & 0.000001 & 0.000001 & 0.999997 & 0 \\ 0.0000001 & 0.0000001 & 0.0000001 & 0 & 0.9999997 \end{pmatrix} \tag{7.1}$$

A partir de ce qui a été dit plus haut, la matrice $\mathbf{\Lambda}$ a la forme suivante:

$$\mathbf{\Lambda} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \tag{7.2}$$

A titre de comparaison, nous allons considérer deux autres modèles bien connus, le modèle poissonien et la chaîne de Markov à deux états (ON-OFF). Le modèle de Poisson est caractérisé par les matrices suivantes:

$$\mathbf{A} = \begin{pmatrix} 0.99 & 0.01 \\ 0.01 & 0.99 \end{pmatrix} \tag{7.3}$$

$$\mathbf{\Lambda} = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} \tag{7.4}$$

La chaîne de Markov à deux états, est caractérisée par les matrices suivantes:

$$\mathbf{A} = \begin{pmatrix} 0.999999 & 0.000001 \\ 0.000099 & 0.999901 \end{pmatrix} \tag{7.5}$$

$$\mathbf{\Lambda} = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} \tag{7.6}$$

pour les figures 7.2, 7.3, 7.4 et par les matrices suivantes:

$$\mathbf{A} = \begin{pmatrix} 0.999 & 0.001 \\ 0.099 & 0.0901 \end{pmatrix} \tag{7.7}$$

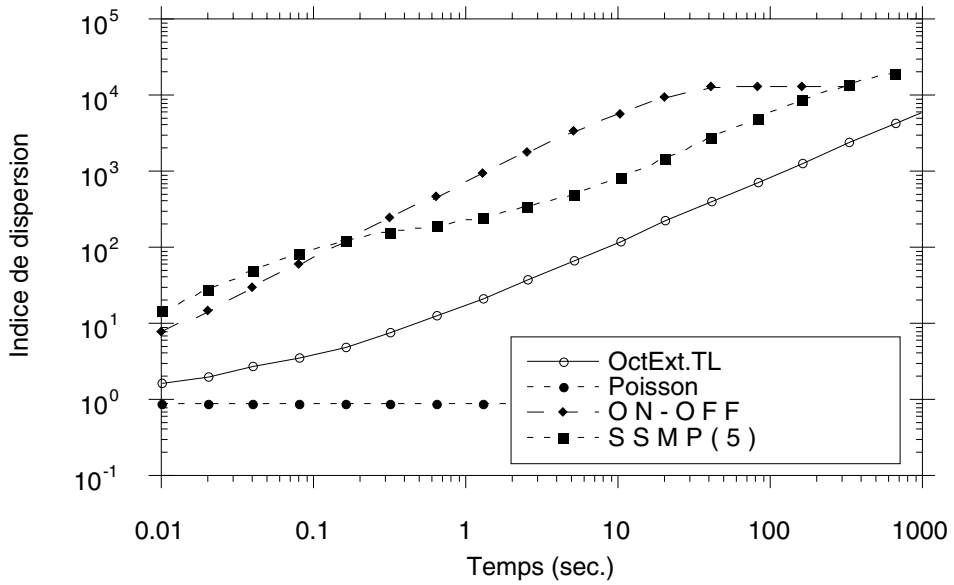

Figure 7.2: Indices de dispersion pour les mesures et les différents modèles

$$\mathbf{\Lambda} = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} \tag{7.8}$$

pour la figure 7.7

7.5 Comparaison des différents modèles

Dans cette section, nous allons comparer la granularité (*burstiness*) des différents modèles sur plusieurs échelles de temps. Les mesures conventionnelles de la granularité sont [15, 48] le coefficient de variation et l'indice de dispersion. Dans [14], ces mesures de la granularité sont remplacées par l'évaluation du paramètre de Hurst et le test visuel qui est beaucoup plus intuitif. La figure 7.2 montre l'évolution de l'indice de dispersion des arrivées des trois modèles en fonction de la longueur de la fenêtre m. Alors que les indices de dispersion des chaînes de Markov à cinq états et à deux états augmentent sur plusieurs échelles de temps, l'indice de dispersion du processus de Poisson reste constant. Rappelons que l'indice de dispersion (section 4.9.2) des arrivées d'un processus pour une longueur de fenêtre m est donné par le rapport de la variance du nombre d'arrivées pendant une période de temps égale à m divisé par la moyenne des arrivées pendant

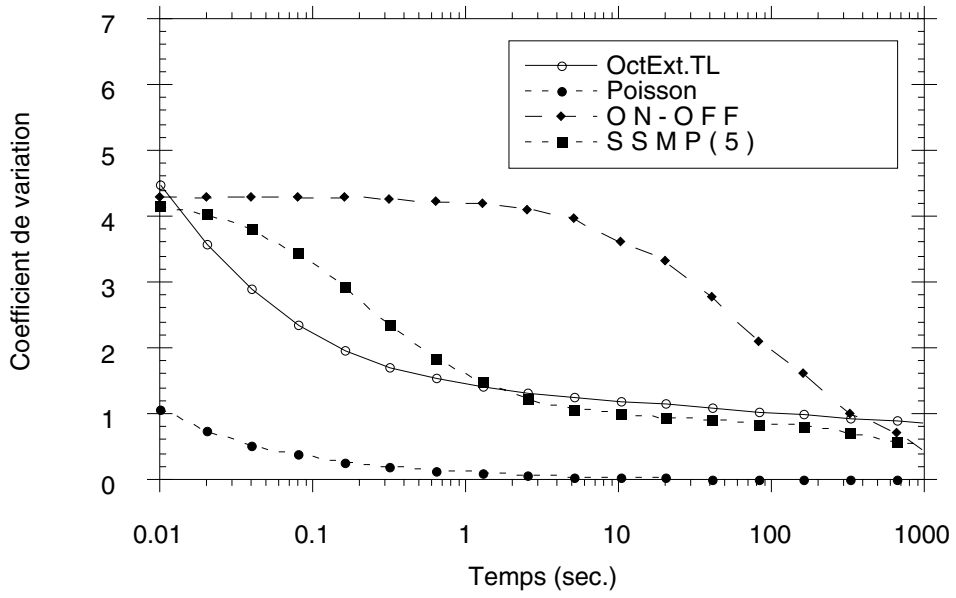

Figure 7.3: Coefficients de variation pour les mesures et les différents modèles

l'intervalle m. L'observation que nous pouvons faire est que la variance $Var(mX^{(m)})$ augmente plus rapidement que l'espérance mathématique multipliée par la taille de la fenêtre $\overline{mX^{(m)}}$. Notons que les deux processus sont limités par leur échelle de temps. A partir d'une certaine échelle de temps, l'indice de dispersion qui augmentait régulièrement va se stabiliser. Tous les processus markoviens sont limités par leur échelle de temps propre. Néanmoins, il est intéressant de constater que cette limite peut être éloignée autant que l'on veut.

On va comparer maintenant le coefficient de variation des trois processus. Le coefficient de variation des arrivées d'un processus pour une longueur de fenêtre m (section 3.10) est donné comme étant le rapport de l'écart-type du nombre d'arrivées pendant une période de temps égale à m divisé par la moyenne des arrivées pendant l'intervalle m. Nous avons observé le coefficient de variation des trois processus sur plusieurs échelles de temps également et il est intéressant de constater que le comportement de la chaîne de Markov à deux états n'est pas le même que celui de la chaîne à cinq états. En effet, si la valeur du coefficient de variation pour $m = 1$ est à peu près la même, l'évolution des deux courbes n'est pas identique. Le comportement de la chaîne à cinq états ressemble à celui des mesures (OctExt.TL) sauf pour les petites valeurs de m. La figure 7.4

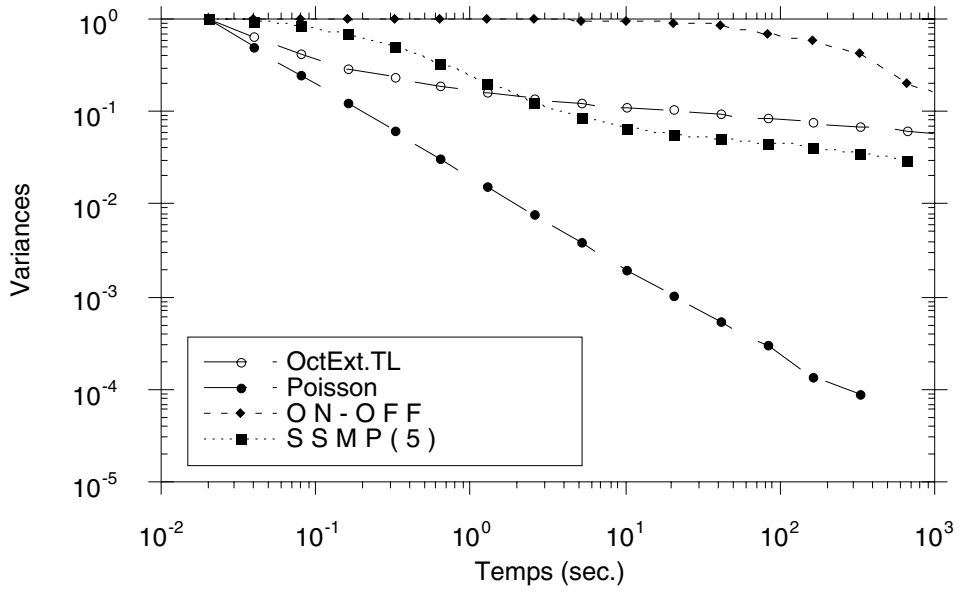

Figure 7.4: Paramètres de Hurst

montre le graphe "temps-variances". A l'aide de ce graphe, il est possible de déterminer le paramètre de Hurst local et par conséquent faire la distinction entre les dépendances à long terme et à court terme. Pour les processus dont les dépendances sont à court terme, le paramètre de Hurst $H = 0.5$, autrement dit, la pente de la courbe $-\beta = -1$ car $\beta = 1 - H/2$. Pour les processus dont les dépendances sont à long terme, $0 < \beta < 1$. Comme nous l'avons décrit à la section 4.8 , sur ce graphe nous traçons la variance normalisée de la série aggrégée en fonction de la grandeur de la fenêtre en échelle $log - log$. Nous remarquons ici également un phénomène intéressant. La figure 7.4 nous montre que β varie de 0.55 (de 0.05 à 1 sec.) à 0.17 (1 sec. à 1000 sec.), ce qui nous donne un paramètre de Hurst local $= 0.72 - 0.91$ pour les données mesurées sur le réseau de Bellcore. Le processus de Poisson a une pente de -1 et un paramètre de Hurst $H = 0.5$, ce qui est très caractéristique des processus dont les dépendances sont à court terme. Pour la source de Markov modulée à deux états, la pente de la courbe varie de 0 à -1 alors que pour la chaîne à cinq états, la courbe suit mieux la courbe des données mesurées. Dans ce cas, la pente de la courbe varie entre -0.6 (de 0.1 à 5 sec.) et -0.17 (de 5 à 1000 sec.). Le paramètre de Hurst local vaut $H_l = 0.7$ à 0.91. De nouveau, on remarque que la chaîne de Markov à deux états a un comportement différent de celui de la chaîne de

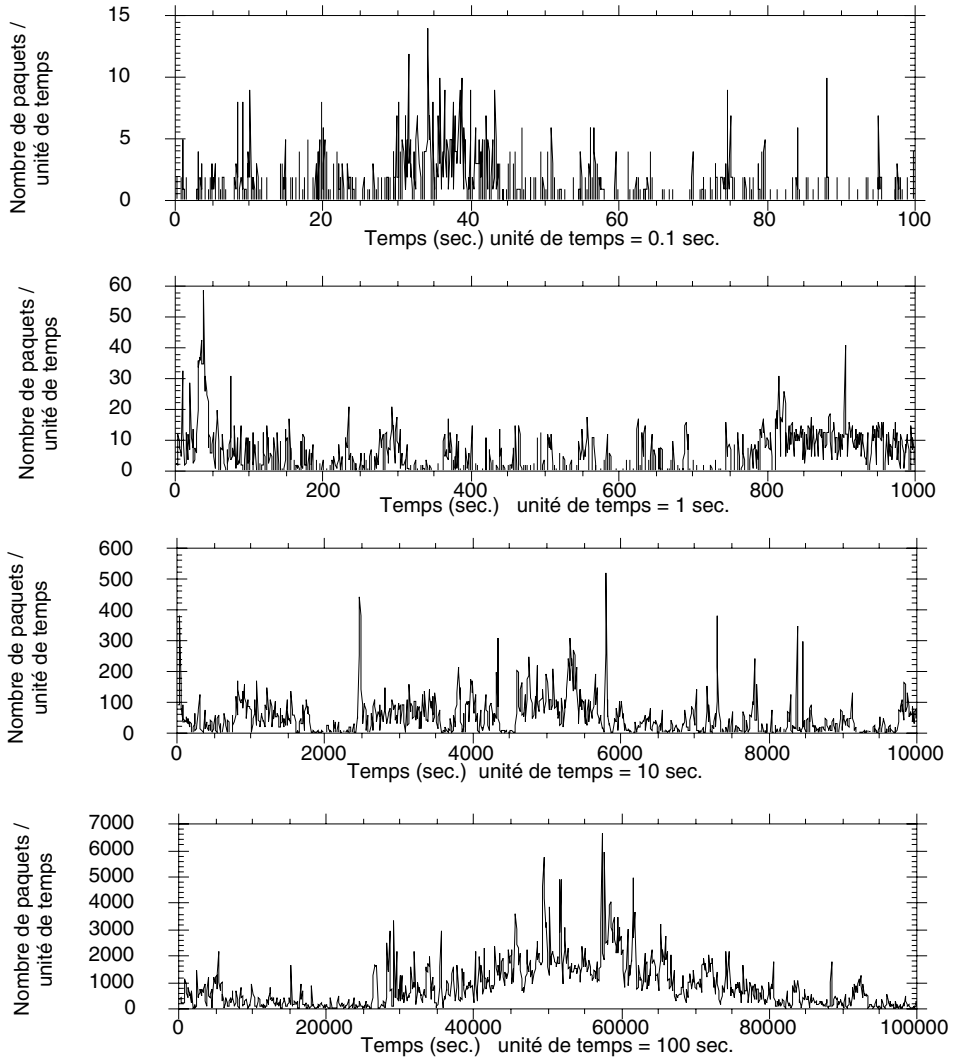

Figure 7.5: Mesures de Bellcore OctExt.TL

Markov à cinq états. Les figures 7.5, 7.6, 7.7 et 7.8 décrivent une séquence du nombre de paquets arrivant par unité de temps en fonction du temps. La figure 7.5 montre 27 heures consécutives de trafic Ethernet mesuré à Bellcore (OctExt.TL). La figure 7.6 montre du trafic poissonien, simulé. L'observation faite par l'équipe de Bellcore [15, 14] est que, contrairement au trafic Ethernet, le trafic poissonien n'a pas la même allure sur toutes les échelles de temps. Le trafic Ethernet est très variable, et ceci sur un grand nombre d'échelles de temps. Ça signifie que les périodes chargées sont constituées elles-même de périodes chargées et de périodes moins chargées, ceci à plusieurs échelles de temps. On notera tout de même que le cycle journalier est visible sur la figure 7.5. Les figures 7.7

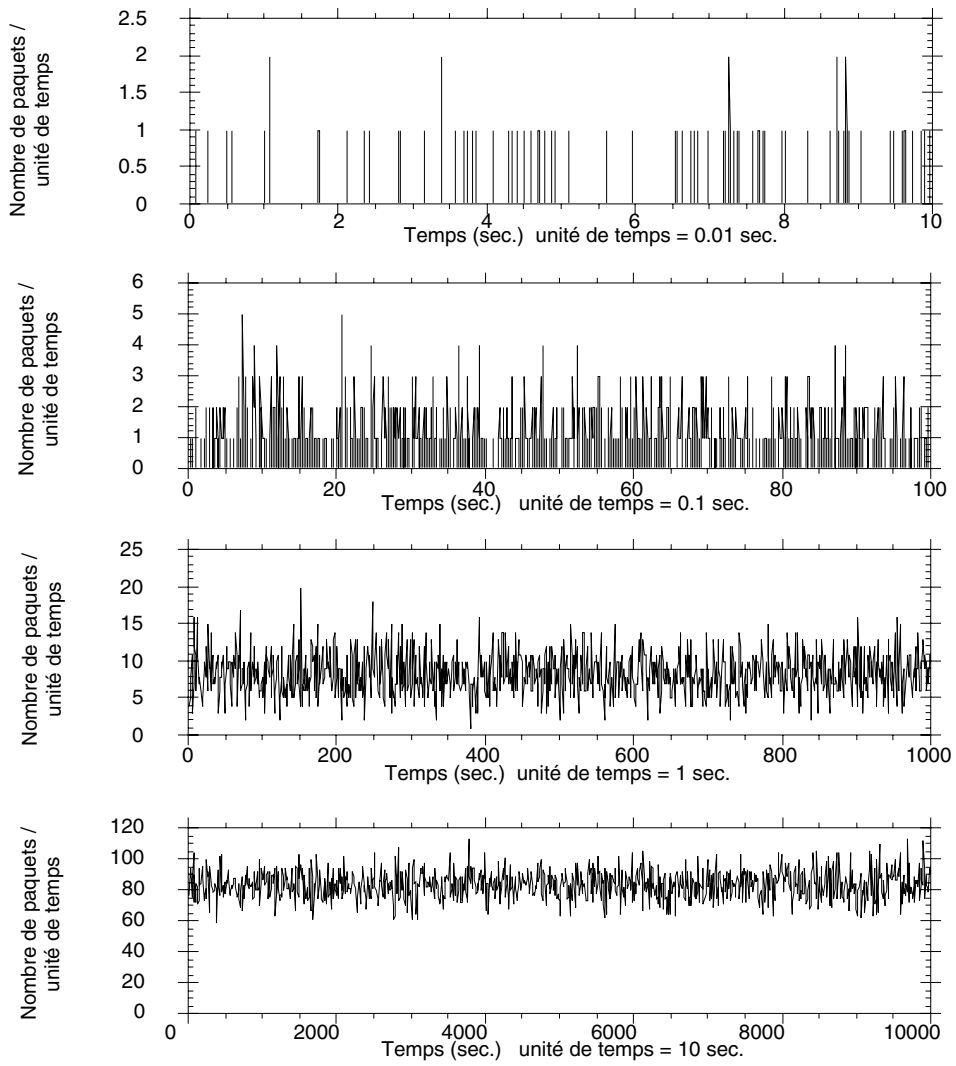

Figure 7.6: Trafic poissonien simulé

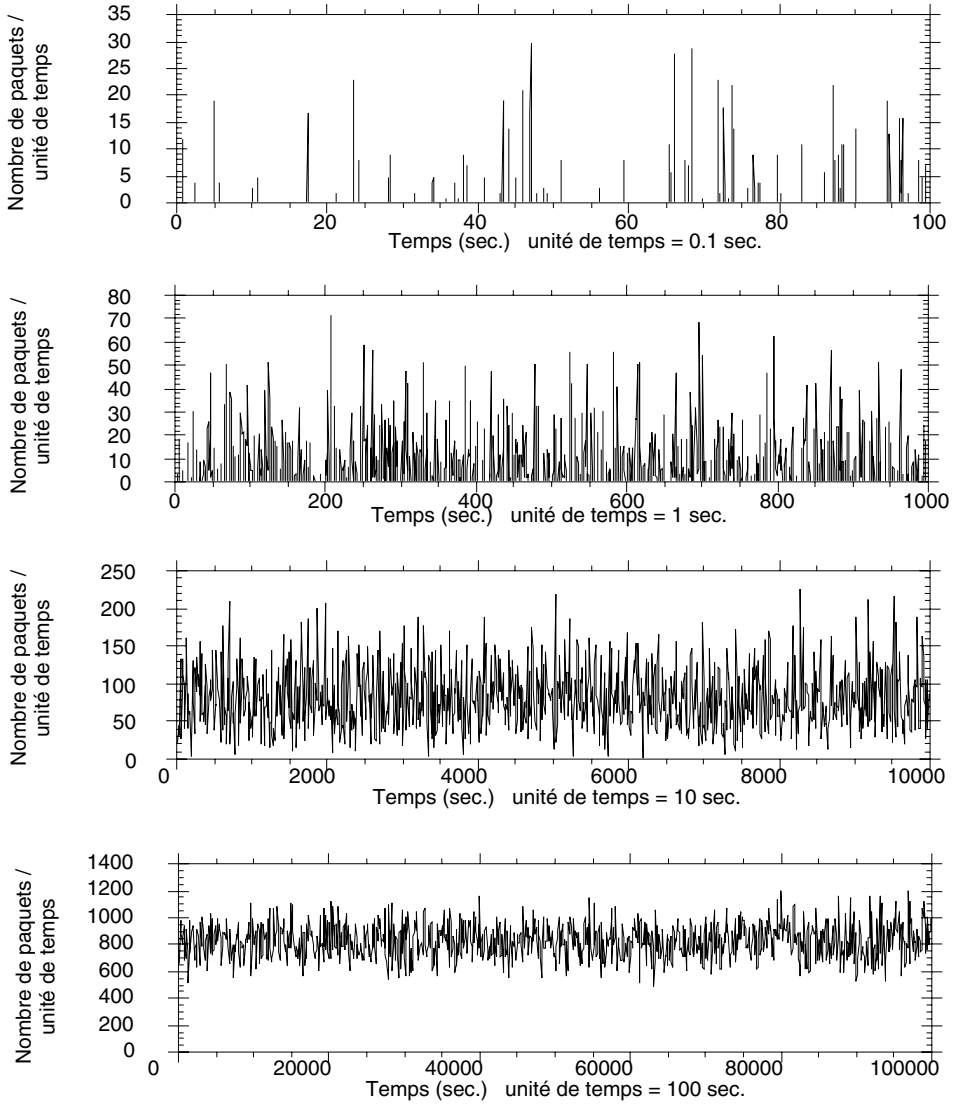

Figure 7.7: Trafic simulé par une source ON-OFF

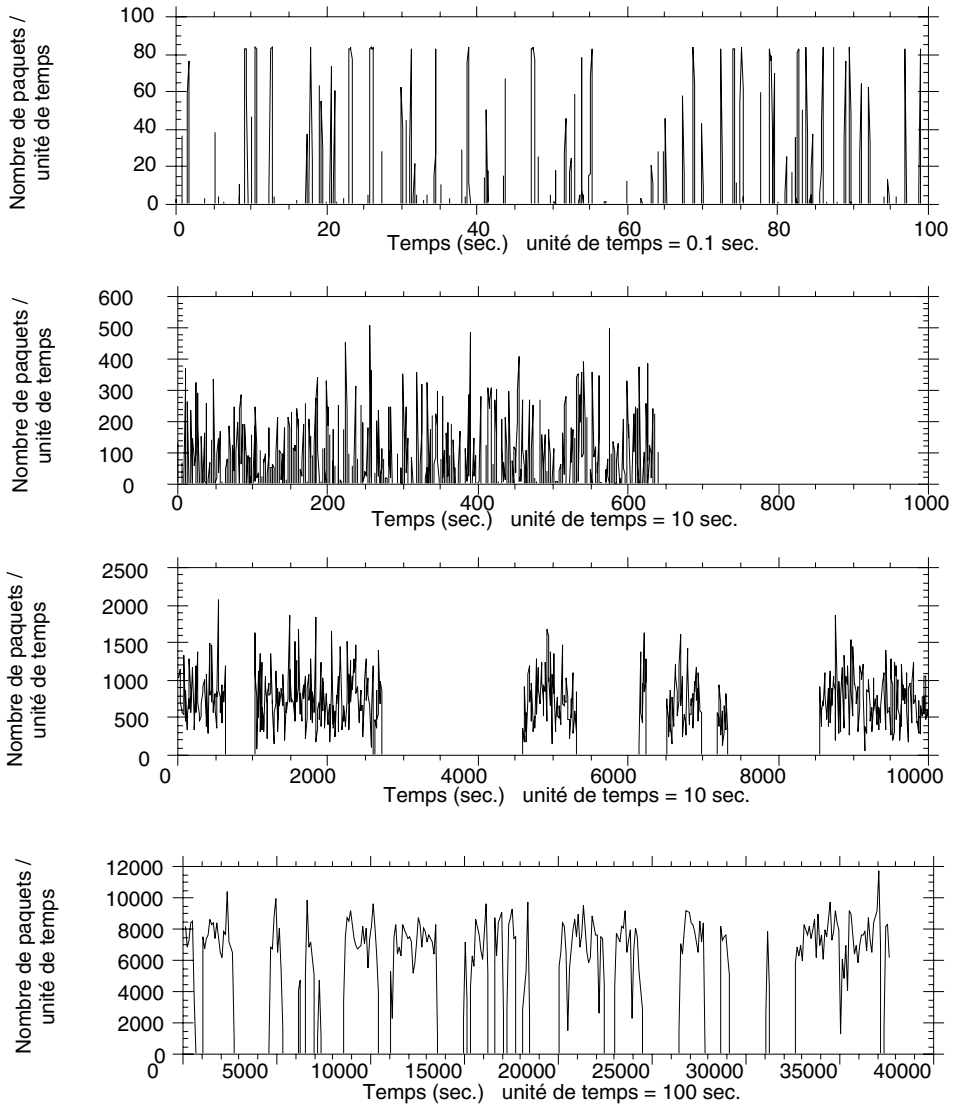

Figure 7.8: Trafic simulé par une chaîne de Markov modulée à 5 états, SSMP(5)

et 7.8 montrent le comportement des deux autres processus (chaînes de Markov à deux et à cinq états). Nous remarquons que le processus de la chaîne de Markov à deux états (figure 7.7) s'uniformise lorsque la fenêtre s'agrandit. Même si le trafic est très grumeleux (*bursty*) pour de petites échelles de temps, il n'est pas suffisant pour reproduire des caractéristiques réelles de trafic mesuré à cause de l'absence de corrélations à des échelles de temps plus importantes. Si nous regardons maintenant (figure 7.8) le comportement du processus de la chaîne de Markov à 5 états, nous remarquons que même si le trafic simulé apparaît comme étant un peu synthétique, il possède des corrélations à plusieurs échelles de temps, il n'y a plus de "longueur naturelle" de rafale (*burst*). Ce modèle, comme tous les modèles markoviens ayant un nombre d'états fini, est limité par sa propre échelle de temps.

Chapitre 8

Modèle markovien modulé ayant peu de paramètres

8.1 Introduction du concept de pseudo-dépendance à long terme

Dans cette section, nous allons introduire le concept de pseudo-dépendance à long terme. Nous avons vu dans le chapitre précédent que certains processus markoviens étaient capables de produire du trafic auto-similaire sur une certaine période de temps, finie. Le comportement de tels processus est tellement différent des processus conventionnels qu'il nous semble raisonnable de les classer différemment. D'après Cox [58], la différence entre les dépendances à long terme et les dépendances à court terme est exprimée par les définitions suivantes:

Définition 8.1 *Un processus à dépendance à court terme est caractérisé par*

- $\sum_{\tau=0}^{\infty} C(X_t, X_{t+\tau})$ *est convergent*
- *le spectre du processus en zéro est fini,* $\Phi(0)$ *est fini*
- *la variance du processus agrégé* $var(X_k^{(m)})$ *est de la forme* $var(X_k^1)/m \sim 1/m$ *pour de grandes valeurs de* m

Définition 8.2 *(Rappel) Un processus à dépendance à long terme est caractérisé par*

- $\sum_{\tau=-\infty}^{\infty} C(X_t, X_{t+\tau})$ *est divergent*
- *le spectre du processus en zéro est infini,* $\Phi(0) = \infty$
- *la variance du processus aggrégé* $var(X_k^{(m)})$ *est de la forme* $\sigma^2 m^{-\beta}$ *pour de grandes valeurs de* m

Il y a encore une autre catégorie de processus, les processus ayant des dépendances à long terme d'indice β. Ils n'ont pas de structure de corrélation dégénérée lorsque $m \to \infty$. Donc tous les processus ARMA d'ordre fini, toutes les chaînes de Markov (en incluant les processus semi-markoviens) finies satisfont à la première définition tandis que les mouvements Brownien fractionnaires [92], les processus ARIMA [14], *chaotic maps* [20] ont des dépendances à long terme et satisfont à la deuxième définition. D'après les définitions ci-dessus, un processus qui serait capable de reproduire du trafic auto-similaire sur plusieurs échelles de temps (tel que le trafic Ethernet mesuré par les chercheurs de Bellcore) mais sur un horizon fini seulement sera classé comme étant un processus ayant des dépendances à court terme. Ainsi, en accord avec les définitions, un processus ayant des dépendances à court terme serait suffisant pour modéliser le trafic LAN. La différence avec les autres processus de la même catégorie est frappante, c'est pourquoi nous proposons de les appeler: **processus ayant des dépendances à pseudo-long terme** (*pseudo long-range dependent processes*). Nous proposons également de parler de paramètre de Hurst local plutôt que de paramètre de Hurst tout court pour nos chaînes de Markov car le paramètre de Hurst est défini pour $m \to \infty$. Il faut bien avouer d'autre part que le temps de mesure reste fini, même s'il est très long, surtout dans le contexte des réseaux informatiques où le trafic augmente de jour en jour. Un modèle ayant des dépendances à pseudo-long terme est capable de modéliser (autant bien qu'un processus ayant des dépendances à long terme) du trafic aggrégé sur plusieurs échelles de temps. En fait, la définition des processus ayant des dépendances à long terme vient historiquement de l'importance des processus auto-similaires qui sont capables de donner une explication élégante à une loi empirique (l'effet de Hurst). En pratique, nous avons toujours un nombre fini de données. Nous proposons de modéliser le trafic LAN à l'aide

de chaînes de Markov. Comme nous l'avons vu au chapitre 3, la fonction d'autocovariance de la chaîne de Markov modulée que nous considérons est une somme d'exponentielles. Or la fonction d'autocovariance d'un processus réellement auto-similaire décroit selon une hyperbole. Il s'agira donc d'approcher cette fonction par une somme d'exponentielles. Nous pourrions donc partir de là pour résoudre notre problème mais il faudrait résoudre un problème très difficile de la résolution inverse des valeurs propres [45], qui n'est d'ailleurs toujours pas résolu théoriquement. Dans ce chapitre, nous allons prendre un autre chemin qui nous permettra d'arriver à nos fins. Le problème peut paraître très compliqué au premier abord car le nombre de paramètres inconnus dans la chaîne de Markov augmente selon $n^2 + n$. Les chercheurs de Bellcore ont d'entrée de jeu condamné cette manière de faire parce qu'elle semblait trop complexe [14]. C'est certainement à cause de ces difficultés que les processus markoviens n'ont pas été considérés jusqu'à présent. En cherchant une structure dans la chaîne de Markov, nous verrons comment il est possible de reproduire du trafic auto-similaire sur un horizon fini à l'aide d'une chaîne de Markov modulée.

8.2 Matrices du modèle

Sur la base des expériences du chapitre 7, nous proposons une nouvelle famille de modèles markoviens dont la matrice de transition vaut

$$\mathbf{A} = \begin{pmatrix} 1 - 1/a - 1/a^2 - ... - 1/a^{n-1} & 1/a & 1/a^2 & \cdots & 1/a^{n-1} \\ b/a & 1 - b/a & 0 & \cdots & 0 \\ (b/a)^2 & 0 & 1 - (b/a)^2 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ (b/a)^{n-1} & 0 & \cdots & 0 & 1 - (b/a)^{n-1} \end{pmatrix} \quad (8.1)$$

et $\phi_{11} = prob(X_t = 1|Y_t = 1) = 1$, $\phi_{i1} = prob(X_t = 1|Y_t = i) = 0 \ \forall i = 2, 3, ..., n$. Avec ce modèle $X_t \in \{0, 1\}$, ce qui nous permet de déduire $\mathbf{\Lambda}$

$$\mathbf{\Lambda} = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & \cdots & 0 \end{pmatrix} \tag{8.2}$$

8.3 Caractéristiques du modèle

8.3.1 Espérance mathématique et moments

La chaîne de Markov que nous nous proposons d'étudier est finie, irréductible et ayant tous ses états récurrents. Il existe donc une unique distribution stationnaire et la solution au système $\vec{\pi} = \vec{\pi}\mathbf{A}$ existe. En résolvant ce système pour la chaîne de Markov proposée, nous voyons facilement que $\pi_i = \pi_1(1/b)^{i-1}$ avec $i = 2, \cdots, n$ mais d'autre part $\sum_{i=1}^{n} \pi_i = 1$ donc finalement nous trouvons que

$$\pi_1 = \frac{1 - 1/b}{1 - (1/b)^n} \tag{8.3}$$

et $E[X] = \pi_1$ avec la chaîne proposée, à cause de la forme particulière de $\mathbf{\Lambda}$. Avec cette chaîne, tous les moments sont égaux, $E[X] = E[X^2] = E[X^3] = \cdots$. La variance vaut $E[X^2] - E^2[X]$. La figure 8.1 nous montre l'évolution de $E[X]$ en fonction de b pour différentes valeurs de n. Nous remarquons que pour $n = 1000$, les valeurs de $E[X]$ sont extrêmement faibles pour $b < 1$. A la limite, lorsque $n \to \infty$, $b \geq 1 \ \forall E[X]$. Par contre, pour un nombre d'états plus faible, b peut être compris entre 0 et 1 sans que $E[X]$ soit ridiculement faible. Il se pose alors un problème puisque $\pi_1 = E[X]$ n'est pas déterminé pour $b = 1$. En appliquant le théorème de l'Hospital, nous trouvons que pour $n \geq 2$, $\pi_1 = E[X] = 1/n$ pour $b = 1$.

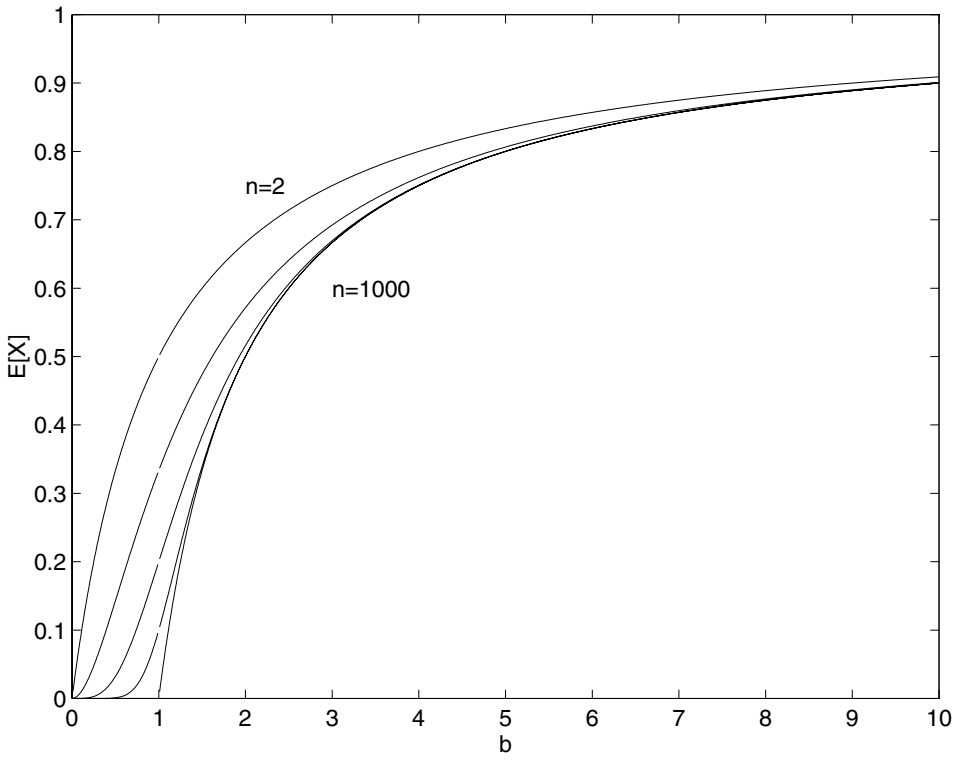

Figure 8.1: $E[X]$ en fonction de b pour $n = 2/3/5/10/1000$

8.3.2 Interarrivées

La distribution des interarrivées T_A est la suivante, pour $k = 1$

$$prob(T_A = 1|Y_t = 1) = 1 - \sum_{i=1}^{n-1}(1/a)^i$$

pour $k \geq 2$

$$prob(T_A = k|Y_t = 1) = \sum_{i=1}^{n-1}(b/a^2)^i(1 - (b/a)^i)^{k-2}$$

cette distribution est une somme de distributions géométriques (qui a son équivalent en temps continu, la **distribution hyperexponentielle** qui est connue pour sa grande variabilité. L'espérance mathématique des interarrivées est donnée (voir annexe, section 11.6) par

$$E[T_A|Y_t = 1] = \frac{1 - (1/b)^n}{1 - 1/b} \tag{8.4}$$

Nous remarquons que si $n \to \infty$, $b > 1$. Le second moment des interarrivées est donné par

$$E[T_A^2|Y_t = 1] = 1 + \frac{1 - (1/b)^n}{1 - 1/b} + 2\frac{1 - (a/b^2)^n}{1 - a/b^2} \tag{8.5}$$

Valeurs propres
1
0.999999999543
0.999999994859
0.999999942030
0.999999343287
0.999992506549
0.999913467916
0.998978686492
0.987354551402
0.803749298055

Tableau 8.1: Valeurs propres exactes

Nous remarquons que si $b > 1$ et $a/b^2 < 1$ alors

$$E[T_A^2|Y_t = 1] = 1 + \frac{1}{1 - 1/b} + 2\frac{1}{1 - a/b^2} \tag{8.6}$$

pour $n \to \infty$ alors que si $b < 1$ ou $a/b^2 > 1$, $E[T_A^2|Y_t = 1] \to \infty$ si $n \to \infty$. La distribution de la longueur des rafales T_R est géométrique.

8.3.3 Valeurs propres

Une technique très simple et très intuitive va nous permettre d'obtenir une bonne approximation des valeurs propres. Elle consiste à chercher les valeurs propres de plusieurs chaînes de Markov à deux états ayant les mêmes probabilités de transition que la chaîne de Markov suggérée entre le premier état et l'état i. Ainsi, les valeurs propres sont 1 et $(1 - (1/a)^{i-1} - (b/a)^{i-1})$ pour $i = 2, \cdots, n$. L'ensemble des valeurs propres a été calculé pour une matrice 10×10 avec a=10 et b=0.9, elles sont données au tableau 8.1 alors que les valeurs propres approximées sont reportées au tableau 8.2.

8.3.4 Autocovariance

Avec l'approximation faite au paragraphe précédent, on peut écrire l'équation (3.22)

$$C(\tau) = \sum_{j=2}^{n} \lambda_j^{\tau} \vec{\pi} \mathbf{\Lambda} \mathbf{G}^{-1} \mathbf{P}_j \mathbf{G} \mathbf{\Lambda} \vec{e} \tag{8.7}$$

Valeurs propres
1
0.999999998612
0.999999985695
0.999999852170
0.999998468550
0.999984095100
0.999834389999
0.998271000000
0.981899999999
0.810000000000

Tableau 8.2: Valeurs propres approximées

de la façon suivante

$$C(\tau) = \sum_{j=2}^{n}(1-(1/a)^j-(b/a)^j)^\tau \vec{\pi}\mathbf{\Lambda}\mathbf{G}^{-1}\mathbf{P}_j\mathbf{G}\mathbf{\Lambda}\vec{e} \tag{8.8}$$

8.3.5 Transformée de Fourier

De même, on peut rappeler la formule que nous avions trouvée pour la transformée de Fourier

$$\Phi(\omega) = C(0) + \sum_{j=2}^{n} 2\beta_j(\frac{1-|\lambda_j|e^{-i(\vartheta_j)}cos(\omega)}{1+|\lambda_j|^2e^{-2i(\vartheta_j)}-2|\lambda_j|e^{-i(\vartheta_j)}cos(\omega)}-1) \tag{8.9}$$

$$\Phi(\omega) = C(0)+\sum_{j=2}^{n} 2\beta_j(\frac{1-(1-(1/a)^j-(b/a)^j)e^{-i(\vartheta_j)}cos(\omega)}{1+(1-(1/a)^j-(b/a)^j)^2e^{-2i(\vartheta_j)}-2(1-(1/a)^j-(b/a)^j)e^{-i(\vartheta_j)}cos(\omega)}-1) \tag{8.10}$$

si maintenant nous posons $\omega = 0$, de plus toutes les valeurs propres de cette approximation sont réelles, donc

$$\Phi(0) = C(0) + \sum_{j=2}^{n} 2\beta_j(\frac{1-(1-(1/a)^j-(b/a)^j)}{1+(1-(1/a)^j-(b/a)^j)^2-2(1-(1/a)^j-(b/a)^j)}-1) \tag{8.11}$$

$$\Phi(0) = C(0) + \sum_{j=2}^{n} 2\beta_j(\frac{1-(1-(1/a)^j-(b/a)^j)}{(1-(1-(1/a)^j-(b/a)^j))^2}-1) \tag{8.12}$$

$$\Phi(0) = C(0) + \sum_{j=2}^{n} 2\beta_j(\frac{1}{(1/a)^j+(b/a)^j}-1) \tag{8.13}$$

or il est intéressant de voir que si $1/a < b/a$, $\sum_{i=0}^{n}((a/b)^i-1)$ diverge lorsque $n \to \infty$ car $b/a < 1$ mais que si $1/a > b/a$, $\sum_{i=0}^{n}(a^i-1)$ diverge aussi lorsque $n \to \infty$ car $1/a < 1$.

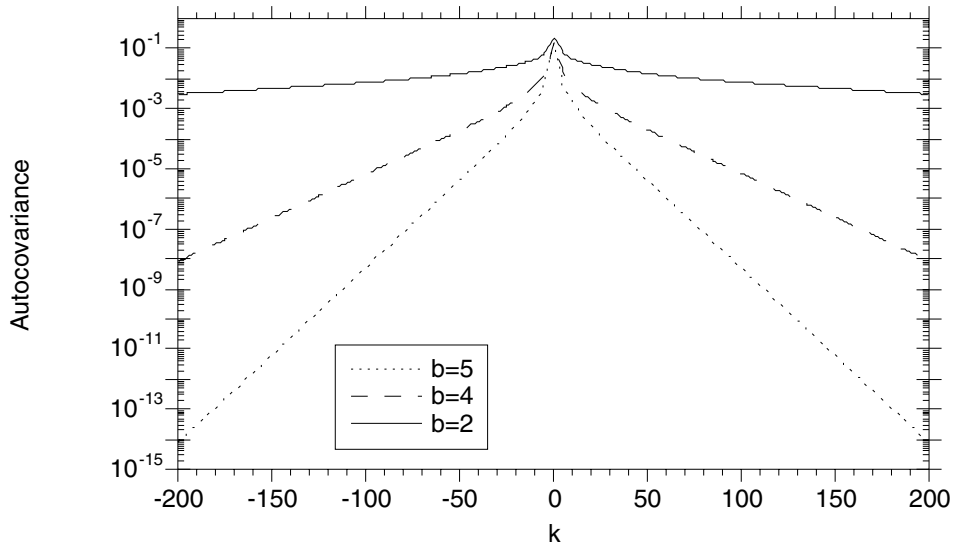

Figure 8.2: Autocovariance, a=10

8.3.6 Exemples

Pour nous familiariser avec cette chaîne de Markov, nous proposons de calculer son autocovariance (figure 8.2) pour différentes valeurs de b en fixant a (a=10). Lorsque b augmente, la fonction d'autocovariance diminue plus rapidement si a reste constant. La densité spectrale de la même chaîne de Markov est montrée à la figure 8.3. On remarque que plus b diminue, plus le pic à l'origine (=$\Phi(0)$) devient important. Pour $a = 10$, $b = 5$ et $n = 4$, les valeurs propres de la matrice de transition valent {1,0.39846,0.742876,0.874258} alors que si $b = 4$, elles valent {1,0.496736, 0.832953, 0.935287} et pour $b = 2$, elles valent {1, 0.675667, 0.953834, 0.991453}.

Les figures 8.4, 8.5 et 8.6 montrent le diagramme "temps-variances" pour la chaîne de Markov décrite à la section 8.2 en variant les paramètres a, b et n. En regardant ces figures, nous remarquons que le paramètre de Hurst local varie avec le nombre d'états de la chaîne de Markov mais il faut aussi remarquer que l'espérance mathématique du processus varie également. Il est intéressant de noter qu'on peut obtenir deux paramètres de Hurst locaux différents sur deux échelles de temps différentes avec ce type de chaînes de Markov. La figure 8.7 nous en montre un exemple; si $m = 1$ à 100 environ, le paramètre de Hurst local vaut environ 0.7 alors qu'entre 1000 et 10000, il vaut environ 1. En combinant plusieurs progressions différentes de a et de b, il est possible d'obtenir

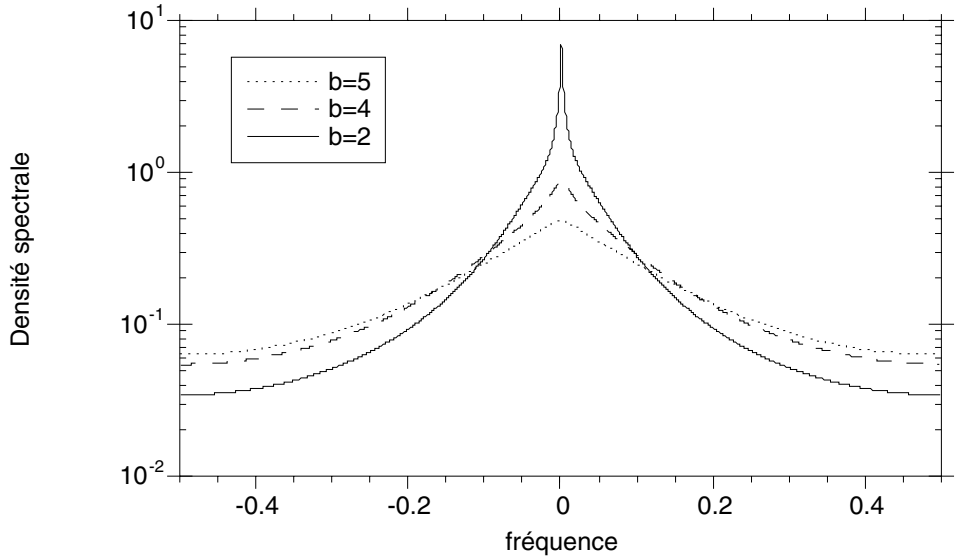

Figure 8.3: Densité spectrale, a=10

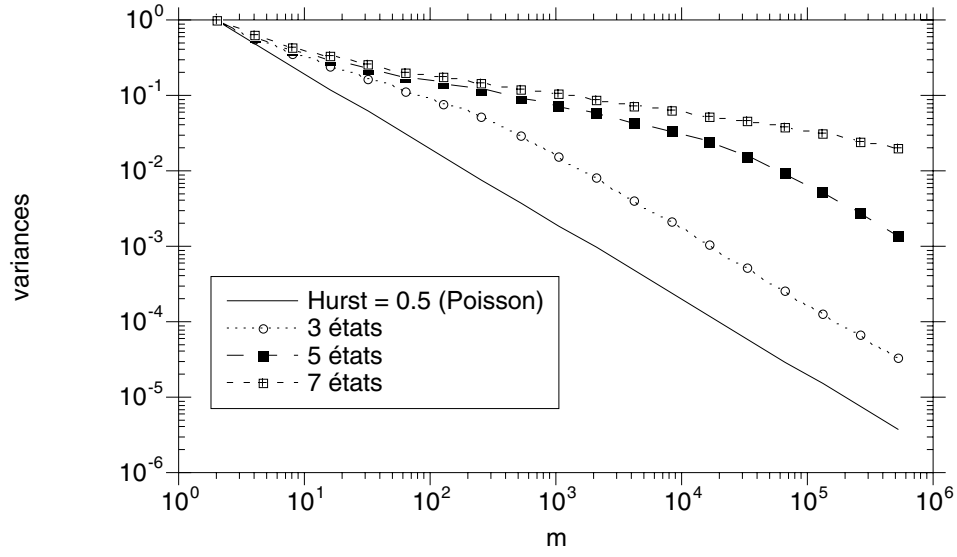

Figure 8.4: a=10, b=0.9, nombre d'états=n=3/5/7

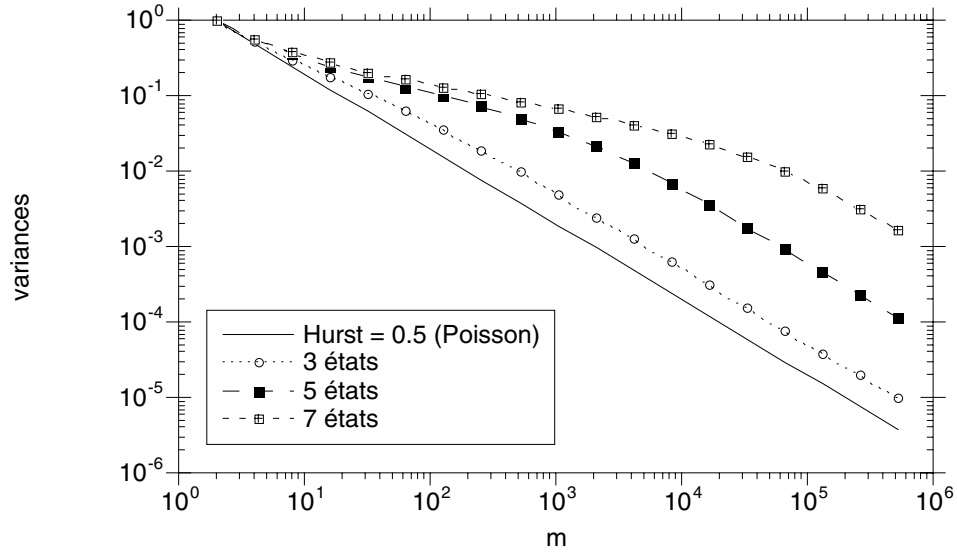

Figure 8.5: a=5, b=0.9, nombre d'états=n=3/5/7

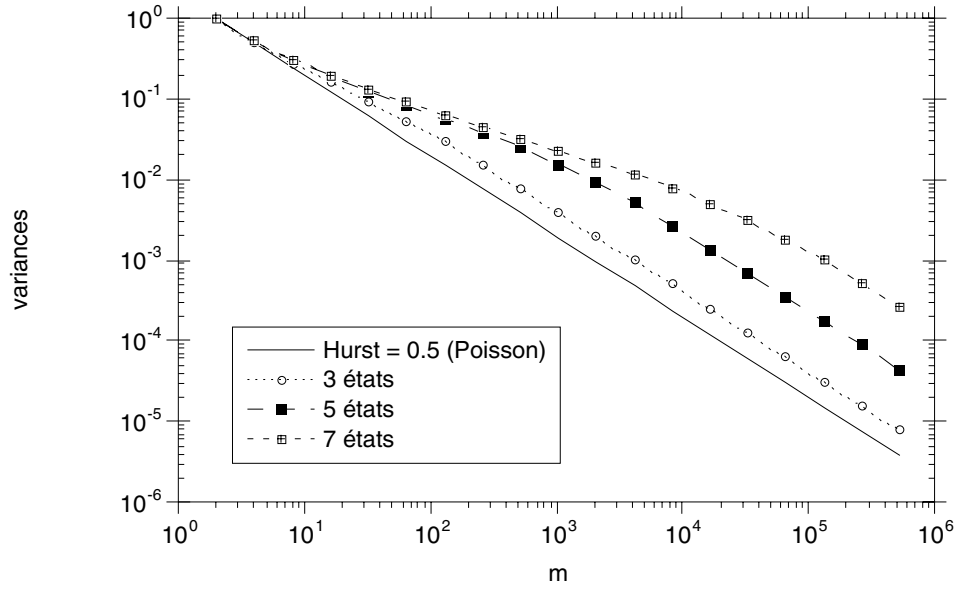

Figure 8.6: a=10, b=2, nombre d'états=n=3/5/7

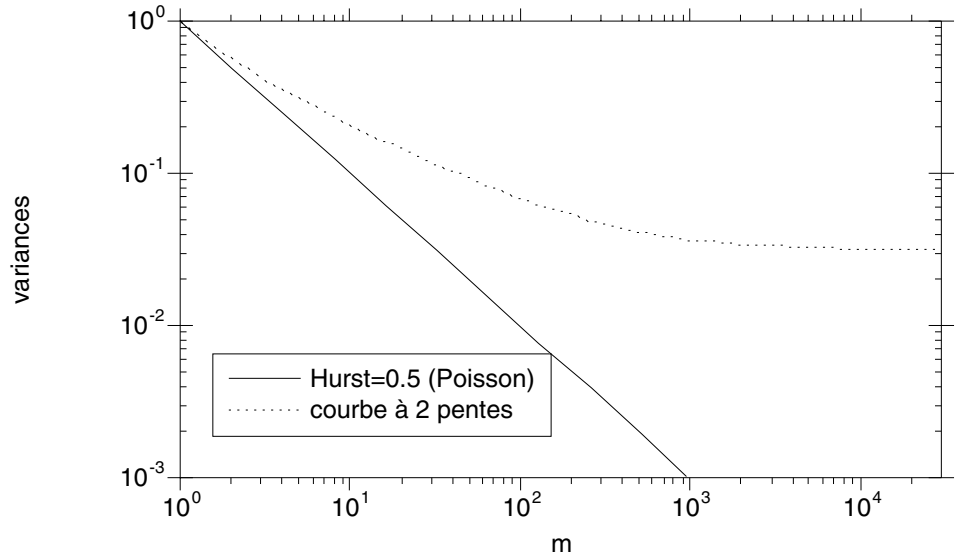

Figure 8.7: Exemple de courbe à 2 pentes différentes

une courbe qui ait différents paramètres de Hurst locaux sur des échelles de temps différentes, ce qu'il n'est pas possible d'obtenir avec des processus à mouvement Brownien fractionnaire par exemple.

8.4 Domaine de validité dans lequel le processus se comporte comme un processus auto-similaire

Après une introduction à la théorie de la décomposabilité de Courtois, nous voulons analyser des matrices à peu près complètement décomposables et complètement décomposables. En comparant les diagrammes "temps-variances", on sera à même de déterminer le domaine dans lequel le processus se comporte comme un processus auto-similaire.

Les pionniers de ce domaine sont Simon et Ando [87] qui ont appliqué leur théorie à plusieurs cas d'études, en économie et en physique. Ce qu'ils prétendent est qu'il est nécessaire de séparer l'étude à court terme de l'étude à long terme lorsqu'on a une agrégation de variables dans un système à peu près décomposable. Ils ont prouvé deux théorèmes majeurs. Le premier dit qu'un système à peu près décomposable peut être ana-

lysé par un système complètement décomposable si les dépendances entre les différentes parties du système sont relativement faibles en comparaison des dépendances à l'intérieur d'un même groupe. Le deuxième théorème dit que même à long terme, les résultats obtenus pour le court terme restent valables dans la mesure où le comportement relatif des variables d'un même groupe reste à peu près identique. Dans notre étude, nous postulons que le trafic LAN est composé de différentes échelles de temps. Ensuite il s'agit de trouver la chaîne de Markov à partir de ce postulat. La chaîne que nous avons proposée à la section 8.2 est plus régulière et possède une structure interne. En regardant sa matrice de transition, nous remarquons que cette dernière est décomposable à plusieurs niveaux mais pour trouver le domaine de validité où le processus a un comportement auto-similaire, nous allons la décomposer à un seul niveau. Le développement qui suit s'inspire des développements de Simon, Ando et Courtois. La matrice de transition $n \times n$ $\mathbf{A}$ de la chaîne de Markov introduite à la section 8.2 est à peu près complètement décomposable à condition que $(b/a)^{n-1}$ soit petit. Soit $\mathbf{A}^*$ complètement décomposable, alors $\mathbf{A}^*$ est composée de sous-matrices placées sur la diagonale

$$\mathbf{A}^* = \begin{pmatrix} \mathbf{A}_1^* & \cdots & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_2^* & \cdots & \mathbf{0} \\ \mathbf{0} & \cdots & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{A}_N^* \end{pmatrix} \tag{8.14}$$

Les éléments qui n'appartiennent pas à ces sous-matrices sont nuls. $\mathbf{A}^*$ est ainsi constituée de blocs de taille variable. Soit $\mathbf{A}_{IJ}^*$ une sous-matrice de $\mathbf{A}^*$ à l'intersection du I^{e} ensemble de lignes et du J^{e} ensemble de colonnes et soit $a_{i_I j_J}$ l'élément se situant dans la sous-matrice $\mathbf{A}_{IJ}^*$ à l'intersection de la i^{e} ligne avec la j^{e} colonne. Un nouveau vecteur de probabilité d'état est associé à la matrice $\mathbf{A}^*$, $\vec{\pi}_{t+1}^* = \vec{\pi}_t^* \mathbf{A}^*$. Le vecteur horizontal $\vec{\pi}^*$ est de la même taille que $\mathbf{A}_{II}^*$ avec $I = 1, ..., N$. Comme tous les blocs sont placés sur la diagonale on peut écrire $\mathbf{A}_{II} = \mathbf{A}_i$, $i = 1, ..., N$ étant une matrice carrée $n(i) * n(i)$ avec $\sum_{i=1}^{N} n(i) = n$. Chaque sous-matrice de $\mathbf{A}_i^*$ a son propre ensemble de valeurs propres $\lambda^*(i_I)$. Par commodité, nous supposerons qu'elles sont ordonnées, $\lambda^*(1_I) = 1 > \lambda^*(2_I) \geq ... \geq \lambda^*(n(I)_I)$ avec $I = 1, ..., N$. Nous savons que chaque bloc est stochastique, donc $\lambda^*(1_I) = 1$, $\forall I = 1, 2, \cdots, N$. Avec la matrice $\mathbf{A}$, la situation est différente car une seule

valeur propre est égale à 1, $\lambda(1_1)$. Nous supposons que les valeurs propres sont ordonnées, ici aussi. Supposons que $\mathbf{A}$ soit diagonalisable, alors

$$\mathbf{A} = \mathbf{G}^{-1}\mathbf{D}\mathbf{G} \tag{8.15}$$

$\mathbf{G}$ est la matrice de passage et $\mathbf{D}$ peut être écrit

$$\mathbf{D} = \sum_i \lambda_i \mathbf{P}_i = \sum_{i=1}^{n(I)} \sum_{I=1}^{N} \lambda(i_I)\mathbf{P}_I(i) \tag{8.16}$$

avec $\mathbf{P}_l$ = projecteur (i.e. $p_{ij} = 0\ \forall i, j \neq l$, $p_{ll} = 1$). Ainsi

$$\mathbf{A} = \mathbf{G}^{-1}\mathbf{P}_1(1)\mathbf{G} + \sum_{I=2}^{N} \lambda(1_I)\mathbf{G}^{-1}\mathbf{P}_I(1)\mathbf{G} + \sum_{I=1}^{N} \sum_{i=2}^{n(I)} \lambda(i_I)\mathbf{G}^{-1}\mathbf{P}_I(i)\mathbf{G} \tag{8.17}$$

$\lambda(i_I)\mathbf{G}^{-1}\mathbf{P}_I(i)\mathbf{G}$ peut être remplacé par $\mathbf{Z}(i_I)$. Les propriétés de $\mathbf{Z}(i_I)$ sont données dans [86].

De manière similaire, avec $\mathbf{A}^*$ nous avons

$$\mathbf{A}^* = \mathbf{G}^{-1}\mathbf{P}_1(1)\mathbf{G} + \sum_{I=2}^{N} \lambda^*(1_I)\mathbf{G}^{-1}\mathbf{P}_I(1)\mathbf{G} + \sum_{I=1}^{N} \sum_{i=2}^{n(I)} \lambda^*(i_I)\mathbf{G}^{-1}\mathbf{P}_I(i)\mathbf{G} \tag{8.18}$$

Ici nous allons donner le premier théorème de Simon et Ando [87] sans démonstration.

Théorème 8.1 *Pour un nombre positif arbitraire ς, il existe un nombre ϵ_ς tel que pour $\epsilon \leq \epsilon_\varsigma$,*

$$\max_{k,l} |z_{kl}(i_I) - z^*_{kl}(i_I)| < \varsigma \tag{8.19}$$

avec $2 \leq i \leq n(I)$, $1 \leq I \leq N$, $1 \leq k, l \leq n$

Concentrons-nous maintenant sur l'implication de ce théorème. La discussion qui va suivre est intuitive mais très importante dans notre contexte. Le comportement de $\vec{\pi}_t$ en fonction du temps et la comparaison de son comportement avec le vecteur $\vec{\pi}^*_t$ en fonction du temps est au centre du débat. A cause de l'ordonnancement des valeurs propres, les premiers termes de (8.17) n'engendrent pas de grandes variations dans le court terme ($t < T_1$) car les valeurs propres $\lambda(1_I)$ avec $I = 1, ..., N$ sont proches de l'unité. Donc, pour $t < T_1$, le terme variant de façon prédominante de $\mathbf{A}$ est le dernier. Ainsi $\vec{\pi}_t$ et $\vec{\pi}^*_t$ évoluent de la même façon. Pour $T_1 < t < T_2$, les évolutions de $\vec{\pi}_t$ et de $\vec{\pi}^*_t$ sont définies

par les derniers termes de $\mathbf{A}$ et de $\mathbf{A}^*$ respectivement. Un équilibre semblable est atteint dans chaque sous-système de $\mathbf{A}$ et de $\mathbf{A}^*$. Pour $T_2 < t < T_3$, le terme variant de manière prépondérante de $\mathbf{A}$ est le second. Pour $t > T_3$, le premier terme de $\mathbf{A}$ domine tous les autres. Un équilibre global est atteint pour le système. Le comportement dynamique de la matrice à peu près décomposable peut être séparé en quatre périodes bien distinctes que Simon et Ando appelent [87] respectivement 1) la dynamique à court terme, lorsque $t < T_1$. 2) l'équilibre à court terme, lorsque $T_1 < t < T_2$. 3) la dynamique à long terme, lorsque $T_2 < t < T_3$ et 4) l'équilibre à long terme, lorsque $t > T_3$.

8.5 Application à notre problème

A partir de la matrice décrite dans la section 8.2, nous nous proposons de créer une nouvelle matrice, décomposable. Rappelons que $1/a^{n-1}$ et $(b/a)^{n-1}$ sont supposés être suffisamment petits ($b < a$ nécessairement). Nous avons vu que la matrice proposée a la forme suivante:

$$\mathbf{A} = \begin{pmatrix} 1 - 1/a - 1/a^2 - ... - 1/a^{n-1} & 1/a & 1/a^2 & \cdots & 1/a^{n-1} \\ b/a & 1 - b/a & 0 & \cdots & 0 \\ (b/a)^2 & 0 & 1 - (b/a)^2 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ (b/a)^{n-1} & 0 & \cdots & 0 & 1 - (b/a)^{n-1} \end{pmatrix} \tag{8.20}$$

Ici, nous allons considérer uniquement deux sous-matrices placées sur la diagonale, $\mathbf{A}_1^*$ et $\mathbf{A}_2^*$. $\mathbf{A}_1^*$ est une matrice carrée de dimension $(n-1) \times (n-1)$ et $\mathbf{A}_2^*$ une matrice carrée de dimension 1. Ainsi $\mathbf{A}^*$ peut s'écrire de la façon suivante

$$\mathbf{A}^* = \begin{pmatrix} 1 - 1/a - 1/a^2 - ... - 1/a^{n-2} & 1/a & 1/a^2 & \cdots & 0 \\ b/a & 1 - b/a & 0 & \cdots & 0 \\ (b/a)^2 & 0 & 1 - (b/a)^2 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & \cdots & 1 \end{pmatrix} \tag{8.21}$$

$$\mathbf{A}^* = \begin{pmatrix} \mathbf{A}_1^* & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_2^* \end{pmatrix} \tag{8.22}$$

Dans ce cas, $n(1) = n - 1$ et $n(2) = 1$ avec $n(1) + n(2) = n$ et $N = 2$. Notons que $\mathbf{A}^*$ n'est pas ergodique. Il est intéressant maintenant de prendre un exemple concret pour étayer cette théorie. Prenons par exemple $a = 6.7$, $b = 0.5764$ et $n = 5$ (nous avons dans ce cas $E[X] = 0.05$ et $H = 0.8$). Traçons le graphe "temps-variances" pour deux chaînes de Markov différentes, dont les matrices de transition sont $\mathbf{A}$ et $\mathbf{A}^*$ respectivement. La matrice $\mathbf{\Lambda}$ est la même pour les deux chaînes de Markov, à savoir celle qui est définie à la section 8.2

$$\mathbf{\Lambda} = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & \cdots & 0 \end{pmatrix} \tag{8.23}$$

La figure 8.8 nous montre l'évolution temporelle des probabilités d'état pour ces deux chaînes de Markov différentes. Les conditions initiales sont identiques pour les deux chaînes de Markov: $\pi_1 = 1$, $\pi_2 = \cdots = \pi_5 = 0$. Leurs évolutions sont identiques jusqu'au début de l'équilibre à long terme. Ensuite, toutes les probabilités d'état de la chaîne décomposée se séparent de celles de la chaîne décomposable. La figure 8.9 nous montre le diagramme "temps-variances" pour ces deux chaînes de Markov différentes. Ici encore, leurs évolutions sont identiques jusqu'au début de l'équilibre à long terme. Ensuite, elles se séparent.

8.6 Test visuel

La figure 8.10 illustre le concept exposé à la section 4.9.4. Comme le concept de l'auto-similarité est étroitement lié au concept d'invariance statistique à travers les échelles de temps, le test le plus facile à faire pour la détecter est de reporter le processus moyenné $X_t^{(m)}$ sur plusieurs échelles de temps. L'unité de temps m vaut successivement 10, 100, 1000 et 10000 créneaux.

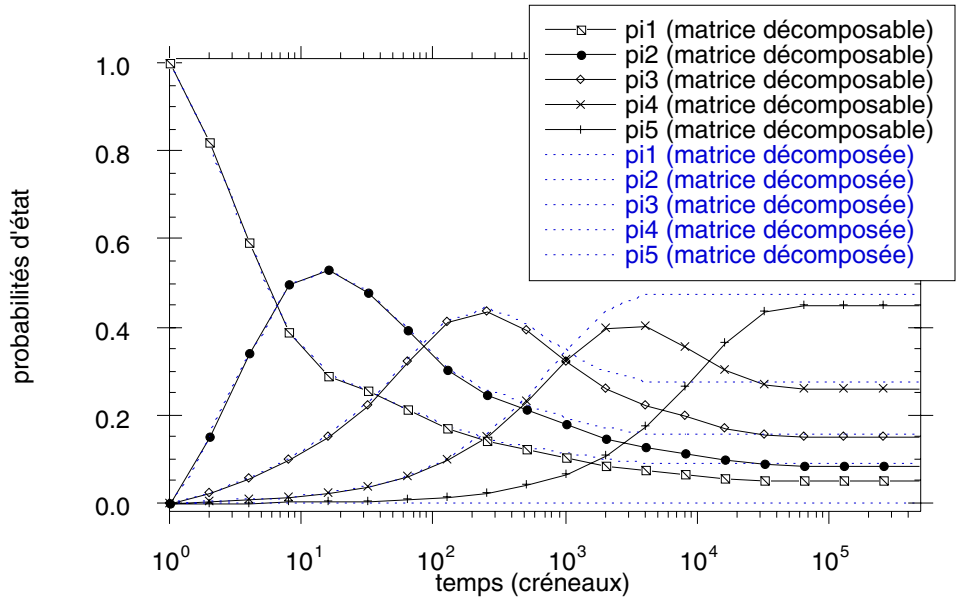

Figure 8.8: Evolution temporelle des probabilités d'état pour la chaîne de Markov ayant une matrice de transition décomposable, a=6.7, b=0.5764, n=5 et pour la chaîne de Markov décomposée

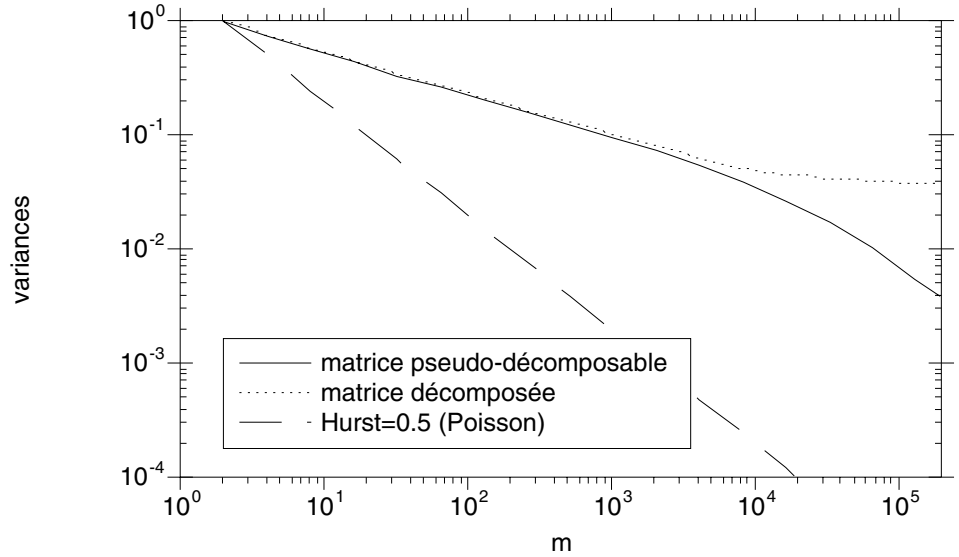

Figure 8.9: Diagramme "temps-variances" pour la chaîne de Markov ayant une matrice de transition décomposable, a=6.7, b=0.5764, n=5 et pour la chaîne de Markov ayant une matrice de transition décomposée

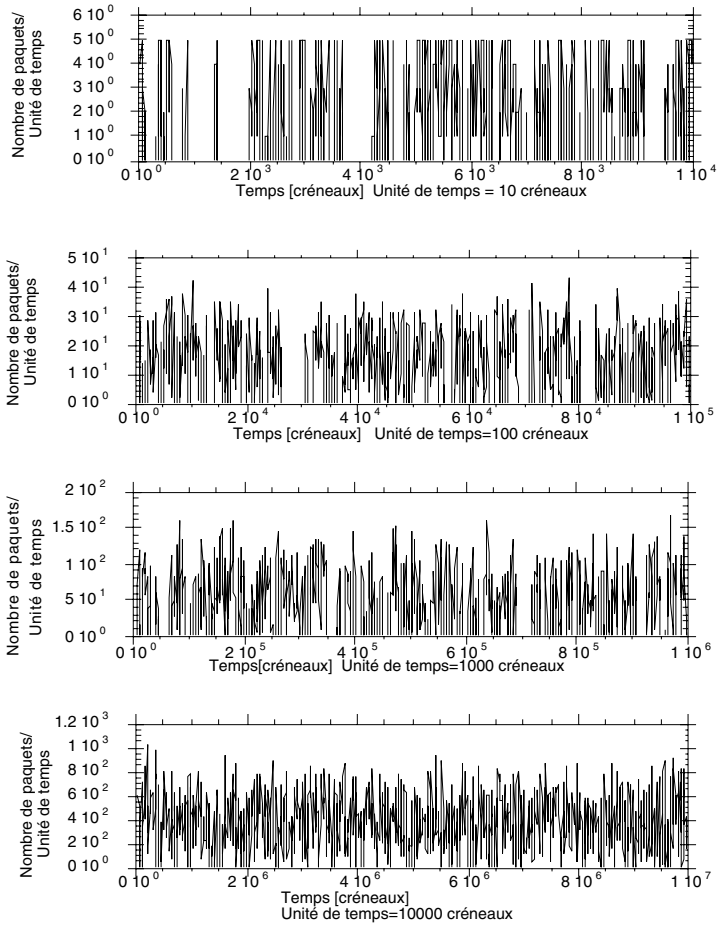

Figure 8.10: Evolution temporelle du trafic généré sur plusieurs échelles de temps, $H_l = 0.78$

8.7 *Fitting*

Notre procédure de *fitting* est basée sur la chaîne de Markov décrite à la section 8.2 qui ne dépend que de trois paramètres, son espérance mathématique, son paramètre de Hurst local et son nombre d'états. Rappelons que son espérance mathématique est connue ($E[X] = (1-(1/b))/(1-(1/b)^n)$. Pour $E[X]$ connu, nous pouvons appliquer l'algorithme suivant:

Algorithme 8.1 Algorithme Recherche de b

Initialisation

entrée de $E[X]$

$b = b_{init} = b_{old}$

répéter

$E[X]_{trouve} = (1-1/b)/(1-(1/b)^n)$

si $(E[X]_{trouve} > E[X])$ **alors**

$b_{old} = b$

$b = b/2$

sinon

$b = (b_{old} - b)/2$

jusqu'à ce que $(E[X] - E[X]_{trouve} < \epsilon)$

fin

Il faut éviter de prendre b_{init} trop petit. Le domaine de validité où le processus a un comportement auto-similaire est donné à la section 8.5. Ce domaine peut être augmenté en ayant plus d'états dans la chaîne de Markov; a est utilisé pour ajuster la pente de la courbe β pour avoir le paramètre de Hurst local H_l désiré (a est trouvé itérativement à l'aide d'une méthode de Newton-Raphson), $\hat{\beta}$ est estimée à l'aide d'une procédure de moindres carrés. La figure 8.11 montre un diagramme "temps-variances" pour une chaîne de Markov à cinq états dont le paramètre de Hurst local varie de 0.6 à 0.95. Il faut relever que le domaine de validité augmente lorsque le paramètre de Hurst local devient plus important, quand $H_l \to 1$. En d'autres termes, si nous voulons que la chaîne de Markov ait un comportement auto-similaire sur un intervalle donné, il faudra qu'elle ait

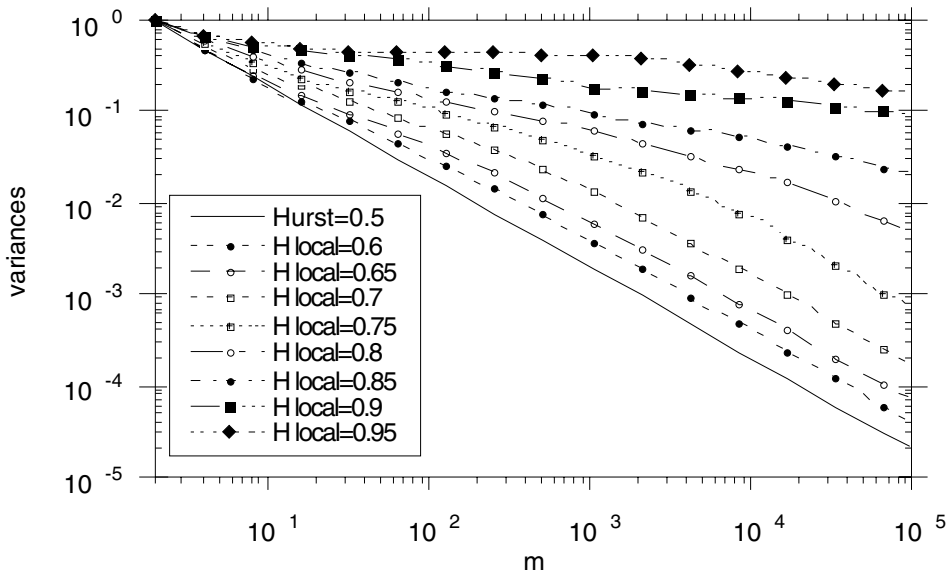

Figure 8.11: Diagramme "temps-variances" pour une chaîne de Markov à 5 états ayant un paramètre de Hurst local variant entre 0.6 à 0.95, $E[X] = 0.05$

plus d'états si le paramètre de Hurst local est faible que s'il est élevé. Le domaine de validité est très apparent à la figure 8.11. Prenons par exemple la courbe où $H_l = 0.8$, le coude y est très visible, il sépare le domaine où le processus a un comportement auto-similaire du domaine où il a un comportement de type "poissonien". Remarquons que pour des valeurs extrêmes du paramètre de Hurst local ($H_l = 0.9, 0.95$), la courbe comporte quelques vagues. Le tableau 8.3 donne quelques valeurs trouvées à l'aide de l'algorithme.

8.8 Simulateur

La simulation d'une telle chaîne de Markov exige d'être prudent à cause des grandes différences qu'on peut trouver entre deux probabilités de transition. Pour illustrer le problème auquel nous pouvons être confrontés, prenons par exemple une chaîne de Markov du type décrit à la section 8.2 ayant 8 états, avec $a = 10$ et $b = 0.9$, nous trouvons que $a_{12} = 10^{-1}$ mais $a_{18} = 10^{-7}$, donc le rapport entre ces deux probabilités de transition est de 10^6. Si nous simulons la chaîne de Markov de manière classique, il faudra un excellent générateur de nombres aléatoires. Si le rapport des probabilités de transi-

n	H_l	a	b
4	0.70	2.65	0.4418
	0.75	4.97	
	0.80	10.27	
	0.85	28.49	
5	0.70	2.60	0.5764
	0.75	4.38	
	0.80	6.70	
	0.85	12.97	
	0.90	85.61	
6	0.70	1.99	0.6737
	0.75	3.22	
	0.80	4.80	
	0.85	8.21	
	0.90	50.02	

Tableau 8.3: Valeurs de a et de b en fonction de H_l et de n, $E[X] = 0.05$

tion est plus grand, aucun générateur de nombres aléatoires ne sera capable de générer correctement du trafic selon la chaîne de Markov donnée. C'est pourquoi nous avons utilisé une technique itérative. Supposons par exemple que $a = 10$ et que la chaîne de Markov se trouve dans le premier état, alors un chiffre entre 0 et 1 est généré à l'aide d'un générateur de nombres aléatoires fiable [93]. Si le nombre obtenu est inférieur à $1/a = 1/10$, nous en déduisons que le prochain état n'est pas l'état 2 mais qu'il est compris entre 3 et n. La procédure se poursuit jusqu'à ce que le nombre obtenu par le générateur de nombres aléatoires soit supérieur à $1/a$ ou que le dernier état soit atteint. Pour résumer, la probabilité de se retrouver dans l'état i en sachant que nous nous trouvons dans l'état 1 est $prob(Y_{t+1} = i|Y_t = 1) = (1/a)^i$ avec $i = 2, 3, ..., n$ et la probabilité de rester dans l'état 1 est donnée par $prob(Y_{t+1} = 1|Y_t = 1) = 1 - \sum_{i=1}^{n-1}(1/a)^i$. Ainsi, nous ne sommes pas limités par le nombre d'états de la chaîne de Markov. Nous avons vu comment changer d'état alors que nous sommes dans le premier état mais lorsque nous sommes dans un autre état, le rapport des probabilités de transition est très important également. En reprenant l'exemple précédent, dans le 8^e état, la probabilité de transition $prob(Y_{t+1} = 1|Y_t = 8) = (b/a)^8 = 4.3 \times 10^{-9}$ et $prob(Y_{t+1} = 8|Y_t = 8) = 1 - prob(Y_{t+1} = 1|Y_t = 8) = 1 - 4.3 \times 10^{-9}$. De nouveau, le problème se pose quant aux grandes différences entre les probabilités de transition. Une technique est d'utiliser la

méthode précédente pour savoir si nous changeons d'état ou non. On peut par exemple se fixer un seuil arbitraire et générer autant de nombres avec le générateur de nombres aléatoires qu'il est nécessaire. Si nous reprenons l'exemple précédent et que nous fixons le seuil à 1/100, il faudra que quatre fois de suite, le nombre aléatoire soit inférieur à 1/100 et la dernière fois inférieur à 0.43 par exemple pour que le modulateur passe du dernier état au premier. Au lieu d'utiliser cette méthode, nous avons procédé différemment car même si elle est précise et fiable, elle présente l'inconvénient d'être lente. Au lieu de faire un calcul à chaque créneau pour savoir dans quel état la chaîne de Markov se trouvera au créneau suivant, nous profitons du fait que cette dernière va rester de longues périodes dans le même état pour calculer ce temps-là. La longueur des temps de séjour dans un même état est distribuée géométriquement. Comme les probabilités de rester dans le même état peuvent être voisines de 1, il convient de tester la distribution géométrique générée. Pour ce faire, nous l'avons testée avec des paramètres variant de 0.5 à $1-0.5\times10^{-8}$. Pour chaque paramètre, nous avons généré 10000 longueurs de temps de séjour et avons trouvé une erreur d'au plus 2.3% entre l'espérance mathématique de la distribution et la moyenne obtenue par simulation. Le modulateur change d'état selon la méthode itérative discutée plus haut et lorsque le modulateur est dans un état, le temps de séjour est trouvé à l'aide du générateur de nombres aléatoires. Ainsi, à chaque itération, la chaîne de Markov change d'état.

8.9 Problème de la file d'attente

Dans cette section, nous allons observer le comportement d'une file d'attente ayant un temps de service fixé de K créneaux (le processus des arrivées est discret) pour chaque cellule entrant dans le système; une seule cellule peut être servie à la fois, la file ne possède qu'un serveur. Aucune priorité n'est prise en compte et la stratégie de service de la file d'attente est du type FCFS. La file d'attente possède c places. Si une cellule arrive alors que la file d'attente est pleine, alors elle est éliminée. Les arrivées de cellules sont modulées par la chaîne de Markov décrite à la section 8.2. Pour le calcul de cette file d'attente, nous introduisons le concept de **mini-créneau**, le temps de K mini-

créneaux étant égal au temps de service d'une cellule. Le rapport des débits entre l'entrée et la sortie est alors donné par K. Maintenant, nous allons décrire comment obtenir la distribution de la variable aléatoire $X(K)_t$ en fonction de K. La variable aléatoire $X(K)_t$ est le nombre de cellules arrivant pendant un intervalle de temps de K créneaux.

Cette distribution peut être calculée par une formule de récurrence. K est le nombre de mini-créneaux. k est le nombre de 1 obtenus pendant l créneaux et i l'état du modulateur. Observons les instants $K, 2\times K, 3\times K, ...$ et soit $D_k^l(i)$ la distribution des arrivées de cellules entre deux instants (K et $2\times K$ par exemple) quand l'état du modulateur est dans l'état i au commencement du créneau, alors la formule de récurrence est donnée par

$$\begin{aligned} D_k^{l+1}(i) &= prob(X_t = 1|Y_t = i)\sum_{j=1}^{n} a(i,j)D_{k-1}^l(j) + \\ &\quad (1 - prob(X_t = 1|Y_t = i))\sum_{j=1}^{n} a(i,j)D_k^l(j) \end{aligned} \tag{8.24}$$

avec les conditions initiales suivantes:

$$D_k^1(i) = \begin{cases} k = 0 & 1 - prob(X_t = 1|Y_t = i) \\ k = 1 & prob(X_t = 1|Y_t = i) \\ k \geq 1 & 0 \end{cases} \tag{8.25}$$

La matrice de transition, observée aux instants $K, 2\times K, 3\times K, ...$ est donnée par $\mathbf{A}^K$. La matrice de transition de la file d'attente $\mathbf{Q}$ est du type *upper-block Hessenberg*. La probabilité de perte se calcule facilement à l'aide de l'algorithme MBH par la procédure décrite dans la section 3.13. La figure 8.12 nous montre l'évolution du *Cell Loss Ratio* (CLR) (nombre de cellules perdues/nombre de cellules émises) en fonction de l'utilisation ($K \times E[X]$) de la file d'attente pour différentes chaînes de Markov. Ici, $E[X] = 0.05$ et la longueur de la file d'attente est de 200 places pour toutes les chaînes de Markov. La source à 5 états a un paramètre de Hurst local $H_l = 0.8$ ($a = 6.7$, $b = 0.5764$). Avec la source à 2 états, nous avons cherché un point commun avec la source à 5 états. Lorsque l'utilisation =0.1, les valeurs du CLR des deux sources sont égales mais ensuite, leur évolution avec des utilisations plus élevées est complètement différente. L'augmentation du paramètre a de la source à deux états ne résoud pas le problème car son évolution

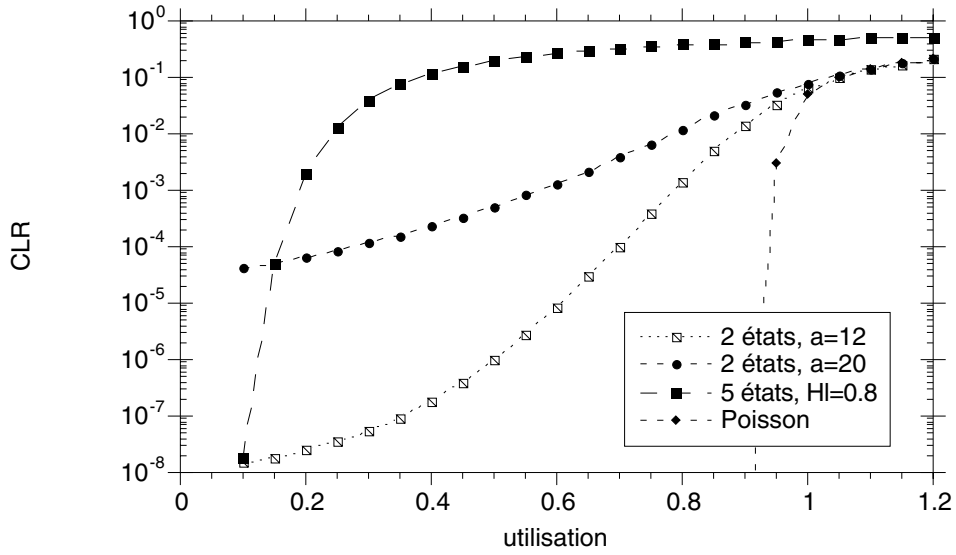

Figure 8.12: Comparaison du CLR pour différentes chaînes de Markov, à 2 et à 5 états, $c = 200$

reste toujours la même. On observe seulement un décalage de la courbe vers le haut. Sur la même figure, l'évolution du CLR d'une source poissonienne est montrée à titre illustratif. Il est intéressant de voir que le comportement du CLR en fonction de l'utilisation est le même pour une chaîne à 3 états que celui de la chaîne à 5 états. Ceci est montré à la figure 8.13. Le paramètre de Hurst local a une grande influence sur le comportement de la file d'attente. Ici, nous avons considéré une chaîne de Markov à 5 états en variant le paramètre de Hurst local mais il ne faut pas oublier que le domaine de validité où le processus se comporte comme un processus auto-similaire augmente avec H_l. La figure 8.14 nous montre comment le paramètre de Hurst local influence le comportement de la file d'attente. Pour voir si le paramètre de Hurst est vraiment un paramètre fondamental pour le comportement de la file d'attente, on peut construire une nouvelle chaîne de Markov en inversant les états de telle sorte à ce que la chaîne de Markov produise des cellules lorsqu'elle se trouve dans les états $2, 3, 4, \cdots, n$. Autrement dit $\phi_{11} = prob(X_t = 1|Y_t = 1) = 1$, $\phi_{10} = 1$ et $\phi_{k1} = 1$, $\phi_{k0} = 0$ $\forall k = 1, 2, \cdots, n$ et qu'elle ait le même paramètre de Hurst local. La figure 8.15 nous montre que l'évolution du CLR avec l'utilisation n'est pas identique à celle de la chaîne originale. Donc le paramètre de Hurst local n'est pas un paramètre fondamental pour le comportement de la file d'attente. Comme nous l'avons dit au chapitre 6, les deux paramètres les plus importants

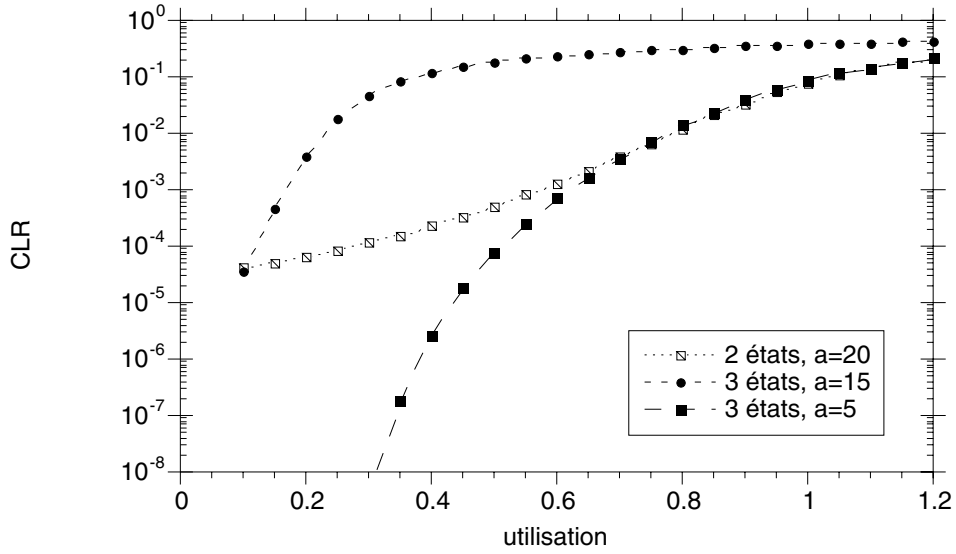

Figure 8.13: Comparaison du CLR pour des chaînes de Markov à 2 et à 3 états, $c = 200$

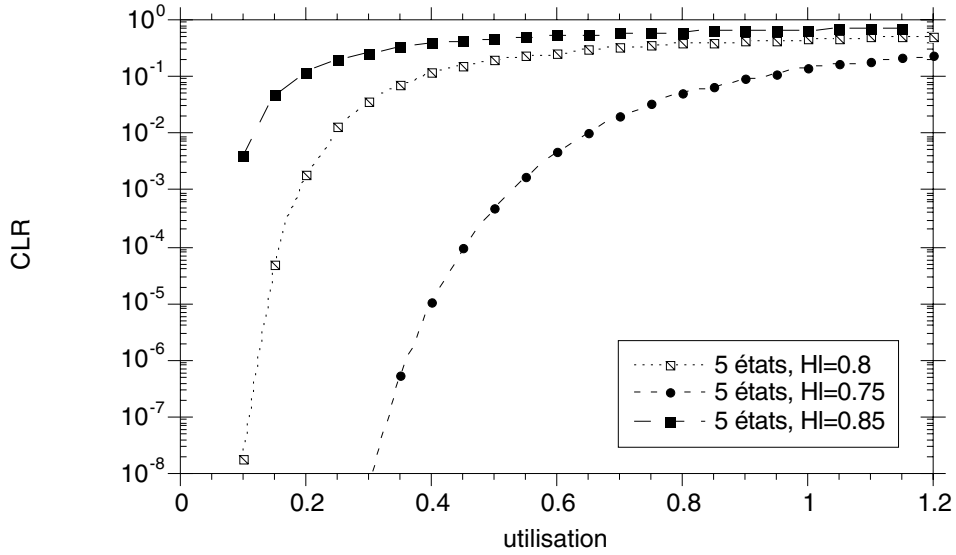

Figure 8.14: Variation du paramètre de Hurst local pour une chaîne de Markov à 5 états, $E[X] = 0.05$, $c = 200$

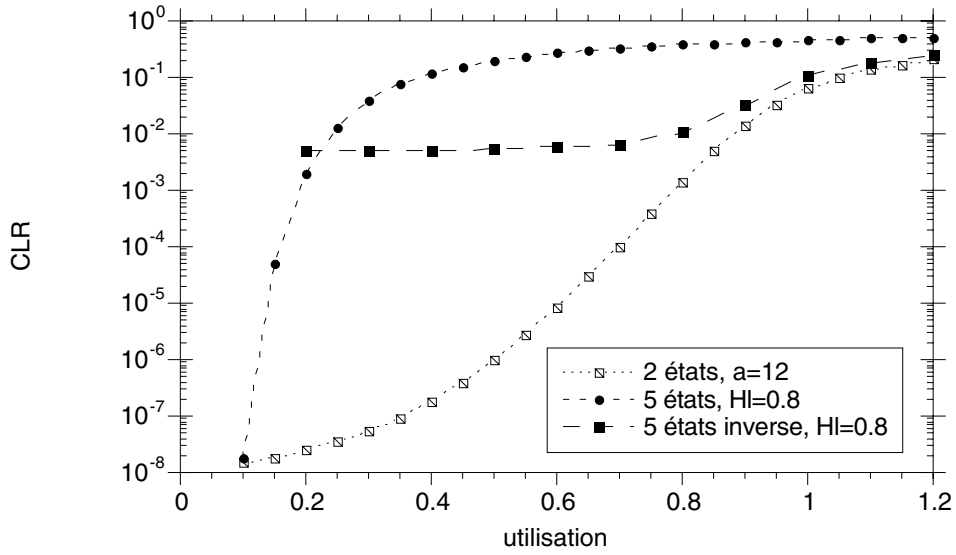

Figure 8.15: Comportement de la chaîne de Markov à 5 états et de son inverse, $c = 200$

pour le comportement de la file d'attente sont la distribution et la densité spectrale de $U(t)$ ($U(t)$=arrivées(t)-départs(t)). Or, lors de l'évaluation du paramètre de Hurst par la méthode des variances, on évalue la variance du processus et on la normalise (ainsi on néglige la valeur absolue de cette dernière). Si on considère la valeur absolue de la variance sur plusieurs échelles de temps alors cet indice nous renseigne sur le deuxième moment de la distribution sur différentes échelles de temps, qui est un élément important pour le comportement de la file d'attente. Maintenant, le tout est de savoir comment se comporte le trafic réel dans la même file d'attente pour voir si les sources que nous considérons ne sont pas trop irréalistes. Pour cela, nous avons pris un fichier de données de Bellcore que nous avons discrétisé pour ensuite faire entrer les cellules dans la même file d'attente. Le fichier comprend environ 10 millions de cellules. En comparant le CLR de notre source avec celui obtenu par simulation, à partir d'une trace réelle (pAug.TL), nous remarquons une bonne correspondance (figure 8.16).

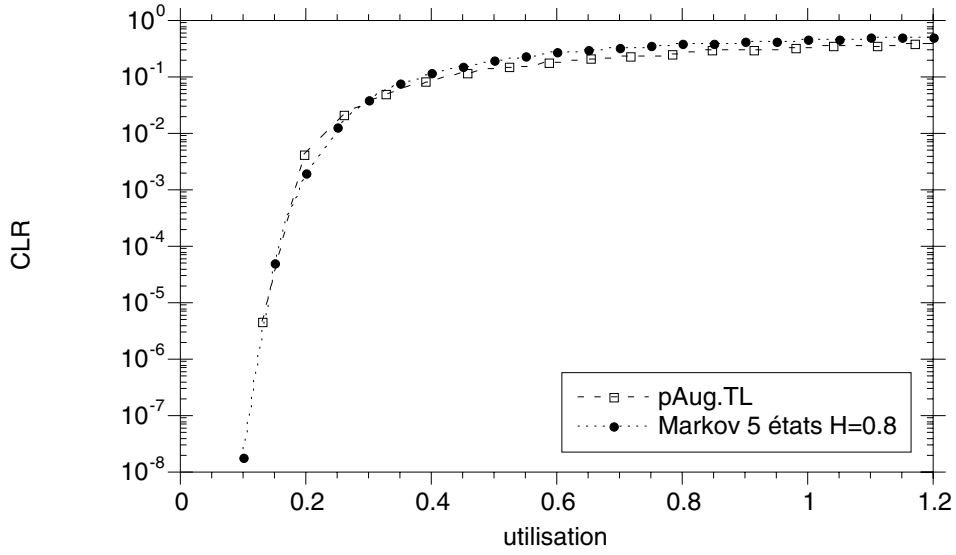

Figure 8.16: Comparaison du comportement de la file d'attente pour la chaîne de Markov à 5 états et des mesures de Bellcore, $c = 200$

Chapitre 9

Multiplexage avec du trafic auto-similaire

9.1 Introduction

Dans ce chapitre, nous allons nous intéresser au multiplexage statistique qu'il est possible d'obtenir avec des sources markoviennes ayant des pseudo-dépendances à long terme. Le but est de savoir s'il est possible ou non de faire du multiplexage statistique malgré le fait que les sources peuvent avoir des comportements qui diffèrent de ceux qui ont été généralement admis dans ce genre d'études [75]. Ce sujet est encore en pleine mouvance [94, 95] car à l'aube du déploiement de l'ATM, on s'interroge encore quant au gain de multiplexage statistique qu'il est possible d'obtenir suite aux affirmations des chercheurs de Bellcore. Il n'est peut-être pas inutile de rappeler qu'un avantage promis par l'ATM était qu'il était possible de gagner beaucoup de largeur de bande avec ce nouveau type de réseau, à cause du multiplexage. D'autre part, nous faisons l'analyse du cas le pire qui peut se présenter sur le réseau, lorsqu'il n'est pas possible de faire du multiplexage statistique. Le contexte dans lequel le problème du multiplexage est étudié est celui d'un projet, *scalability enhancements for connection-oriented networks* (SCONE), dans lequel une nouvelle architecture pour les réseaux ATM est proposée. Cette architecture vise à réduire le coût des contrôles dû au maintien des connexions. Ce coût est associé au temps de calcul nécessaire et au stockage d'informations. La solution proposée avec

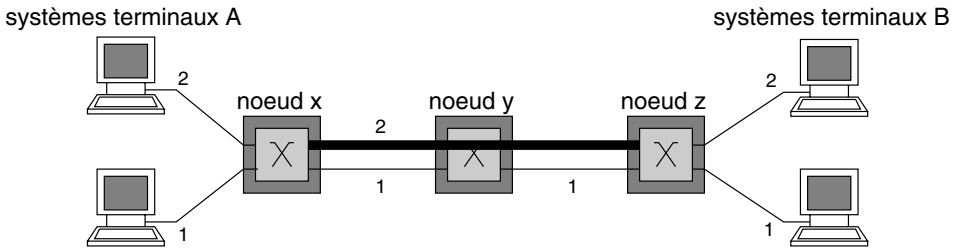

Figure 9.1: Concept de SCONE

SCONE est de grouper les connexions au sein d'un même réseau de transit, autrement dit, en utilisant des connexions dynamiques (chemins virtuels) entre les noeuds d'accès ATM. Les caractéristiques de SCONE concernant l'idée de base sont présentées dans [96]. Dans [97], une proposition concernant la signalisation dans SCONE est présentée. Elle concerne les réseaux ATM qui doivent supporter simultanément l'établissement de connexion d'un *Virtual Channel Connection* (VCC) et du *Virtual Path* (VP) en une seule opération. Dans [98], c'est le contrôle de trafic dans la solution SCONE qui est présenté. Dans les pages qui vont suivre, nous allons nous intéresser au contrôle de trafic également, mais en prenant comme sources de trafic des sources markoviennes produisant du trafic auto-similaire sur une certaine échelle de temps. Dans SCONE, le concept de *Virtual Path Trunk* (VPT) est introduit. Un VPT est la connexion d'un chemin (*path*) établi par le réseau pour réduire la connaissance (*awareness*) de cette dernière aux noeuds de transit. Les VCC sont établis sur des VPT. Ce concept est montré à la figure 9.1. Dans SCONE, le CAC est fait au niveau des VPT. Les connexions VCC sont de classe *Variable Bit Rate* (VBR). Les connexions VPT peuvent être de classe *Constant Bit Rate* (CBR), VBR, *Available Bit Rate* (ABR) ou *Unspecified Bit Rate* (UBR). Dans les cas qu'on va étudier, on va considérer uniquement les classes CBR et VBR. CBR est un cas particulier de VBR où le paramètre de la connexion est simplement donné par le *Peak Cell Rate* (PCR). Ces modèles ont déjà été étudiés dans la littérature mais à notre connaissance pas avec du trafic de type auto-similaire. Lorsque la connexion VPT est de classe CBR, on ne peut évidemment pas avoir de multiplexage statistique entre VPTs. SCONE donne la possibilité de transporter un trafic d'une autre classe que CBR pour pouvoir en bénéficier. L'utilisation de la classe VBR pour les connexions VPT permet

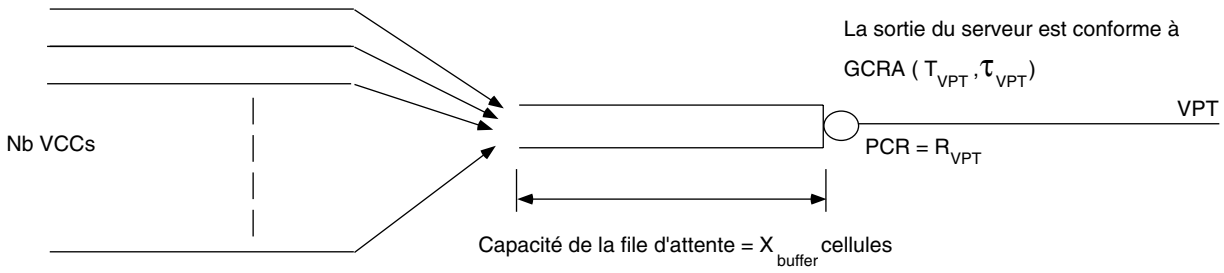

Figure 9.2: Multiplexeur

de faire un meilleur usage de la largeur de bande.

9.2 Problème à étudier

Avec de telles connexions, le problème est de savoir s'il est possible d'obtenir un gain de multiplexage en mélangeant les différentes sources, même si ce trafic est auto-similaire sur une certaine période. La figure 9.2 montre une description de la situation. Nb connexions identiques, de classe VBR, ayant les mêmes attributs de connexion (section 9.3), entrent dans le multiplexeur constitué d'une file d'attente et d'un serveur. Le temps est discret dont l'unité est le créneau ATM. La taille de la file d'attente est X_{buffer}. La connexion VPT est de classe VBR, le serveur sert les cellules conformément au GCRA(T_{VPT}, τ_{VPT}) et GCRA(R_{VPT}, CDV_{VPT}) (section 9.3). Nous allons étudier premièrement les connexions VCC avec du trafic auto-similaire sur une certaine échelle de temps et ensuite étudier le multiplexage de toutes ces sources identiques et faire une comparaison avec le cas le pire (*worst case*).

9.3 Entrée du multiplexeur

9.3.1 Modèle du GCRA pour les VCCs

Dans ce paragraphe, nous voulons modéliser le *Generic Cell Rate Algorithm* (GCRA) car à l'entrée du multiplexeur, nous avons Nb connexions de type VCC qui doivent se conformer aux paramètres du GCRA. Or, pour des paramètres de source définis, nous

voulons pouvoir régler les paramètres du GCRA de telle sorte à ce que la perte soit inférieure à un certain taux. Pour commencer, rappelons le rôle du GCRA. L'ATM Forum [99] et l'ITU [29] ont spécifié un mécanisme pour le *User Network Interface* (UNI) qui contrôle le trafic s'écoulant à travers une connexion VCC. Ce mécanisme est appelé GCRA qui est utilisé pour définir une relation entre le PCR et le *Cell Delay Variation* (CDV) et une relation entre le *Sustained Cell Rate* (SCR) et *Burst Tolerance* (BT). Le GCRA dépend de deux paramètres, son incrément (noté T_{VCC} ici) et sa limite (notée τ_{VCC} ici). Cet algorithme définit un *Theoretical Arrival Time* (TAT) qui est le temps d'arrivée "nominal" d'une cellule [99]. Si le temps d'arrivée n'est pas inférieur à $TAT - \tau_{VCC}$, alors la cellule est déclarée conforme et l'algorithme ajourne sa valeur de TAT à $max(t, TAT) + \tau_{VCC}$, sinon, la cellule est déclarée non-conforme. Les équations du compteur du GCRA et du "travail inachevé" (*unfinished work*) d'une file d'attente $G/D/1/c$ sont les mêmes. Le comportement du GCRA est le même que celui d'une file d'attente en ce qui concerne le rejet de cellules si on admet que le débit maximum à l'entrée du GCRA est le même que le débit maximum à la sortie de ce dernier. Une grande différence, cependant, entre les deux mécanismes est que le GCRA ne modifie pas la forme du trafic alors que la file d'attente le fait. Ici, nous examinons uniquement le rejet ou le marquage des cellules. L'occupation de la file d'attente est $\lceil$travail inachevé$/D\rceil$ avec D étant le temps de service. Dans la file d'attente, on rejette des cellules lorsque la file d'attente est pleine et que $\lceil$travail inachevé$/D\rceil$ excède la grandeur de la file d'attente tandis qu'avec le GCRA, on marque les cellules quand le compteur dépasse une certaine limite. Or nous savons que les équations du compteur du GCRA et du "travail inachevé" de la file d'attente sont les mêmes. Pour une file d'attente suffisamment longue (3.3% d'erreur au maximum pour une file d'attente d'au moins 30 places), on peut dire que la limite du GCRA = longueur de la file d'attente$\times D$. Ici $D = T_{VCC}$ et la limite du GCRA est donnée par τ_{VCC}.

9.3.2 Paramètres du GCRA pour VBR et pour CBR

ATM-Forum autorise la transmission de trafic de classe CBR ou de classe VBR sur une connexion VCC. Pour les connexions de classe CBR, l'usager ne déclare pas de

SCR mais uniquement un PCR. En d'autres termes, les cellules qui arrivent doivent être conformes à $GCRA(T_{VCC} = 1/R_{VCC}, CDV_{VCC})$ avec CDV_{VCC} étant petit. Pour du trafic VBR, la source devra déclarer PCR_{VCC} $(= R_{VCC})$, SCR_{VCC} $(= 1/T_{VCC})$, BT_{VCC} $(= \tau_{VCC})$ et CDV_{VCC}. En d'autres termes, les cellules qui arrivent devront se conformer à deux GCRA consécutifs, le premier étant $GCRA(R_{VCC}, CDV_{VCC})$ et le second $GCRA(T_{VCC}, \tau_{VCC})$. L'avantage de faire usage de connexions VBR est qu'il est possible d'envoyer un flux de cellules collées les unes aux autres durant une période de temps qui dépend de la valeur de BT. Il ne faut cependant pas oublier que le mécanisme à mettre en oeuvre dans le réseau est plus complexe.

9.3.3 Calcul de la perte dans un GCRA sur le VCC

Dans le paragraphe 9.3.1, nous avons vu que dans ce contexte, le GCRA pouvait être remplacé par une file d'attente. Dans ce paragraphe, nous allons évaluer la probabilité de perte de cellules dans le GCRA en fonction de ses paramètres avec une source déterminée (chaîne de Markov à 5 états avec différents paramètres de Hurst local). Les figures 9.3, 9.4 et 9.5 nous montrent, pour une probabilité de perte allant de 10^{-6} à 10^{-10} dans le GCRA, quel τ_{VCC} il faut déclarer pour chaque SCR. La figure 9.6 montre comment un plus faible CLR influence la valeur de τ_{VCC}.

9.4 Multiplexeur

9.4.1 Cas le pire (*worst case*)

Dans cette section, le cas le pire est envisagé pour calculer la grandeur de la file d'attente nécessaire afin qu'il ne se produise aucune perte de cellules suite à un éventuel débordement. Les sources émettent toutes en même temps à plein régime jusquà ce qu'elles n'en aient plus l'autorisation (à cause du GCRA placé sur les connexions VCC). On admet que la file d'attente est vide au départ et que les attributs des connexions VCC sont tous identiques. La file d'attente est supposée être relativement grande et le CDV petit (qui est ignoré pour la suite des calculs). Les attributs de la connexion sont les suivants

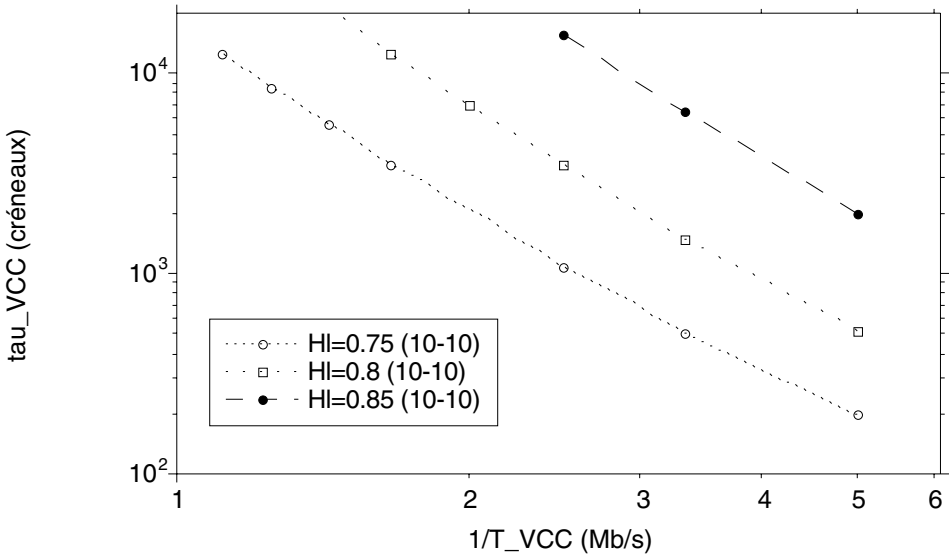

Figure 9.3: τ_{VCC} en fonction de $1/T_{VCC} = SCR_{VCC}$ pour différentes valeurs de paramètres de Hurst local, CLR= 10^{-10}

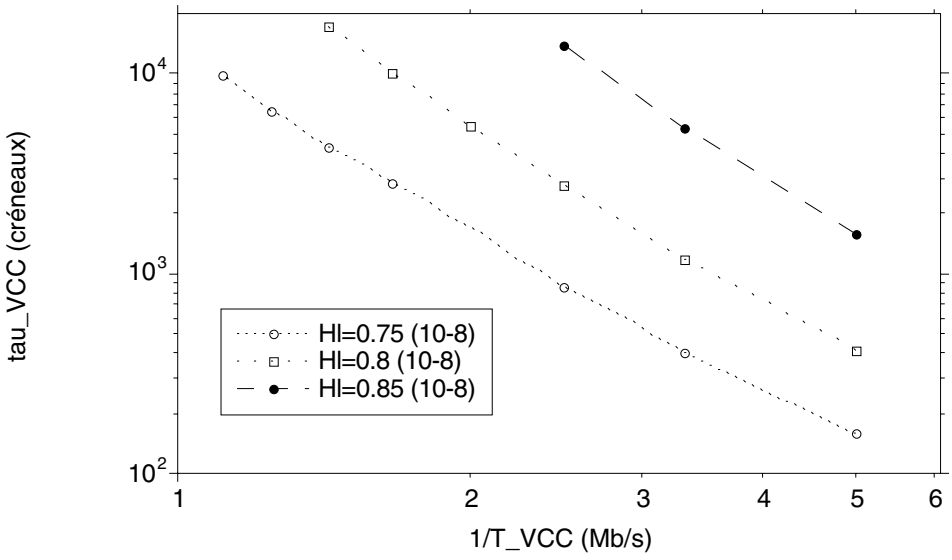

Figure 9.4: τ_{VCC} en fonction de $1/T_{VCC} = SCR_{VCC}$ pour différentes valeurs de paramètres de Hurst local, CLR=10^{-8}

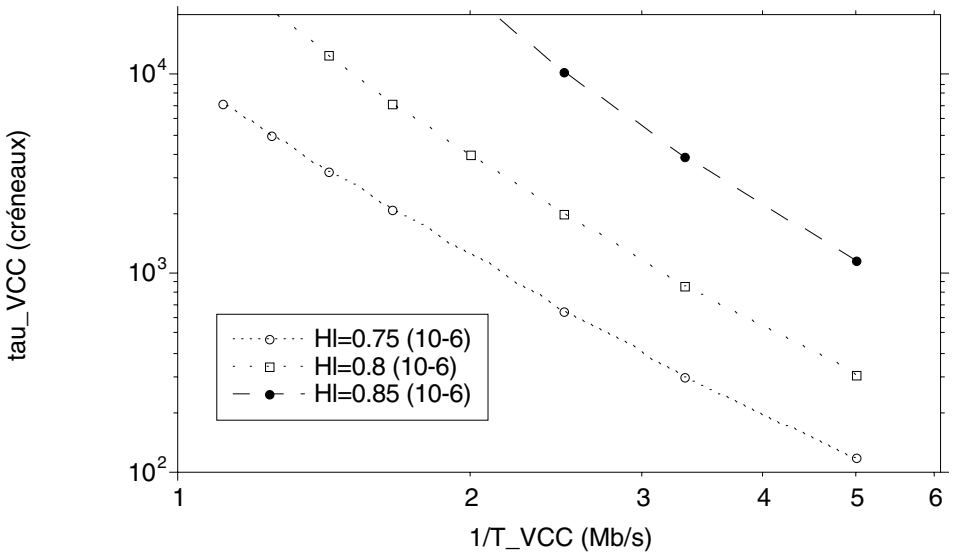

Figure 9.5: τ_{VCC} en fonction de $1/T_{VCC} = SCR_{VCC}$ pour différentes valeurs de paramètres de Hurst local, CLR=10^{-6}

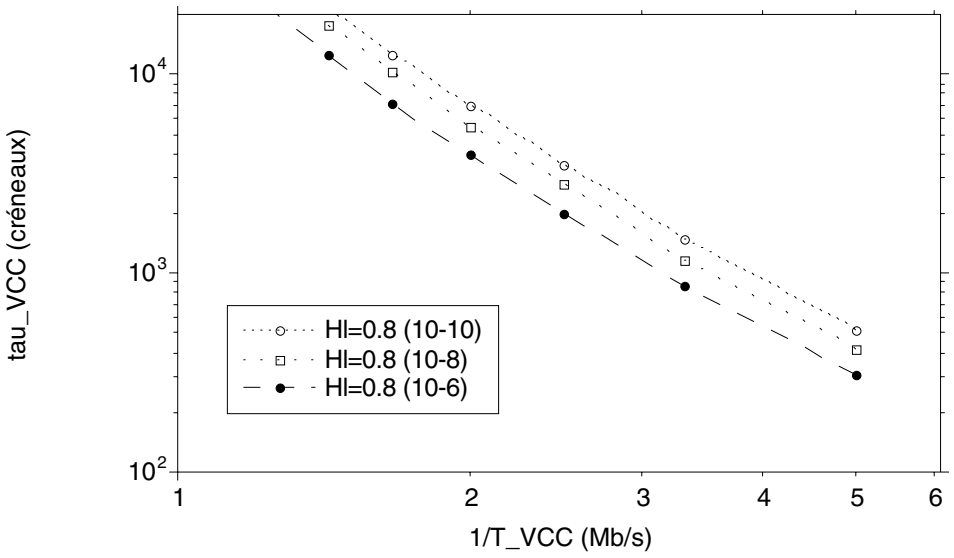

Figure 9.6: BT en fonction de $1/T_{VCC} = SCR_{VCC}$ pour différentes valeurs de CLR, allant de 10^{-6} à 10^{-10}

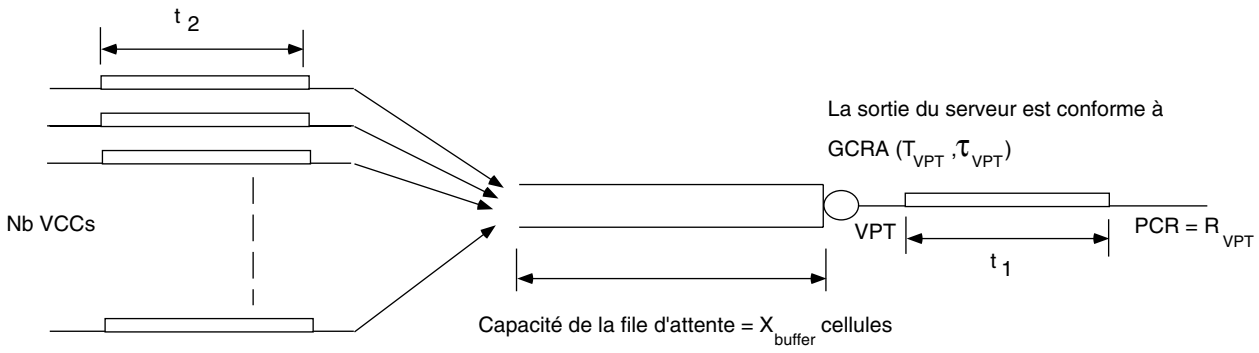

Figure 9.7: Multiplexeur

$CA_{VPT} = (T_{VPT}, \tau_{VPT}, R_{VPT})$ et ceux du *trunk* sont $TA_{VPT} = (Nb, T_{VCC}, \tau_{VCC}, R_{VCC})$. Le trafic de la connexion VPT est lissé par la file d'attente de taille X_{buffer} dont la sortie est conforme au $GCRA(T_{VPT}, \tau_{VPT})$. Le calcul de la taille minimale qu'il faudrait (Req_Buf) pour la file d'attente X_{buffer} se fait en séparant les différentes situations qui peuvent se présenter. Tout d'abord, deux cas sont triviaux, premièrement lorsque $Nb \times 1/T_{VCC} > 1/T_{VPT}$, il faut une file d'attente de longueur infinie, Req_Buf $= \infty$ (Cas 1), et deuxièmement lorsque $Nb \times R_{VCC} < 1/T_{VPT}$ alors il n'y a même pas besoin de file d'attente, Req_Buf= 0 (Cas 2). Outre ces deux cas, examinons ce qui peut se passer. Sur les connexions VCC comme sur la connexion VPT, les longueurs des rafales sont limitées à cause des GCRA. On sait [99] que la longueur d'une rafale est donnée par la formule suivante: $t_{rafale} = 1/R + \tau/(R \times T - 1)$ avec R étant le PCR, $1/T$ le SCR et τ la BT de la connexion. Soit la rafale sur la connexion VPT (t_1) est plus longue que celle de la connexion VCC (t_2), soit c'est l'inverse. D'autre part, il faut considérer le cas où $Nb \times R_{VCC} > R_{VPT}$ et le cas où $Nb \times R_{VCC} < R_{VPT}$. On a ainsi 4 combinaisons à considérer séparement.

- $Nb \times R_{VCC} < R_{VPT}$, $t_1 < t_2$ (Cas 3, figure 9.8)

  La longueur de la rafale t_1 est donnée par

$$t_1 = \frac{1}{Nb \times R_{VCC}} + \frac{\tau_{VPT}}{Nb \times R_{VCC} \times T_{VPT} - 1} \tag{9.1}$$

  et celle de t_2 par

$$t_2 = \frac{1}{R_{VCC}} + \frac{\tau_{VCC}}{R_{VCC} \times T_{VCC} - 1} \tag{9.2}$$

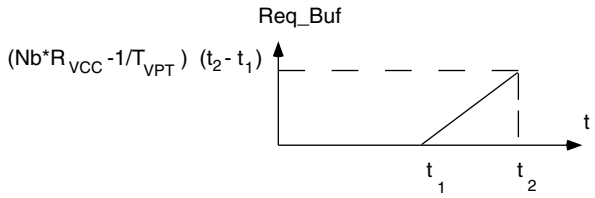

Figure 9.8: Remplissage de la file d'attente quand $Nb \times R_{VCC} < R_{VPT}$, $t_1 < t_2$

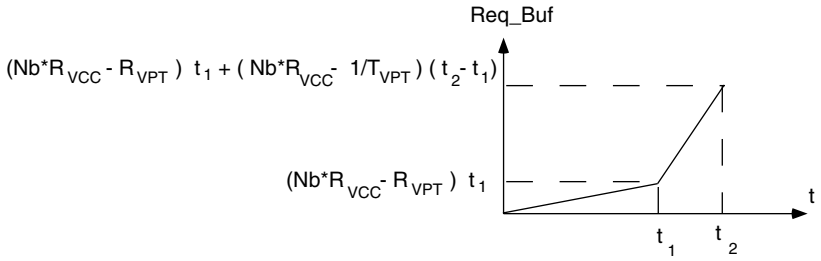

Figure 9.9: Remplissage de la file d'attente quand $Nb \times R_{VCC} > R_{VPT}$, $t_1 < t_2$

Nous en déduisons facilement que

$$Req\_Buf = (Nb \times R_{VCC} - \frac{1}{T_{VPT}})(t_2 - t_1) \tag{9.3}$$

- $Nb \times R_{VCC} < R_{VPT}$, $t_1 > t_2$ (Cas 4)
  $Req\_Buf = 0$

- $Nb \times R_{VCC} > R_{VPT}$, $t_1 < t_2$ (Cas 5, figure 9.9) D'après la figure nous voyons immédiatement que

$$Req\_Buf = (Nb \times R_{VCC} - R_{VPT})t_1 + (Nb \times R_{VCC} - \frac{1}{T_{VPT}})(t_2 - t_1) \tag{9.4}$$

  Si maintenant nous remplaçons $(1 + \tau_{VPT}/(T_{VPT} - 1/R_{VPT}))$ par M_{VPT} alors

$$Req\_Buf = M_{VPT}(\frac{Nb \times R_{VCC}}{R_{VPT}} - 1) + (\frac{M_{VCC}}{R_{VCC}} - \frac{M_{VPT}}{R_{VPT}})(Nb \times R_{VCC} - \frac{1}{T_{VPT}}) \tag{9.5}$$

- $Nb \times R_{VCC} > R_{VPT}$, $t_1 > t_2$ (Cas 6, figure 9.10) D'après la figure 9.10, nous voyons immédiatement que

$$Req\_Buf = (Nb \times R_{VCC} - R_{VPT})(t_2) \tag{9.6}$$

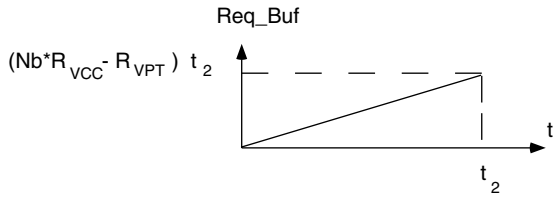

Figure 9.10: Remplissage de la file d'attente quand $Nb \times R_{VCC} > R_{VPT}$, $t_1 > t_2$

Si maintenant nous remplaçons $(1 + \tau_{VCC}/(T_{VCC} - 1/R_{VCC}))$ par M_{VCC} alors il vient

$$Req\_Buf = M_{VCC} \times Nb\left(1 - \frac{R_{VPT}}{Nb \times R_{VCC}}\right) \tag{9.7}$$

9.4.2 Algorithme du cas le pire (*worst case*)

Pour résumer, la fonction qui nous donne la longueur du tampon (*buffer*) nécessaire pour chaque cas qui se présente au multiplexeur est donné par:

Algorithme 9.1 Algorithme Req_Buf

si $Nb \times m_{VCC} > m_{VPT}$ **alors** $Req\_Buf = \infty$ (cas 1)

sinon si $Nb \times R_{VCC} \leq m_{VPT}$ **alors** $Req\_Buf = 0$ (cas 2)

sinon si $Nb \times R_{VCC} \leq R_{VPT}$ **alors**

si $t_1 < t_2$ **alors** $Req\_Buf = (Nb \times R_{VCC} - m_{VPT})(t_2 - t_1)$ (cas 3)

sinon $Req\_Buf = 0$ (cas 4)

sinon si $M_{VCC}/R_{VCC} \geq M_{VPT}/R_{VPT}$ **alors**

$Req\_Buf = M_{VPT}(Nb \times R_{VCC}/R_{VPT} - 1) + (M_{VCC}/R_{VCC} - M_{VPT}/R_{VPT})(Nb \times R_{VCC} - 1/T_{VPT})$ (cas 5)

sinon $Req\_Buf = M_{VCC} \times Nb(1 - R_{VPT}/(Nb \times R_{VCC}))$ (cas 6)

fin

Le tableau 9.1 donne les expressions utilisées dans l'algorithme.

9.4.3 Exemples

Dans ce paragraphe, nous allons examiner le comportement de l'algorithme au travers de quelques exemples. Commençons par examiner la figure 9.11. La première courbe montre l'évolution de Req_Buf en fonction de R_{VPT} lorsque $1/T_{VPT} = SCR_{VPT} = 200$

Expressions utilisées dans l'algorithme
$m_{VPT} = 1/T_{VPT}$
$m_{VCC} = 1/T_{VCC}$
$M_{VPT} = (1 + \tau_{VPT}/(T_{VPT} - 1/R_{VPT}))$
$M_{VCC} = (1 + \tau_{VCC}/(T_{VCC} - 1/R_{VCC}))$
$t_1 = 1/(Nb \times R_{VCC}) + \tau_{VPT}/(Nb \times R_{VCC} \times T_{VPT} - 1)$
$t_2 = 1/R_{VCC} + \tau_{VCC}/(R_{VCC} \times T_{VCC} - 1)$

Tableau 9.1: Expressions utilisées dans l'algorithme

Mb/s. R_{VPT} ne peut évidemment pas être inférieur à 200 Mb/s mais d'autre part $1/T_{VPT}$ ne peut pas être inférieur à 150 Mb/s = $Nb \times 1/T_{VCC}$. Dans la première partie de la courbe, $M_{VPT}/R_{VPT} < M_{VCC}/R_{VCC}$ avec $Nb \times R_{VCC} > R_{VPT}$, ce qui signifie que $t_1 < t_2$. Req_Buf augmente lorsque R_{VPT} diminue. A la limite, lorsque $R_{VPT} = 1/T_{VPT}$, nous nous retrouvons dans le cas où VPT est de classe CBR. Il faut que la file d'attente puisse absorber toutes les rafales (*bursts*) issues des VCC. Dans la deuxième partie de la courbe, $M_{VPT}/R_{VPT} > M_{VCC}/R_{VCC}$ avec $Nb \times R_{VCC} > R_{VPT}$, ce qui signifie que $t_1 > t_2$. Il est intéressant de noter ici que Req_Buf reste presque constant pour n'importe quelle valeur de R_{VPT}. Il est donc inutile d'augmenter R_{VPT} (et par conséquent de gaspiller de la largeur de bande) dès que $t_1 > t_2$ car Req_Buf reste à peu près constant. La limite se situe à $R_{VPT} = 260$ Mb/s dans ce cas. La deuxième courbe nous montre l'évolution de Req_Buf pour le même τ_{VPT} mais pour $1/T_{VPT} = 160$ Mb/s. R_{VPT} pourra commencer plus bas cette fois-ci, à 160 Mb/s. La valeur constante du buffer est plus élevée que dans le cas précédent (32.5 Mb contre 26.5 Mb). La troisième courbe reste à peu près constante lorsque R_{VPT} augmente car $\tau_{VPT} = 0$ sec., donc ce type de connexion est de classe CBR ayant un débit de $1/T_{VPT}$. Comme $R_{VPT} > 1/T_{VPT}$, le changement de R_{VPT} ne change rien au débit de la connexion car à aucun moment, on a l'autorisation de transmettre des cellules à ce débit. La figure 9.12 nous montre l'évolution de Req_Buf en fonction de $1/T_{VPT}$. Il est intéressant de constater que lorsque $R_{VPT} = 500$ Mb/s, Req_Buf= 0 Mb si $1/T_{VPT} \geq 360$ Mb/s, pour $t_1 > t_2$ car τ_{VPT} est suffisamment grand pour absorber les rafales en provenance des VCCs. Ici aussi, nous remarquons qu'à partir de $SCR_{VPT} = 1/T_{VPT} = 370$ Mb/s, l'augmentation de $SCR_{VPT} = 1/T_{VPT}$ est un gaspillage de largeur de bande. Pour les deux courbes suivantes, nous avons également

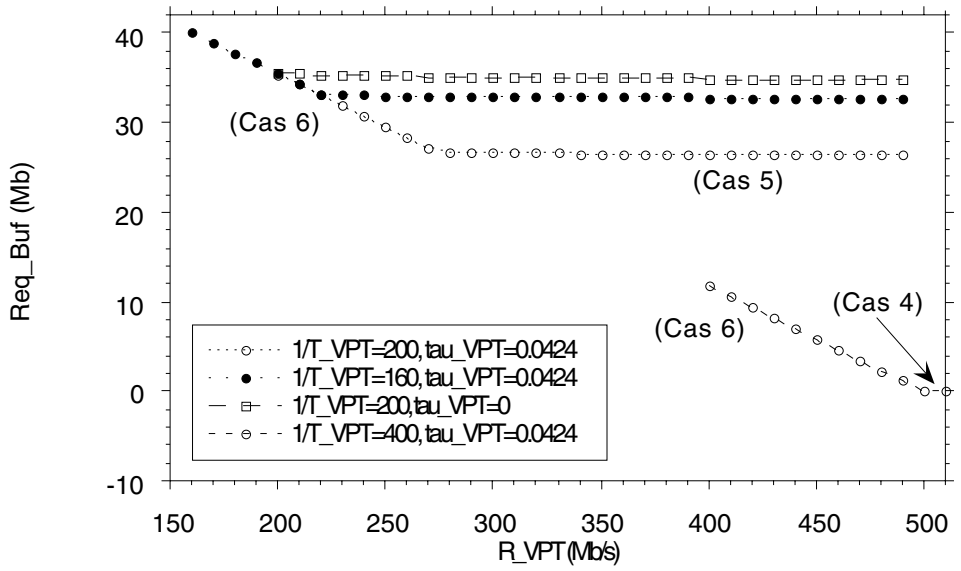

Figure 9.11: Evolution de Req_Buf en fonction R_{VPT} dans le cas le pire avec $T_{VCC} = 1/SCR_{VCC} = 0.333$ s/Mb, R_{VCC}=10 Mb/s, τ_{VCC}=0.0424 sec.

$1/T_{VPT} \leq R_{VPT}$. Notons que pour $1/T_{VPT} = 300$ Mb/s et $R_{VPT} = 300$ Mb/s, τ_{VPT} n'apporte plus rien car VPT peut être considéré comme une connexion de classe CBR. La figure 9.13 nous montre l'évolution de Req_Buf en fonction de τ_{VPT}. Si $R_{VPT} = 500$ Mb/s, alors Req_Buf= 0 Mb pour $\tau_{VPT} > 0.17$ sec. Si $R_{VPT} < 500$ Mb/s, il est intéressant de constater que Req_Buf est nécessairement non nul car la connexion VPT ne peut pas écouler tout le trafic en provenance des connexions VCC et ceci quel que soit τ_{VCC}. Quand $1/T_{VPT}$ diminue, le coude de la courbe (séparant le cas 5 du cas 6) se déplace vers la droite. Autrement dit, si on diminue le SCR sur la connection VPT, il faut compenser par BT si on ne change pas le débit maximal (R_{VPT}) de la connexion. Comme pour les autres figures, nous avons des régions dans lesquelles il n'est pas intéressant d'augmenter τ_{VPT}. Pour la première courbe par exemple, il n'est pas intéressant que $\tau_{VPT} > 0.17$ sec. La figure 9.14 nous montre la surface "minimale" des paramètres τ_{VPT}, R_{VPT} et $1/T_{VPT}$ pour une certaine grandeur de tampon (*buffer*). Si par exemple $X_{buffer} = 0$ Mb alors $R_{VPT} \geq 500$ et les deux degrés de liberté qu'il nous reste sont $1/T_{VPT}$ et τ_{VPT}. Si nous voulons que $\tau_{VPT} = 0$, alors $1/T_{VPT} > 500$. Par contre, si nous voulons que $1/T_{VPT} = 170$, il faudra que $\tau_{VPT} > 0.22$. Nous remarquons que l'augmentation de R_{VPT} au-delà de 500 Mb/s ne sert à rien. Cette dernière figure conclut ce paragraphe.

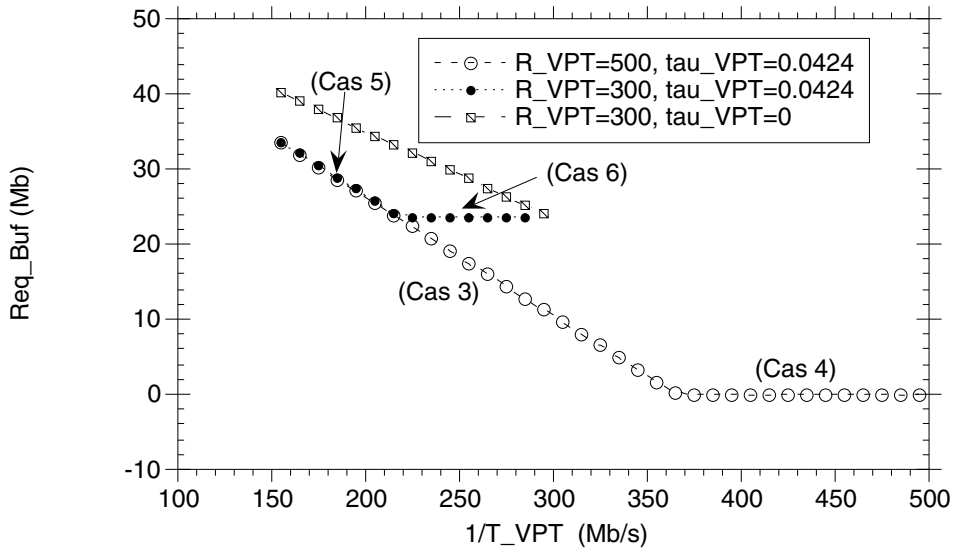

Figure 9.12: Evolution de Req_Buf en fonction $1/T_{VPT} = SCR_{VPT}$ dans le cas le pire avec $T_{VCC} = 1/SCR_{VCC} = 0.333$ s/Mb, R_{VCC}=10 Mb/s, τ_{VCC}=0.0424 sec.

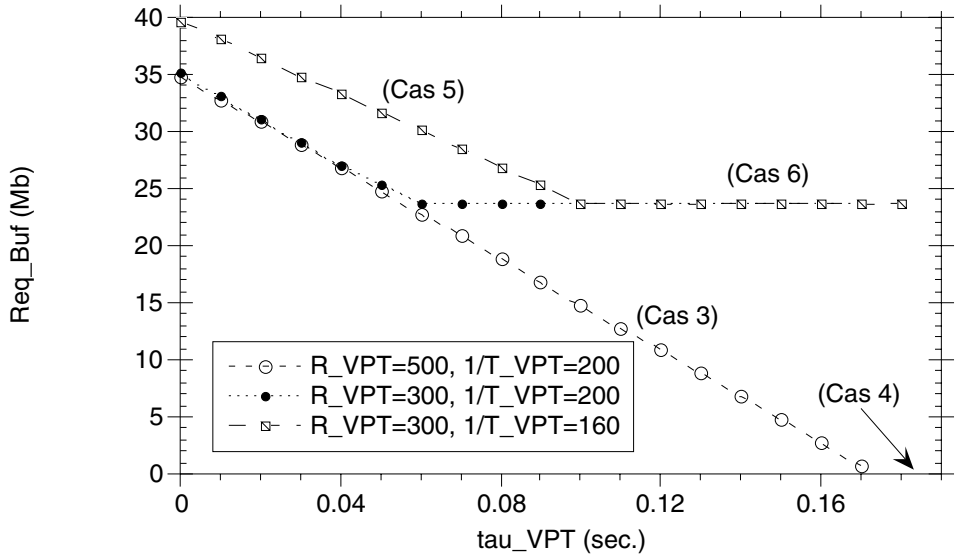

Figure 9.13: Evolution de Req_Buf en fonction τ_{VPT} dans le cas le pire avec $T_{VCC} = 1/SCR_{VCC} = 0.333$ s/Mb, R_{VCC}=10 Mb/s, τ_{VCC}=0.0424 sec.

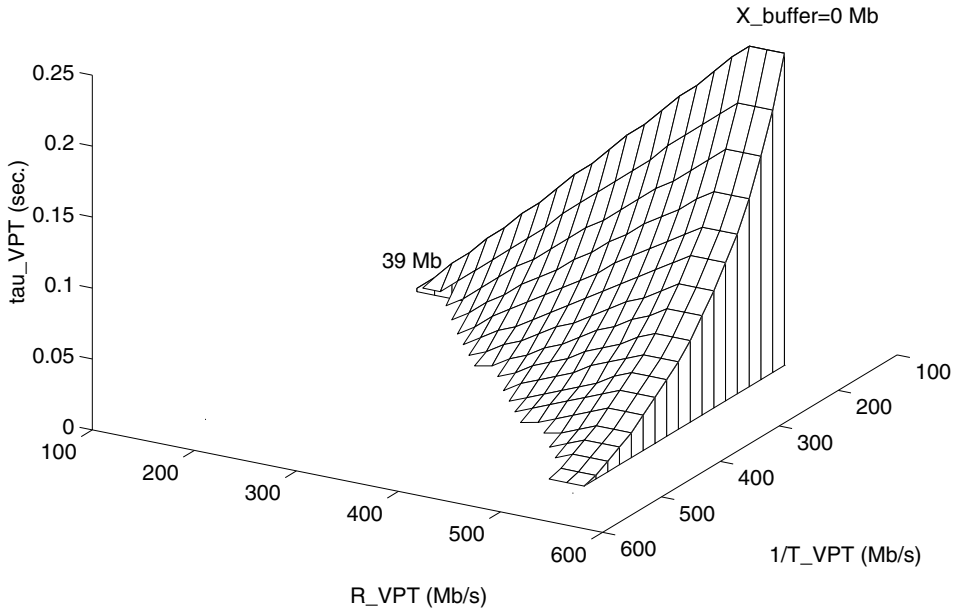

Figure 9.14: Surface "minimale" des paramètres R_{VPT}, $1/T_{VPT} = SCR_{VPT}$ et τ_{VPT} pour une grandeur de tampon donnée

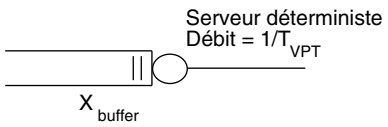

Figure 9.15: Modèle du multiplexeur lorsque la connexion VPT est de classe CBR

Au travers des exemples que nous avons vus, il apparaît clairement que des ajustements de paramètres peuvent être mauvais. Pour bien faire, il faudrait toujours se situer sur la surface qui est décrite par la figure 9.14.

9.4.4 Modélisation du multiplexeur

En ce qui concerne la perte de cellules, il est possible de modéliser le multiplexeur de la manière suivante: lorsque la connexion VPT est de classe CBR, le serveur de la file d'attente est un serveur déterministe (figure 9.15), par contre, lorsque la connexion est de classe VBR, le modèle se complique un petit peu. Nous avons une succession de deux files d'attente dont le débit du premier serveur est égale au débit de la connexion VPT,

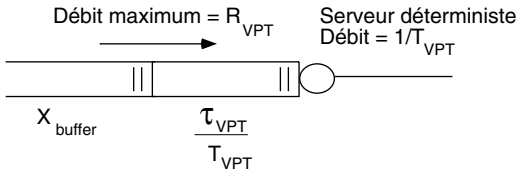

Figure 9.16: Modèle du multiplexeur lorsque la connexion VPT est de classe VBR

à savoir R_{VPT} et dont le débit du deuxième serveur est égal au SCR de la connexion VPT, à savoir $1/T_{VPT}$. La taille de la deuxième file d'attente est approximativement égale à τ_{VPT}/T_{VPT} (figure 9.16). La vitesse de transfert des cellules de la première file d'attente (file d'attente "physique" de taille X_{buffer}) à la seconde (file d'attente virtuelle) est limitée par le débit maximum de la connexion VPT. Quelle est la signification de ceci ? Nous remarquons que lorsque le débit instantané est inférieur au débit maximum (R_{VPT}) de la connexion VPT, alors le gain d'une connexion de classe VBR par rapport à une connexion de classe CBR est qu'elle offre une file d'attente virtuelle de longueur approximativement égale à τ_{VPT}/T_{VPT} en plus de la file d'attente physique X_{buffer}. Mais par contre si le débit instantané peut être plus important que le débit maximum de la connexion VPT, le gain de la classe VBR par rapport à la classe CBR peut être nul, ça va dépendre du trafic. Imaginons par exemple un trafic sporadique ou si les Nb sources à l'entrée du multiplexeur sont fortement corrélées par exemple. Il est donc important de constater qu'on y gagne beaucoup avec une connexion VPT de classe VBR lorsque le débit instantané des sources est inférieur au débit maximum de la connexion VPT mais le débit maximum des sources peut par contre être plus élevé, en d'autres termes $Nb \times R_{VCC}$ peut être supérieur à R_{VPT}.

9.4.5 VPT de classe CBR

Les résultats des simulations lorsque VPT est de classe CBR sont montrés à la figure 9.17 et à la figure 9.18. Pour faire ces simulations, nous avons envisagé la situation suivante: $Nb = 50$ sources de classes VBR entrent dans le multiplexeur. Leur débit maximum est de 10 Mb/s ($R_{VCC} = 10$ Mb/s). Les sources sont toutes identiques ($n = 5$, $H_l = 0.75 - 0.85$, 0.5 Mb/s). Pour la simulation, cinq séries de 10^7 créneaux ont été

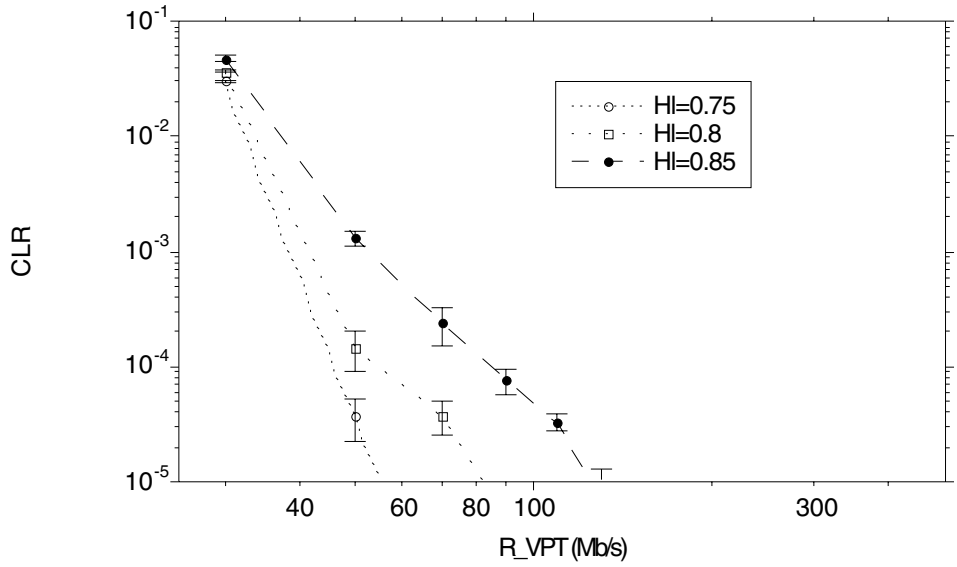

Figure 9.17: CLR du multiplexeur pour une file d'attente de 500 places, int. de conf. 95%, $Nb = 50$ sources ($n = 5$, 0.5 Mb/s), la connexion VPT est de classe CBR

générées pour chaque point. La figure 9.17 montre que si nous acceptons un taux de pertes de 10^{-5} dans la file d'attente X_{buffer}, il est possible d'obtenir un gain de multiplexage statistique d'environ 5-6 pour des sources markoviennes ayant des paramètres de Hurst local, $H_l = 0.8$. Par contre si nous extrapolons la courbe obtenue et qu'on exige un taux de pertes de 10^{-8}, alors le gain n'est plus que de 2.5 environ. La situation est évidemment pire avec des sources markoviennes ayant des paramètres de Hurst local plus élevés. La figure 9.18 montre que pour un taux de pertes exigé, ici 10^{-5}, l'augmentation de la longueur de la file d'attente n'est pas tellement avantageuse, surtout pour des sources markoviennes ayant des paramètres de Hurst local élevés. L'étude est limitée à des files d'attente relativement courtes, ce qui limite la généralisation de cette affirmation.

9.4.6 VPT de classe VBR

La simulation s'est faite de la même façon qu'au paragraphe 9.4.5 mais ici la source est la même pour toutes les simulations ($n = 5$, $H_l = 0.8$, 0.5 Mb/s), leur débit maximum est de 10 Mb/s ($R_{VCC} = 10$ Mb/s). Pour la simulation, cinq séries de 10^7 créneaux ont été générées pour chaque point. La première comparaison que nous pouvons faire à partir

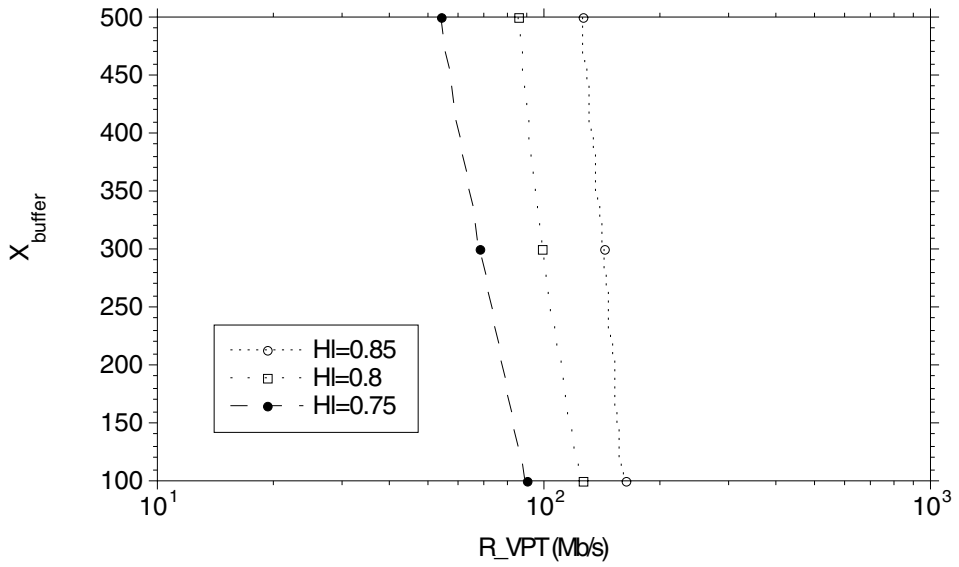

Figure 9.18: Grandeur de la file d'attente (cellules) en fonction de R_{VCC}, CLR=10^{-5}

de la figure 9.19 est celle des deux premières courbes. Nous remarquons que lorsque τ_{VPT}, en nombre de créneaux, reste constant, le taux de perte (CLR) augmente si R_{VPT} augmente, ce qui est logique car ici τ_{VPT} est exprimé en créneaux et pas en secondes. La deuxième comparaison que nous voulons faire est celle des deux dernières courbes qui sont pratiquement identiques. La première courbe indique le taux de pertes (CLR) dans la file d'attente X_{buffer} lorsque celle-ci a 500 places et que $\tau_{VPT} = 1000$ créneaux (=2.83 ms si $R_{VPT} = 150$ Mb/s) et la seconde indique le taux de pertes dans la file d'attente X_{buffer} lorsque celle-ci a 1000 places et que $\tau_{VPT} = 0$ créneau. Les cinq traces pour la génération de trafic sont les mêmes pour les deux cas (la racine de chacun des cinq générateurs de nombres aléatoires est identique pour les deux séries des simulations). Ainsi la comparaison en est facilitée puisque ce ne sont que les différences inhérentes au mécanisme de contrôle qui apparaissent et non des différences dûes aux générateurs de nombres aléatoires. Ces deux courbes sont presque identiques, ce qui indique que dans le cadre de nos simulations, l'approximation du service VBR par une file d'attente supplémentaire est bonne. R_{VPT} a été arbitrairement choisi deux fois plus grand que $1/T_{VPT}$. Le résultat de ci-dessus indique que la trafic issu du multiplexeur ne dépasse pas souvent cette valeur mais il ne faut pas perdre de vue que nous avons considéré Nb sources indépendantes, non corrélées entre elles. On peut deviner que le résultat ne serait

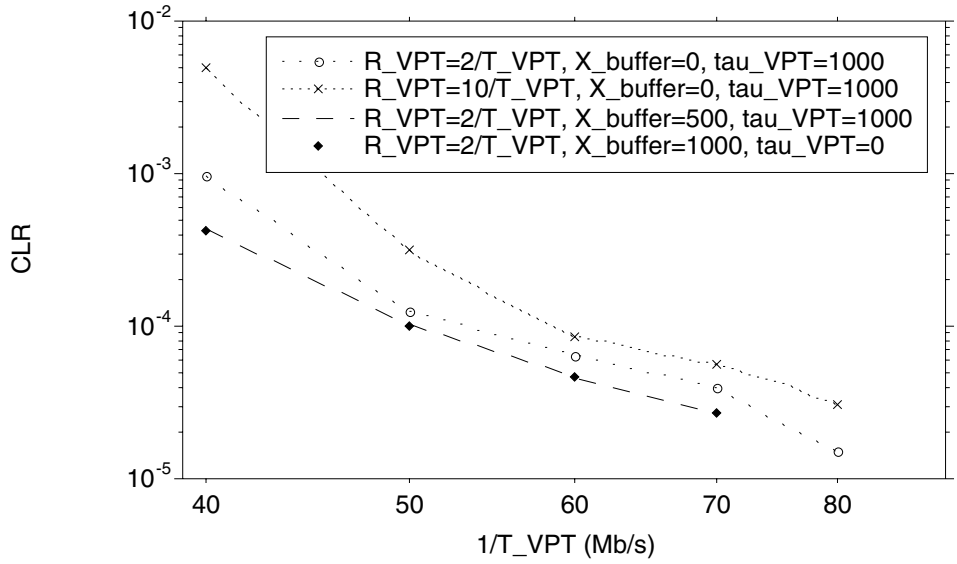

Figure 9.19: CLR du multiplexeur pour une file d'attente de 500 places, $Nb = 50$ sources ($n = 5$, $H_l = 0.8$, 0.5 Mb/s), la connexion VPT est de classe VBR

pas été le même si elles l'avaient été.

Chapitre 10

Conclusion

Le chapitre 1 présente le contexte actuel dans lequel a été effectué cette thèse et le débat animé que les récentes découvertes des chercheurs de Bellcore a suscité. Au chapitre 2, ATM et Ethernet sont très brièvement introduits.

Au chapitre 3, le modèle markovien (SSMP), très général, que nous utilisons tout au long de la thèse est présenté ainsi que ses caractéristiques. La résolution du problème de la file d'attente est tout à fait intéressante. L'algorithme MBH est utilisé à cette fin et permet de calculer très efficacement la probabilité de perte et la probabilité d'occupation de la file d'attente.

Au chapitre 4, des résultats de mesures sur les réseaux de l'EPFL et de Bellcore sont exposés. Les mesures sur le réseau de l'EPFL ont été effectuées à l'aide d'un logiciel développé dans les laboratoires LBL à Berkeley en Californie, *Tcpdump* tandis que les mesures de Bellcore ont été effectuées à l'aide d'un mesureur dédié à cette tâche. Plusieurs métriques ont été évaluées, l'indice de dispersion, le coefficient de variation et le paramètre de Hurst par la méthode des variances. Le trafic mesuré est très variable au cours du temps, sur de très longues périodes.

Au chapitre 5, une méthode pour interconnecter Ethernet à DQDB en mode message est présentée (ça signifie que la trame Ethernet doit être complètement arrivée au noeud DQDB avant de pouvoir être transmise sur le réseau). Pour ce type d'interconnexion, des formules pour l'évaluation du délai et de l'écart-type de ce dernier ont été élaborées en considérant le cas le pire pour le trafic s'écoulant sur Ethernet. Le cas le pire sur Ethernet

considère toutes les cellules (après conversion de trafic) collées les unes aux autres dont les rafales (*bursts*) sont de longueur variable. L'évaluation de la performance de cette interconnexion montre que le délai moyen d'accès au bus DQDB dépend fortement de la distance entre la station et le générateur de créneaux alors que l'écart-type du retard y est insensible. Des formules simples permettant de calculer le temps moyen et l'écart-type du retard de l'accès au bus sont données.

Au chapitre 6, nous avons considéré les paramètres qui avaient beaucoup d'influence sur le comportement de la file d'attente, à savoir la moyenne de la différence entre les arrivées et les départs dans la file d'attente, de la variance et de la densité spectrale autour de zéro de cette différence. Des exemples numériques illustrent l'importance de ces paramètres. La connaissance de ces paramètres permet de réduire grandement une chaîne de Markov modulée produisant du trafic destiné à être stocké dans une file d'attente (comme c'est le cas dans le multiplexage par exemple) si nous nous intéressons à l'occupation et à la perte dans cette dernière. Nous avons appliqué cette méthode à du trafic mesuré sur le réseau de l'EPFL pour trouver un modèle de trafic. La comparaison entre la moyenne des pertes du modèle et des mesures dans la file d'attente est encourageante mais le nombre de paramètres à manipuler est très important. Une méthode d'optimisation efficace, le *Tabu Search*, a été utilisée à cet effet.

Au chapitre 7, nous comparons du trafic mesuré (à l'EPFL et à Bellcore) avec différents modèles normalement utilisés pour la représentation du trafic de données. Les mesures faites sur les deux réseaux Ethernet ont montré une très grande variabilité du trafic en comparaison avec les modèles conventionnels. En nous basant sur la théorie de la décomposabilité de Courtois, nous trouvons qu'un modèle markovien modulé à cinq états est capable de reproduire une grande variabilité du trafic sur un intervalle de temps raisonnable (de 20 ms à plus de 20 minutes). Nous avons évalué et comparé l'indice de dispersion, le coefficient de variation, le paramètre de Hurst local du modèle markovien modulé à cinq états avec les mesures de Bellcore, un modèle poissonnien et un modèle de source ON-OFF.

Au chapitre 8, nous introduisons le concept de pseudo-dépendances à long terme pour tenir compte de l'intervalle fini sur lequel l'auto-similarité est observée. Le nouveau

modèle présenté est capable de produire un comportement auto-similaire sur un intervalle de temps fini. A la différence de celui décrit au chapitre 7, celui-ci ne dépend que de trois paramètres et est aisément manipulable. L'avantage des modèles markoviens est qu'il est possible de réutiliser toutes les techniques bien connues de la théorie des files d'attente développées dans le passé pour évaluer la performance des réseaux de communication. Une méthode quantitative, basée sur la théorie de la décomposabilité de Courtois est donnée pour évaluer le domaine sur lequel le processus a un comportement auto-similaire.

Au chapitre 9, nous nous intéressons au problème du multiplexage statistique. L'architecture de base (SCONE) sur laquelle sont faites les études de performances est novatrice et nécessite une étude préliminaire pour savoir quel est le gain qu'on peut espérer obtenir en faisant du multiplexage statistique. Une étude préliminaire considérant le cas le plus défavorable (*worst case*) a été menée pour bien comprendre le système et pour dégager des principes importants du multiplexage de connexions de classe VBR sur une connexion de classe VBR.

Pour résumer, nous avons, au cours de ce long périple, essayé de répondre à la question suivante: "la modélisation markovienne a-t-elle encore un sens dans l'analyse du trafic des réseaux informatiques? ". Nous y avons répondu par l'affirmative suite à un certain nombre d'études. Le modèle markovien qui a été trouvé ne dépend que de trois paramètres et permet de représenter du trafic ayant les caractéristiques du trafic mesuré par les chercheurs de Bellcore. Ce modèle trouvé a le grand avantage d'être simple et facilement manipulable. Il est ainsi possible de réutiliser les théories des files d'attente qui ont été développées dans le passé pour évaluer la performance des réseaux de communication.

Chapitre 11

Annexes

11.1 Calcul de la probabilité d'occupation de la file d'attente

Pour le calcul de la probabilité d'occupation de la file d'attente SSMP/G/1/c, nous allons tout d'abord séparer deux cas distincts, lorsque la file est vide ($d = 0$) et lorsqu'elle ne l'est pas ($d \geq 1$)

pour $d \geq 1$

$$
\begin{aligned}
prob(N_{t+1} \; & = min(d,c) \cap R_{t+1} = r \cap Y_{t+1} = j | N_0 = 0 \cap Y_0 = i) = \\
& \textstyle\sum_{s=1}^{n} \{ \sum_{k=1}^{min(d,c)} prob((N_t = k \cap R_t = r+1 \cap Y_t = s | N_0 = 0 \cap Y_0 = i) \cap \\
& (X_t = d - k | Y_t = s) \cap (Y_{t+1} = j | Y_t = s)) + \\
& \textstyle\sum_{k=1}^{min(d+1,c)} prob((N_t = k \cap R_t = 1 \cap Y_t = s | N_0 = 0 \cap Y_0 = i) \cap \\
& (X_t = d - k + 1 | Y_t = s) \cap (Y_{t+1} = j | Y_t = s) \cap (H = r)) + \\
& prob((N_t = 0 \cap Y_t = s | N_0 = 0 \cap Y_0 = i) \cap \\
& (X_t = d | Y_t = s) \cap (Y_{t+1} = j | Y_t = s) \cap (H = r)) \}
\end{aligned}
\tag{11.1}
$$

$d = 0$

$$\begin{aligned} prob(N_{t+1} &= 0 \cap Y_{t+1} = j | N_0 = 0 \cap Y_0 = i) = \\ &\sum_{s=1}^{n} \{prob((N_t = 1 \cap R_t = 1 \cap Y_t = s | N_0 = 0 \cap Y_0 = i) \cap \\ &(X_t = 0 | Y_t = s) \cap (Y_{t+1} = j | Y_t = s)) + \\ &prob((N_t = 0 \cap Y_t = s | N_0 = 0 \cap Y_0 = i) \cap \\ &(X_t = 0 | Y_t = s) \cap (Y_{t+1} = j | Y_t = s))\} \end{aligned} \tag{11.2}$$

en introduisant temporairement les notations suivantes

$$\begin{aligned} prob_{ij} \quad & (min(d,c), r, t+1) \\ & \equiv prob(N_{t+1} = min(d,c) \cap R_{t+1} = r \cap Y_{t+1} = j | N_0 = 0 \cap Y_0 = i) \end{aligned}$$

$$prob_{ij}(0, t+1) \equiv prob(N_t = 0 \cap Y_t = j | N_0 = 0 \cap Y_0 = i)$$

Ainsi les équations (11.2) et (11.1) peuvent s'écrire

$d \geq 1$

$$\begin{aligned} prob_{ij} \quad & (min(d,c), r, t+1) = \sum_{s=1}^{n} \{\sum_{k=1}^{min(d,c)} \\ & prob_{is}(k, r+1, t) \phi_{s,d-k} a_{sj} + h_r \sum_{k=1}^{min(d+1,c)} p_{is}(k, 1, t) \phi_{s,d-k+1} a_{sj} + \\ & h_r prob_{is}(0, t) \phi_{sd} a_{sj}\} \end{aligned} \tag{11.3}$$

$d = 0$

$$prob_{ij}(0, t+1) = \sum_{s=1}^{n} (p_{is}(1, 1, t) + prob_{is}(0, t)) \phi_{s0} a_{sj} \tag{11.4}$$

Nous supposons l'existence de probabilités stationnaires pour $t \to \infty$, ça signifie que la chaîne de Markov est homogène et irréductible.

$d \geq 1$

$$\begin{aligned} \lim_{t \to \infty} \quad & (prob_{ij}(min(d,c), r, t+1)) = prob_j(min(d,c), r) = \\ & \sum_{s=1}^{n} \{\sum_{k=1}^{min(d,c)} prob_s(k, r+1) \phi_{s,d-k} a_{sj} + \\ & h_r \sum_{k=1}^{min(d+1,c)} p_s(k, 1) \phi_{s,d-k+1} a_{sj} + \\ & h_r prob_s(0) \phi_{sd} a_{sj}\} \end{aligned} \tag{11.5}$$

$d = 0$

$$\lim_{t \to \infty} (prob_{ij}(0, t+1)) = prob_j(0) = \sum_{s=1}^{n} (p_s(1, 1) + prob_s(0)) \phi_{s0} a_{sj} \tag{11.6}$$

Il est important de noter que ces limites sont indépendantes de l'état initial de la chaîne de Markov. Pour la suite, nous admettons que le temps résiduel de service est indépendant de l'occupation du système, ça signifie que pour $r \geq 1$, $prob_j(k,r) = prob_j(k)prob(R=r)$. Mais rappelons que $r > 0$ s'il y a au moins une cellule dans le système de file d'attente et que $\sum_{r=1}^{\infty} prob(R=r) = 1$. Ainsi les équations (11.5) et (11.6) deviennent

$d \geq 1$

$$\begin{aligned} prob_j \quad & (min(d,c)) = \sum_{s=1}^{n}\{\sum_{k=1}^{min(d,c)} prob_s(k)\phi_{s,d-k}a_{sj}(1-prob(R=1))+ \\ & \sum_{k=1}^{min(d+1,c)} p_s(k)\phi_{s,d-k+1}a_{sj}prob(R=1) + prob_s(0)\phi_{sd}a_{sj}\} \end{aligned} \tag{11.7}$$

$d = 0$

$$prob_j(0) = \sum_{s=1}^{n}(prob(R=1)p_s(1) + prob_s(0))\phi_{s0}a_{sj} \tag{11.8}$$

Introduisons la notation suivante

$$\vec{P} = (\vec{P_0}\ \vec{P_1}\ \cdots\ \vec{P_c}) \tag{11.9}$$

avec $\vec{P_i}$, $i = 1, 2, \cdots, c$ étant un vecteur horizontal de n éléments

$$\vec{P} = (P_{01}\ P_{02}\ \cdots\ P_{0n}\ P_{11}\ P_{12}\ \cdots P_{1n}\ \cdots\ P_{c1}\ Pc2\ \cdots\ Pcn) \tag{11.10}$$

$\vec{P}$ représente le vecteur des probabilités stationnaires du système de la file d'attente avec $P_{ij} = prob(N_t = i | Y_t = j)$ et $\mathbf{Q}$ représente la matrice de transition du même système. Les équations du système de file d'attente (équations (11.7) et (11.8)) peuvent se mettre sous la forme suivante: $\vec{P} = \vec{P}\mathbf{Q}$ avec

$$\mathbf{Q} = \begin{pmatrix} \mathbf{B}_0 & \mathbf{B}_1 & \mathbf{B}_2 & \cdots & \cdots & \mathbf{B}_{c-1} & \sum_{l\geq c}\mathbf{B}_l \\ \mathbf{H}_0 & \mathbf{H}_1 & \mathbf{H}_2 & \cdots & \cdots & \mathbf{H}_{c-1} & \sum_{l\geq c}\mathbf{H}_l \\ \mathbf{0} & \mathbf{H}_0 & \mathbf{H}_1 & \cdots & \cdots & \mathbf{H}_{c-2} & \sum_{l\geq c-1}\mathbf{H}_l \\ \mathbf{0} & \mathbf{0} & \mathbf{H}_0 & \cdots & \cdots & \mathbf{H}_{c-3} & \sum_{l\geq c-2}\mathbf{H}_l \\ \vdots & & & & \cdots & & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & & \mathbf{H}_0 & \sum_{l\geq 1}\mathbf{H}_l \end{pmatrix} \tag{11.11}$$

et

$$\begin{aligned} (\mathbf{B}_k)_{ij} = \phi_{ik}a_{ik} \qquad & 0 \leq k \leq c-1 \\ & 1 \leq i,j \leq n \end{aligned} \tag{11.12}$$

$$(\sum_{l\geq c}\mathbf{B}_l)_{ij} = a_{ij} - \sum_{k=0}^{c-1}(\mathbf{B}_k)_{ij} \qquad 1 \leq i,j \leq n \tag{11.13}$$

$$(\mathbf{H}_k)_{ij} = a_{ij}(\gamma_1\phi_{ik} + (1-\gamma_1)\phi_{i,k-1} \qquad \begin{array}{l} 0 \leq k \leq c-1 \\ 1 \leq i,j \leq n \end{array} \tag{11.14}$$

$$(\sum_{l\geq c}\mathbf{H}_l)_{ij} = a_{ij} - \sum_{k=0}^{c-1}(\mathbf{H}_k)_{ij} \qquad 1 \leq i,j \leq n \tag{11.15}$$

11.2 Calcul de la moyenne et de la variance

Pour trouver la formule (5.8), nous allons partir de la formule (5.4). Par définition, la moyenne (μ_u) est donnée par

$$\mu_u = \lim_{N'\to\infty} \frac{1}{N'} \sum_{k=1}^{N'} W_k \tag{11.16}$$

mais si nous admettons que la file d'attente est vide chaque fois que le premier créneau de la rafale arrive, alors il est suffisant de ne considérer qu'une seule rafale.

$$\mu_u = \frac{1}{n_c} \sum_{k=0}^{n_c-1} W_k \tag{11.17}$$

or nous connaissons W_k

$$W_k = S_0 + \sum_{j=1}^{k-1} S_j - kT_b \tag{11.18}$$

et $U_k = W_k + S_k$ donc

$$U_k = S_0 + \sum_{j=1}^{k} S_j - kT_b \tag{11.19}$$

et

$$\mu_u = \frac{1}{n_c} \sum_{k=0}^{n_c-1} \left(S_0 + \sum_{j=1}^{k} S_j - kT_b \right) \tag{11.20}$$

$$= S_0 + \frac{1}{n_c} \sum_{k=0}^{n_c-1} (\sum_{j=1}^{k} S_j - kT_b) \tag{11.21}$$

$$= S_0 + \frac{1}{n_c} \sum_{k=0}^{n_c-1} (kE[S] - kT_b) \tag{11.22}$$

mais comme $\sum_{i=1}^{n_c-1} i = \frac{n_c}{2}(n_c-1)$, il vient

$$\mu_u = S_0 + \frac{(n_c-1)}{2}(E[S] - T_b) \tag{11.23}$$

Pour trouver la formule (5.9), les développements sont presque identiques à ceux que nous avons faits pour trouver la moyenne. Nous allons partir de la formule (5.4). Par définition, la variance (σ_u^2) est donnée par

$$\sigma_u^2 = \lim_{N' \to \infty} \frac{1}{N'} \sum_{k=1}^{N'} (W_k)^2 \tag{11.24}$$

mais si nous admettons que la file d'attente est vide chaque fois que le premier créneau de la rafale arrive, alors il est suffisant de ne considérer qu'une seule rafale.

$$\mu_u^2 = \frac{1}{n_c} \sum_{k=0}^{n_c-1} (W_k)^2 \tag{11.25}$$

avec μ^k étant le k$^{\text{e}}$ moment. Or nous connaissons W_k

$$W_k = S_0 + \sum_{j=1}^{k-1} S_j - kT_b \tag{11.26}$$

et $U_k = W_k + S_k$ donc

$$U_k = S_0 + \sum_{j=1}^{k} S_j - kT_b \tag{11.27}$$

et

$$\mu_u^2 = \frac{1}{n_c} \sum_{k=0}^{n_c-1} \left(S_0 + \sum_{j=1}^{k} S_j - kT_b \right)^2 \tag{11.28}$$

avec $\left(\sum_{j=1}^{k} S_j\right)^2 = k(E[S^2] - E[S]^2) + k^2 E[S]^2$ et en sachant que $\sum_{i=1}^{n_c-1} i = (n_c/2)(n_c - 1)$ et que $\sum_{i=1}^{n_c-1} i^2 = (n_c/6)(n_c - 1)(2n_c - 1)$, nous montrons facilement que

$$\sigma_u^2 = \frac{(n_c^2 - 1)}{12}(E[S] - T_b)^2 + \frac{(n_c - 1)}{2} Var^2[S] \tag{11.29}$$

11.3 Calcul des moments du MMPP(2)

Soit X est une variable aléatoire à valeurs entières non négatives, $X \in \mathbb{N}$, alors sa fonction génératrice est définie par

$$f(z) = E[z^X] = \sum_{k=0}^{\infty} prob(X = k) z^k \tag{11.30}$$

où $k \in \mathbb{N}$ et où z est une variable complexe. Nous savons que la loi de probabilité est définie de manière unique par sa fonction génératrice associée. Pour la distribution de Poisson,

$$f(z) = \sum_{k=0}^{\infty} \frac{\alpha}{k!} e^{-\alpha} z^k = e^{-\alpha} e^{\alpha z} = e^{\alpha(z-1)} \tag{11.31}$$

en dérivant, nous obtenons

$$f'(z) = \sum_{k=0}^{\infty}(k)prob(X = k)z^{k-1} \tag{11.32}$$

$$f''(z) = \sum_{k=0}^{\infty} k(k-1)prob(X = k)z^{k-2} = \sum_{k=0}^{\infty}(k^2 - k)prob(X = k)z^{k-2} \tag{11.33}$$

$$f'''(z) = \sum_{k=0}^{\infty} k(k-1)(k-2)prob(X = k)z^{k-3} = \sum_{k=0}^{\infty}(k^3 - 3k^2 + 2k)prob(X = k)z^{k-3} \tag{11.34}$$

en posant $z = 1$, nous voyons immédiatement que $f'(1) = \alpha$, $f''(1) = \alpha^2$, $f'''(1) = \alpha^3$. A partir, de là il est très facile de déterminer les différents moments de la loi de Poisson, $E[X] = \alpha$, $E[X^2] = \alpha + \alpha^2$, $E[X^3] = \alpha^3 + \alpha + 3\alpha^2$. Avec notre processus,

$$E[X] = \frac{1}{p+q}(q\alpha_1 + p\alpha_2) \tag{11.35}$$

$$E[X^2] = \frac{1}{p+q}(q(\alpha_1 + \alpha_1^2) + p(\alpha_2 + \alpha_2^2)) \tag{11.36}$$

$$E[X^2] = E[X] + \frac{1}{p+q}(q\alpha_1^2 + p\alpha_2^2) \tag{11.37}$$

$$E[X^3] = \frac{1}{p+q}(q(\alpha_1^3 + 3\alpha_1 + \alpha_1) + q(\alpha_2^3 + 3\alpha_2 + \alpha_2)) \tag{11.38}$$

$$E[X^3] = -2E[X] + 3E[X^2] + \frac{1}{p+q}(q\alpha_1^3 + p\alpha_2^3) \tag{11.39}$$

Maintenant calculons $\Phi(0)$. Pour une chaîne de Markov de type SSMP à deux états, la fonction d'autocovariance est donnée par $C(k) = \beta_2\lambda_2^k$ (section 3.4), le facteur constant β_2 vaut, après quelques calculs

$$\beta_2 = \vec{\pi}\mathbf{\Lambda}\mathbf{G}^{-1}\mathbf{P}_2\mathbf{G}\mathbf{\Lambda}\vec{e} \tag{11.40}$$

$$\beta_2 = \frac{pq}{(p+q)^2}(\alpha_1 - \alpha_2)^2 \tag{11.41}$$

et comme $\sum_{k=0}^{\infty}(1-p-q)^k = 1/(p+q)$, nous déduisons

$$\Phi(0) = \beta_2(\frac{2}{p+q} - 1) = \frac{pq}{(p+q)^2}(\frac{2}{p+q} - 1)(\alpha_1 - \alpha_2)^2 \tag{11.42}$$

11.4 SMP

Le processus SMP est plus général que le processus SSMP; ce n'est plus à un état qu'on associe une loi de probabilité mais à un état du modulateur et à celui qui l'a précédé. Lorsque le modulateur se trouve dans l'état i à l'instant t et était dans l'état j à l'instant $t-1$, la probabilité que la chaîne de Markov génère k cellules est $prob(X_t = k|Y_t = i \cap Y_{t-1} = j) \neq prob(X_t = k|Y_t = i)$. L'autocovariance d'un tel processus s'écrit

$$C(X^s_{t+\tau}, X^k_t) = E[X^s_{t+\tau} X^k_t] - E[X^s_{t+\tau}]E[X^k_t] \tag{11.43}$$

avec

$$E[X^r_t] = \sum_{x_i \geq 0} x^r_i prob(X_t = x_i) \tag{11.44}$$

mais

$$E[X^s_{t+\tau} X^r_t] = \sum_{x_1, x_2 \geq 0} x^r_1 x^s_2 prob(X_{t+\tau} = x_1 \cap X_t = x_2) = \tag{11.45}$$

$$\begin{aligned} = \textstyle\sum^n_{i,j,k,l} \quad & E[X^s_{t+\tau}(X^s_{t+\tau}|Y_{t+\tau} = i \cap Y_{t+\tau-1} = j)E[X^r_t|Y_t = k \cap Y_{t-1} = l) \\ & prob(Y_{t-1} = l)prob(Y_t = k|Y_{t-1} = l)prob(Y_{t+\tau-1} = j|Y_t = k) \\ & prob(Y_{t+\tau} = i|Y_{t+\tau-1}) \end{aligned} \tag{11.46}$$

car

$$\begin{aligned} prob(X_{t+\tau} \quad & = x_2 \cap X_t = x_1) = \textstyle\sum^n_{i,j,k,l} prob(X_{t+\tau} = x_2 \cap X_t = x_1|Y_{t+\tau} \\ & = i \cap Y_{t+\tau-1} = j \cap Y_t = k \cap Y_{t-1} = l). \\ & prob(Y_{t+\tau} = i \cap Y_{t+\tau-1} = j \cap Y_t = k \cap = l) = \end{aligned} \tag{11.47}$$

$$\begin{aligned} \textstyle\sum^n_{i,j,k,l} prob \quad & ((X_{t+\tau} = x_2 \cap X_t = x_1|Y_{t+\tau} = i \cap Y_{t+\tau-1} = j) \cap \\ & (X_{t+\tau} = x_2 \cap X_t = x_1|Y_t = k \cap Y_{t-1} = l)) \\ & prob(Y_{t+\tau} = i \cap Y_{t+\tau-1} = j \cap Y_t = k \cap Y_{t-1} = l) = \end{aligned} \tag{11.48}$$

$$\begin{aligned} \textstyle\sum^n_{i,j,k,l} prob \quad & ((X_{t+\tau} = x_2|Y_{t+\tau} = i \cap Y_{t+\tau-1} = j) \cap (X_t = x_1|Y_t = k \cap Y_{t-1} = l)) \\ & prob(Y_{t+\tau} = i \cap Y_{t+\tau-1} = j \cap Y_t = k \cap Y_{t-1} = l) = \end{aligned} \tag{11.49}$$

$$\begin{aligned} \textstyle\sum^n_{i,j,k,l} prob \quad & (X_{t+\tau} = x_2|Y_{t+\tau} = i \cap Y_{t+\tau-1} = j) \\ & prob(X_t = x_1|Y_t = k \cap Y_{t-1} = l) \\ & prob(Y_{t+\tau} = i \cap Y_{t+\tau-1} = j \cap Y_t = k \cap Y_{t-1} = l) \end{aligned} \tag{11.50}$$

Maintenant on va calculer $prob(Y_{t+\tau} = i \cap Y_{t+\tau-1} = j \cap Y_t = k \cap Y_{t-1} = l)$

$$\begin{aligned} prob(Y_{t+\tau} &= i \cap Y_{t+\tau-1} = j \cap Y_t = k \cap Y_{t-1} = l) = \\ & prob(Y_{t-1} = l) prob(Y_t = k | Y_{t-1} = l) \\ & prob(Y_{t+\tau-1} = j | Y_t = k \cap Y_{t-1} = l) \\ & prob(Y_{t+\tau} = i | Y_{t+\tau-1} = j \cap Y_t = k \cap Y_{t-1} = l) = \end{aligned} \tag{11.51}$$

$$prob(Y_{t-1} = l) prob(Y_t = k | Y_{t-1} = l) prob(Y_{t+\tau-1} = j | Y_t = k) prob(Y_{t+\tau} | Y_{t+\tau-1} = j) \tag{11.52}$$

donc

$$\begin{aligned} E[X_{t+\tau}^s X_t^r] &= \textstyle\sum_{x_1, x_2 \geq 0} x_1^r x_2^s \\ & (\textstyle\sum_{i,j,k,l}^n prob(X_{t+\tau} = x_2 | Y_{t+\tau} = i \cap Y_{t+\tau-1} = j) \\ & prob(X_t = x_1 | Y_t = k \cap Y_{t-1} = l) \\ & prob(Y_{t-1} = l) prob(Y_t = k | Y_{t-1} = l) prob(Y_{t+\tau-1} = j | Y_t = k) \\ & prob(Y_{t+\tau} | Y_{t+\tau-1} = j)) \end{aligned} \tag{11.53}$$

$$\begin{aligned} E[X_{t+\tau}^s X_t^r] &= \textstyle\sum_{i,j,k,l}^n E[X_{t+\tau} | Y_{t+\tau} = i \cap Y_{t+\tau-1} = j] \\ & E[X_t | Y_t = k \cap Y_{t-1} = l] prob(Y_{t-1} = l) \\ & prob(Y_t = k | Y_{t-1} = l) prob(Y_{t+\tau-1} = j | Y_t = k) \\ & prob(Y_{t+\tau} | Y_{t+\tau-1} = j) \end{aligned} \tag{11.54}$$

on peut mettre cette expression sous forme matricielle

$$E[X_{t+\tau}^s X_t^r] = \vec{\pi_{t-1}} \mathbf{\Lambda}_t^{(r)} \mathbf{A}(t+\tau-1, t) \mathbf{\Lambda}_{t+\tau}^{(s)} \vec{e} \tag{11.55}$$

avec

$$\vec{\pi_t} = (prob(Y_t = 1) \cdots prob(Y_t = n)) \tag{11.56}$$

$$(\mathbf{A}(t+\tau-1, t))_{kj} = a_{kj}(t+\tau-1, t) = prob(Y_{t+\tau-1} = j | Y_t = k) \quad 1 \leq k, j \leq n \tag{11.57}$$

$$\mathbf{\Lambda}_t^{(r)} = \begin{pmatrix} E[X_t^r|Y_t = 1 \cap Y_{t-1} = l] & \cdots & E[X_t^r|Y_t = n \cap Y_{t+\tau-1} = l] \\ prob(Y_{t-1} = 1|Y_t = 1) & & prob(Y_{t-1} = n|Y_t = 1) \\ \cdots & \cdots & \cdots \\ E[X_t^r|Y_t = 1 \cap Y_{t-1} = n] & \cdots & E[X_t^r|Y_t = n \cap Y_{t+\tau-1} = n] \\ prob(Y_{t-1} = 1|Y_t = n) & & prob(Y_{t-1} = n|Y_t = n) \end{pmatrix} \tag{11.58}$$

Maintenant, on peut calculer $E[X_t^r]$

$$E[X_t^r] = \vec{\pi}_{t-1}\mathbf{\Lambda}_t^{(r)}\vec{e} \tag{11.59}$$

car

$$\begin{aligned} E[X_t^r] = \quad & \textstyle\sum_{x_1 \geq 0} x_1^k prob(X_t = x_1) = \\ & \textstyle\sum_{k,l=1}^{n} (\sum_{x_1 \geq 0} x_1^r prob(X_t = x_1|Y_t = k \cap Y_{t-1} = l) \\ & prob(Y_t = k \cap Y_{t-1} = l)) = \end{aligned} \tag{11.60}$$

$$\sum_{k,l=1}^{n} (E[X_t = x_1|Y_t = k \cap Y_{t-1} = l]prob(Y_t = k \cap Y_{t-1} = l)) = \tag{11.61}$$

$$\sum_{k,l=1}^{n} (E[X_t = x_1|Y_t = k \cap Y_{t-1} = l]prob(Y_t = k|Y_{t-1} = l)prob(Y_{t-1} = l)) \tag{11.62}$$

11.5 MBH

Pour nous aider à comprendre l'algorithme MBH, nous allons partir de l'équation $\vec{P} = \vec{P}\mathbf{Q}$. Considérons une file d'attente de 3 places, alors

$$(\ \vec{P_1}\quad \vec{P_2}\quad \vec{P_3}\ ) = \begin{pmatrix} \mathbf{Q}_{11} - \mathbf{I} & \mathbf{Q}_{12} & \mathbf{Q}_{13} \\ \mathbf{Q}_{21} & \mathbf{Q}_{22} - \mathbf{I} & \mathbf{Q}_{23} \\ \mathbf{0} & \mathbf{Q}_{32} & \mathbf{Q}_{33} - \mathbf{I} \end{pmatrix} = \vec{0} \tag{11.63}$$

$$\vec{P_1}(\mathbf{Q}_{11} - \mathbf{I}) + \vec{P_2}\mathbf{Q}_{21} = \vec{0} \tag{11.64}$$

$$\vec{P_1}\mathbf{Q}_{12} + \vec{P_2}(\mathbf{Q}_{22} - \mathbf{I}) + \vec{P_3}\mathbf{Q}_{32} = \vec{0} \tag{11.65}$$

$$\vec{P_1}\mathbf{Q}_{13} + \vec{P_2}\mathbf{Q}_{23} + \vec{P_3}(\mathbf{Q}_{33} - \mathbf{I}) = \vec{0} \tag{11.66}$$

de l'équation (11.64), nous tirons que $\vec{P_1} = \vec{P_2}\mathbf{Q}_{21}(\mathbf{I} - \mathbf{Q}_{11})^{-1}$, si nous posons $\mathbf{C}_2 = \mathbf{Q}_{11}$, $\mathbf{R}_2 = (\mathbf{I} - \mathbf{C}_2)^{-1}$, nous trouvons $\vec{P_1} = \vec{P_2}\mathbf{R}_2$.

De l'équation (11.65), nous tirons

$$\vec{P}_2\mathbf{R}_2\mathbf{Q}_{12} + \vec{P}_2(\mathbf{Q}_{22} - \mathbf{I}) + \vec{P}_3\mathbf{Q}_{32} = \vec{0}$$

ce qui donne

$$\vec{P}_2 = \vec{P}_3\mathbf{Q}_{32}(\mathbf{I} - (\mathbf{Q}_{22} - \mathbf{R}_1\mathbf{Q}_{12}))^{-1} \tag{11.67}$$

posons $\mathbf{C}_3 = \mathbf{Q}_{22} - \mathbf{R}_1\mathbf{Q}_{12}$, $\mathbf{R}_3 = \mathbf{Q}_{32}(\mathbf{I} - \mathbf{C}_3)^{-1}$ et $\vec{P}_2 = \vec{P}_3\mathbf{R}_3$ alors nous voyons se profiler les équations utilisées dans l'algorithme. En généralisant, nous trouvons que $\vec{P}_{i-1} = \vec{P}_i\mathbf{R}_i$, $\mathbf{R}_i = \mathbf{Q}_{i,i-1}(\mathbf{I} - \mathbf{C}_i)^{-1}$ avec $\mathbf{C}_i = \mathbf{Q}_{i-1,i-1} + \mathbf{R}_{i-1}(\mathbf{Q}_{i-2,i-1} + \cdots + \mathbf{R}_1(\mathbf{Q}_{1,i-1})\cdots))$

Tous les détails concernant le mécanisme d'inversion et la précision peuvent être trouvés dans [51] (on peut aussi voir dans [35]). Ici nous allons encore exposer brièvement le calcul de la constante de normalisation. Soit

$$G_c = \sum_{d=0}^{c} \vec{P}_d\vec{e} \tag{11.68}$$

et soit $G_c = G$ alors

$$G = \vec{P}_1\vec{e} + \vec{P}_1\vec{e} + \cdots + \vec{P}_c\vec{e} \tag{11.69}$$

$$G = \vec{P}_c(\mathbf{R}_c \cdots \mathbf{R}_1\vec{e} + \cdots + \mathbf{R}_c\vec{e} + \vec{e}) \tag{11.70}$$

$$G = \vec{P}_c(\vec{e} + \mathbf{R}_c(\vec{e} + \mathbf{R}_{c-1}(\vec{e} + \cdots + \mathbf{R}_2(\vec{e} + \mathbf{R}_1\vec{e})\cdots)) \tag{11.71}$$

en posant $\vec{S}_0 = \vec{e}$, $\vec{S}_1 = \vec{e} + \mathbf{R}_1\vec{e}$, $\vec{S}_2 = \vec{e} + \mathbf{R}_2\vec{S}_1$, ..., $\vec{S}_i = \vec{e} + \mathbf{R}_i\vec{S}_{i-1}$ donc

$$G = \vec{P}_c\vec{S}_c \tag{11.72}$$

11.6 Calcul des temps d'interarrivées pour le modèle markovien dépendant de peu de paramètres

La distribution des interarrivées T_A est la suivante, pour $k = 1$

$$prob(T_A = 1|Y_t = 1) = 1 - \sum_{i=1}^{n-1}(1/a)^i$$

pour $k \geq 2$

$$prob(T_A = k|Y_t = 1) = \sum_{i=1}^{n-1}(b/a^2)^i(1 - (b/a)^i)^{k-2}$$

L'espérance mathématique de cette distribution se calcule de la façon suivante

$$E[T_A|Y_t=1] = 1 - \sum_{i=1}^{n-1} 1/a^i + \sum_{k=2}^{\infty}\left(\sum_{i=1}^{n-1}(b/a^2)^i(k+2)(1-(b/a)^i)^{k-2}\right) \tag{11.73}$$

avec

$$\sum_{k=0}^{\infty} k(1-\varrho)^k = \frac{1-\varrho}{\varrho^2} \tag{11.74}$$

$$\sum_{k=0}^{n-1} \varrho^k = \frac{1-\varrho^n}{1-\varrho} \tag{11.75}$$

avec $\varrho < 1$ et quelques calculs, il vient

$$E[T_A|Y_t=1] = \frac{1-(1/b)^n}{1-1/b} \tag{11.76}$$

Si nous voulons que $E[T_A|Y_t=1] < \infty$, nous remarquons que $b > 1$ lorsque $n \to \infty$. Pour calculer le deuxième moment, nous devons en plus savoir que

$$\sum_{k=0}^{\infty} k^2(1-\varrho)^k = \frac{2(1-\varrho^n)^2}{\varrho^3} + \frac{1-\varrho}{\varrho^2} \tag{11.77}$$

$$E[T_A^2|Y_t=1] = 1 - \sum_{i=1}^{n-1} 1/a^i + \sum_{k=2}^{\infty}\left(\sum_{i=1}^{n-1}(b/a^2)^i(k+2)^2(1-(b/a)^i)^{k-2}\right) \tag{11.78}$$

$$E[T_A^2|Y_t=1] = 1 - \sum_{i=1}^{n-1} 1/a^i + \sum_{k=2}^{\infty}\left(\sum_{i=1}^{n-1}(b/a^2)^i(k^2+4k+4)(1-(b/a)^i)^{k-2}\right) \tag{11.79}$$

$$E[T_A^2|Y_t=1] = 1 - \sum_{i=1}^{n-1} 1/a^i + \sum_{i=1}^{n-1} 2(a/b^2)^i(1-(b/a)^i)^2 + \sum_{i=1}^{n-1} 5/b^i(1-(b/a)^i) + \sum_{i=1}^{n-1} 4/a^i \tag{11.80}$$

$$E[T_A^2|Y_t=1] = 1 + \sum_{i=1}^{n}(1/b)^i + \sum_{i=1}^{n}(a/b^2)^i \tag{11.81}$$

$$E[T_A^2|Y_t=1] = 1 + \frac{1-(1/b)^n}{1-1/b} + 2\frac{1-(a/b^2)^n}{1-a/b^2} \tag{11.82}$$

Nous remarquons que si $b > 1$ et $a/b^2 < 1$ alors

$$E[T_A^2|Y_t=1] = 1 + \frac{1}{1-1/b} + 2\frac{1}{1-a/b^2} \tag{11.83}$$

pour $n \to \infty$ alors que si $b < 1$ ou $a/b^2 > 1$, $E[T_A^2|Y_t=1] \to \infty$ si $n \to \infty$.

Chapitre 12

Mémo

12.1 Sigles

ACF	Access Control Field
ARIMA	Autoregressive Integrated Moving Average
ARMA	Autoregressive Moving Average
ATM	Asynchronous Transfer Mode
B-ISDN	Broadband Integrated Services Digital Network
CAC	Connection Acceptance Control
CATV	Community Antenna TV
CD_CNT	Count Down
CRC	Cyclic Redundancy Check
D-BMAP	Discrete Batch Markovian Arrival Process
DQDB	Dual Queue Distributed Queue
EGP	Exterior Gateway Protocol
F+E	Forschung und Entwicklung
FIFO	First In First Out
HMM	Hidden Markov Model
IEEE	Institute of Electrical and Electronics Engineers
ICMP	Internet Control Message Protocol
IM-PDU	Initial MAC PDU
IP	Internet Protocol
ITU	International Telecommunication Union
LAN	Local Area Network
LBL	Lawrence Berkeley Laboratory
MAC	Medium Access Control
MBH	Matrix Block Hessenberg
MMPP	Markov Modulated Poisson Process

PDU	Protocol Data Unit
QoS	Quality of Service
RD	Research and Developpment
REQ_CNT	Request Counter
SSMP	Special Semi-Markov Process
TCP	Transmission Control Protocol
UDP	User Datagram Protocol

12.2 Symboles

Dans cette section, nous n'avons mentionné que les symboles les plus courants, qui peuvent se retrouver dans plusieurs chapitres différents.

β	valeur absolue de la pente du graphe "temps-variances"
c	nombre de places dans la file d'attente
$C(\tau)$	autocovariance pour un processus stationnaire au sens large
E	ensemble des nombres entiers non négatifs où X prend ses valeurs
ϵ	nombre positif arbitrairement petit
ϵ_ς	nombre positif arbitrairement petit
FO	fonction objectif
γ_r	probabilité que $R_t = r$
H_l	paramètre de Hurst
L	probabilité de perte
λ_i	$i$ème valeur propre
λ_i^*	$i$ème valeur propre de la matrice décomposable
m	taille de fenêtre (aussi utilisée comme variable d'incrémentation)
μ_u	moyenne du temps d'attente
n	nombre d'états de la chaîne de Markov
N	nombre de blocs dans la matrice décomposable (aussi utilisé pour DQDB)
$\mathbb{N}$	ensemble des nombres entiers non négatifs
$n(i)$	nombre d'éléments dans un bloc i de la matrice décomposable
N_m	nombre d'arrivées dans une fenêtre de taille m
p, q	probabilité de transition de la chaîne de Markov à deux états
ϕ_{ij}	$prob(X_t = j \mid Y_t = i)$
$\Phi(\omega)$	transformée de Fourier de $C(\tau)$
R_t	temps résiduel de service
σ_u	variance du temps d'attente
S_k	temps de service du $k$ième client

τ	entier positif
W_k	temps d'attente du k^{e} client
X_t	variable aléatoire (arrivées de cellules)
$x_1, x_2, \cdots$	valeurs de X_t
Y_t	état du modulateur de la chaîne de Markov
$\mathbf{A}$	matrice de transition
$\mathbf{A}^*$	matrice de transition décomposable, solution optimale avec TS
$\mathbf{B}_i$	blocs de taille $n \times n$
$\mathbf{C}_i$	blocs de taille $n \times n$
$\mathbf{D}$	matrice diagonale
$\vec{e}$	vecteur colonne
$\mathbf{G}$	matrice de passage
$\mathbf{H}_i$	blocs de taille $n \times n$
$\mathbf{\Lambda}$	matrice diagonale contenant les espérances mathématiques
$\vec{P}$	vecteur des probabilités stationnaires de la file d'attente
$\mathbf{P}_i$	projecteur i

12.3 Conventions

Ce volume est divisé en chapitres repérés par un nombre arabe. Chaque chapitre est divisé en sections repérées par deux nombres arabes séparés par un point (section 2.3). Chaque section est divisée en paragraphes repérés par trois nombres arabes séparés par deux points (paragraphe 3.4.10). Les références bibliographiques sont numérotées continûment et repérées par un seul nombre arabe entre crochets [54]. Un terme apparaît en **gras** la première fois qu'il apparaît dans le texte. Les équations numérotées le sont continûment par chapitres et repérées par deux nombres arabes placés entre parenthèses et séparés par un point (10.4). Les figures sont numérotées continûment par chapitres et repérées par deux nombres arabes séparés par un point (figure 2.1). Les tableaux sont numérotées continûment par chapitres et repérées par deux nombres arabes séparés par un point (tableau 8.2).

Chaque terme anglais utilisé dans le texte est en *italique*. Lorsqu'un mot, traduit de l'anglais au français, peut prêter à confusion, il sera en général suivi du terme anglais entre parenthèses (...horodateur (*timestamps*)...). Chaque fois qu'une abréviation est introduite pour la première fois, elle est introduite entre parenthèses après la suite de mots complète (... un algorithme de *Distributed Queue Dual Bus* (DQDB)...). Si la traduction en français existe, alors l'abréviation est mise entre parenthèses suivie du terme anglais

(.. fonction de lissage (UPC, *Usage Parameter Control*)...). Ensuite l'abréviation est utilisée librement comme un autre mot. Le seul mot utilisé qui ne soit ni français, ni anglais est le verbe *fitter*.

Liste des figures

Liste des tableaux

Index

Bibliographie

[1] J. Roberts, *COST 224, Performance evaluation and design of multiservice networks.* Luxembourg: Commission of the European Communities, 1992.

[2] A. K. Erlang, "The Theory of Probabilities and Telephone Conversations," *Nyt Tidsskrift Matematik*, no. B 20, pp. 33–39, 1909.

[3] A. K. Erlang, "Solutions of Some Problems in the Theory of Probabilities of Significance in Automatic Telephone Exchanges," *Electroteknikeren*, no. 13, pp. 5–13, 1917.

[4] E. C. Molina, "Application of the Theory of Probability to Telephone Trunking Problems," *Bell Syst. Tech. J.*, no. 6, pp. 461–494, 1927.

[5] F. Pollaczek, "Losung eines Geometrischen Wahrscheinlichkeits-problems," *Math. Z.*, no. 35, pp. 230–278, 1932.

[6] F. Pollaczek, "Uber das Warteproblem," *Math. Z.*, no. 38, pp. 492–537, 1934.

[7] W. Feller, *An Introduction to Probability Theory and Its Applications, vol. II.* New-York: Wiley, 1966.

[8] D. Boggs, J. Schoch, E. Taft, and R. Metcalfe, "Pup: An internetwork architecture," *IEEE Transactions on Communications*, vol. 4, pp. 612–624, April 1980.

[9] G. Almes and E. Lazowska, "The Behavior of Ethernet-Like Computer Communications Networks," in *ACM SIGCOM*, (Asilomar, California), December 1979.

[10] W. Bux, "Local-Area Subnetworks: A Performance Comparison," *IEEE Transactions on Communications*, vol. 34, pp. 1465–1473, October 1981.

[11] F. A. Tobagi and B. Hunt, "Performance Analysis of Carrier Sense Multiple Access with Collision Detection," *Computer Networks*, vol. 4,5, pp. 245–259, October/November 1980.

[12] S. Tasaka, "Dynamic Behavior of a csma-cd System with a Finite Population of Buffered Users," *IEEE Transactions on Communications*, vol. 34, pp. 576–586, June 1986.

[13] J. Shoch and J. Hupp, "Measured Performance of an Ethernet Local Network," *Communications of the ACM*, vol. 23, pp. 711–721, December 1980.

[14] W. E. Leland, "On the Self-Similar Nature of Ethernet Traffic (Extended Version)," *IEEE/ACM Transactions on Networking*, vol. 2, February 1994.

[15] W. Leland and D. Wilson, "High Time–Resolution Measurement and Analysis of LAN Traffic: Implications for LAN Interconnection," in *IEEE Infocom*, (Bal Harbor, FL), April 1991.

[16] H. Fowler and W. Leland, "Local Area Network Traffic Characteristics, with Implications for Broadband Network Congestion Management," *IEEE Journal on Selected Areas in Communications*, vol. 9, pp. 1139–1149, September 1991.

[17] C. Partridge, "The End of Simple Traffic Models," *IEEE Network*, p. 3, September 1993.

[18] I. Norros, "On the Use of Fractional Brownian Motion in the Theory of Connectionless Networks," *IEEE Journal on Selected Areas in Communications*, vol. 13, pp. 953–962, August 1995.

[19] P. Droz, *A High-Speed Self-Similar ATM Traffic Generator with QoS Parameters*. IBM, Ruschlikon, Switzerland, 1995.

[20] A. Erramilli, "Chaotic maps as models of packet traffic," in *ITC 14, The Fundamental Role of Teletraffic in the Evolution of Telecommunications Networks* (J. Labetoulle and J. Roberts, eds.), (Antibes Juan-Les-Pins, France), Elsevier Science Publishers B.V. (North-Holland), June 6-10 1994.

[21] D. Veitch, "Novel Models of Broadband Traffic," in *IEEE Globecom*, (Houston, USA), November 29 - December 2 1993.

[22] F. Brichet, J. Roberts, A. Simonian, and D. Veitch, *Heavy Traffic Analysis of a Storage Model Long Range Dependent On/Off Sources*, 1995.

[23] V. Paxson, "Fast Approximation of Self-Similar Network Traffic," Tech. Rep. LBL-36750, Lawrence Berkeley Laboratory, University of California, Berkeley, California, USA, April 1995.

[24] A. S. Tanenbaum, *Computer Networks*. Prentice-Hall, 1989.

[25] ITU, Geneva, Switzerland, *Recommendation I.121: Aspects Large Bande du RNIS*.

[26] M. de Prycker, *Asynchronous Transfer Mode Solution for Broadband ISDN*. Ellis Horwood Limited, 1991.

[27] M. Luoni, *ATM traffic characterization with applications to connection acceptance control*. No 979, EPFL, Lausanne, Switzerland, 1991.

[28] J. Y. Hui, "Resource allocation for broadband networks," *IEEE Journal on Selected Areas in Communications*, vol. 6, pp. 1598–1608, September 1988.

[29] ITU, Geneva, Switzerland, *Recommendation I.371: Traffic Control and Congestion Control in B-ISDN*.

[30] M. T. Le, "The IEEE 802.6 Metropolitan Area Network Distributed-Queue, Dual-Bus Protocol," *Journal of Data and Computer Communications*, 1990.

[31] J. F. Mollenauer, "Standards for Metropolitan Area Networks," *IEEE Communications Magazine*, vol. 26, pp. 15–19, April 1988.

[32] R. Newmann, Z. Budrikis, and J. Hullett, "The QPSX Man," *IEEE Communications Magazine*, vol. 16, pp. 20–28, April 1988.

[33] P. T.-G. et al., "Approximate Performance Analysis of the DQDB Access Protocol," in *ITC Specialist Seminar*, pp. 231–240, September 1989.

[34] R. Grünenfelder and S. Robert, "Wich Arrival Law Parameters Are Decisive for Queuing System Performance? ," in *ITC 14, The Fundamental Role of Teletraffic in the Evolution of Telecommunications Networks* (J. Labetoulle and J. Roberts, eds.), (Antibes Juan-Les-Pins, France), pp. 377–386, Elsevier Science Publishers B.V. (North-Holland), June 6-10, 1994.

[35] S. Robert, "Calcul de la probabilité de perte dans une file d'attente de type SSMP(N)/G/1/c," tech. rep., EPFL-TCOM, Lausanne, Switzerland, March 1993.

[36] S. Robert, "Modélisation de trafic réel avec chaînes de markov," tech. rep., EPFL-TCOM, Lausanne, Switzerland, September 1993.

[37] S. Robert, "Modelisation of the Ethernet-DQDB interconnection based on measured data," tech. rep., EPFL-TCOM, Lausanne, Switzerland, April 1994.

[38] R. Grünenfelder and S. Robert, "Decisive arrival law parameters and a general finite capacity queueing problem," *Performance Evaluation*, vol. 23, pp. 199–215, September 1995.

[39] S. Robert and C. van den Branden Lambrecht, "An SSMP Traffic Model Based on Network Measurements," in *IEEE International Phoenix Conference on Computers and Communications*, (Phoenix, USA), March 28-31 1995.

[40] L. Baum and T. Petrie, "Statistical Inference for probabilistic functions of finite state Markov Chains," *Ann. Math. Stat.*, vol. 37, pp. 1554–1563, 1966.

[41] L. R. Rabiner, "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," *Proceedings of the IEEE*, vol. 77, pp. 257–285, February 1989.

[42] C. Blondia and O. Casals, "Statistical Multiplexing of VBR sources: A matrix-analytic approach," *Performance Evaluation*, vol. 16, pp. 5–20, 1992.

[43] F. de Coulon, *Théorie et Traitement des Signaux*. EPFL, Lausanne, Switzerland: Presses Polytechniques Romandes, 1988.

[44] S. Friedberg, A. Insel, and L. Spence, *Linear Algebra*. Illinois State University: Prentice-Hall, 1989.

[45] H. Minc, *Nonnegative Matrices*. John Wiley and Sons, 1988.

[46] J. Swift, "The Location of Characteristic Roots of Stochastic Matrices," Master's thesis, Mc Gill University, Montreal, 1972.

[47] J. Brewer, "Kronecker Products and Matrix Calculus in System Theory," *IEEE Circuits and Systems*, vol. 25, pp. 772–781, September 1978.

[48] R. Gusella, "A Measurement Study of Diskless Workstation Traffic on an Ethernet," *IEEE Transactions on Communications*, vol. 38, September 1990.

[49] S. Dafermos and M. Neuts, "A Single Server Queue in Discrete Time," *Cahiers du Centre d'Etudes en Recherche Opérationnelle*, vol. 13, pp. 23–40, 1971.

[50] O. Hashida, Y. Takahashi, and S. Shimog, "Switched batch Bernoulli Process (SBBP) and the Discrete-Time SBBP/G/1 Queue with Application to Statistical Multiplexer Performance," *IEEE Journal on Selected Areas in Communications*, vol. 9, pp. 394–401, April 1991.

[51] J.-Y. L. Boudec, "An Efficient Solution Method for Markov Models of ATM Links with Loss Priorities," *IEEE Journal on Selected Areas in Communications*, vol. 9, pp. 408–417, April 1991.

[52] H. C. Tijms, *Stochastic Modelling and Analysis: A Computational Approach*. Wiley, 1986.

[53] R. Metcalfe and D. Boggs, "Ethernet: Distributed packet switching for local computer networks," *Communications of the ACM*, vol. 19, pp. 395–404, July 1976.

[54] R. Jain and S. A. Routier, "Packet Trains Measurements and a New Model for Computer Network Traffic," *IEEE Journal on Selected Areas in Communications*, vol. 4, pp. 986–995, September 1986.

[55] A. Kolmogorov, "Local structure of turbulence in fluid for very large Reynolds numbers," *Trans. in Turbulence*, pp. 151–155, 1941.

[56] B. Mandelbrot and J. V. Ness, "Fractional Brownian Motions, Frational Noises and Applications," *SIAM Review*, vol. 10, October 1968.

[57] J. Beran, *Statistics for Long-Memory Processes.* Chapman and Hall, 1994.

[58] D. R. Cox, "Long-range dependance: A review," *Statistics: An Appraisal*, 1984.

[59] I. Norros, *The New COSTbook. The sections on long range dependence.* COST242, 1996.

[60] H. Heffes and D. Lucantoni, "A Markov Modulated Characterization of Packetised Voice and Data Traffic and Related Statistical Multiplexer Performance," *IEEE Journal on Selected Areas in Communications*, vol. 4, September 1986.

[61] R. Grünenfelder, S. Robert, J.-P. Hubaux, and F. Braun, "A Performance Study of the Ethernet/DQDB Interconnection," in *Interworking in Broadband Networks*, pp. 378–387, IOS Press, 1993.

[62] L. Kleinrock, *Queueing Systems, Volume I: Theory.* John Wiley and Sons, 1975.

[63] S. Robert, "Interconnexion Ethernet-DQDB," tech. rep., EPFL-TCOM, Lausanne, Switzerland, September 1992.

[64] M. Fogiel, *Handbook of Mathematical, Scientific, and Engineering.* Research and Education Association, 1986.

[65] M. Livny, B. Melamed, and A. Tsiolis, "The Impact of Autocorrelation on Queueing Systems," tech. rep., University of Wisconsin-Madison, September 1990.

[66] S. Li and J. Mark, "Traffic Characterization for Integrated Services Networks," *IEEE Transactions on Communications*, vol. 38, pp. 1231–1243, August 1990.

[67] S. Li and C.-L. Hwang, "Queue Response to Input Correlation Functions: Discrete Spectral Analysis," in *IEEE Infocom*, (Florence, Italy), May 1992.

[68] S. Li and C.-L. Hwang, "Queue Response to Input Correlation Functions: Continuous Spectral Analysis," *IEEE/ACM Transactions on Networking*, vol. 1, pp. 678–692, December 1993.

[69] R. Addie and M. Zuckermann, "A Gaussian Characterization of Correlated ATM Multipexed Traffic and Related Queueing Studies," in *ICC'93*, (Geneva), pp. 1404–1408, May 23-26 1993.

[70] C. Courcoubetis, G. Fouskas, and R. Weber, "On the Performance of an Effective Bandwidth Formula," in *ITC 14, The Fundamental Role of Teletraffic in the Evolution of Telecommunications Networks* (J. Labetoulle and J. Roberts, eds.), (Antibes Juan-Les-Pins, France), pp. 201–222, Elsevier Science Publishers B.V. (North-Holland), June 6-10 1994.

[71] A. Mukherjee, "On the Dynamics and the Significance of Low Frequency Components of Internet Load," *Internetworking, Research and Experience*, vol. 5, pp. 163–205, 1994.

[72] W. Ding, "A Unified Correlated Input Process Model for Telecommunication," in *ITC 13, Teletraffic and Datatraffic in a Period of Change* (A. Jensen and V. Iversen, eds.), (Copenhagen, Denmark), pp. 539–544, Elsevier Science Publishers B.V. (North-Holland), June 1991.

[73] J. Andrade, "ATM Source Traffic Descriptor Based on the Peak, Mean and Second Moment of the Cell Rate," in *ITC 14, The Fundamental Role of Teletraffic in the Evolution of Telecommunications Networks* (J. Labetoulle and J. Roberts, eds.), (Antibes Juan-Les-Pins, France), pp. 223–232, Elsevier Science Publishers B.V. (North-Holland), June 6-10 1994.

[74] R. Grünenfelder, *Stochastic Modelling of the Traffic and its Properties in an ATM Network.* No 912, EPFL, Lausanne, Switzerland, 1991.

[75] A. Baiocchi, N. Melazzi, M. Listanti, A. Roveri, and R. Winkler, "Loss Performance Analysis of an ATM Multiplexer loaded with High-Speed ON-OFF Sources," *IEEE Journal on Selected Areas in Communications*, vol. 9, pp. 388–393, April 1991.

[76] A. Papoulis, *Probability, Random Variables, and Stochastic Processes.* Mc Graw Hill, 2nd edition, 1989.

[77] S. Kirkpartick, C. D. Gelatt, and M. P. Vecchi, "Optimization by Simulated Annealing," *IBM Research Report RC 9355*, 1982.

[78] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, and A. H. Teller, "An efficient General Cooling Schedule for Simulated Annealing," *the Journal of Chemical Physics*, vol. 21, June 1953.

[79] S. Kirkpartick, C. D. Gelatt, and M. P. Vecchi, "Optimization by Simulated Annealing," *Science*, vol. 220, May 1983.

[80] S. Geman and D. Geman, "Stochastic Relaxation, Gibbs Distribution, and the Bayesian Restoration of Images," *IEEE Proc. Pattern Analysis and Machine Intelligence*, PAMI-6 1984.

[81] M. Lundy and A. Mees, "Convergence of an Annealing Algorithm," *Math. Prog.*, no. 34, pp. 111–124, 1986.

[82] M. D. Huang, F. Romeo, and A. L. Sangiovanni-Vincentelli, "An efficient General Cooling Schedule for Simulated Annealing," in *IEEE Proc. Internal Conference on Computer-Aided Design*, (Santa Clara), November 1986.

[83] F. Glover, "Future paths for Inter Programming and Links to Artificial Intelligence," *Comput. Oper. Res.*, vol. 13, pp. 533–549, 1986.

[84] F. Glover, "Tabu Search," *CAAI Report*, vol. 88-3, 1988.

[85] P. Hansen and B. Jaumard, "Algorithms for the Maximum Satisfability Problem," *RUTCOR Research Report*, vol. 43-87, 1987.

[86] P. J. Courtois, *Decomposability.* ACM Monograph Series, 1977.

[87] H. Simon and A. Ando, "Aggregation of variables in dynamic systems," *Econometrica*, no. 29, 1961.

[88] H. A. Simon, "The Architecture of complexity," *Proc. Amer. Phil. Soc.*, no. 106, pp. 467–482, 1962.

[89] H. A. Simon, "The sciences of the artificial," *MIT Press, Cambridge, Massachussetts*, 1969.

[90] D. Comer, *Internetworking with TCP/IP, Volume 1: Principles, Protocols and Architecture.* Prentice-Hall, 1991.

[91] S. Manthorpe and X. Garcia, "TCP Performance over ATM Based LAN Interconnection Services," in *Interop'95, Engineers Conference*, (Las Vegas, USA), April 28-30 1995.

[92] I. Norros, "A storage model with self-similar input," *Queueing Systems*, vol. 16, pp. 387–396, 1994.

[93] P. L'Ecuyer, "Efficient and Portable Combined Random Number Generators," *Communications of the ACM*, vol. 31, pp. 742–774, June 1988.

[94] D. Tse, R. Gallager, and J. Tsitsiklis, "Statistical Multiplexing of Multiple Time-Scale Markov Streams," *IEEE Journal on Selected Areas in Communications*, vol. 13, pp. 1028–1038, August 1995.

[95] A. Simonian and J. Guibert, "Large Deviations Approximation for Fluid Queues Fed by a Large Number of On/Off Sources," *IEEE Journal on Selected Areas in Communications*, vol. 13, pp. 1017–1027, August 1995.

[96] E. Gauthier and J.-Y. L. Boudec, "Scalability Enhancements for Connection-Oriented Networks," Tech. Rep. TR 95/127, EPFL-LRC, Lausanne, Switzerland, 1995.

[97] E. Gauthier, S. Giordano, and J.-Y. L. Boudec, "Reduce Connection Awareness," Tech. Rep. TR 95/145, EPFL-LRC, Lausanne, Switzerland, 1995.

[98] S. Giordano and J.-Y. L. Boudec, "Control Architecture for Reducing Connection Awareness," tech. rep., EPFL-LRC, Lausanne, Switzerland, 1995.

[99] Technical Comittee, Signaling Subworking Group, ATM-Forum, *UNI Specification.*

Chapitre 13

Curriculum Vitae

Stephan Robert est né à Boudevilliers, dans une petite commune du Val-de-Ruz dans le canton de Neuchâtel (Suisse) en 1965. Après avoir suivi ses classes aux Ponts-de-Martel et au Locle, il commence des études d'ingénieur ETS en microtechnique en 1980 pour les terminer en 1986. A l'obtention de son diplôme (septembre 1986), il reçoit un prix du club Rotary (meilleure moyenne de la section microtechnique). Après ses études, il s'engage chez ABB à Baden pour développer deux relais de protection: les relais UKT 910 et UKT 911. Ces relais servent à la protection d'appareils contre les surtensions en régime continu et alternatif. Après un an, en octobre 1987, il reprend ses études (octobre 1987) à l'école polytechnique fédérale de Lausanne (EPFL) en électricité (orientation communications) pour les terminer en janvier 1991. En parallèle, il enseignera à l'Ecole Nouvelle de la Suisse Romande (ENSR). Pendant l'été 1989, il fera un stage très enrichissant à la poste en Allemagne (Stuttgart) où il aura l'occasion de découvrir toutes les facettes d'un opérateur. A la remise de son diplôme, il est récompensé par l'entreprise Nokia-Maillefer pour l'originalité de son travail pratique, qui a consisté à développer et à construire un prototype d'antennes planaires SSFIP en collaboration avec Huber & Suhner à Hérisau. Ensuite, il sera employé par le laboratoire de dynamique des fluides pour analyser des mesures de turbulence atmosphérique mesurées par un avion Tiger. Ce projet servira à mieux comprendre ces phénomènes atmosphériques dans le but de créer un simulateur de vol dont les caractéristiques se rapprocheraient de la réalité observée. Pendant cette période, il a l'occasion de se familiariser avec les techniques de calcul de

la dynamique des fluides, notamment à l'aide de cours qu'il a suivi à l'école centrale de Lyon. Un an plus tard, il va retourner dans le domaine des télécommunications pour étudier le comportement du trafic issu des réseaux de données. En avril 1992, il va être engagé par le Professeur Hubaux du laboratoire des télécommunications de l'EPFL. Le projet sur lequel il va travailler est financé par Telecom PTT. En octobre 1994, suite à une restructuration du laboratoire des télécommunications, il va changer de laboratoire pour désormais travailler sous la direction du professeur Le Boudec, au laboratoire des réseaux de communication. Pendant cette période, il va faire du *consulting* pour le compte des Telecom PTT en même temps que sa recherche. Entre autres, il va s'impliquer dans un projet européen, COST 242 qui rassemble plusieurs spécialistes du domaine du télétrafic. Après l'obtention de son doctorat, il a l'intention de passer deux ans à l'université de Californie à Berkeley sous la direction du professeur Walrand. Sur le plan privé, il est marié et l'heureux père d'un petit garçon.

Publications

- J. Sanford, J.-F. Zürcher and S.Robert, "Shaped beam patch arrays for mobile communication base stations", *Microwave Engineering Europe Journal*, June/July, pp. 31-33, 1991.

- J. Sanford, J.-F. Zürcher and S. Robert, "Optimized Antennas for Mobile Communication Base Stations", *Proceedings of the 21th European Microwave Conference*, paper 1.17, vol. 1, pp. 780-786, July 1991.

- R.Grünenfelder, P.Rogl and S.Robert, "Performance evaluation of DQDB/ATM interconnection with adaptative prevention mechanism", *EFOC/LAN'92 Conference*, Paris, France, pp. 241-245, June 24-26, 1992.

- R.Grünenfelder, S.Robert, J.-P.Hubaux and F.Braun, "Ethernet/DQDB Interconnection", *Interworking' 92*, Bern, November 18-20, 1992.

- R.Grünenfelder, S.Robert, J.-P.Hubaux and F.Braun, "Ethernet/DQDB Interconnection", *Interworking in Broadband Networks*, ed: S. Rao, IOS Press, Amsterdam,

Oxford, Washington, Tokyo, 1993.

- R.Grünenfelder and S.Robert, "Which Arrival Law Parameters Are Decisive for Queueing System Performance", in *ITC14*, Antibes Juan-les-Pins, France, June 6-10, 1994.

- S.Robert and C.van den Branden Lambrecht, "An SSMP Traffic Model Based on Network Measurements", *1995 IEEE International Phoenix Conference on Computers and Communications*, Phoenix, USA, March 28-31, 1995.

- S.Robert, "Fitting an SSMP to measured data", *Engineer's Conference 1995, Interop95*, Las Vegas, USA, March 27-31, 1995.

- S.Robert and J.-Y. Le Boudec, "Can self-similar traffic be modeled by Markovian processes? ", *COST 242*, TD95_26, Stockholm, May 10-11, 1995.

- R.Grünenfelder and S.Robert, "Decisive Arrival Parameters and a General Finite Capacity Queueing Problem", *Performance Evaluation Journal*, vol. 23, no. 3, September 1995.

- S.Robert and J.-Y. Le Boudec, "Stochastic processes for self-similar traffic", *COST242*, TD95_46, Bratislava, September 13-14, 1995.

- S.Robert and J.-Y. Le Boudec, "A Modulated Markov Model for Self-Similar Traffic", *Internationales Begegnungs und Forschungszentrum fuer Informatik*, Schloss Dagsthul, Saarbrücken, Germany, September 24-29, 1995.

- S.Robert and J.-Y. Le Boudec, "Properties of a new class of models designed for self-similar traffic", *IFIP, WG.6.2 Broadband Communication*, Paris, France, pp. 131-138, December 6-8, 1995.

- S.Robert and J.-Y. Le Boudec, "Can self-similar traffic be modeled by Markovian processes? ", *International Zurich Seminar on Digital Communications*, ETH Zurich, Switzerland, February 19-23, 1996.

- S. Giordano, J.-Y Le Boudec and S. Robert, "VBR multiplexed over VBR", to be submitted
- S. Robert and J.-Y. Le Boudec, "New models for self-similar traffic", submitted to *Performance Evaluation Journal*
- S. Robert and J.-Y. Le Boudec, "On a Markov Modulated Chain Exhibiting Self-Similarities Over Finite Timescale", submitted to *Performance'96*
- S. Robert and J.-Y. Le Boudec, "A Modulated Markov Chain with Pseudo-Long Range Dependences", submitted to *IEEE/ACM Transaction on Networking*

Rapports

- S. Robert, "Etude d'antennes SSFIP", EPFL/LEMA Research Report, *MSc. Thesis*, Lausanne, Switzerland, January 1991.
- S. Robert, "Etude de turbulences atmospheriques", *EPFL/IMHEF-GDA Research Report*, Lausanne, March 1992.
- S. Robert, "Interconnexion Ethernet-DQDB", *EPFL/TCOM-PTT Telecom Research report*, F&E Projekt 240, Lausanne, Septembre 1992
- S. Robert, "Calcul de la probabilite de perte dans une file d'attente de type SMP(N)/G/1/c", *EPFL/TCOM-PTT Telecom Research report*, F&E Projekt 240, Lausanne, Mars 1993.
- S. Robert, "Modelisation de trafic reel avec chaînes de Markov", *EPFL/TCOM-PTT Telecom Research report*, F&E Projekt 240, Lausanne, Septembre 1993.
- C. van den Branden Lambrecht and S. Robert, "Measurement and Analysis of Computer Traffic on Ethernet and FDDI Networks", *EPFL/TCOM Research report*, Lausanne, September 1993
- S. Robert, "Ethernet-DQDB Interconnection Model Based on Measured Data", *EPFL/TCOM-PTT Telecom Research report*, F&E Projekt 240, Lausanne, April 1994.

- S. Robert, "UPC Parameters and CLS Topology", *EPFL/TCOM-PTT Telecom Research report*, R&D Project 303, Lausanne, October 1994.

- S. Robert, "SMDS/CBDS Over ATM", *EPFL/LRC-PTT Telecom Research report*, R&D Project 303, Lausanne, May 1995.