UNIVERSITÉ DE VERSAILLES ST-QUENTIN-EN-YVELINES

Département d'informatique
UFR des Sciences

École doctorale "STV"

Chaînes de Markov Incomplètement Spécifiées : analyse par comparaison stochastique et application à l'évaluation de performance des réseaux

THÈSE

présentée et soutenue publiquement le 3 octobre 2014

pour l'obtention du

Doctorat de l'Université de Versailles Saint-Quentin

Spécialité : Informatique

par

AIT SALAHT Farah

Composition du jury

Président :	Véronique VEQUE, Professeur, Université Paris-Sud 11
Rapporteurs :	Gerardo RUBINO, Directeur de Recherche, à INRIA Rennes - IRISA Patrice MOREAUX, Professeur, LISTIC, Polytech Annecy-Chambéry, Université de Savoie
Directeurs :	*Pr Jean-Michel FOURNEAU, Professeur, Université de Versailles Saint-Quentin en Yvelines* *Pr Hind CASTEL-TALEB, Professeur, TSP-Télécom SudParis* *Pr Nihal PEKERGIN, Professeur, Université Paris-Est Créteil Val de Marne*

Laboratoire PRISM

Remerciements

Je tiens d'abord et avant tout, à adresser ma profonde gratitude à mes directeurs de thèse : Jean-Michel FOURNEAU, Professeur à l'Université de Versailles Saint-Quentin, Hind CASTEL-TALEB, Professeur à TSP-Télécom SudParis et Nihal PEKERGIN, Professeur à l'Université Paris-Est Créteil, qui ont dirigé mes travaux. Merci pour tout le temps consacré, les conseils et le soutien que vous m'avez apporté, mais également pour la confiance témoignée tout au long de la thèse. Je vous remercie sincèrement de m'avoir permis de progresser sur de nombreux plans avec bienveillance. Je vous en suis énormément reconnaissante.

Je tiens à remercier Gerardo RUBINO, Directeur de Recherche, à INRIA Rennes, et Patrice MOREAUX, Professeur à LISTIC, Polytech Annecy-Chambéry de l'Université de Savoie, qui m'ont fait l'honneur de rapporter cette thèse et qui, par leurs remarques m'ont permis d'en améliorer la qualité. Je tiens à vous exprimer toute ma gratitude pour avoir consacré une partie de votre temps à lire et juger mes travaux.

Je remercie également Véronique VEQUE, Professeur à l'Université Paris-Sud 11, d'avoir accepté la présidence de mon jury de thèse.

Un grand merci à tous les membres du laboratoire PRISM de l'université de Versailles, du laboratoire LACL de l'université de Créteil ainsi que les membres du laboratoire Samovar de INT d'Évry. J'ai eu vraiment beaucoup de plaisir à y côtoyer les gens et à travailler avec eux. Les échanges ont toujours été source d'enrichissement et dans un respect mutuel. Merci à toutes celles et à tous ceux qui ont contribué à mon intégration dans ces laboratoires.

Je tiens à remercier également mes amis et mes collègues du PRiSM : Lydia, Faiza, Hanane, Nora, Karim, Samir et tous les autres, qui m'ont apporté de la bonne humeur, leurs conseils et qui ont rendu mon séjour au PRiSM particulièrement agréable, ainsi qu'à tous les autres membres des laboratoires que j'ai eu l'occasion de connaître pendant ma thèse.

Ce travail résulte également d'un grand soutien familial. Un immense merci à mes chers parents et à mon frère Mennad, pour leur soutien inconditionnel et sans limite, leur confiance en moi et leurs encouragements sans faille. Je remercie également les autres membres de ma famille, particulièrement : ma grand-mère Feroudja, mes oncles : Karimo, Hamid et Kakou, mes tantes : Fatiha, Nedjima et Fadila et mes cousins : Amina, Amine, Said, Samy et Yelena. Vous m'avez tous à votre manière soutenue, encouragée et aidée pendant cette thèse. Pour cela, mais aussi pour tout le reste, merci.

J'ai sûrement oublié de citer certains d'entre vous qui m'ont aidé dans mon travail, même indirectement, du fond du coeur je vous dis MERCI.

Résumé

Nous traitons les problèmes d'incertitudes dans les modèles probabilistes et tentons de déterminer leur impact sur l'analyse de performances et le dimensionnement des systèmes. Nous considérons dans cette thèse deux aspects du problème d'imprécision. Le premier, consiste à étudier des chaînes de Markov dont les probabilités ou taux de transition ne sont pas parfaitement connus. L'absence de précision pour certaines transitions dans le système exact à cause de la complexité peut se résoudre par la construction de systèmes bornants où des transitions pire des cas sont définies pour obtenir une borne supérieure ou inférieure sur l'indice de performance à calculer. Nous déterminons ainsi une chaîne "pire" et une chaîne "meilleure" dont l'analyse exacte nous permet d'apporter des réponses sur le fonctionnement "au pire" ou au " au mieux " du système. Nous développons à cet effet de nouveaux algorithmes de calcul de bornes par éléments sur les vecteurs de distribution stationnaires de chaînes partiellement spécifiées ainsi que sur les temps moyens d'absorption de chaînes absorbantes partiellement spécifiées. Ces algorithmes sont plus rapides que les algorithmes existants dans la littérature et permettent de déterminer des bornes par élément à chaque étape de calcul.

Le second aspect revient à considérer les mesures de trafic réel dans les réseaux. Souvent très volumineuses, la modélisation du trafic est généralement impossible à effectuer de façon suffisamment précise et l'adéquation avec une loi de probabilité connue n'est pas assez réaliste. Sous l'hypothèse de stationnarité des flux, nous définissons des histogrammes (distributions) pour caractériser le trafic. Afin de faciliter l'analyse, nous simplifions les distributions empiriques en construisant des bornes stochastiques optimales pour une fonction de récompense positive croissante sur ces distributions. Nous présentons une méthode d'évaluation de performance des réseaux reposant sur diverses techniques complémentaires. Premièrement, nous montrons comment obtenir des distributions bornantes au sens de l'ordre stochastique fort des traces de trafic à l'entrée des réseaux. Ainsi, nous obtenons des histogrammes encadrants le trafic et nous pouvons par conséquent paramétrer la taille, ce qui impacte à la fois la complexité et la qualité des calculs. Nous montrons également que les éléments de réseaux sont stochastiquement monotones par rapport aux histogrammes, ce qui permet d'obtenir un encadrement stochastique très pertinent des mesures de performance. L'intérêt et l'impact de notre méthode est montré sur différentes applications : éléments de réseaux (files d'attente, élément de routage), file d'attente avec AQM (Active Queue Management), réseaux de files d'attente et file d'attente avec entrée non stationnaire. La méthode proposée fournit des résultats très pertinents et offre un compromis très intéressant entre le temps de calcul et la précision. De plus, les bornes de performance calculées sont très utiles dans le dimensionnement du réseau.

Mots-clés : Chaînes de Markov, Incertitude dans les paramètres, Comparaison stochastique, Évaluation de performance, Encadrements des QoS, Traces de Trafic, Réseaux.

Abstract

This thesis is devoted to the uncertainty in probabilistic models, how it impacts their analysis and how to apply these methods to performance analysis and network dimensioning. We consider two aspects of the uncertainty. The first consists to study a partially specified Markov chains. We assume that the specifications of the model come from partial observations and from incomplete information on the transitions. The precision missing of some transitions in the exact system because of its complexity, can be solved by constructing bounding systems where worst-case transitions are defined to obtain an upper or a lower bound on the performance measures. We specify a "worst" chain and a "best" chain whose exact analysis allows us to provide answers on "worst" or "best " functioning of the system. We develop here new algorithms which gives element-wise bounds of the steady-state distribution for the partially specified Markov chain and also the mean time to failure (MTTF) for a set of absorbing discrete-time Markov chain partially specified. These algorithms are faster than the existing ones in the literature and allow us to compute element-wise bounds at each iteration.

The second aspect studied concerns the problem of the measurements of real traffic trace in networks. Exact analysis of queueing networks under real traffic becomes quickly intractable due to the state explosion. Assuming the stationarity of flows, we propose to apply the stochastic comparison method to derive performance measure bounds under histogram-based traffics. We apply an algorithm based on dynamic programming to derive optimal bounding traffic histograms on reduced state spaces according to a positive increasing reward function. Using the stochastic bound histograms and the monotonicity of the networking elements, we show how we can obtain, in a very efficient manner, guarantees on performance measures. We indeed obtain easier bounding stochastic processes providing stochastic upper and lower bounds on buffer occupancy histograms, response time distributions, losses, etc.. The interest and the impact of our method is shown on various applications: elements of networks (queue, routing elements), queue with Active Queue Management (AQM) mechanisms, queueing networks and queue with non-stationary arrival process. The proposed method provides very accurate results with a tradeoff between computation time and accuracy. Moreover, the derived performance bounds are very relevant in network dimensioning.

Keywords: Markov chains, Uncertainty, Stochastic Ordering, Performance analysis, Bounding methods, Traces, Networks.

Table des matières

Table des figures

Chapitre 1

Introduction

1.1 Motivations et objectifs

L'évolution croissante des systèmes informatiques, des systèmes de communication et des lignes de productions entraîne de nos jours des problèmes critiques de performances. Ainsi, afin de les garantir, une analyse fine du comportement est nécessaire pour identifier les problèmes et les résoudre. L'évaluation des performances d'un système s'effectue par un modèle reposant en général sur l'utilisation de chaînes de Markov. Ces modèles servent à représenter d'une manière aussi fidèle que possible l'état du système et permettent de prédire leur comportement par le calcul des mesures de performances. L'utilisation de ces modèles mathématiques représente la solution souvent adoptée dans la prise de décisions (choix de dimensionnement, techniques de partage de ressources, politique de maintenance, architectures, etc.). Il est donc fondamental de disposer de méthodes d'analyse permettant de comprendre le comportement de systèmes complexes, afin de répondre à des questions de performance et de coût. Plusieurs formalismes reposant sur l'utilisation de chaînes de Markov décrivant des systèmes complexes ont été développés au cours de ces dernières années. Parmi eux, nous citons les files d'attente [Kle76], les réseaux de Petri [Don93], les réseaux d'automates stochastiques (RAS) [Pla85] ou encore le formalisme d'algèbre de processus stochastiques [HK01]. Les chaînes de Markov peuvent également être enrichies avec des structures de récompenses sur les états et/ou les transitions afin d'exprimer des mesures fonctionnelles relatives à ces récompenses.

Malgré la diversité des méthodes d'analyses et l'évolution des technologies, l'étude de systèmes complexes présentant un facteur d'incertitude demeure un sujet exploratoire. En effet, le problème d'imprécision dans l'évaluation de performances des systèmes réels est souvent observé et rend très difficile l'analyse exacte de ces derniers. Cette problématique peut se manifester sous deux aspects. Le premier est retranscrit dans la difficulté de déterminer tous les paramètres du modèle. En effet, la modélisation d'un système stochastique quantitatif par une chaîne de Markov en temps discret nécessite la spécification d'une distribution initiale et de la matrice de probabilité de transition. Cependant, parfois, il peut être impossible ou irréaliste ou même trop coûteux de déterminer précisément ces probabilités. Dans les études d'évaluation des performances et de fiabilité, nous rencontrons souvent plusieurs situations dans lesquelles il n'est pas possible de préciser exactement les paramètres du modèle qui décrivent la chaîne

de Markov. L'information partielle peut concerner l'espace d'états et/ou les probabilités de transition. En outre, dans la phase de construction d'un modèle, les paramètres peuvent être estimés avec des intervalles de confiance à travers des mesures répétées. Par conséquent, les valeurs des paramètres sont définies par des intervalles, et ne sont pas appréhendées par une connaissance précise. De plus, ces modèles sont en général l'abstraction d'interactions complexes ou de dépendance des paramètres du système. Ainsi, une description par intervalles de valeurs pour les probabilités de transition serait plus appropriée.

Le second aspect réside dans le problème de mesures. En effet, les études ont globalement montré que les mesures de trafic réel sont particulièrement instables, et engendrent un nombre important de données, non toutes significatives, mais qui rendent difficile la caractérisation du modèle. Jusqu'à présent, la plupart des méthodes développées dans la littérature dépendaient principalement de la caractérisation statistique (le trafic est souvent décrit par une distribution connue, comme la distribution gaussienne (normale), la distribution de Poisson, etc.) engendrant ainsi des résultats inexacts et approximatifs. En effet, de nombreux travaux sont parvenus à montrer que certains trafics (trafic Internet) sont très loin des modèles simples poissonniens et markoviens utilisés pour modéliser les processus d'arrivée. En effet, le processus de Poisson, même s'il a été utilisé pour modéliser les réseaux téléphoniques (car les appels sont générés indépendamment les uns des autres), est incapable de représenter les rafales et les relations de dépendance qui existent entre les flux, les paquets et les pertes dans le réseau. Les propriétés d'auto-similarité [1] et de queue lourde [2] ont été montrées pour les trafics Internet. Plusieurs distributions ont été proposées pour s'adapter à ces caractéristiques de trafic. Les distributions de Pareto et Weibull sont souvent utilisées afin de refléter la distribution à queue lourde. La propriété d'auto-similarité peut être modélisée par superposition d'un grand nombre de sources ON/OFF dont les périodes ON et OFF sont à queue lourde. D'autres lois telles que les : "Markov Modulated Poisson Process" (MMPP) [KLL+02] ou "Switched Batch Bernoulli Process" (SBBP) [HTS91], plus complexes, permettent également de représenter une certaine variabilité du trafic. L'intérêt de ces lois d'arrivées est de pouvoir utiliser des modèles probabilistes tels que les chaînes de Markov afin de calculer des indices de performance (probabilités de perte, temps de réponse, etc.). Malheureusement, ces lois ne définissent pas forcément une représentation fidèle du comportement du réseau. De plus, lorsque le nombre de paramètres est élevé (souvent le cas pour les lois telles qu'une MMPP ou une loi de type phase), les modèles deviennent généralement insolubles, et recourent donc à une réduction du nombre de paramètres à utiliser. Cette démarche implique une perte de précision et les modèles construits ne vont par conséquent pas refléter de manière fiable de trafic réel. En effet, quand le nombre d'événements entraînant des changements d'état du système augmente, le problème devient certainement plus difficile. Par exemple, en performabilité, on tient compte à la fois des performances et de la fiabilité du système. Prenons l'exemple simple de la probabilité de rejet d'un appel téléphonique dans un commutateur obtenue par la formule d'Erlang B. Trivedi [Tri02] a considéré que chaque canal téléphonique peut-être sujet à des pannes avec réparation.

1. Auto-similarité : la structure des variations d'amplitude du signal analysé (par exemple le nombre de bits transférés par unité de temps) se reproduit de manière similaire quelle que soit la finesse temporelle avec laquelle il est représenté [WVMS96].

2. Queue lourde(heavy tail) : ce phénomène implique qualitativement la présence de dépendance longue dans la série des points analysés [WVMS96].

On se rend compte alors que l'étude de ce système devient vite complexe du fait que pour analyser le trafic, il faut considérer à la fois l'arrivée des appels et des pannes.

Certes, il existe des algorithmes d'approximation (appelés *fitting*) qui calculent les paramètres pour une famille de lois à partir d'un ensemble de mesures. Approche largement et souvent utilisée, le *fitting* a été développée, depuis plusieurs années et permet au travers des différentes données mesurées et récoltées, d'obtenir des lois de probabilité. Cependant, il s'agit fondamentalement d'une cause d'erreurs, car on cherche la loi la plus proche des mesures au sens d'un certain nombre d'estimateurs. La plupart des approches d'approximation citées dans la littérature sont fondées (au plus) sur les deux premiers moments de la distribution des mesures. Gupta et *al.* dans [GHBDZ10], ont prouvé que cette considération pouvait engendrer des erreurs importantes sur les performances d'un système. Considérant le temps d'attente moyen dans une file d'attente multi-serveur M/G/K, les auteurs ont montré qu'il était impossible de développer une approximation, fondée uniquement sur les deux premiers moments, qui soit précise pour toutes les distributions de service. Ils ont par exemple considéré une gamme de distributions (Weibull, log-normale, Pareto tronquée) utilisées dans la littérature pour modéliser les charges de travail des systèmes informatiques et comparer le temps d'attente moyen obtenu via des simulations et la méthode d'approximation fondée sur les deux premiers moments. Ils ont remarqué une énorme différence entre le temps moyen d'attente simulé et l'approximation des deux moments. Une étude approfondie a amené les auteurs a affirmer que les moments ne représentent pas une caractérisation idéale, sur laquelle on peut se reposer pour approximer les temps d'attente moyens. La séquence des moments peut s'avérer être utile uniquement si l'un des moments normalisés (de manière appropriée) est faible [GHBDZ10]. Par exemple, si la distribution des services dispose d'un troisième moment normalisé faible alors, une approximation fondée sur les deux premiers moments sera susceptible d'être précise. Cependant, il existe de nombreuses distributions de service telles que la distribution log-normale (dont les moments sont tous élevés), pour qui la méthode des moments n'est pas utile pour prédire avec précision le temps d'attente moyen.

Concrètement, dans une telle approche, nous n'avons aucun contrôle sur l'approximation et nous ne disposons d'aucune preuve de précision spécifiant que si l'on considère les trois premiers moments, les quatre premiers moments ou plus, nous pouvons garantir que la loi caractérisée représente correctement le comportement du trafic. Ce que nous proposons dans cette thèse, c'est d'employer une démarche radicalement différente et tout à fait nouvelle qui consiste à effectuer des réductions sur l'espace d'états en utilisant les bornes stochastiques. Cette réduction de taille va bien évidemment introduire un bruit de quantification, mais grâce à notre approche, nous garantissons l'obtention de bornes stochastiques, même si les mesures ont été biaisées avec un bruit positif. Pour résumer, notre approche va offrir la possibilité de contrôler la qualité d'un modèle abstrait fondé sur des mesures et va permettre d'équilibrer la complexité et la précision pendant le processus de résolution.

Ainsi, reposant sur la modélisation de systèmes par des chaînes de Markov, notre démarche vise à apporter des solutions de modélisation de systèmes complexes et de dimensionnement. L'objectif est de définir de nouvelles méthodes et de nouveaux algorithmes de résolution. Ainsi, dans le but de trouver des réponses à cette problématique, nous proposons d'organiser nos travaux selon les deux axes suivant :

– Une première partie consacrée à l'analyse des chaînes de Markov partiellement spécifiées. Cette partie est plutôt théorique et part du principe que les taux de transition ou des probabilités du processus étudié résultent de mesures et ne sont qu'approximativement connus sous forme de limites inférieures et supérieures. Dans ce cas, le calcul de bornes de la distribution stationnaire est une alternative au calcul exact des mesures du processus. Lorsqu'on est confronté à ce genre de problème (description par intervalles des taux de transition / probabilités), un certain nombre de questions se posent telles que : Que peut-on dire sur le modèle ? Quel impact a cette imprécision sur l'analyse du modèle ? Quelle méthode aborder pour l'analyser ? Quelles indications ou résultats peut on émettre sur les mesures de performance, de fiabilité ou de performabilité du modèle, etc. ?, autant de questions auxquelles nous allons tenter d'apporter des réponses.
– Et une deuxième partie dédiée à l'étude des réseaux. Nous considérons des traces de trafic réelles obtenues à partir de mesures du réseau et nous essayons d'étudier le problème d'imprécision du trafic à quantifier dû à la taille importante des traces. Nous allons tenter de développer une nouvelle méthode permettant de simplifier et d'analyser ces systèmes.

Contrairement à d'autres études qui reposent essentiellement sur la connaissance de tous les paramètres de la chaîne de Markov et qui ont pour but de trouver des méthodes ou des approches de résolution simplifiée des modèles complexes, nous nous intéressons dans cette thèse non seulement à la difficulté de résolution, mais également à l'impossibilité de construire le modèle. Notre objectif consiste à chercher un encadrement du processus original et définir une chaîne "pire" cas et une chaîne "meilleur" cas dont l'analyse nous permet d'apporter des informations sur le fonctionnement "au pire" ou "au mieux" du système. Notre but se résume donc à trouver un encadrement pertinent en construisant des systèmes bornants et en obtenant des bornes supérieures et/ou inférieures sur les indices de performance escomptés.

1.1.1 Partie I

Dans la première partie de cette thèse, nous proposons d'étudier des chaînes en temps discret ou temps continu dont les probabilités ou taux de transition ne sont pas parfaitement connus. Nous partons du principe que les spécifications du modèle proviennent d'observations partielles et d'informations incomplètes sur les transitions et nous voulons évaluer certaines mesures de performance et calculer, par exemple le taux moyen d'arrivée ou le taux d'utilisation du serveur dans une file d'attente. Typiquement les probabilités sont dans un intervalle ou elles proviennent de l'interaction de divers processus d'arrivée ou de service dont seuls la moyenne et le support sont connus. Une description partielle du système permet de préciser une borne inférieure sur les probabilités de transition en exhibant des conditions nécessaires pour qu'elles se produisent. De façon symétrique, des conditions suffisantes fournissent une borne supérieure. L'absence de certaines transitions dans le système exact à cause de sa complexité peut se résoudre par la construction de systèmes bornants où des transitions "pires des cas" sont définies pour obtenir une borne supérieure ou inférieure sur l'indice de performance à calculer. Nous pouvons donc dire que la chaîne de Markov partiellement spécifiée appartient à un ensemble de matrices stochastiques respectant toutes les spécifications, noté $\mathcal{B}$.

Cette problématique peut par exemple être observée dans le domaine de la fiabilité dans

un contexte légèrement différent. Les experts fiabilistes ont souvent besoin d'exprimer leurs incertitudes dans l'énoncé des valeurs de probabilités de défaillances et autre paramètre des systèmes. Cette incertitude se traduit donc par des probabilités de transition des chaînes de Markov définies par des intervalles. La connaissance de cette imprécision peut avoir beaucoup d'intérêt pour l'ingénieur lors de la conception de dispositif de sécurité. Dans les études d'évaluation des performances également, il est important de déterminer le comportement d'un modèle dans le "pire" et/ou dans le "meilleur" des cas. Par exemple, dans le dimensionnement de réseau, nous considérons des bornes de performance pire des cas, afin de garantir la qualité de service requise.

Nous cherchons donc à construire une borne supérieure (resp. inférieure) de tout élément de l'ensemble $\mathcal{B}$. Une analyse exhaustive n'est pas possible, car $\mathcal{B}$ est en général infini et non dénombrable. Le but est donc de généraliser le calcul d'une borne d'un élément selon un ordre donné à une borne d'un ensemble selon le même ordre. De nombreux travaux effectués dans ce sens existent dans la littérature. Nous citons principalement :

• L'algorithme proposé par Haddad et Moreaux [HM07] pour calculer une borne supérieure pour l'ordre stochastique fort ("st") quand l'ensemble est défini par deux bornes supérieures et inférieures par élément. Cette hypothèse ne correspond pas initialement à une chaîne mal spécifiée, mais à une hypothèse sur la difficulté d'obtenir les probabilités exactes de transition dans une approche numérique généralisant l'algorithme de Muntz [MDG89, LM94]. Plutôt que de déterminer exactement les probabilités (les formules sont connues), Haddad et Moreaux ont préféré les encadrer par des calculs plus rapides. Leur algorithme fournit cependant la réponse à notre problème si nous supposons que le modèle imprécis est spécifié par des encadrements de probabilités de transition. Le point fort de cet algorithme est de calculer avec la même complexité que l'algorithme de Vincent [AAV98] une borne pour un ensemble infini de matrices.

• La méthode de détermination de bornes sur les récompenses stationnaires. Cette approche a été proposée par Courtois et Semal dans [CS84, CS85, CS86], puis améliorée et généralisée par de nombreux auteurs [MDG89, LM94, Sem95, Buc06, Mah97, MR01]. Cette méthode ne s'intéresse qu'à l'état stationnaire et le calcul des bornes peut donc être vu comme un problème d'algèbre linéaire. Les bornes inférieures et supérieures sont obtenues en utilisant des techniques polyédrales.

• La comparaison des distributions stationnaires et/ou transitoires dont nous allons développer les aspects algorithmiques dans la première partie de cette thèse.

Notre idée consiste à borner un système imparfaitement connu par un autre système parfaitement connu. Pour ce faire, nous proposons de nouveaux algorithmes et développons des outils d'analyse quantitative.Ainsi, nous élaborons dans cette partie de nouveaux travaux dans ce contexte en proposant différents algorithmes de construction de bornes par élément sur le vecteur de distribution stationnaire de chaînes de Markov partiellement spécifiées. Des améliorations apportées à d'anciens algorithmes présentés dans [AFP12], nous ont permis de fournir à chaque itération des bornes supérieures et inférieures de la distribution stationnaire. En outre, des bornes sur la probabilité d'absorption et le temps moyen de premier passage de la chaîne originale sont également déterminés et présentés dans [AFP13].

1.1.2 Partie II

Au cours des dernières années, nous avons connu une croissance accrue du trafic dans les réseaux Internet. Essentiellement due à une forte demande des applications multimédia, des flux importants de trafic sont congestionnés dans les nœuds du réseau et entraînent des délais conséquents et des pertes considérables de paquets. Dans le but d'évaluer les performances de ces réseaux, une multitude de mesures de trafic sont disponibles pour tester différentes hypothèses. Cependant, ces mesures sont relativement difficiles à utiliser pour la modélisation de performance d'une manière efficace. En effet, les mesures de trafic sont extrêmement volumineuses et empêchent leur utilisation directes dans un modèle. Des problèmes d'incertitudes des paramètres d'entrée dus au volume des mesures peuvent également subvenir.

Nous proposons d'illustrer un exemple de trace de trafic réelle, nous considérons à cet effet la trace de trafic MAWI [SC00] correspondant à une heure de mesure de trafic IP à 150 Mb/s sur une ligne transpacifique, réalisée le 9 janvier 2007 entre 12 h et 13 h.

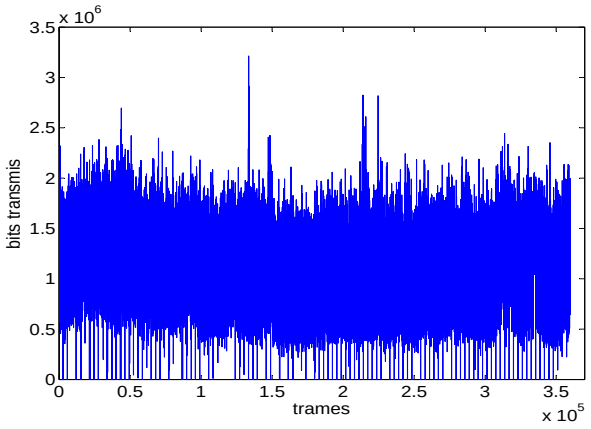


FIGURE 1.1 – Trafic MAWI pour une période de 10 ms.

Représentée pour un échantillonnage de période de 10 ms, cette trace est décrite sur un support de taille de 359.992 états. Le problème est bien entendu que la taille des traces est très grande et que cela peut avoir un impact négatif sur les algorithmes d'analyse des files d'attente.

Parmi les méthodes de résolution, il est toujours possible d'utiliser la simulation, mais cela ne représente pas vraiment un modèle abstrait, d'autant plus que le temps de résolution est relativement long.

Dans le cadre de cette thèse, nous proposons de nous intéresser aux modèles décrits par des histogrammes. Plusieurs travaux reposant sur cette approche ont été développés pour l'étude de performance des systèmes. Les modèles par histogrammes ont été introduits pour la première fois par Skelly et al. [SSD93] dans le domaine du *network calculus*, pour modéliser les sources vidéo, prévoir les distributions d'occupation du tampon ainsi que les taux de perte des cellules pour les flux multiplexés. Ensuite, Hernández et al. [HOVC07a, HOVC07b, HOVC09, HOVC10] ont proposé une nouvelle analyse de performance pour déterminer l'histogramme de la distribution d'occupation du tampon. Ils ont proposé une méthode d'approximation permettant d'analyser une file d'attente simple à travers un nouveau processus stochastique

appelé HBSP (Histogram Based Stochastic Process) dont l'espace d'état de la trace de trafic en entrée a été réduit. La méthode de réduction consiste à diviser l'espace d'état en sous-intervalles de longueur égale, afin d'en tirer une distribution de probabilité sur un espace d'état plus petit. Une autre approche reposant également sur la réduction de l'histogramme initial a été présentée par Tancrez et al. [TSC09]. Partant d'une distribution continue, les auteurs ont proposé de construire une distribution discrète borne supérieure qui modélise la durée de service dans une ligne de production. Le procédé utilisé par cette méthode permet de fournir des bornes pour l'ordre $\leq_{st}$.

Nous définissons dans cette thèse une nouvelle approche d'analyse. Partant d'un modèle pouvant être décrit par des distributions de probabilité discrètes, nous proposons de définir des modèles bornants plutôt que des approximations avec des supports réduits. Cette démarche peut alors laisser envisager la détermination de mesures de performance bornantes pouvant apporter des réponses sur le fonctionnement du système. Cette nouvelle méthode s'appuie sur l'utilisation de la comparaison stochastique pour déterminer des distributions bornantes, au sens de l'ordre stochastique fort, de la distribution stationnaire ou transitoire de la chaîne de Markov modélisant le système initial. L'avantage des méthodes de comparaison stochastique par rapport aux bornes utilisées habituellement, c'est justement de pouvoir donner des indications plus fines puisque l'on calcule une distribution et non pas le pire cas déterministe.

Nous avons pour ce faire développé un algorithme d'agrégation de distribution de probabilités permettant de générer des bornes optimales pour une fonction de récompense positive croissante. Fondé sur la programmation dynamique, cet algorithme présenté dans [ACC+12] permet d'obtenir un histogramme bornant sur un espace d'états réduit. Il permet également de contrôler la taille de l'espace d'états et de calculer la borne la plus précise au sens d'une fonction de récompense. En outre, les bornes correspondent à une analyse "meilleure" ou "pire" cas, et donnent également un aperçu quant aux problèmes d'incertitudes des paramètres d'entrée. Nous avons appliqué cette nouvelle méthode pour différents cas concrets d'évaluation des performances. Partant de l'hypothèse de stationnarité du trafic en entrée des systèmes, nous avons à cet effet, analysé les performances des réseaux informatiques en considérant les éléments de réseau suivant : file d'attente, élément de fusion (merge) et élément de division (split) des flux ainsi que certains mécanismes de gestion active de files d'attente. Ces études ont été présentées respectivement dans [ACFP13c, ACFP13d], [ACFP13a] et [ACFP13b]. La contribution apportée dans cette thèse consiste à développer une approche originale fondée sur la monotonie des éléments de réseau. En effet, nous considérons les histogrammes bornants ayant un espace d'état plus petit, afin de simplifier l'analyse, et de dériver des bornes sur des mesures de performance comme la moyenne des temps de réponse, les probabilités de perte, et la longueur du tampon. Nous proposons une solution très intéressante au problème de dimensionnement qui consiste à faire un compromis entre la précision et la complexité de calcul. En outre, nos bornes s'avèrent être très utiles lorsqu'on s'intéresse à la vérification de certaines contraintes sur la qualité de service (QoS).

Nous avons également omis l'hypothèse de stationnarité du trafic, en proposant une extension pour des trafics modulés (c.-à-d. une version simplifiée du type Switch Bernoulli Batch Process). Nous étudions l'impact de la méthode de bornes stochastique sur les mesures de performance d'une file d'attente simple.

Pour résumer, le dimensionnement d'un réseau requiert une bonne caractérisation du trafic offert à l'entrée du système. Le sujet est compliqué à cause de la difficulté d'obtenir des paramètres statistiques de la demande. L'approche proposée dans cette thèse est innovante. En effet, l'approche classique qui consiste à partir avec des hypothèses markoviennes plus ou moins justifiées conduit également à des processus aléatoires difficiles à paramétrer. L'approche proposée ici est nouvelle dans le sens où la caractérisation de la demande sera obtenue via l'étude de bornes stochastiques ayant de bonnes propriétés. L'étude vise des objectifs qui présentent un intérêt aussi bien théorique que pratique avec une mise en oeuvre d'une implémentation informatique des algorithmes. Les réseaux et les services devenant de plus en plus complexes, les retombées de cette étude dans le domaine de dimensionnement de réseau de télécommunication seront indéniablement très intéressantes.

1.2 Organisation du document

Les études présentées dans cette thèse sont de nature diverse et requièrent des outils mathématiques différents. Dans le chapitre 2, nous présentons les définitions et les principaux résultats préliminaires utilisés dans cette thèse. Nous nous intéressons essentiellement à la description des ordres stochastiques sur lesquels repose la notion de borne stochastique. Nous y rappelons les résultats existants qui permettent, à partir de la comparaison stochastique de deux variables aléatoires, d'aboutir à la comparaison et à la monotonie stochastique des chaînes de Markov à partir de leurs matrices de transitions. Ces résultats ont été utilisés par Trémolières, Vincent et Plateau pour l'élaboration de l'algorithme de construction de borne stochastique [TVP92] que nous décrivons en détail à la fin de ce chapitre.

Afin d'investiguer les deux problématiques énoncées précédemment, ce document comporte deux parties. La première expose les études effectuées sur les chaînes de Markov partiellement spécifiées et la deuxième décrit en détail l'approche que nous proposons pour analyser et résoudre les modèles dont les mesures de trafic sont très importantes. Cette nouvelle méthode est mise à l'épreuve de manière systématique dans les chapitres 6, 7, 8, 9 et 10.

Partie I : Chaînes de Markov partiellement spécifiées

Nous résumons dans les chapitres 3 et 4 les principaux résultats obtenus et développés dans le cadre de construction de bornes pour une chaîne de Markov partiellement définie.

Chapitre 3 : Méthodes de bornes pour les chaînes de Markov partiellement spécifiées. Les incertitudes sur les probabilités de transition conduisent à un ensemble de chaînes de Markov exprimées par une matrice sous-stochastique $\mathbf{L}$ pour les bornes inférieures sur les probabilités et une matrice sur-stochastique, $\mathbf{U}$ pour les bornes supérieures sur les probabilités. Ces matrices représentant, dans un certain sens, le meilleur et le pire des cas dans l'ensemble et ne correspondent pas à de bons candidats pour modéliser le comportement d'une chaîne de Markov dont les probabilités de transition sont dans cet intervalle. En effet, elles ne sont pas stochastiques. Le but serait donc d'utiliser ces deux matrices pour déterminer des bornes pertinentes sur les distributions stationnaires de la chaîne partiellement définie.

Dans la littérature, plusieurs méthodes de bornes des chaînes de Markov imprécises ont été développées. Nous présentons dans ce chapitre un état de l'art des méthodes existantes répondant à cette problématique. Utilisant des critères complètement différents, les algorithmes proposés par les différentes méthodes énoncées ne sont pas optimisés selon les mêmes critères et ne renvoient donc pas le même type de résultats. Ainsi, nous proposons d'effectuer une étude comparative entre les différents algorithmes, démarche qui n'a pas était abordée auparavant. Nous considérons pour ce faire, des bornes sur une fonction de récompense positive croissante des vecteurs de distribution stationnaires, et nous les comparons en terme de précision et de complexité de calcul.

Chapitre 4 : Nouvelles bornes par élément pour les chaînes de Markov. Pour calculer des bornes sur la distribution stationnaire d'une DTMC imprécise, nous pouvons recourir aux méthodes existantes à savoir : l'approche développée par Muntz [FM94] fondée sur la théorie polyhédrale de Courtois et Semal [CS84] est basée sur la résolution de n chaînes de Markov où n est la taille de l'espace d'état. La résolution de ces n chaînes nécessite un calcul numérique via un algorithme précis. Si nous utilisons l'algorithme GTH [GTH85] pour la résolution de chaque chaîne, nous obtenons une complexité totale de n^4 vu que ce dernier est une variante de l'élimination de Gauss et a une complexité cubique. L'approche de Buchholz proposé dans [Buc05] peut également être envisagée. Cette approche utilise à la fois les matrices $\mathbf{L}$ et $\mathbf{U}$ pour définir l'ensemble des matrices. L'utilisation des deux matrices $\mathbf{L}$ et $\mathbf{U}$, permet ainsi de définir des bornes plus précises. Cependant, cet algorithme a été rapporté par l'auteur dans [Buc10] être numériquement instable. De plus, cet algorithme nécessite beaucoup plus d'étapes de calcul que l'algorithme que nous proposons ici.

Nous proposons dans ce chapitre, de nouveaux algorithmes permettant de déterminer des bornes supérieures et inférieures par élément de la distribution stationnaire d'une chaîne de Markov ergodique. Fondés sur l'approche de Muntz [MDG89, LM94] et sur le nouvel algorithme itératif de Bušić et Fourneau [BF11], nos algorithmes fournissent à chaque itération des bornes de la distribution stationnaire de la chaîne de Markov imprécise. Ainsi, nos algorithmes sont beaucoup plus rapides et si nous itérons jusqu'à la convergence, nous obtenons les mêmes bornes que celles obtenues par Muntz. Il est également à noter que nos algorithmes sont numériquement stables, car ils sont seulement basés sur le produit de vecteurs non négatifs. Nous abordons également dans ce chapitre le cas de chaînes de Markov absorbantes et imprécises. Nous présentons à cet effet un algorithme pour trouver des bornes supérieures et inférieures sur les temps moyens avant panne (MTTF) pour un ensemble de DTMCs absorbantes.

Partie II : Simplification des distributions discrètes par l'approche des bornes stochastiques

Chapitre 5 : Spécification des variables aléatoires discrètes et leur réduction. Nous introduisons dans ce chapitre les modèles par histogramme. Cette caractérisation permet de représenter par des variables aléatoires discrètes (également appelées histogrammes) les trafics en entrée du système. Une brève description des méthodes récentes développées dans ce sens est présentée.

Chapitre 6 : Complexité ou précision : une approche liée à la comparaison stochastique. Ce chapitre est consacré au développement et à la description d'une nouvelle méthode de bornes stochastiques. Partant d'une distribution de probabilité discrète définie sur N états, nous construisons des distributions bornantes de tailles de supports plus petites. Différents algorithmes sont ainsi développés et ont pour but de calculer une distribution borne stochastique inférieure et une distribution borne stochastique supérieure de la distribution initiale. Nous nous focalisons plus particulièrement sur l'algorithme optimal. Fondé sur la programmation dynamique, cet algorithme fournit des bornes optimales au sens d'une récompense donnée et permet de définir un encadrement très pertinent de la distribution exacte et cela pour des tailles de réduction relativement petite.

Nous illustrons cette approche en proposant d'étudier deux problèmes. Le premier est un problème de recherche opérationnelle et consiste à déterminer la distribution de la durée d'exécution dans un graphe de tâches stochastique et le second traite le problème de la détermination du temps d'attente décrit par l'équation de Loynes pour l'analyse rapide d'une file d'attente FIFO. L'étude de ces deux exemples via notre méthode, a permis de montrer que les distributions de supports réduits développées, offrent la possibilité de faire un compromis entre la précision et la complexité de calcul.

Chapitre 7 : Analyse d'une file d'attente simple. Nous étendons l'application de notre approche de bornes au domaine des files d'attente. Pour le cas d'une file d'attente simple, nous présentons une méthode d'évaluation de performance reposant sur la détermination de distributions bornantes au sens de l'ordre stochastique fort. Reposant sur la stationnarité du trafic d'entrée, nous étudions la propriété de monotonie de l'élément de réseau FIFO (file d'attente isolée) et utilisons les histogrammes réduits dans les équations d'évolution afin d'obtenir des bornes sur les mesures de performance. Ainsi, nous obtenons des bornes qui fournissent des valeurs supérieures et inférieures pour les mesures de performance pertinentes pour la vérification des contraintes de qualité de service.

Nous utilisons pour nos évaluations numériques des traces de trafic réelles en entrée. Sachant que les histogrammes associés à ces traces sont définis sur un nombre d'états très important, l'analyse exacte de ces derniers peut s'avérer très coûteuse. L'utilisation de notre approche de bornes stochastiques nous permet alors de définir des histogrammes bornants, dont nous pouvons paramétrer la taille, ce qui impacte à la fois la complexité et la qualité des calculs.

Chapitre 8 : Modélisation des éléments de réseaux. Considérant toujours le cas d'une file d'attente isolée, le chapitre 8 propose de s'intéresser à deux extensions. La première concerne les éléments de routage : division (split : division d'un flux en entrée en plusieurs flux de sortie) et fusion (merge : combiner plusieurs flux en un seul), et la seconde porte sur l'étude de certains mécanismes de gestion active de files d'attente (AQM).

Pour commencer, nous décrivons dans ce chapitre les chaînes de Markov via la construction de leurs matrices de transition. Nous démontrons la propriété de monotonie de certaines mesures de performance telles que la distribution des temps de réponse, la distribution des pertes, etc. Nous passons ensuite à l'étude des deux extensions énoncées. La première extension

nous permet d'envisager l'application de notre méthode de bornes à l'analyse de réseaux feed-forward. En effet, en garantissant la propriété de monotonie de ces éléments, nous pouvons préserver l'ordre de comparaison stochastique du (des) flux en sortie. Ainsi, en considérant un réseau de topologie DAG (*Directed Acyclic Graph*), nous pouvons assurer et préserver la relation d'ordre des mesures de performances dans le réseau. Pour la deuxième extension, notre but consiste à étudier la propriété de monotonie et l'apport que peut offrir notre méthode de bornes stochastiques à la résolution d'un modèle de file d'attente avec mécanisme de gestion.

Chapitre 9 : Analyse des réseaux de files d'attente. Nous abordons l'étude des réseaux de files d'attente ayant une topologie feed-forward (DAG). Nous utilisons dans ce chapitre deux approches distinctes d'analyse. La première consiste à utiliser une technique par décomposition. Elle revient à analyser, selon l'ordre topologique, chaque élément de réseau de façon isolée puis de recomposer à la fin les différents résultats calculés pour déterminer les performances globales du réseau. Cette approche est souvent utilisée pour évaluer les performances des réseaux ouverts, même si elle n'est qu'une approche approximative. La seconde approche revient à utiliser une analyse de bornes sur les paramètres de sortie du réseau. Cette approche a été développée dans le but de définir des bornes plutôt que des approximations sur les paramètres de sortie du réseau (chose que fait la première approche).

Distinguant ces deux approches, notre objectif reste évidemment clair et consiste à étudier et démontrer l'avantage qu'offre notre méthode de bornes à l'analyse des réseaux de files d'attente.

Chapitre 10 : Analyse d'une file d'attente simple avec processus d'arrivée non-stationnaire. L'utilisation de notre méthode de bornes stochastiques nous a permis d'obtenir de résultats très intéressants sur les systèmes de files d'attente ayant des processus d'arrivée stationnaires. N'ayant pas étudié son applicabilité aux processus non stationnaires, nous proposons de nous y intéresser dans ce chapitre. Considérant une file d'attente isolée, nous développons notre analyse pour le cas d'un processus d'entrée *Switched Batch Bernoulli Process* (SBBP). Les résultats de monotonie stochastique prouvés pour le modèle ainsi que les évaluations numériques effectuées nous permettent alors d'aboutir aux mêmes conclusions énoncées dans les chapitres précédents. À savoir que l'utilisation de notre méthode peut s'avérer très intéressante dans l'analyse de performance des files d'attente.

Le dernier chapitre est consacré aux conclusions de nos travaux, ainsi qu'à des idées d'extensions possibles des travaux actuels.

Liste des publications

La plupart des résultats de la Partie I ont été publiés dans :

- Farah Aït-Salaht, Jean-Michel Fourneau, et Nihal Pekergin. Computing entry-wise bounds of the steady-state distribution of a set of markov chains. In *27th International Symposium on Computer and Information Sciences*, *Paris, France*, pages 115-122. Springer, 2012. [AFP12]

- Farah Aït-Salaht, Jean-Michel Fourneau, et Nihal Pekergin. Computing bounds of the MTTF for a set of Markov chains. In *28th International Symposium on Computer and Information Sciences, Paris, France*, Lecture Notes in Electrical Engineering, pages 67-76. Springer, 2013. [AFP13]

Les résultats présentés dans la Partie II sont publiés dans :

- F. Aït-Salaht, J. Cohen, H. Castel Taleb, J. M. Fourneau, et N. Pekergin. Accuracy vs. complexity : the stochastic bound approach. In *11th International Workshop on Disrete Event Systems (WODES 2012)*, pages 343-348, 2012. [ACC+12]
- F. Aït-Salaht, H. Castel Taleb, J. M. Fourneau, et N. Pekergin. Stochastic bounds and histograms for network performance analysis. In *10th European Workshop on Performance Engineering (EPEW'13)*, volume 8168, pages 13-27, 2013. [ACFP13c]
- F. Aït-Salaht, H. Castel Taleb, J. M. Fourneau, et N. Pekergin. Une approche combinant bornes stochastiques, traces et histogrammes pour l'analyse de performance des réseaux. In *Journal européen des systèmes automatisés Volume 47 N° 1-2-3/Janvier-Mai 2013 : Modélisation des systèmes réactifs (MSR 2013)*, pages 165-179. Germes Lavoisier, 2013. [ACFP13d]
- F. Aït-Salaht, H. Castel Taleb, J. M. Fourneau, et N. Pekergin. A bounding histogram approach for network performance analysis. In *15th IEEE International Conference on High Performance Computing and Communications (HPCC'13)*, China, 2013. [ACFP13a]
- F. Aït-Salaht, H. Castel Taleb, J. M. Fourneau, et N. Pekergin. Modeling networks and active queues management with stochastic bounds and histograms. In *7th International Workshop on Verification and Evaluation of Computer and Communication Systems (VeCoS 2013), Florence, Italy*, 2013. [ACFP13b]
- J. M. Fourneau and F. Aït-Salaht. Bornes sur les histogrammes et équation de Loynes pour l'analyse rapide d'une file FIFO. In *ROADEF - 15ème congrès annuel de la Société française de recherche opérationnelle et d'aide à la décision, Bordeaux, France*, 2014. [FACP14]
- F. Aït-Salaht, H. Castel Taleb, J. M. Fourneau, et N. Pekergin. Approche par bornes stochastiques et histogrammes pour l'analyse de performance des réseaux. In *10ième Atelier en Évaluation de Performances, Sophia-Antipolis, France*, 2014. [AFCP14]

[HTB05]

Chapitre 2

Pré-requis

Dans ce chapitre, nous introduisons les notions préliminaires utilisées tout au long de la thèse. Le lecteur est supposé être familier avec la théorie classique des probabilités qui peuvent être trouvées dans [Bil95, Dud02]. Nous commençons dans la section 2.1 par introduire les modèles Markoviens, les chaînes de Markov à temps discret et à temps continu, à espace d'états fini. Les chaînes de Markov à temps discret sont les principaux modèles, auxquels nous nous sommes intéressés dans notre recherche. Nous discutons également des probabilités transitoires et stationnaires des chaînes de Markov et des méthodes permettant de les calculer. Nous présentons dans la section 2.2 les notions de base et les principaux résultats concernant la méthode de comparaison stochastique. Pour plus d'informations sur cette méthode, nous nous référons aux livres de Muller et Stoyan [MS02] et de Shaked et Shantikumar [SS94]. Nous nous intéressons particulièrement aux propriétés de l'ordre stochastique fort noté $\leq_{st}$. Dans le cadre de nos travaux de thèse, nous considérons uniquement un espace d'états fini et totalement ordonné. Ensuite nous présentons la comparaison des DTMCs. Nous présentons également la monotonie stochastique [KK77] qui est une propriété clé pour la comparaison des chaînes de Markov homogènes. Nous donnerons un algorithme simple de construction d'une borne supérieure et inférieure $\leq_{st}$ monotone d'une matrice stochastique arbitraire.

2.1 Modèles Markoviens

Les chaînes de Markov sont un cas particulier des processus stochastiques. Par conséquent, nous allons d'abord présenter brièvement ces derniers, et expliquer l'ensemble des conditions nécessaires pour qu'un processus stochastique soit appelé une chaîne de Markov. Puis, nous procédons à une classification des états de la chaîne et donnons les définitions des chaînes de Markov à temps discret et continu ainsi que les méthodes de calcul de leurs probabilités transitoires et stationnaires. Pour plus d'informations sur les chaînes de Markov nous nous référons aux ouvrages tels que [Tij03, BGDMT98]. La plupart des résultats présentés dans cette section peuvent être trouvés dans [Ste95].

Un *processus stochastique* est une famille de variables aléatoires $\{X(t) \mid t \in \mathcal{T}\}$ définies sur un espace de probabilité et indexées par un paramètre t qui peut prendre ces valeurs dans $\mathcal{T}$. Généralement, t est supposé représenter le temps. Les valeurs de $X(t)$ sont appelées états.

L'ensemble des états possibles du processus stochastique est appelé *espace d'état* et est noté $\mathcal{S}$. L'espace d'état peut être continu ou discret. Dans un premier temps, nous traitons le processus stochastique en temps continu, le processus stochastique en temps discrets sera traité après, ce dernier est appelé une chaîne et pour plus de commodité, nous supposons que $\mathcal{S} = \{0, 1, 2, \ldots\}$ est l'espace d'état. Une classification similaire peut être faite en ce qui concerne l'ensemble des indices $\mathcal{T}$. Un ensemble dénombrable mène au processus stochastique discret dans le temps alors qu'un ensemble continu engendre le processus stochastique à temps continu.

Un processus stochastique est appelé un processus de Markov si pour tout $t_0 < \ldots < t_n < t_{n+1}$ la distribution de $X(t_{n+1})$, pour les valeurs de $X(t_0), \ldots, X(t_n)$ ($s_0, \ldots, s_n \in \mathcal{S}$ respectivement), ne dépend que de $X(t_n)$, *i.e.,* pour tout $s_{n+1} \in \mathcal{S}$:

$$\mathrm{Prob}(X(t_{n+1}) \leq s_{n+1} \,|\, X(t_n) = s_n) = \mathrm{Prob}(X(t_{n+1}) \leq s_{n+1} \,|\, X(t_0) = s_0, \ldots, X(t_n) = s_n). \tag{2.1}$$

Cette équation est généralement connue comme étant *la propriété de Markov*. Le plus souvent, les processus de Markov utilisés pour le modèle sont invariants aux changements de temps, *i.e.,* pour tout $t, t' \in \mathcal{T}$, tel que $t' > t$, et $s', s \in \mathcal{S}$, nous avons :

$$\mathrm{Prob}(X(t') \leq s \,|\, X(t) = s') = \mathrm{Prob}(X(t' - t) \leq s \,|\, X_0 = s'). \tag{2.2}$$

Dans ce cas, nous avons un processus de Markov homogène dans le temps pour lequel l'état suivant ne dépend que de l'état actuel, ni sur les états antérieurs, ni sur combien de temps nous avons déjà été dans la situation actuelle.

Dans cette thèse, nous considérons une chaîne de Markov comme étant un processus de Markov homogène dans le temps avec un espace d'état discret $\mathcal{S}$ et un ensemble d'indices $\mathcal{T} = \mathbb{R}_{\geq 0}$ pour le temps continu ou $\mathcal{T} = \mathbb{N}_{\geq 0}$ pour le temps discret. En plus, sauf indication contraire, nous supposons que l'espace d'état est fini $\mathcal{S} = \{1, \ldots, N\}$ avec $|\mathcal{S}| = N$. Les conditions (2.1) et (2.2) signifient que dans un processus de Markov en temps homogène, les temps de séjour d'un état doivent être des variables aléatoires qui ont une distribution sans mémoire. Avant de présenter plus en détails les chaînes de Markov en temps continu et en temps discret, nous présentons quelques définitions utiles. Les chaînes de Markov comportent des propriétés très intéressantes, qui sont définies à travers les états qui les constituent.

Définition 2.1 *Une chaîne de Markov est dite irréductible si et seulement si de tout état s on peut atteindre tout état s' en un nombre fini d'étapes :*

$$\forall\, s, s' \in \mathcal{S},\ \exists\, t \in \mathcal{T} \mid \mathrm{Prob}(X(t) = s' \mid X(0) = s) > 0.$$

En d'autre termes, l'irréductibilité signifie que chaque état est accessible à partir de tout autre état de la chaîne.

Définition 2.2 *Un état $s \in \mathcal{S}$ d'une chaîne de Markov est dit absorbant si pour tout $t \in \mathcal{T}$ et $s' \in \mathcal{S}$ tels que $s \neq s'$ nous avons* $\mathrm{Prob}(X(t) = s' \,|\, X(0) = s) = 0$.

Clairement, un état absorbant est un état pour lequel on a une probabilité nulle d'en sortir.

Définition 2.3 *Une chaîne de Markov est appelée absorbante si pour tout état non-absorbant $s \in \mathcal{S}$ il existe un état absorbant $s' \in \mathcal{S}$ et $t \in \mathcal{T}$ tel que* $\mathrm{Prob}(X(t) = s' \mid X(0) = s) > 0$.

Ce qui revient à dire qu'une chaîne de Markov est absorbante si elle a au moins un état absorbant, et si à partir de chaque état, il est possible de passer à un état absorbant (pas nécessairement en une seule étape). Remarquons qu'une chaîne de Markov qui contient au moins un état absorbant n'est pas irréductible.

Définition 2.4 *Un état $s \in \mathcal{S}$ d'une chaîne de Markov est dit transitoire si nous avons :*

$$\lim_{t\to\infty}(\mathrm{Prob}(X(t) = s \mid X(0) = s)) < 1.$$

La limite ci-dessus indique que la probabilité de revenir à un état après l'avoir quitté est inférieure à 1.

Définition 2.5 *Un état $s \in \mathcal{S}$ d'une chaîne de Markov est dit récurrent si nous avons :*

$$\lim_{t\to\infty}(\mathrm{Prob}(X(t) = s \mid X(0) = s)) = 1.$$

En fonction de la durée de retour à un état s, nous distinguons les états récurrents nuls où la durée moyenne de retour à s est infinie, et les états récurrents non nuls où la durée moyenne de retour en s est finie. Dans une chaîne irréductible finie, tous les états sont récurrents non nuls.

Après avoir introduit les principales définitions, nous allons procéder à présent à la représentation formelle des chaînes de Markov à temps discret et à temps continu. Nous allons également présenter les méthodes généralement utilisées pour calculer leurs probabilités transitoires et stationnaires.

2.1.1 Chaînes de Markov à temps discret

Une Chaîne de Markov à temps discret (notée DTMC, en anglais *Discrete Time Markov Chain*) $\{X(n), n \geq 0\}$ est un processus de Markov dont l'espace d'état est fini ou infini dénombrable et le temps est discret, $n \in \mathbb{N}$.

Définition 2.6 (Chaîne de Markov à temps discret) *Une chaîne de Markov à temps discret, notée DTMC, est définie par :*

- *$\mathcal{S}$: un ensemble d'états fini ou infini dénombrable,*
- *Une matrice $\mathbf{P} : \mathcal{S} \times \mathcal{S} \mapsto [0; 1]$ appelée matrice de transition, telle que la probabilité de passage de l'état s vers l'état s' correspond à $\mathbf{P} = (\mathbf{P}[s, s'])_{s,s'\in\mathcal{S}} = \mathrm{Prob}(X(n+1) = s' \mid X(n) = s),\ \forall s, s' \in \mathcal{S}$. De plus, la matrice de transition $\mathbf{P} = \mathbf{P}[s, s']_{s,s'\in\mathcal{S}}$ associée à la DTMC vérifie les propriétés suivantes :*
  - *Positive :* $\forall s, s' \in \mathcal{S}\ \ 0 \leq \mathbf{P}[s, s'] \leq 1,$
  - *Stochastique :* $\sum_{s'\in\mathcal{S}} \mathbf{P}[s, s'] = 1,\ \forall s \in \mathcal{S}$
- *Un vecteur $\boldsymbol{\pi} : \mathcal{S} \mapsto [0; 1]$ appelé distribution initiale, tel que :*

$$\sum_{s\in\mathcal{S}} \boldsymbol{\pi}[s] = 1$$

Soit $X = \{X(n), n \in \mathbb{N}\}$ la chaîne de Markov en temps continu associé à une matrice de transition $\mathbf{P}$ et une distribution initiale $\boldsymbol{\pi}$. La loi de X est définie par les équations suivantes :

$$\forall\, s \in \mathcal{S},\ \ \mathrm{Prob}(X(0) = s) = \boldsymbol{\pi}[s]$$

$$\forall\, s,\, s' \in \mathcal{S},\ \ \mathrm{Prob}(X(1) = s' \,|\, X(0) = s) = \mathbf{P}[s,\, s']$$

Une DTMC peut se représenter par un graphe orienté appelé *graphe représentatif*, ou *graphe de transition*, de la chaîne de Markov. Les sommets de ce graphe représentent les états de la chaîne et les valeurs sur les arcs (transitions) représentent la probabilité avec laquelle la transition est prise.

Exemple 2.1 *La matrice de transition d'une chaîne de Markov est :*

$$\mathbf{P} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1/3 & 1/6 & 1/4 & 1/4 & 0 \\ 0 & 1/2 & 0 & 0 & 1/2 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}.$$

L'ensemble des états de la chaîne est $\mathcal{S} = \{1, 2, 3, 4, 5\}$. Le graphe représentatif est donné par :

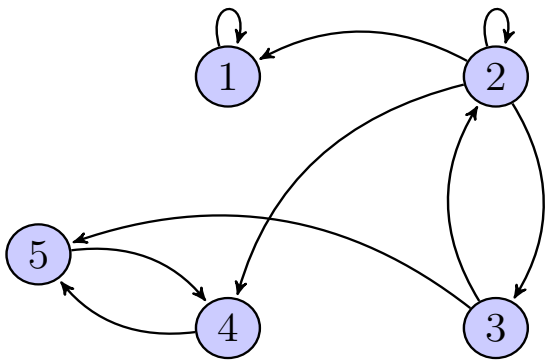


FIGURE 2.1 – Graphe de transition associé à la chaîne de Markov, de matrice de transition $\mathbf{P}$.

Propriété 2.1 *Soit $\mathbf{P}$ une matrice stochastique de taille finie, alors toute puissance $\mathbf{P}^m,\ m \geq 0$, de $\mathbf{P}$ est une matrice stochastique.*

Définition 2.7 (Mesures de probabilité) *Une DTMC est définie par des distribution transitoires qui décrivent le comportement de la chaîne à un instant n et une distribution stationnaire (si elle existe) qui décrit le comportement de la chaîne à l'infini. Soit $\boldsymbol{\pi}(n)$ la distribution transitoire de la DTMC considérée à l'instant n, et soit $\boldsymbol{\pi}(0)$ sa distribution initiale. La distribution transitoire s'obtient par :*

$$\boldsymbol{\pi}(n) = \boldsymbol{\pi}(n-1)\,\mathbf{P} = \boldsymbol{\pi}(0)\,\mathbf{P}^n. \tag{2.3}$$

La distribution stationnaire décrit le comportement de la chaîne à l'infini, $lim_{n\to\infty}\boldsymbol{\pi}(n)$. Le comportement transitoire d'une DTMC n'est pas difficile à définir. En effet, la distribution transitoire peut être calculée à travers la relation de récurrence de l'équation 2.3. En prenant en compte la propriété de Markov et d'homogénéité que la probabilité d'atteindre un état s' depuis s en k étapes se calcule ainsi :

$$\mathrm{Prob}(X(n+k) = s' \,|\, X(n) = s) = \mathbf{P}^k[s,\, s']$$

où $\mathbf{P}^k$ est la k-ième puissance associée au produit matriciel. En revanche, l'analyse du comportement stationnaire dépend des caractéristiques de la chaîne considérée. Avant d'énoncer le théorème 2.1 qui stipule que quand la DTMC admet une unique limite et une distribution stationnaire, nous devons définir la notion d'apériodicité de la DTMC. Cette dernière est effectuée en utilisant la notion d'états périodiques.

Définition 2.8 *Un état s de la DTMC est dit périodique si pour un $d > 1$, il s'avère que pour tout $n \in \mathbb{N}_{\geq 1}$, tel que $n \bmod d \neq 0$, la probabilité de revenir à l'état s en n étapes est 0.*

Définition 2.9 *La DTMC est appelée périodique si un de ses états est périodique.*

Définition 2.10 *La DTMC est appelée apériodique si elle n'est pas périodique.*

Notant qu'une condition suffisante pour une DTMC irréductible (définition 2.1) pour être apériodique est qu'il existe au moins un état avec une boucle.

Théorème 2.1 *Une chaîne de Markov à temps discret $\{X(n), n \geq 0\}$ qui est irréductible, apériodique et homogène dans le temps est dite ergodique. Pour une chaîne de Markov ergodique, les probabilités limites :*

$$\boldsymbol{\pi}[j] = \lim_{n\to\infty} \mathbf{P}[X(n) = j] \qquad j = 0, 1, \ldots$$

existent et sont indépendantes de la distribution de probabilité initiale. Les probabilités stationnaires $\boldsymbol{\pi}[j]$ sont uniques et sont déterminées par les équations suivantes :

$$\begin{aligned} \textstyle\sum_j \boldsymbol{\pi}[j] &= 1 \\ \boldsymbol{\pi}[j] = \textstyle\sum_i \boldsymbol{\pi}[i]\, \mathbf{P}[i,\, j] &\quad j = 0,\, 1,\, 2, \ldots,\, N. \end{aligned} \tag{2.4}$$

La résolution du système 2.4 pour le calcul de la distribution de probabilités stationnaires $\boldsymbol{\pi}$ peut être effectuée en utilisant plusieurs techniques numériques (voir par exemple [Ste95]).

Résolution numérique des équations d'équilibres

Il existe principalement deux types de méthodes utilisées pour calculer le vecteur de distribution stationnaire d'une DTMC : les méthodes directes et les méthodes itératives. Les méthodes directes permettent de déterminer les solutions exactes et sont généralement adaptées aux DTMCs ayant un espace d'état relativement petit. Les méthodes itératives sont plus fréquemment utilisées pour la plupart des DTMCs qui ont un grand espace d'état, ce qui se présente assez souvent dans les systèmes actuels. Dans cette thèse, nous nous sommes particulièrement intéressés à l'utilisation de deux de ces méthodes à savoir : la méthode GTH et la méthode des puissances que nous présentons ci-après.

a) Algorithme GTH (méthode directe)

L'algorithme GTH (Grassmann-Taksar-Heyman) [GTH85] représente une variante de l'élimination de Gauss et est connu pour sa stabilité numérique, car il n'utilise ni nombres négatifs ni soustractions. Cette méthode repose sur la suppression des états (sommets) à chaque itération depuis le dernier état N.
La procédure commence par l'élimination de $\boldsymbol{\pi}[N]$ de la N-ième équation afin d'obtenir

$$\boldsymbol{\pi}[N] = \sum_{i=0}^{N-1} \boldsymbol{\pi}[i]\,\mathbf{P}[i,\,N]/(1-\mathbf{P}[N,\,N]).$$

En remplaçant $1 - \mathbf{P}[N, N]$ par $\sum_{j=0}^{N-1} \mathbf{P}[N, j]$ et en utilisant la propriété $\boldsymbol{\pi}[j] = \sum_{i=0}^{N-1} \boldsymbol{\pi}[i]\,\mathbf{P}^{(N-1)}[i, j]$, $j = 0, 1, 2, \ldots, N-1$, nous avons :

$$\mathbf{P}^{(N-1)}[i,\,j] = \mathbf{P}[i,\,j] + \frac{\mathbf{P}[i,\,N]\,\mathbf{P}[N,\,j]}{(1-\mathbf{P}[N,\,N])} = \mathbf{P}[i,\,j] + \frac{\mathbf{P}[i,\,N]\,\mathbf{P}[N,\,j]}{\sum_{j=0}^{N-1}\mathbf{P}[N,\,j]}.$$

C'est le principe selon lequel la réduction de l'état fonctionne. Généralement, $\boldsymbol{\pi}[n] = \sum_{i=1}^{n-1} \boldsymbol{\pi}[i]\,\mathbf{R}[i, n]$ pour $n = N, N-1, N-2, \ldots, 2, 1$ où $\mathbf{R}[i, n] = \mathbf{P}^n[i, n]/\sum_{j=0}^{n-1}\mathbf{P}^n[n, j]$.
L'algorithme peut être exprimé comme suit :

ALGORITHME GTH

---

1. **Pour** $n = N, N-1, N-2, \ldots, 2, 1$, **faire**
   $\mathbf{R}[i, n] = \mathbf{P}[i, n]/\sum_{j=0}^{n-1}\mathbf{P}[n, j], \quad i = 0, 1, \ldots, n-1;$
   $\mathbf{P}[i, j] = \mathbf{P}[i, j] + \mathbf{R}[i, n]\,\mathbf{P}[n, j], \quad i, j = 0, 1, \ldots, n-1;$
   $\mathbf{g}[1] = 1;$

2. **Pour** $j = 1, 2, \ldots, N, \quad \mathbf{g}[j] = \sum_{i=0}^{j-1}\mathbf{g}[j]\,\mathbf{R}[i, j];$

3. **Pour** $j = 2, 3, \ldots, N, \quad \boldsymbol{\pi}[j] = \mathbf{g}[j]/\sum_{i=0}^{N}\mathbf{g}[i].$

Nous remarquons que cette méthode a une complexité cubique en temps et quadratique en espace pour les matrices pleines. Et donc peu adaptée aux matrices ayant des tailles très grandes (>1000).

b) Méthode des puissances (méthode itérative)

Les méthodes itératives sont généralement plus efficaces et plus efficientes que les méthodes directes lorsque la taille de la matrice est grande. Cependant, le nombre d'itérations nécessaires pour obtenir la convergence souhaitée n'est généralement pas connu à l'avance et les distributions résultantes ont une certaine marge d'erreur qui leur est associée.

Nous abordons ici la méthode des puissances appliquée à la matrice de transitions [Var09]. Cette méthode repose sur un procédé simple qui consiste à partir d'un vecteur de probabilité

$\boldsymbol{\pi}(0)$ qui satisfait la condition $\boldsymbol{\pi}(0)\mathbb{1} = 1$, d'appliquer la relation $\boldsymbol{\pi}(n+1) = \boldsymbol{\pi}(n)\,\mathbf{P}$, $n \geq 0$ jusqu'à ce que la différence entre les vecteurs $\boldsymbol{\pi}(n+1)$ et $\boldsymbol{\pi}(n)$ soit suffisamment petite (selon une norme donnée), voir page 173 dans [Doo53] par exemple.
Cette méthode est décrite comme suit :

MÉTHODE DES PUISSANCES

---

1. Choisir un vecteur initial $\boldsymbol{\pi}(0) \geq 0$, $\boldsymbol{\pi}(0) \neq 0$ et calculer pour $n = 0, 1, \ldots$

$$\boldsymbol{\pi}(n+1) = \boldsymbol{\pi}(n)\,\mathbf{P},$$

2. Jusqu'à ce que $\boldsymbol{\pi}(n+1) - \boldsymbol{\pi}(n)$ soit suffisamment petit.

Le critère d'arrêt est basé sur la différence entre les vecteurs $\boldsymbol{\pi}(n+1)$ et $\boldsymbol{\pi}(n)$. Par exemple, le critère d'arrêt peut être considéré comme :

$$\|\boldsymbol{\pi}(n+1) - \boldsymbol{\pi}(n)\| = \sum_{i=0}^{N} |\boldsymbol{\pi}(n+1)[i] - \boldsymbol{\pi}(n)[i]| \leq \epsilon,$$

où ϵ est un nombre positif très petit. Nous notons que, si $\boldsymbol{\pi}(0)$ est une distribution de probabilité initiale, alors $\boldsymbol{\pi}(n)$ est la distribution de probabilité de la chaîne de Markov de matrice de probabilité de transition $\mathbf{P}$, après n transitions. De plus, si la chaîne est apériodique, cette méthode converge géométriquement.

2.1.2 Chaînes de Markov à temps continu

Une Chaîne de Markov à temps continu (CTMC, en anglais *Continuous Time Markov Chain*) $\{X(t), t \geq 0\}$ est un processus de Markov dont l'espace d'état est fini ou infini dénombrable et dont le temps est continu, $t \in \mathbb{R}^+$. Une CTMC peut être représentée par un graphe orienté. Les sommets de ce graphe représentent les états et les valeurs sur les arcs représentent le taux avec lequel une transition est prise.

Définition 2.11 (Chaîne de Markov à temps continu) *Une chaîne de Markov à temps continu, notée CTMC, est définie par :*

- $\mathcal{S}$ *: un ensemble fini ou infini dénombrable d'états,*
- *Une matrice* $\mathbf{Q} : \mathcal{S} \times \mathcal{S} \mapsto \mathbb{R}$*, le* générateur infinitésimal*, telle que :*

$$\forall\, s \in \mathcal{S},\ \mathbf{Q}[s, s'] = -\sum_{s' \in \mathcal{S}_{s' \neq s}} \mathbf{Q}[s, s']$$

- *Un vecteur* $\boldsymbol{\pi} : \mathcal{S} \mapsto [0, 1]$*, la* distribution initiale *telle que :*

$$\sum_{s \in \mathcal{S}} \boldsymbol{\pi}[s] = 1$$

Soit $X = \{X(t), t \in \mathbb{R}_+\}$ *le processus markovien associé au générateur infinitésimal* $\mathbf{Q}$ *et à la distribution initiale* $\boldsymbol{\pi}$*. La loi de* X *est définie par les équations ci-dessous :*

$$\forall\, s \in \mathcal{S},\ \mathrm{Prob}(X(0) = s) = \boldsymbol{\pi}[s]$$

$\forall\, s,\, s' \in \mathcal{S}\ \forall\, t, \Delta t \in \mathbb{R}_+,$

$$\mathrm{Prob}(X(t+\Delta t) = s' \,|\, X(t) = s) = \begin{cases} \mathbf{Q}[s,\, s'] \times \Delta t + o(\Delta t) & si \quad s \neq s' \\ 1 - \mathbf{Q}[s,\, s] \times \Delta t + o(\Delta t) & si \quad s = s' \end{cases}$$

avec $lim_{\Delta t \to 0} \frac{o(\Delta t)}{\Delta t} = 0$.

Propriété 2.2 *Le générateur infinitésimal* $\mathbf{Q}$ *d'une CMTC possède les propriétés suivantes :*

- *Les éléments non diagonaux sont positifs ou nuls :*

$$\mathbf{Q}[s,\, s'] \geq 0, \quad \forall s' \neq s.$$

- *Les éléments diagonaux sont négatifs :*

$$\mathbf{Q}[s,\, s'] < 0.$$

- *La somme des éléments de chacune des lignes est nulle :*

$$\mathbf{Q}[s,\, s] = -\sum_{s' \neq s} \mathbf{Q}[s,\, s'].$$

L'analyse d'une CTMC à travers une DTMC peut être effectuée par une méthode générique appelée *uniformisation*.

Définition 2.12 *Une CTMC est dite uniformisable si son générateur infinitésimal* $\mathbf{Q}$ *vérifie :*

$$\max_{x \in \mathcal{S}} |\mathbf{Q}[x,\, x]| \leq +\infty.$$

Les CTMCs à espace d'états fini sont uniformisables [Kij97]. Pour une CTMC uniformisable, ses matrices de transition $\mathbf{P}^{(t)}, t \geq 0$ peuvent être obtenues à partir du générateur infinitésimal $\mathbf{Q}$ comme suit ([Kij97], théorème 4.18)

Définition 2.13 *Soit* $\{X(t), t \geq 0\}$ *une CTMC uniformisable de générateur* $\mathbf{Q} = (\mathbf{Q}[x,\, x'])_{x,\, x' \in \mathcal{S}}$.

$$\forall\, q \geq \max_{x \in \mathcal{S}} |\mathbf{Q}[x,\, x']|,\ \mathbf{P}_q = \mathbf{I} + \frac{1}{q}\mathbf{Q}. \tag{2.5}$$

q *est appelé constante d'uniformisation. De plus, il est facile de vérifier que* $\mathbf{P}_q$ *est stochastique.*

Théorème 2.2 *Les deux CTMCs* $\{X(t), t \geq 0\}$ *et* $\{X_q(n), n \geq 0\}$ *ont la même loi.*

Définition 2.14 *Soit* $\{X(t), t \geq 0\}$ *une CTMC uniformisable. Nous notons par* $\{X_q(n), n \geq 0\}$ *la DTMC ayant la matrice de transition* $\mathbf{P}_q$ *(voir équation 2.5) et qui a la même distribution initiale que la CTMC* $\{X(t), t \geq 0\}$*. La chaîne* $\{X_q(n), n \geq 0\}$ *est appelée la DTMC uniformisée correspondante à la CTMC* $\{X(t), t \geq 0\}$.

Théorème 2.3 *Soit* $\{X(t), t \geq 0\}$ *une CTMC uniformisable et* $\{X_q(n), n \geq 0\}$ *sa DTMC associée. Si* $\{X_q(n), n \geq 0\}$ *est ergodique pour un* $q \geq \max_{x \in \mathcal{S}} |\mathbf{Q}[x, x]|$*, alors* $\{X(t), t \geq 0\}$ *est aussi ergodique et les distributions stationnaires de* $\{X(t), t \geq 0\}$ *et* $\{X_q(n), n \geq 0\}$ *sont égales.*

2.2 Comparaison stochastique

La comparaison stochastique permet de comparer des variables aléatoires ou des processus stochastiques. Nous nous intéressons dans cette thèse aux ordres dits intégraux, définis par une famille de fonctions et plus particulièrement à l'ordre stochastique fort $\leq_{st}$. Nous commençons par donner des généralités sur les ordres stochastiques et en particulier l'ordre $\leq_{st}$. Ensuite, nous présentons la comparaison des DTMCs. Nous mettons en évidence la monotonie stochastique [KK77], qui représente une propriété clé pour la comparaison des chaînes de Markov homogènes.

2.2.1 Ordres Stochastiques

Les ordres stochastiques peuvent se définir selon deux approches. La première approche consiste à les définir à partir de familles de fonctions [Sto83], ces ordres sont alors appelés ordres intégraux. Les plus connus dans cette catégorie, sont l'ordre stochastique fort noté "st" pour des fonctions croissantes, et l'ordre "icx" défini par rapport aux fonctions croissantes convexes. La deuxième approche les définit à partir de familles d'ensembles croissants [Sto83, Tal96]. Ces ordres sont appelés ordres ensemblistes, parmi ces ordres on cite : l'ordre "wk", l'ordre "wk*" et l'ordre "st". Cette approche est essentiellement utilisée dans l'étude d'espaces partiellement ordonnés. Nous présenterons dans la suite que l'ordre "st" définit selon la première approche.

2.2.2 Comparaison stochastique de variables aléatoires selon l'ordre $\leq_{st}$

L'ordre stochastique fort (en anglais *Strong Stochastique Order*) noté $\leq_{st}$ est l'ordre le plus connu pour la comparaison de variables aléatoires. Il équivaut à une comparaison de toutes les trajectoires de ces variables aléatoires, l'ordre $\leq_{st}$ est défini ici comme étant un ordre intégral. Nous présentons dans cette section certaines définitions de base et théorèmes nécessaires et utilisés dans notre étude.

La relation d'ordre stochastique fort a été initialement considérée pour la comparaison des fonctions de répartition, comme une généralisation de la comparaison des nombres réels. Ainsi, définir l'ordre $\leq_{st}$ entre deux fonctions de répartition, revient à faire la comparaison par point de ces deux fonctions.

Définition 2.15 *Soient F_X et F_Y deux fonctions de répartition sur $\mathbb{R}$, F_X est plus petite que F_Y au sens de l'ordre stochastique fort (noté $\leq_{st}$), si et seulement si*

$$\forall\, a \in \mathbb{R} \;:\; F_X[a] \geq F_Y[a].$$

Si on considère deux variables aléatoires X et Y, ayant respectivement les fonctions de répartition F_X et F_Y, alors $F_X[a] \geq F_Y[a]$, $\forall\, a \in \mathcal{S}$ est équivalent à dire que $\mathrm{Prob}(X > a) \leq \mathrm{Prob}(Y > a)$, $\forall\, a \in \mathbb{R}$.

En d'autres termes, il est plus probable pour Y de prendre des valeurs plus grandes que pour X. En outre, $X =_{st} Y$ signifie que X et Y ont la même distribution.

Cela signifie que la probabilité d'avoir de plus grandes valeurs est plus petite pour la variable aléatoire X que pour la variable aléatoire Y.

De la même manière, le théorème suivant montre que l'ordre $\leq_{st}$ défini entre deux variables aléatoires, revient à faire la comparaison par point de ses deux variables.

Théorème 2.4 *Soient X et Y deux variables aléatoires à valeurs dans $\mathcal{S}$, et soient F_X et F_Y leurs fonctions de répartition respectives. Les propositions suivantes sont équivalentes.*

- $X \leq_{st} Y$;
- *Il existe un espace de probabilité $(\Omega, \mathcal{A}, \mathcal{P})$ et des variables aléatoires $\hat{X}$ et $\hat{Y}$, définies sur ce même espace et ayant F_X et F_Y comme fonctions de répartition, tel que $\hat{X}[\omega] \leq \hat{Y}[\omega]$ pour tous $\omega \in \Omega$.*

L'ordre stochastique $\leq_{st}$ peut être vu comme un ordre intégral, i1 est défini à partir de fonctions croissantes.

Théorème 2.5 *Soient X et Y deux variables aléatoires à valeurs dans $\mathcal{S}$. On dira que :*

$$X \leq_{st} Y \;\Leftrightarrow\; E\,f(X) \leq E\,f(Y).$$

pour toute fonction f de $\mathcal{S} \to \mathbb{R}$ $\leq$-croissante, telle que les deux espérances existent.

Sur un espace totalement ordonné, [KK77, AA95], définissent l'ordre intégral $\leq_{st}$ à partir de matrices K_{st}, génératrices de cônes. L'ordre $\leq_{st}$ est généré par la matrice K_{st}, dont les colonnes représentent les fonctions en escalier.

$$K_{st} = \begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ 1 & 1 & 0 & \cdots & 0 \\ 1 & 1 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & 1 & \cdots & 1 \end{bmatrix}.$$

Définition 2.16 *Soient X et Y deux variables aléatoires à valeurs dans $\mathcal{S} = \{1, 2, \ldots, n\}$ et soient $\mathbf{p} = [p_1, p_2, \ldots, p_n]$ et $\mathbf{q} = [q_1, q_2, \ldots, q_n]$ leurs distributions de probabilités.*

$$p_i = \mathrm{Prob}(X = i) \;\text{ et }\; q_i = \mathrm{Prob}(Y = i),\;\; \forall\, i \in \mathcal{S}.$$

$$X \leq_{st} Y \Longrightarrow \mathbf{p}\,K_{st} \leq \mathbf{q}\,K_{st} \Longrightarrow \sum_{k=i}^{n} \mathbf{p}[k] \leq \sum_{k=i}^{n} \mathbf{q}[k], \quad 1 \leq i \leq n$$

Nous pouvons également écrire :

$$X \leq_{st} Y \Longrightarrow \mathbf{p}\,K_{st} \leq \mathbf{q}\,K_{st} \Longrightarrow \sum_{k=1}^{i} \mathbf{p}[k] \geq \sum_{k=1}^{i} \mathbf{q}[k], \quad 1 \leq i \leq n.$$

Exemple 2.2 *Soient X et Y deux variables aléatoires définies sur l'espace d'état discret et totalement ordonné $\mathcal{S} = \{1, 2, 3, 4, 5, 6, 7\}$. Tel que $\mathbf{p}_X = [0.1, 0.2, 0.1, 0.2, 0.05, 0.1, 0.25]$ et $\mathbf{p}_Y = [0, 0.25, 0.05, 0.1, 0.15, 0.15, 0.3]$.*

Les distributions $\mathbf{p}_X$ et $\mathbf{p}_Y$ sont st-comparable $\mathbf{p}_X \leq_{st} \mathbf{p}_Y$

Car

$$\begin{cases} 0.25 & \leq & 0.3 \\ 0.25+0.1 & \leq & 0.3+0.15 \\ 0.25+0.1+0.05 & \leq & 0.3+0.15+0.15 \\ 0.25+0.1+0.05+0.2 & \leq & 0.3+0.15+0.15+0.1 \\ 0.25+0.1+0.05+0.2+0.1 & \leq & 0.3+0.15+0.15+0.1+0.05 \\ 0.25+0.1+0.05+0.2+0.1+0.2 & \leq & 0.3+0.15+0.15+0.1+0.05+0.25 \\ 0.25+0.1+0.05+0.2+0.1+0.2+0.1 & \leq & 0.3+0.15+0.15+0.1+0.05+0.25 \end{cases}$$

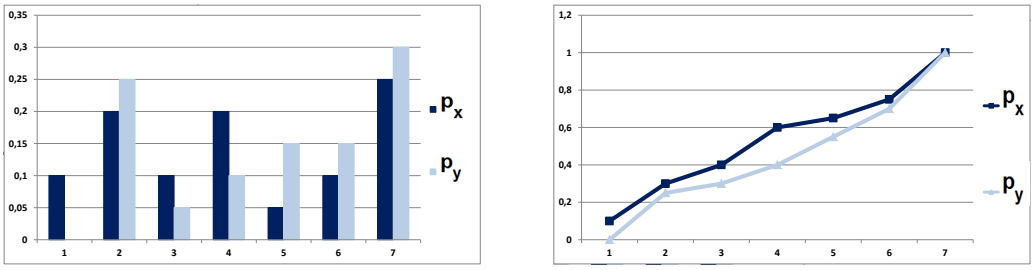


FIGURE 2.2 – $\mathbf{p}_X \leq_{st} \mathbf{p}_Y$: Leurs pmfs (à gauche) ainsi que leurs fonctions de répartition (à droite).

La masse de probabilité de $\mathbf{p}_Y$ est concentrée sur les états plus grands, ainsi, le cumule de la distribution de $\mathbf{p}_Y$ est toujours en dessous de la distribution cumulé de $\mathbf{p}_X$.

L'ordre stochastique $\leq_{st}$ peut se définir aussi comme un ordre ensembliste, à partir des ensembles croissants définis sur l'espace d'état.

2.2.3 Comparaison stochastique de chaînes de Markov

La comparaison des variables aléatoires selon un ordre stochastique est généralisée pour la comparaison des processus stochastiques et particulièrement les Chaînes de Markov. D'une manière générale, la comparaison de chaînes selon l'ordre stochastique $\leq_{st}$ consiste à vérifier la conservation de l'ordre stochastique initial $\leq_{st}$ dans le temps. Nous noterons par $\{X(n),\, n \geq 0\}$ une DTMC où n représente le temps qui prend des valeurs dans $\mathbb{N}$.

Définition 2.17 *Nous dirons qu'une DTMC $\{X(n),\, n \geq 0\}$ est inférieure à une autre DTMC $\{Y(n),\, n \geq 0\}$ au sens de l'ordre stochastique $\leq_{st}$, noté par :*

$$\{X(n),\, n \geq 0\} \leq_{st} \{Y(n),\, n \geq 0\};$$

si et seulement si,

$$X(n) \leq_{st} Y(n),\ \ \forall n \geq 0.$$

Nous allons définir à présent la monotonie stochastique. Pour les chaînes de Markov, la propriété de monotonie stochastique permet d'obtenir des conditions suffisantes très simples pour la comparaison des chaînes de Markov homogènes.

Définition 2.18 *Une DTMC* $\{X_n,\, n \geq 0\}$ *est dite monotone au sens de l'ordre* $\leq_{st}$*, et*

– *Croissante si :*

$$X(n) \leq_{st} X(n+1),\ \forall\, n \geq 0.$$

– *Décroissante si :*

$$X(n+1) \leq_{st} X(n),\ \forall\, n \geq 0.$$

Nous considérons que les chaînes (resp. processus) de Markov sont homogènes, c'est à dire que leurs probabilités de transition (resp. taux de transition) ne dépendent pas du temps, permettant ainsi d'exprimer la monotonie et la comparaison stochastique en termes de matrices de transition dans le cas des DTMCs (resp. des générateurs infinitésimaux dans le cas des CTMCs).

2.2.3.1 Comparaison et monotonie stochastique des DTMCs

Nous allons aborder la comparaison stochastique ainsi que la monotonie des matrices de transition pour les chaînes de Markov à temps discret. Rappelons que la matrice de transition $\mathbf{P} = \{\mathbf{P}[s,\, s']\}_{s,\, s' \in \mathcal{S}}$ d'une DTMC est la matrice dont les éléments $\mathbf{P}[s,\, s']$ correspondent à des probabilités de passage de l'état s vers l'état s'.

La comparaison stochastique des matrices de transition se définit à partir de la conservation de l'ordre stochastique lors de la pré-multiplication de ces matrices par des vecteurs de probabilités identiques [Sto83].

Définition 2.19 (Comparaison stochastique des matrices de transition) *Soient* $\mathbf{P}$ *et* $\mathbf{Q}$ *deux matrices de transition sur* $\mathcal{S}$*. On dira que*

$$\mathbf{P} \leq_{st} \mathbf{Q},$$

si et seulement si, pour tout vecteur de probabilité $\mathbf{v}$ *sur* $\mathcal{S}$*,*

$$\mathbf{v}\,\mathbf{P} \leq_{st} \mathbf{v}\,\mathbf{Q}.$$

Une matrice de transition $\mathbf{P}$ est $\leq_{st}$-monotone si elle préserve la relation d'ordre entre deux vecteurs de probabilités. La monotonie d'une matrice de transition est une condition suffisante et nécessaire pour la monotonie de sa DTMC, cette propriété sera vue prochainement dans la proposition 2.2.

Définition 2.20 *Soit* $\mathbf{P}$ *une matrice de transition sur* $\mathcal{S}$*.* $\mathbf{P}$ *est* $\leq_{st}$*-monotone, si pour tous les vecteurs de probabilités* $\mathbf{u}$ *et* $\mathbf{v}$ *sur* $\mathcal{S}$ *:*

$$\mathbf{u} \leq_{st} \mathbf{v} \Longrightarrow \mathbf{u}\,\mathbf{P} \leq_{st} \mathbf{v}\,\mathbf{P}.$$

Dans la suite, nous allons énoncer les conditions de comparaison et de monotonie des matrices de transition à partir de la comparaison de leurs lignes. Soit $\mathbf{P}$ une matrice de transition. $\mathbf{P}$ peut être vue comme un ensemble de lignes. Chaque ligne est un vecteur de probabilités, représentant les probabilités de transition d'un état x de $\mathcal{S}$ vers les états de $\mathcal{S}$. Elle est notée par :

$$\mathbf{P}[x, \bullet], \ \forall\, x \in \mathcal{S}.$$

La matrice de transition $\mathbf{P}$ peut s'écrire :

$$\mathbf{P} = \{\mathbf{P}[x, \bullet], \ \forall\, x \in \mathcal{S}\}.$$

Nous avons également la proposition suivante :

Proposition 2.1 *Soit $\mathbf{P}$ une matrice de transition définie sur $\mathcal{S} = \{1, 2, \ldots, n\}$. $\mathbf{P}$ est dite $\leq_{st}$-monotone si elle vérifie :*

$$\mathbf{P}[x, \bullet] \leq_{st} \mathbf{P}[x+1, \bullet], \ \forall\, 1 \leq x < n.$$

qui est équivalent à :

$$\forall 1 \leq i \leq j \leq n, \quad \forall k, \quad \sum_{l=k}^{n} \mathbf{P}[i, l] \leq \sum_{l=k}^{n} \mathbf{P}[j, l]$$

La monotonie d'une chaîne de Markov selon l'ordre $\leq_{st}$ se définit comme la croissance stochastique de ces lignes.

Propriété 2.3 *La st-monotonie de matrices positives est assurée lors d'une addition et d'une multiplication par une constante positive.*

Proposition 2.2 (Monotonie d'une DTMC) *Nous dirons qu'une DTMC $\{X(n), n \geq 0\}$ est $\leq_{st}$-monotone si sa matrice de transition $\mathbf{P}$ est $\leq_{st}$-monotone.*

Le théorème suivant montre que la comparabilité et la monotonie des matrices de transitions constituent des conditions suffisantes pour comparer stochastiquement les DTMCs.

Théorème 2.6 (Comparaison) *Soient **P** (resp. **Q**) la matrice de probabilité transitoire associée à la chaîne de Markov homogène $\{X(n), n \geq 0\}$ (resp. $\{Y(n), n \geq 0\}$). La comparaison des chaînes de Markov est établie ($\{X(n), n \geq 0\} \leq_{st} \{Y(n), n \geq 0\}$), si les conditions suivantes sont vérifiées :*

- $X(0) \leq_{st} Y(0)$,
- *au moins une des matrices de transition est monotone, qui est, soit **P** ou **Q** (on dit **P**) est $\leq_{st}$-monotone, si pour tout vecteur de probabilité **u** et **v**,*

$$\mathbf{u} \leq_{st} \mathbf{v} \Longrightarrow \mathbf{u}\boldsymbol{P} \leq_{st} \mathbf{v}\boldsymbol{P}$$

ce qui est équivalent à

$$1 \leq i \leq n-1, \quad \boldsymbol{P}[i, \bullet] \leq_{st} \boldsymbol{P}[i+1, \bullet]$$

- *les matrices de probabilité transitoire sont comparables au sens de l'ordre $\leq_{st}$ si :*

$$\boldsymbol{P} \leq_{st} \boldsymbol{Q} \Leftrightarrow 1 \leq i \leq n,\ \boldsymbol{P}[i,\bullet] \leq_{st} \boldsymbol{Q}[i,\bullet]$$

De plus, si les deux chaînes sont ergodique, alors $\boldsymbol{\pi_P} \leq_{st} \boldsymbol{\pi_Q}$ (où $\boldsymbol{\pi_P}$ et $\boldsymbol{\pi_Q}$ représentent les distributions stationnaires).

Propriété 2.4 *Si $X \leq_{st} Y$, Alors*

i) *$-Y \leq_{st} -X$;*

ii) *$f(X) \leq_{st} f(Y)$, pour toute fonction croissante f ;*

iii) *$\leq_{st}$ est conservé pour l'addition : si $W \leq_{st} Z$, alors $X + W \leq_{st} Y + Z$.*

Dans le cas de chaînes de Markov absorbantes, les temps d'absorption peuvent également être comparés [BBFP06].

Proposition 2.3 (Comparaison des temps d'absorption) *Soit $\{X(n), n \geq 0\}$ (resp. $\{Y(n), n \geq 0\}$) une DTMC avec un seul état absorbant et correspondant à l'état le plus petit, et T_X (resp. T_Y) représente le temps d'absorption.*

Si $\{X(n), n \geq 0\} \leq_{st} \{Y(n), n \geq 0\}$, alors $T_X \leq_{st} T_Y$.

Nous donnons également le lemme suivant nécessaire dans l'une de nos démarches de preuve qu'on retrouvera dans la suite du manuscrit.

Lemme 2.1 *Soient $\mathbf{p}$ et $\mathbf{q}$ deux vecteurs non négatifs de même taille n et de même norme $N1$ ($\|\mathbf{p}\|_1 = \|\mathbf{q}\|_1$). Sans ambiguïté, nous utilisons l'ordre stochastique fort pour comparer les vecteurs $\mathbf{p}$ et $\mathbf{q}$. Soit γ_i une distribution de probabilité discrète pour tout i entre 1 et n. Supposons que :*

1. $\mathbf{p} \leq_{st} \mathbf{q}$,
2. *pour tout* $1 \leq i \leq n-1$, $\gamma_i \leq_{st} \gamma_{i+1}$,

alors il existe une décomposition de $\sum_i \gamma_i(\mathbf{p}[i] - \mathbf{q}[i])$ en $\sum_{j,k} \lambda_{j,k}(\gamma_j - \gamma_k)$ avec $j < k$ et $\lambda_{j,k} > 0$. Et cette décomposition n'est pas unique.

Ces résultats sont valables pour des chaînes de Markov à temps discret. La comparaison et la monotonie stochastique des processus de Markov à temps continu peuvent s'obtenir à partir de la comparaison et la monotonie stochastique des chaînes de Markov grâce à des méthodes d'uniformisation[Kij97]. Les résultats précedents peuvent alors être utilisés dans le cas du temps continu.

2.2.3.2 Construction de bornes optimale $\leq_{st}$-monotone

Abu-Amsha et Vincent ont proposé dans [AAV98] un algorithme simple pour construire une borne supérieure $\leq_{st}$-monotone pour une matrice stochastique arbitraire $\mathbf{P}$ définie sur un espace d'états fini et totalement ordonné $\mathcal{S} = \{1, 2, \ldots, n\}$. Pour cela la matrice borne doit vérifier deux contraintes :

- $\mathbf{P}^+$ est une borne supérieure au sens $\leq_{st}$ de $\mathbf{P}$, $\mathbf{P} \leq_{st} \mathbf{P}^+$ se traduit par :

$$\sum_{k=j}^{n} \mathbf{P}[i,\, k] \leq \sum_{k=j}^{n} \mathbf{P}^+[i,\, k], \quad \forall\, 1 \leq i,\, j \leq n. \tag{2.6}$$

- $\mathbf{P}^+$ est $\leq_{st}$-monotone, ce qui se traduit par :

$$\sum_{k=j}^{n} \mathbf{P}^+[i,\, k] \leq \sum_{k=j}^{n} \mathbf{P}^+[i+1,\, k], \quad \forall\, 1 \leq i \leq n-1,\, \forall\, 1 \leq j \leq n. \tag{2.7}$$

En transformant les inégalités 2.6 et 2.7 en égalités et en les ordonnant selon l'ordre croissant des indices des lignes et selon l'ordre décroissant des indices de colonnes nous obtenons l'algorithme de Vincent.

Algorithme 1 : (Algorithme de Vincent) Construction d'une borne supérieure $\leq_{st}$-monotone $\mathbf{P}^+$ pour une matrice stochastique $\mathbf{P}$ d'ordre n.

ENTRÉES : $\mathbf{P}$.
SORTIES : $\mathbf{P}^+$.

1: $\mathbf{P}^+[1,\, n] = \mathbf{P}[1,\, n]$.
2: **Pour** $i = 2,\, 3, \ldots,\, n$ **faire**
3: $\quad \mathbf{P}^+[i,\, n] = \max(\mathbf{P}^+[i-1,\, n], \mathbf{P}[i,\, n])$.
4: **Fin pour**
5: **Pour** $j = n-1,\, n-2, \ldots,\, 2$ **faire**
6: $\quad \mathbf{P}^+[1,\, j] = \mathbf{P}[1,\, j]$.
7: $\quad$ **Pour** $i = 2,\, 3, \ldots,\, n$ **faire**
8: $\quad\quad \mathbf{P}^+[i,\, j] = \max(\sum_{k=j}^{n} \mathbf{P}^+[i-1,\, k], \sum_{k=j}^{n} \mathbf{P}[i,\, k]) - \sum_{k=j+1}^{n} \mathbf{P}^+[i,\, k]$.
9: $\quad$ **Fin pour**
10: **Fin pour**
11: **Pour** $i = 2,\, 3, \ldots,\, n$ **faire**
12: $\quad \mathbf{P}^+[i,\, 1] = 1 - \sum_{k=2}^{n} \mathbf{P}^+[i,\, k]$.
13: **Fin pour**
14: **Retourner** $\mathbf{P}^+$.

De la même manière, nous pouvons calculer une borne inférieure et $\leq_{st}$-monotone $\mathbf{P}^-$ pour la matrice $\mathbf{P}$. Pour cela la matrice $\mathbf{P}^-$ doit respecter les deux contraintes suivantes :

– $\mathbf{P}^-$ est une borne inférieure au sens $\leq_{st}$ de $\mathbf{P}$, $\mathbf{P}^- \leq_{st} \mathbf{P}$ qui se traduit par :

$$\sum_{k=j}^{n} \mathbf{P}^-[i,\, k] \leq \sum_{k=j}^{n} \mathbf{P}[i,\, k], \quad \forall\, 1 \leq i,\, j \leq n. \tag{2.8}$$

– $\mathbf{P}^-$ est $\leq_{st}$-monotone qui se traduit par :

$$\sum_{k=j}^{n} \mathbf{P}^-[i,\, k] \leq \sum_{k=j}^{n} \mathbf{P}^-[i+1,\, k], \quad \forall\, 1 \leq i \leq n-1,\, \forall\, 1 \leq j \leq n. \tag{2.9}$$

Algorithme 2 : (Algorithme de Vincent) Construction d'une borne inférieure $\leq_{st}$-monotone $\mathbf{P}^-$ pour une matrice stochastique $\mathbf{P}$ d'ordre n.

ENTRÉES : $\mathbf{P}$.
SORTIES : $\mathbf{P}^-$.

1: $\mathbf{P}^-[n, n] = \mathbf{P}[n, n]$.
2: **Pour** $i = n-1, n-2, \ldots, 1$ **faire**
3: $\quad \mathbf{P}^-[i, n] = \min(\mathbf{P}^-[i+1, n], \mathbf{P}[i, n])$.
4: **Fin pour**
5: **Pour** $j = n-1, n-2, \ldots, 2$ **faire**
6: $\quad \mathbf{P}^-[n, j] = \mathbf{P}[n, j]$.
7: $\quad$ **Pour** $i = n-1, n-2, \ldots, 1$ **faire**
8: $\quad\quad \mathbf{P}^-[i, j] = \min(\sum_{k=j}^{n} \mathbf{P}^-[i+1, k], \sum_{k=j}^{n} \mathbf{P}[i, k]) - \sum_{k=j+1}^{n} \mathbf{P}^-[i, k]$.
9: $\quad$ **Fin pour**
10: **Fin pour**
11: **Pour** $i = 2, 3, \ldots, n$ **faire**
12: $\quad \mathbf{P}^-[i, 1] = 1 - \sum_{k=2}^{n} \mathbf{P}^-[i, k]$.
13: **Fin pour**
14: **Retourner** $\mathbf{P}^-$.

L'algorithme de construction d'une borne inférieure $\leq_{st}$-monotone est donné par l'algorithme 2. Il est obtenu en transformant les inégalités 2.8 et 2.2.3.2 en égalités et en les ordonnant selon l'ordre décroissant des indices des lignes et des colonnes.

La complexité des algorithmes 1 et 2 est au pire $O(n^2)$, où n représente la taille de l'espace d'état. De plus, il a été prouvé dans [DFPV06], lemme 3.1 que ces algorithmes construisent une borne optimale. Nous présentons ce résultat dans le théorème 2.7.

Théorème 2.7 ([DFPV06], lemme 3.1) *Soit* $\mathbf{P}$ *une matrice stochastique et* $\mathbf{P}^+$ *la matrice obtenue par l'algorithme 1. La matrice* $\mathbf{P}^+$ *est la plus petite matrice* $\leq_{st}$*-monotone, telle que* $\mathbf{P} \leq_{st} \mathbf{P}^+$. *Formellement, s'il existe une matrice* $\mathbf{R}$ *telle que :*

1. $\mathbf{R}$ *est* $\leq_{st}$*-monotone et*
2. $\mathbf{P} \leq_{st} \mathbf{R}$

Alors, $\mathbf{P}^+ \leq_{st} \mathbf{R}$.

Exemple 2.3 *Nous allons illustrer l'algorithme 1 sur une petite matrice. Nous considérons la matrice stochastique* $\mathbf{P}_{5\times5}$ *:*

$$\boldsymbol{P} = \begin{bmatrix} 0.1 & 0.3 & 0.2 & 0.4 \\ 0.1 & 0.4 & 0.2 & 0.3 \\ 0.2 & 0.1 & 0.5 & 0.2 \\ 0.2 & 0 & 0.4 & 0.4 \end{bmatrix}$$

l'algorithme de Vincent permet de définir la matrice $\leq_{st}$*-monotone borne supérieure suivante :*

$$\boldsymbol{Q} = \begin{bmatrix} 0.1 & 0.3 & 0.2 & 0.4 \\ 0.1 & 0.3 & 0.2 & 0.4 \\ 0.1 & 0.2 & 0.3 & 0.4 \\ 0.1 & 0.1 & 0.4 & 0.4 \end{bmatrix}$$

Leurs distributions stationnaires sont respectivement $\boldsymbol{\pi}_{\mathbf{P}} = [0.1685, 0.1466, 0.3744, 0.3105]$ *et* $\boldsymbol{\pi}_{\mathbf{Q}} = [0.1000, 0.1889, 0.3111, 0.4000]$. *Nous remarquons facilement que* $\boldsymbol{\pi}_{\mathbf{P}} \leq_{st} \boldsymbol{\pi}_{\mathbf{Q}}$.

Première partie

Chaînes de Markov partiellement spécifiées

Chapitre 3

Méthodes de bornes pour les chaînes de Markov partiellement spécifiées

Les chaînes de Markov sont des modèles largement utilisés dans l'étude de performance, de fiabilité ou d'analyse de performabilité des systèmes informatiques et de communication. Offrant diverses méthodes mathématiques puissantes et efficaces, les modèles markoviens sont souvent employés. Cependant, dans de nombreux problèmes d'ingénierie, il est encore difficile voir impossible de définir de manière précise les paramètres qui décrivent le modèle, comme les probabilités de transition. Très souvent, la seule information dont nous disposons est que les probabilités de transition appartiennent à un intervalle. Ce qui est équivalent à dire que le modèle étudié appartient à un ensemble de chaînes de Markov noté $\mathcal{P}_{\mathbf{L},\mathbf{U}}$. L'ensemble $\mathcal{P}_{\mathbf{L},\mathbf{U}}$ est caractérisé par une matrice bornante inférieure par élément notée $\mathbf{L}$, et une matrice bornante supérieure par élément notée $\mathbf{U}$. De plus, pour toute matrice stochastique irréductible et apériodique $\mathbf{P}$ de l'ensemble $\mathcal{P}_{\mathbf{L},\mathbf{U}}$, nous avons $\mathbf{L} \leq \mathbf{P} \leq \mathbf{U}$. Une démarche naturelle revient donc à trouver des bornes supérieures et inférieures de la distribution stationnaire de toutes les matrices de l'ensemble. Donc, nous voulons déterminer deux vecteurs non négatifs $\mathbf{l}$ et $\mathbf{u}$ tels que pour toute matrice $\mathbf{P}$ dans $\mathcal{P}_{\mathbf{L},\mathbf{U}}$, nous avons $\mathbf{l} \leq \boldsymbol{\pi}_{\mathbf{P}} \leq \mathbf{u}$, où $\boldsymbol{\pi}_{\mathbf{P}}$ est la distribution stationnaire de $\mathbf{P}$.

De nombreux résultats ont été proposés pour trouver des bornes sur la distribution stationnaire d'une chaîne de Markov dont les probabilités de transitions sont décrites dans un intervalle. Il existe principalement deux approches. La première a été développée par Courtois & Semal dans [CS84] et consiste à construire une borne supérieure (resp. inférieure) relative à une récompense positive de la distribution stationnaire à travers la théorie polyédrale. Cette approche est communément appelée approche algébrique, puisque le problème est formulé comme un problème d'algèbre linéaire. Cette approche a été étendue par la suite par Buchholz dans [Buc05]. La deuxième approche basée sur la comparaison stochastique a été étudiée par Haddad & Moreaux dans [HM07] pour des matrices sous-stochastiques et facilement généralisées au cas de matrice stochastique par Bušić dans [Bus07]. Nous notons que la comparaison stochastique permet de fournir aussi bien des bornes sur des distributions transitoires et stationnaires que sur des fonctions de récompense croissantes [Sto83].

Dans ce chapitre, nous nous sommes penchés sur la comparaison des méthodes et des

algorithmes de construction de bornes pour des chaînes de Markov définies par intervalles. Nous avons constaté qu'une telle comparaison n'a jamais été étudiée ni publiée, il nous a donc semblé important d'effectuer une telle comparaison. Cependant, nous avons remarqué que ces méthodes ne reposent pas sur le même critère d'optimalité et ne dégagent pas par conséquent les mêmes résultats. En effet, ces algorithmes ne considèrent pas les mêmes hypothèses sur les récompenses : positivité pour l'approche polyédrale, non-décroissance pour les bornes stochastiques. Ainsi, afin de comparer les résultats numériques des différentes méthodes liées aux deux approches, nous avons supposé des récompenses qui satisfont les deux hypothèses, c'est-à-dire, la fonction de récompense est considérée positive et non décroissante. Nous effectuons cette comparaison en tenant compte de la précision et de la complexité des algorithmes.

3.1 Chaînes de Markov imprécises

3.1.1 Notations

Les notations suivantes sont utilisées tout au long de la première partie du manuscrit. Les vecteurs et les matrices sont représentés en caractère gras et en lettres minuscules et majuscules $\mathbf{p}$ et $\mathbf{P}$, respectivement. Les vecteurs sont des vecteurs lignes, sauf indication contraire. Les éléments d'un vecteur (resp. matrice) sont spécifiés par leurs indices entre crochets. Par exemple, $\mathbf{P}[i, j]$ indique l'élément dans la ligne i et la colonne j de la matrice $\mathbf{P}$. De plus, $\mathbf{P}[i, \bullet]$ désigne la ligne i de $\mathbf{P}$ et $\mathbf{P}[\bullet, j]$ décrit la colonne j de la matrice $\mathbf{P}$. La matrice *diag*($\mathbf{p}$) pour un vecteur colonne $\mathbf{p}$ à n-dimensions est une matrice diagonale $n \times n$ avec $\mathbf{p}[i]$ en position $[i,\ i]$. Le vecteur ligne où tous les éléments sont à 1 est noté $\mathbf{e}$, $\mathbf{e}_i$ est un vecteur ligne avec 1 en position i et 0 ailleurs et le vecteur où toutes les entrées sont égales à 0 est noté $\mathbf{0}$. La matrice identité est notée par $\mathbf{Id}$, $\mathbf{A}^t$ et $\mathbf{a}^t$ sont les transposés de la matrice et du vecteur, respectivement. Les ensembles sont représentés par des lettres calligraphiques. Nous notons par "$\leq$" (resp. "$\geq$") la comparaison élément par élément de deux vecteurs ou matrices. Par exemple, pour deux matrices non négatives $\mathbf{A}$ et $\mathbf{B}$, $\mathbf{A} \leq \mathbf{B}$ revient à dire que $\mathbf{A}[i, j] \leq \mathbf{B}[i, j]$ pour toutes les entrées de la matrice. De plus, la matrice $\mathbf{A}$ est dite borne inférieure de $\mathbf{B}$ et $\mathbf{B}$ la borne supérieure de $\mathbf{A}$. La valeur $||\mathbf{x}||$ représente la somme des éléments du vecteur $\mathbf{x}$.

3.1.2 Description du modèle

Soit $\{X(n), n \geq 0\}$ une Chaîne de Markov en Temps Discret (DTMC) mal spécifiée, à partir de laquelle nous voulons évaluer certaines mesures de performance. Une DTMC mal spécifiée est une chaîne dont la matrice de transition n'est pas parfaitement connue. Cette chaîne appartient à un ensemble de matrices stochastiques noté $\mathcal{P}$ respectant toutes les spécifications. L'ensemble $\mathcal{P}$ est décrit par des transitions dont les probabilités sont typiquement données dans un intervalle ou proviennent de l'interaction de divers processus d'arrivée ou de service dont seuls la moyenne et le support sont connus. L'ensemble des chaînes de Markov $\mathcal{P}$ est spécifié par une matrice sous-stochastique notée $\mathbf{L}$ pour les bornes inférieures sur les probabilités de transition et une matrice sur-stochastique, $\mathbf{U}$ pour les bornes supérieures sur les probabilités. Et non pas une matrice de probabilité de transition $\mathbf{P}$. Nous notons que ces matrices ne

représentent pas de bons candidats pour modéliser le comportement d'une chaîne de Markov dont les probabilités de transition sont dans cet intervalle (matrices non stochastiques). Elles sont utilisées dans le but d'obtenir une représentation plus précise de l'ensemble, en définissant ces deux matrices, dans un sens, nous déterminons le "meilleur" et le "pire" cas de l'ensemble.

Définition 3.1 (Matrice sous-stochastique) *Nous appelons matrice sous-stochastique de taille $n \times n$, toute matrice* $\mathbf{L}$ *vérifiant les propriétés suivantes :*

- $\forall\, i \in \{1, \ldots, n\},\ \forall\, j \in \{1, \ldots, n\}\ \ \mathbf{L}[i, j] \geq 0$ *;*
- $\forall\, i \in \{1, \ldots, n\},\ \sum_{j=1}^{n} \mathbf{L}[i, j] \leq 1$ *avec, au moins, une valeur de i où* $\sum_{j=1}^{n} \mathbf{L}[i, j] < 1$.

Définition 3.2 (Matrice sur-stochastique) *Nous appelons matrice sur-stochastique de taille $n \times n$, toute matrice* $\mathbf{U}$ *vérifiant les propriétés suivantes :*

- $\forall\, i \in \{1, \ldots, n\},\ \forall\, j \in \{1, \ldots, n\}\ \ \mathbf{U}[i, j] \geq 0$ *;*
- $\forall\, i \in \{1, \ldots, n\},\ \sum_{j=1}^{n} \mathbf{U}[i, j] \geq 1$ *avec, au moins, une valeur de i où* $\sum_{j=1}^{n} \mathbf{U}[i, j] > 1$.

Nous partons du principe que les spécifications du modèle proviennent d'observations partielles et d'informations incomplètes sur les transitoires (c.-à-d. sur la matrice de transition $\mathbf{P}$). Une description partielle du système nous permet de préciser une borne inférieure sur les probabilités de transition $\mathbf{L} \in \mathbb{R}^{n \times n}$ et une borne supérieure sur les probabilités de transition $\mathbf{U} \in \mathbb{R}^{n \times n}$ tel que $\mathbf{L} \leq \mathbf{P} \leq \mathbf{U}$.

Définition 3.3 *L'ensemble $\mathcal{P}_{\boldsymbol{L},\boldsymbol{U}}$ est défini par le 3-tuple ($\mathcal{F}$, $\mathbf{L}$, $\mathbf{U}$), où $\mathcal{F}$ est l'espace d'état, la matrice* $\mathbf{L}$ *(resp.* $\mathbf{U}$*) est une matrice sous-stochastique (resp. matrice sur-stochastique) vérifiant les conditions suivantes :*
Pour tout $i, j \in \mathcal{F}$, les inégalités suivantes sont satisfaites entre $\boldsymbol{L}$ et $\boldsymbol{U}$:

$$0 \leq \boldsymbol{L}[i, j] \leq \boldsymbol{U}[i, j] \ \ \wedge \sum_{k \in \mathcal{F}} \boldsymbol{U}[i, k] \geq 1 \geq \sum_{k \in \mathcal{F}} \boldsymbol{L}[i, k] \tag{3.1}$$

$$\boldsymbol{U}[i, j] \leq 1 - \sum_{k \in \mathcal{F} \setminus \{j\}} \boldsymbol{L}[i, k] \tag{3.2}$$

$$\boldsymbol{L}[i, j] \geq \max(0, 1 - \sum_{k \in \mathcal{F} \setminus \{j\}} \boldsymbol{U}[i, k]) \tag{3.3}$$

Définition 3.4 *Soit $\{X(n), n \geq 0\}$ une DTMC homogène dans le temps définie par sa matrice de transition* $\mathbf{P}$*. La DTMC $X(n)$ est dite appartenir à l'ensemble $\mathcal{P}_{\boldsymbol{L},\boldsymbol{U}}$ ($\mathbf{P} \in \mathcal{P}_{\boldsymbol{L},\boldsymbol{U}}$), si*

$$\forall i, j \ \ \boldsymbol{L}[i, j] \leq \boldsymbol{P}[i, j] \leq \boldsymbol{U}[i\, j] \tag{3.4}$$

Remarque : Nous distinguons que l'ensemble d'inégalités (3.1) représente une condition nécessaire et suffisante pour que l'ensemble $\mathcal{P}_{\mathbf{L},\mathbf{U}}$ ne soit pas vide. Par ailleurs, une modification peut toujours être apportée aux matrices $\mathbf{L}$ et $\mathbf{U}$ afin de satisfaire les inégalités (3.2) et (3.3) sans affecter l'ensemble $\mathcal{P}_{\mathbf{L},\mathbf{U}}$. Les modifications sont :

$$\mathbf{L}[i, j] = \max(\mathbf{L}[i, j], 1 - \sum_{k \neq j} \mathbf{U}[i, k]) \ \ \text{et}$$

$$\mathbf{U}[i,\, j] = \min(\mathbf{U}[i,\, j],\, 1 - \sum_{k \neq j} \mathbf{L}[i,\, k]).$$

Dans la suite, nous noterons par $X(n)$ (resp. $\mathbf{P}$) la DTMC (resp. la matrice de transition) appartenant à l'ensemble de chaînes de Markov $\mathcal{P}_{\mathbf{L},\mathbf{U}}$.

Connaissant les matrices $\mathbf{L}$ et/ou $\mathbf{U}$, la construction d'une borne inférieure et/ou supérieure de la matrice $\mathbf{P}$ peut être obtenue en utilisant les méthodes développées ci-après. Nous citons respectivement la méthode de Courtois & Semal [CS84, CS86] et de Buchholz [Buc05] pour l'approche polyédrale et la méthode de Haddad & Moreaux [HM07] suivi de la généralisation de Bušić [Bus07] pour l'approche par comparaison stochastique.

3.2 Bornes polyédrales

L'approche polyédrale a été proposée par Courtois & Semal dans [Sem95] et étendue par la suite par Buchholz dans [Buc05]. Ces méthodes sont basées sur l'optimisation de l'espérance d'une récompense positive sur la distribution de probabilité stationnaire. Formellement, l'idée consiste à trouver dans l'ensemble des matrices $\mathcal{P}$ l'élément qui maximise la récompense moyenne. La borne est atteinte par construction et la matrice optimale appartient à l'ensemble $\mathcal{P}$. Nous soulignons que les matrices optimales calculées changent lorsque nous apportons une modification à la fonction de récompense (le vecteur de récompense intervient lors de la procédure d'optimisation). Enfin, nous précisons que les quantités calculées à partir de la matrice optimale (comme le temps de premier passage, la distribution transitoire ou encore les distributions stationnaires) ne constituent pas forcement des bornes, seules celles calculées comme récompense sur la distribution stationnaire le garantissent. De plus, l'optimalité des résultats est seulement assurée sur l'espérance de la récompense considérée.

Nous présentons au préalable, quelques terminologies et résultats liés à la théorie polyédrale.

Théorème 3.1 *Soit* $\mathbf{M}$ *une matrice non négative, alors*

- $\mathbf{M}$ *a une valeur propre* $\beta_{\mathbf{M}}$ *;*
- *Les vecteurs propres associés à la valeur propre* $\beta_{\mathbf{M}}$ *sont non négatifs ;*
- *Soit une matrice* $\mathbf{M}'$ *telle que* $\mathbf{M}' \geq \mathbf{M}$ *alors* $\beta_{\mathbf{M}'} \geq \beta_{\mathbf{M}}$.

Rappelons que dans le cas d'une matrice non négative quelconque, la valeur propre $\beta_{\mathbf{M}}$ peut avoir une multiplicité de vecteurs propres linéairement indépendants. Dans la suite de ce travail, nous appelons vecteur propre (ou vecteur propre à gauche dans les travaux de Courtois & Semal) de $\mathbf{M}$, un vecteur propre associé à $\beta_{\mathbf{M}}$ dont la somme des composantes est égale à 1.

Définition 3.5 *Le vecteur propre* $\mathbf{v}$ *est appelé vecteur propre à gauche d'une matrice non négative* $\mathbf{M}$ *s'il vérifie :*

$$\mathbf{v}\,\mathbf{M} = \beta_{\mathbf{M}}\,\mathbf{v} \;\text{ et }\; \mathbf{v}\,\mathbf{e}^t = 1.$$

Définition 3.6 *Nous appelons polyèdre d'une matrice* $\mathbf{M}$*, l'ensemble* $\mathcal{P}_{\mathbf{M}}$ *des combinaisons convexes des lignes de la matrice* $\mathbf{M}$ *normalisée :*

$$\mathcal{P}_{\mathbf{M}} = \{\mathbf{v} \in \mathbb{R}_n \mid \exists\ \boldsymbol{\rho} \in \mathbb{R}_n^+,\ \boldsymbol{\rho}\mathbf{e}^t = 1,\ \mathbf{v} = \boldsymbol{\rho}\ (diag(\mathbf{M}\,\mathbf{e}^t))^{-1}\,\mathbf{M}\}.$$

Pour chaque polyèdre, il est possible de définir une base. C'est-à-dire, définir un ensemble de vecteurs normalisés linéairement indépendants qui engendrent le polyèdre tout entier. Ces vecteurs sont aussi appelés sommets du polyèdre. Par exemple, si les lignes de $\mathbf{M}$ sont linéairement indépendantes alors elles forment une base de $\mathcal{P}_{\mathbf{M}}$.

3.2.1 Approche de Courtois & Semal

Dans cette section, nous résumons les résultats de Courtois & Semal [CS84, CS86]. Les résultats présentés ici sont propres aux matrices stochastiques et sont donc moins généraux que ceux présentés dans [CS84, CS86].

Soit $\mathbf{P}$ une matrice stochastique et soient $\mathbf{L}$ et $\mathbf{U}$ une borne inférieure et supérieure par élément de la matrice $\mathbf{P}$ ($\mathbf{L} \leq \mathbf{P} \leq \mathbf{U}$). Nous supposons que $\mathbf{P}$, $\mathbf{L}$ et $\mathbf{U}$ sont des matrices irréductibles. Les matrices stochastiques considérées vérifient que la connaissance de la matrice $\mathbf{L}$ impose certaines restrictions sur la taille des éléments de $\mathbf{U}$ et vice versa. Ainsi, nous supposons que les inégalités (3.2) et (3.3) sont toujours satisfaites. Connaissant la matrice $\mathbf{L}$ (resp. $\mathbf{U}$), l'ensemble des matrices stochastiques est défini comme suit :

$$\begin{aligned} \mathcal{P}_{\mathbf{L}} &= \{\mathbf{M} \mid \mathbf{M} \geq \mathbf{L},\ \mathbf{M}\mathbf{e}^t = \mathbf{e}^t \text{ et } \mathbf{M} \text{ irréductible}\} \quad \text{et} \\ \mathcal{P}_{\mathbf{U}} &= \{\mathbf{M} \mid \mathbf{M} \leq \mathbf{U},\ \mathbf{M}\mathbf{e}^t = \mathbf{e}^t \text{ et } \mathbf{M} \text{ irréductible}\}. \end{aligned}$$

En outre, la méthode montre que l'ensemble des vecteurs propres constitue un polyèdre convexe dont les sommets peuvent être calculés comme suit :

Lemme 3.1 *[CS84] Soit* $\mathbf{L}_{n\times n}$ *une matrice de* $\mathbb{R}^+$*, alors nous avons :*

- $(\mathbf{Id} - \mathbf{L})$ *est non singulière (inversible) ;*
- $\mathbf{N} = (\mathbf{Id} - \mathbf{L})^{-1} \geq 0 \ \text{ et } \ (\mathbf{Id} - \mathbf{L})^{-1}\,\mathbf{e}^t \geq 0.$

Théorème 3.2 *[CS84] Soient* $\boldsymbol{P}$ *une matrice stochastique et* $\boldsymbol{L}$ *une borne inférieure par élément de* $\boldsymbol{P}$*. Nous supposons que les deux matrices* $\boldsymbol{P}$ *et* $\boldsymbol{L}$ *sont irréductibles. Le vecteur de probabilité stationnaire* $\boldsymbol{\pi}$ *de* $\boldsymbol{P}$ *appartient au polyèdre dont les sommets sont les lignes de la matrice* $\mathbf{Z}$ *tel que les indices appartiennent à l'ensemble* $J = \{j \in 1, \ldots, n,\ t.q.\ \exists\, i\ :\ \mathbf{P}[i,j] > \mathbf{L}[i,j]\}$*, où*

$$\boldsymbol{Z} = (\mathrm{diag}(\boldsymbol{N}\boldsymbol{e}^t))^{-1}\boldsymbol{N}.$$

i) *Alors, il existe un vecteur propre* $\boldsymbol{\rho}$ *tel que :*

- $\boldsymbol{\rho}\,\mathbf{e}^t = 1$ *;*
- $\boldsymbol{\rho}[j] = 0,\ j \notin J$ *;*
- $\boldsymbol{\pi} = \boldsymbol{\rho}\,\mathbf{Z}$*.*

ii) *et*

$$\boldsymbol{\pi}_i^- \leq \boldsymbol{\pi}_i \leq \boldsymbol{\pi}_i^+ \quad \forall\, i = 1, \ldots, n.$$

où

$$\boldsymbol{\pi}^- = \max\{\min_{k\in J}\mathbf{Z}[k,\, i];\, 1 - \sum_{j\neq i} \max_{k\in J} \mathbf{Z}[k,\, j]\},$$

$$\boldsymbol{\pi}^+ = \min\{\max_{k\in J}\mathbf{Z}[k,\, i];\, 1 - \sum_{j\neq i} \min_{k\in J} \mathbf{Z}[k,\, j]\}.$$

Le théorème suivant donne une façon alternative d'obtenir des bornes sans calculer l'inverse des matrices. Ce théorème est notamment utilisé par Muntz [MDG89] et référencé comme tel.

Théorème 3.3 *[CS84, MDG89] Étant donné une matrice borne inférieure par élément* $\mathbf{L}$ *de* $\mathbf{P}$*, la* i*-ème ligne de la matrice* $\mathbf{Z}$ *correspond au vecteur de probabilité stationnaire de la matrice obtenue en incrémentant la* i*-ème colonne de* $\mathbf{P}$ *jusqu'à la rendre stochastique.*

Ainsi, si nous n'avons que la borne inférieure $\mathbf{L}_{n\times n}$ de la matrice de probabilité de transition d'une DTMC, nous pouvons encore calculer des bornes pour son vecteur de probabilité stationnaire $\boldsymbol{\pi}$. Cela se fait en calculant la solution de n DTMC, la matrice de probabilité de transition $\mathbf{L}^i$ associée à la i-ème DTMC est obtenue à partir de la matrice sous-stochastique $\mathbf{L}$ en augmentant les éléments de la colonne i pour rendre la matrice $\mathbf{L}^i$ stochastique. Soit $\boldsymbol{\pi}^i$ le vecteur de probabilité stationnaire obtenu en résolvant la i-ème DTMC. La borne inférieure (resp. supérieure) sur le vecteur de distribution stationnaire de l'état k est calculée comme étant $\min_i \boldsymbol{\pi}^i[k]$ (resp. $\max_i \boldsymbol{\pi}^i[k]$) : $\min_i \boldsymbol{\pi}^i[k] \leq \boldsymbol{\pi}[k] \leq \max_i \boldsymbol{\pi}^i[k]$.

Le théorème qui suit est analogue au théorème 3.2 quand la borne supérieure par élément $\mathbf{U}$ de $\mathbf{P}$ est connue (au lieu de la borne inférieure). Cependant, ce résultat est légèrement plus faible parce que l'existence de la matrice inverse de $\mathbf{U}$ n'est pas toujours garantie. De plus, même si la matrice inverse existe, certaines contraintes supplémentaires doivent être testées afin d'en déduire des bornes. Dans ce cas, il n'y a pas de résultat semblable au lemme 3.1.

Théorème 3.4 *[CS84] Soient* $\boldsymbol{P}$ *une matrice stochastique et* $\boldsymbol{U}$ *une borne supérieure de* $\boldsymbol{P}$*. Nous supposons que les deux matrices* $\boldsymbol{P}$ *et* $\boldsymbol{U}$ *sont irréductibles. Si la matrice inverse suivante*

$$\overline{\mathbf{N}} = (\mathbf{Id} - \mathbf{U})^{-1}$$

existe et satisfait

$$(\mathbf{Id} - \mathbf{U})^{-1}\,\mathbf{e}^t < \mathbf{0}^t, \quad \mathbf{e}(\mathbf{Id} - \mathbf{U})^{-1} < 0$$

alors, il existe donc un vecteur propre $\boldsymbol{\rho}$ *tel que :*

- $\boldsymbol{\rho}\,\mathbf{e}^t = 1$ *;*
- $\boldsymbol{\pi} = \boldsymbol{\rho}\left(\mathrm{diag}(\overline{\boldsymbol{N}}\boldsymbol{e}^t)\right)^{-1}\overline{\boldsymbol{N}}$.

L'ensemble des vecteurs stationnaires appelés aussi vecteurs propres à gauche correspondant aux matrices des ensembles $\mathcal{P}_\mathbf{L}$ et $\mathcal{P}_\mathbf{U}$ sont regroupés dans les ensembles $\mathcal{V}_\mathbf{L}$ et $\mathcal{V}_\mathbf{U}$:

$$\mathcal{V}_\mathbf{L} = \{\mathbf{v} \geq 0 \mid \exists\, \mathbf{P} \in \mathcal{P}_\mathbf{L}, \mathbf{vP} = \mathbf{v},\ \mathbf{ve}^t = 1\}\ \ et$$

$$\mathcal{V}_\mathbf{U} = \{\mathbf{v} \geq 0 \mid \exists\, \mathbf{P} \in \mathcal{P}_\mathbf{U}, \mathbf{vP} = \mathbf{v},\ \mathbf{ve}^t = 1\}.$$

Ces techniques peuvent être utilisées pour calculer des bornes non seulement sur le vecteur de probabilité stationnaire, mais aussi sur tout type de mesure de performance définie en terme d'une fonction de récompense $\mathbf{r}$. Soient $\mathbf{r}[j]$ la récompense associée à l'état j et $\boldsymbol{\pi}^i$ le vecteur de probabilité stationnaire obtenu en résolvant la i-ème DTMC (avec matrice de transition $\mathbf{L}^i$ obtenue à partir de la matrice $\mathbf{L}$ en augmentant sa i-ème colonne de sorte à la rendre stochastique). La récompense moyenne notée R est définie comme

$$R = \boldsymbol{\pi}\,\mathbf{r}^t = \sum_j \boldsymbol{\pi}[j]\,\mathbf{r}[j]. \tag{3.5}$$

Le terme $\sum_j \boldsymbol{\pi}^i[j]\,\mathbf{r}[j]$ est égale à la récompense moyenne $R^i = \boldsymbol{\pi}^i\,\mathbf{r}^t$ calculée à partir de la matrice $\mathbf{L}^i$. Alors, le $\min_i R^i$ et $\max_i R^i$ correspondent respectivement à la borne inférieure et supérieure de R : $\min_i R^i \leq R \leq \max_i R^i$.
Connaissant la matrice $\mathbf{L}$, nous proposons d'illustrer à travers l'exemple suivant le calcul des bornes inférieure et supérieure de la méthode de Courtois & Semal.

Exemple 3.1 *Soient la matrice sous-stochastique* $\boldsymbol{L}$ *donnée par :*

$$\boldsymbol{L} = \begin{bmatrix} 0.2744 & 0.2765 & 0.3779 & 0 \\ 0.1024 & 0 & 0.2651 & 0.1430 \\ 0.3844 & 0.3912 & 0 & 0.0488 \\ 0.0248 & 0.4571 & 0.2229 & 0.0001 \end{bmatrix}$$

et $\boldsymbol{r} = [1, 2, 3, 4]^t$ *le vecteur de récompense décrit sur les états du système. La méthode de Courtois & Semal permet d'obtenir les résultats suivants :*

$$(\mathbf{Id} - \boldsymbol{L})^{-1} = \begin{bmatrix} 2.1767 & 1.1669 & 1.1820 & 0.2245 \\ 0.5955 & 1.5531 & 0.6938 & 0.2559 \\ 1.0975 & 1.1042 & 1.7618 & 0.2439 \\ 0.5708 & 0.9850 & 0.7392 & 1.1769 \end{bmatrix}$$

et

$$\boldsymbol{Z} = \begin{bmatrix} 0.4582 & 0.2457 & 0.2488 & 0.0473 \\ 0.1922 & 0.5013 & 0.2239 & 0.0826 \\ 0.2609 & 0.2624 & 0.4187 & 0.0580 \\ 0.1644 & 0.2837 & 0.2129 & 0.3390 \end{bmatrix}$$

Nous spécifions que les vecteurs de l'ensemble $\mathcal{V}_{\boldsymbol{L}}$ *correspondent aux lignes de la matrice* $\boldsymbol{Z}$*. Le calcul de la récompense moyenne* R^i *associé à chaque vecteur stationnaire* $\boldsymbol{\pi}^i$ *est regroupé dans le vecteur* $R = [1.8851, 2.1970, 2.2738, 2.7265]$ *(équation 3.5). Nous pouvons voir aisément que la récompense minimum est atteinte pour* i *égal à* 1*, ceci permet de dire que la distribution de probabilité stationnaire associée à la borne inférieure correspond à la première ligne de la matrice* $\boldsymbol{Z}$*. La matrice de transition de la borne inférieure (notée* $\underline{\boldsymbol{P}}_{C\&S}$*) est la matrice obtenue par construction sur* $\boldsymbol{L}$ *en augmentant la colonne* 1 *jusqu'à ce que la matrice devienne stochastique.*

$$\underline{\boldsymbol{P}}_{C\&S} = \begin{bmatrix} 0.3456 & 0.2765 & 0.3779 & 0 \\ 0.5919 & 0 & 0.2651 & 0.1430 \\ 0.5600 & 0.3912 & 0 & 0.0488 \\ 0.3199 & 0.4571 & 0.2229 & 0.0001 \end{bmatrix}$$

En suivant le même raisonnement que précédemment, la détermination de la matrice borne supérieure peut aussi être définie. La récompense maximale est atteinte pour i égal à 4, ceci revient à dire que la distribution de probabilité stationnaire associée à la borne supérieure correspond à la quatrième ligne de la matrice $\mathbf{Z}$*. La matrice de transition borne supérieure (notée* $\overline{\boldsymbol{P}}_{C\&S}$*) est la suivante :*

$$\overline{\boldsymbol{P}}_{C\&S} = \begin{bmatrix} 0.2744 & 0.2765 & 0.3779 & 0.0712 \\ 0.1024 & 0 & 0.2651 & 0.6325 \\ 0.3844 & 0.3912 & 0 & 0.2244 \\ 0.0248 & 0.4571 & 0.2229 & 0.2952 \end{bmatrix}$$

Si nous nous retrouvons dans le cas où seul la matrice $\mathbf{L}$ est connue, alors pour une fonction de récompense donnée, le théorème 3.2 fournit les meilleures bornes inférieures et supérieures possibles. De plus, la borne est atteinte par un des vecteurs de l'ensemble $\mathcal{V}_{\mathbf{L}}$ dont la matrice de transition appartient à $\mathcal{P}_{\mathbf{L}}$.

3.2.1.1 Complexité

Le calcul de la matrice inverse $(\mathbf{Id} - \mathbf{L})^{-1}$ nécessite $O(n^3)$ opérations, où n est la taille de la matrice $\mathbf{L}$. Ce qui signifie que le temps d'exécution global est souvent en $O(n^3)$.

Toutefois, si les deux matrices $\mathbf{L}$ et $\mathbf{U}$ sont connues, Buchholz [Buc05] a introduit une nouvelle méthode pour borner les mesures stationnaires d'une DTMC finie. En effet, Buchholz a proposé une version améliorée de la méthode de borne en utilisant le polyèdre des vecteurs propres de Courtois & Semal [CS84, CS85].

3.2.2 Méthode de Buchholz

Fondée sur la théorie polyédrale de Courtois & Semal, Buchholz a étudié dans [Buc05] une famille de matrices donnée par des intervalles de probabilités de transition. En effet, l'approche considère un ensemble de matrices stochastiques et irréductibles, noté $\mathcal{P}_{\mathbf{L},\mathbf{U}}$ bornées inférieurement et supérieurement par élément par les matrices $\mathbf{L}$ et $\mathbf{U}$ respectivement, tel que $\mathcal{P}_{\mathbf{L},\mathbf{U}} = \mathcal{P}_{\mathbf{L}} \cap \mathcal{P}_{\mathbf{U}} = \{\mathbf{P} \text{ stochastique} \mid \mathbf{0} \leq \mathbf{L} \leq \mathbf{P} \leq \mathbf{U}\}$. Rappelons que la théorie polyédrale de Courtois & Semal permet de monter que, sous certaines conditions sur la matrice $\mathbf{L}$ (resp. $\mathbf{U}$), la distribution stationnaire de la matrice $\mathbf{P}$ appartient au polyèdre des vecteurs propres définis par les lignes de la matrice $(\mathbf{Id} - \mathbf{L})^{-1}$ (resp. $(\mathbf{Id} - \mathbf{U})^{-1}$). Buchholz quant à lui, montre que les bornes obtenues par son algorithme sont atteignables par des matrices de la famille $\mathcal{P}_{\mathbf{L},\mathbf{U}}$. Nous allons présenter dans ce qui suit l'essentiel de la méthode.

L'algorithme de Buchholz a pour but de borner des récompenses moyennes à l'état stationnaire et repose sur des modifications répétées d'une matrice stochastique de l'ensemble $\mathcal{P}_{\mathbf{L},\mathbf{U}}$. L'algorithme est basé sur les résultats du théorème (voir [Buc05], Théorème 3.6) qui

montre qu'une récompense optimale est atteinte si la récompense ne peut pas être améliorée en échangeant les probabilités de transition entre deux colonnes de la matrice.
Pour une description algorithmique de l'approche, une matrice $\mathbf{P} \in \mathcal{P}_{\mathbf{L},\mathbf{U}}$ et un indice colonne $i \in \{1,\ldots,n\}$ avec $\mathbf{L}[\bullet, i] < \mathbf{P}[\bullet, i]$ a été défini.
Avec $\mathbf{P} = \mathbf{D} + \mathbf{d}\,\mathbf{e}_i$ et $\mathbf{d} = \mathbf{P}[\bullet, i] - \mathbf{L}[\bullet, i]$. Soient $\mathbf{N} = (\mathbf{I} - \mathbf{D})^{-1}$ et $\mathbf{Z} = (diag(\mathbf{N}\mathbf{e}^t))^{-1}\mathbf{N}$. Par hypothèse, la matrice inverse existe. Le vecteur stationnaire et la récompense moyenne de la DTMC décrite par la matrice $\mathbf{P}$ peuvent être exprimés respectivement comme $\mathbf{e}_i\ \mathbf{Z}$ et $\mathbf{e}_i\ \mathbf{Z}\ \mathbf{r}$. Nous définissons deux ensembles :

$$\begin{aligned} \mathcal{C}^{+}_{\mathbf{D},i} &= \{j | j \in \{1,\ldots,n\} \wedge j \neq i \wedge (\mathbf{U}[\bullet, j] - \mathbf{D}[\bullet, j])^T\mathbf{d} > 0\}\ \ et \\ \mathcal{C}^{-}_{\mathbf{D},i} &= \{j | j \in \{1,\ldots,n\} \wedge j \neq i \wedge \mathbf{P}[\bullet, j] > \mathbf{L}[\bullet, j]\} \end{aligned} \tag{3.6}$$

L'ensemble $\mathcal{C}^{+}_{\mathbf{D},i}$ contient toutes les colonnes j dans lesquelles la masse de probabilités peut être déplacée en provenance de la colonne i. S'il existe $j \in \mathcal{C}^{+}_{\mathbf{D},i}$, avec une récompense améliorée ($\mathbf{e}_j\mathbf{Z}\mathbf{r} > \mathbf{e}_i\mathbf{Z}\mathbf{r}$), alors la nouvelle matrice $\mathbf{P}'$ existe. Soient

$$\mathbf{D}' = \mathbf{D} + (\mathbf{d} - \mathbf{c})\mathbf{e}_i - (\mathbf{P}[\bullet, j] - \mathbf{L}[\bullet, j])\mathbf{e}_j \quad \text{et} \quad \mathbf{d}' = \mathbf{c} + \mathbf{P}[\bullet, j] - \mathbf{L}[\bullet, j] \tag{3.7}$$

avec $\mathbf{c}[k] = \min(\mathbf{d}[k], \mathbf{U}[k, j] - \mathbf{P}[k, j])$, et substituer $\mathbf{e}_i$ par $\mathbf{e}_j$ tel que $\mathbf{D}'$, $\mathbf{d}'$ et $\mathbf{e}_j$ représente la nouvelle DTMC. Pour poursuivre avec la matrice $\mathbf{P}'$, la matrice inverse $\mathbf{N}' = (\mathbf{I} - \mathbf{D}')^{-1}$ doit être connue. Au lieu de calculer directement l'inverse des matrices, nous pouvons calculer $\mathbf{N}'$ plus efficacement à partir de $\mathbf{N}$ en utilisant la formule de Sherman-Morrison-Woodbury [Hag89]. Appliqué à notre cas particulier, le calcul se fait en deux étapes :

$$\mathbf{N}^* = \frac{\mathbf{N} + \mathbf{N}(\mathbf{P}[\bullet, j] - \mathbf{L}[\bullet, j])\mathbf{e}_j\mathbf{N}}{1 - \mathbf{e}_j\mathbf{N}(\mathbf{P}[\bullet, j] - \mathbf{L}[\bullet, j])} \tag{3.8}$$

et une seconde étape qui donne

$$\mathbf{N}' = \frac{\mathbf{N}^* + \mathbf{N}^*(\mathbf{d} - \mathbf{c})\mathbf{e}_i\mathbf{N}^*}{1 - \mathbf{e}_i\mathbf{N}^*(\mathbf{d} - \mathbf{c})} \tag{3.9}$$

Nous observons que les égalités (3.8) et (3.9) ont une complexité de $O(n^2)$ au lieu de $O(n^3)$ qui serait nécessaire pour calculer l'inverse de la nouvelle matrice.

L'algorithme 3 converge en un nombre fini d'étapes. À chaque itération, la récompense est améliorée en déplaçant autant que possible la masse de probabilités entre deux états où un état est marqué comme *checked*. Finalement, ces étapes se terminent par une matrice qui donne une récompense maximale. Nous précisons que l'algorithme suivant calcule la borne supérieure pour la récompense ainsi que la matrice borne supérieure associée, notée $\overline{\mathbf{P}}_{\text{Buch}}$ ($\overline{\mathbf{P}}_{\text{Buch}} = \mathbf{D} + \mathbf{d}\,\mathbf{e}_i$). Pour le calcul de la borne inférieure, le raisonnement inverse sera alors adopté c'est-à-dire, au lieu de chercher à maximiser la récompense nous minimiserons plutôt la récompense moyenne. Donc, seuls les étapes (**1**) et (**5**) de l'algorithme vont être modifiés.

Remarque 1 *Cet algorithme laisse libre le choix de la matrice initiale ainsi que le choix des indices dans le cas où les ensembles $\mathcal{C}^{-}_{\mathbf{D},i}$ et $\mathcal{C}^{+}_{\mathbf{D},i}$ ne se limitent pas à un seul indice (libre choix de la stratégie à aborder). Lors de notre implémentation, nous considérons toujours l'indice le plus petit possible et la matrice initiale est générée suivant l'algorithme de Courtois & Semal.*

Algorithme 3 : Algorithme de Buchholz - Borne supérieure sur les mesures stationnaires d'un processus de Markov fini

1: Initialisation de $\mathbf{P} \in \mathcal{P}_{\mathbf{L},\mathbf{U}}$ et $i \in \{1, \ldots, n\}$ qui définissent $\mathbf{D}$ et $\mathbf{d}$;
2: Calculer $\mathbf{N} = (\mathbf{I} - \mathbf{D})^{-1}$, $\mathbf{Z} = (diag(\mathbf{N}\,\mathbf{e}^t))^{-1}\mathbf{N}$, $\mathcal{C}^+_{\mathbf{D},i}$ et $\mathcal{C}^-_{\mathbf{D},i}$ (via (3.6)) ;
3: Initialisation de $checked = \varnothing$;
4: **Tant que** $((\mathcal{S} \backslash checked) \cap (\mathcal{C}^-_{\mathbf{D},i} \cup \{i\}) \,! = \varnothing)$ **faire**
5: **Si** $(\exists j \in \mathcal{C}^+_{\mathbf{D},i}\ with\ \mathbf{e}_j\mathbf{Zr} > \mathbf{e}_i\mathbf{Zr})$ **alors**
6: calcul de $\mathbf{D}'$, $\mathbf{d}'$ et j en fonction de (3.7) ;
7: $checked = \varnothing$;
8: **sinon**
9: $checked = checked \cup \{i\}$;
10: **Si** $(\exists j \in \mathcal{C}^-_{\mathbf{D},i} \backslash checked)$ **alors**
11: Calcul de $\mathbf{D}'$, $\mathbf{d}'$ et j en fonction de (3.7) en utilisant **c= 0** ;
12: **Fin si**
13: **Fin si**
14: **Si** (la nouvelle matrice $\mathbf{D}'$ est générée) **alors**
15: calculer $\mathbf{N}'$ selon (3.8) et (3.9) et $\mathbf{Z}'=(diag(N'e^t))^{-1}N'$;
16: Poser **N=N′**, **D= D′**, **Z=Z′**, **d=d′** et i=j ;
17: recalculer $\mathcal{C}^+_{D,i}$ et $\mathcal{C}^-_{D,i}$ suivant (3.6);
18: **Fin si**
19: **Fin Tantque**

3.2.2.1 Complexité

Les observations indiquent que le nombre d'itérations dans la boucle while est souvent égale à n ce qui signifie que le temps d'exécution global de l'algorithme est souvent en $O(n^3)$.

Nous allons illustrer la méthode à travers l'exemple suivant.

Exemple 3.2 *Nous considérons la même matrice* **L** *définie dans l'exemple 3.1 et la matrice* **U** *donnée par :*

$$\boldsymbol{U} = \begin{bmatrix} 0.3456 & 0.3477 & 0.4490 & 0 \\ 0.3067 & 0 & 0.7282 & 0.4282 \\ 0.5085 & 0.5152 & 0 & 0.1004 \\ 0.0544 & 0.7522 & 0.4885 & 0.0003 \end{bmatrix}$$

Pour une récompense $\mathbf{r} = [1, 2, 3, 4]^t$*, l'application de l'algorithme de Buchholz pour la détermination de la matrice borne supérieure (resp. matrice borne inférieure) maximisant (resp. minimisant) la récompense moyenne* (R) *permet d'obtenir les résultats suivants.*
Nous détaillons ici les calculs liés à la détermination de la borne inférieure (le même raisonnement peut être effectué pour le calcul de la borne supérieure). Nous commençons par définir la matrice initiale ainsi que l'indice i *nécessaire à l'initialisation de l'algorithme. Cette étape est réalisée en se basant sur la méthode de Courtois et Semal. En calculant la matrice* $\boldsymbol{Z} = (diag(\boldsymbol{N}\,\boldsymbol{e}^t))^{-1}\boldsymbol{N}$*, la valeur de l'indice* i *est fixée à* 1 *puisque* $\mathbf{Z}[1, \bullet]\,\mathbf{r}$ *est minimal. Le*

vecteur $\mathbf{d}$ *est donné par* $\mathbf{d} = \mathbf{U}[\bullet, 1] - \mathbf{L}[\bullet, 1] = [7.1200\,e\text{-}2,\ 2.0430\,e\text{-}1,\ 1.2410\,e\text{-}1, 2.9600\,e\text{-}2]$ *et la matrice* $\mathbf{D}$ *est* $\mathbf{D} = \mathbf{L} + \mathbf{d}\,\mathbf{e}_1$. *La matrice* $\mathbf{D}$ *calculée est sous-stochastique. Les probabilités des lignes de* $\mathbf{L}$ *sont augmentées pour générer la nouvelle matrice stochastique* $\mathbf{D}$. *L'incrémentation des lignes de* $\mathbf{L}$ *doit respecter les contraintes posées par la matrice* $\mathbf{U}$ *(ne pas dépasser les probabilités de transition de* $\mathbf{U}$*). Après avoir déterminé les quantités nécessaires à l'étape d'initialisation, le déroulement de l'algorithme permet d'obtenir les résultats suivants :*

$$\overline{\mathbf{P}}_{\text{Buch}} = \begin{bmatrix} 0.2745 & 0.2765 & 0.4490 & 0 \\ 0.1288 & 0 & 0.4430 & 0.4282 \\ 0.5085 & 0.4427 & 0 & 0.0488 \\ 0.0544 & 0.4571 & 0.4885 & 0 \end{bmatrix}$$

avec une récompense moyenne de $\overline{R}_{\text{Buch}} = 2.3679$. *Le vecteur de distribution stationnaire associé est donné par* $\boldsymbol{\pi}_{\text{Buch}} = [0.2780,\ 0.2762,\ 0.3124,\ 0.1335]$.

$$\underline{\mathbf{P}}_{\text{Buch}} = \begin{bmatrix} 0.3456 & 0.2765 & 0.3779 & 0 \\ 0.3067 & 0 & 0.5503 & 0.1430 \\ 0.5085 & 0.4427 & 0 & 0.0488 \\ 0.0544 & 0.4571 & 0.4885 & 0 \end{bmatrix}$$

et une récompense moyenne de $\underline{R}_{\text{Buch}} = 1.9955$. *Le vecteur de distribution stationnaire associé est donné par* $\boldsymbol{\pi}_{\text{Buch}} = [0.3706,\ 0.2647,\ 0.3116,\ 0.0531]$.

3.3 Bornes stochastiques

La comparaison stochastique de chaînes de Markov nous permet d'établir des bornes aussi bien sur les distributions transitoires que les distributions stationnaires. Pour notre étude, nous nous sommes intéressés à la comparaison au sens de l'ordre stochastique fort, noté aussi ordre "st" pour construire des bornes sur les chaînes de Markov mal spécifiées. Cette méthode nous offre la possibilité de dériver des bornes sur des fonctions de récompenses croissantes qui peuvent représenter soient des taux de perte, des délais d'attente ou encore les durées entre deux événements "remarquables" (pannes, destructions). De plus, contrairement au méthodes de bornes utilisées habituellement, les méthodes de comparaison stochastique permettent de donner des indications plus fines, car elles définissent des bornes sur les distributions et non pas le pire cas déterministe comme le font certaines méthodes. Nous citons à titre d'exemple les approches de Muntz [LM94], Buchholz [Buc02] ou encore celle de Courtois & Semal [CS86] qui considèrent l'espérance d'une récompense donnée.

Dans cette section, nous introduisons respectivement les algorithmes de construction de bornes pour des matrices sous-stochastiques, présentés par Haddad & Moreaux dans [HM07] suivi par leur généralisation au cas stochastique développés par Bušić dans [Bus07]. Les deux méthodes reposent sur la détermination d' un algorithme permettant de calculer une borne supérieure (resp. borne inférieure) optimale au sens "st" pour toute matrice appartenant à l'ensemble des chaînes de Markov $\mathcal{P}_{\mathbf{L},\mathbf{U}}$. De plus, en partant d'une des bornes stochastiques, la théorie générale sur la comparaison stochastique d'une DTMC permet de tirer de nombreuses comparaisons par exemple : comparaison des distributions transitoires, des distributions

stationnaires, l'espérance de récompenses non décroissantes calculées sur ces distributions ainsi que les temps de premier passage.

3.3.1 Méthode de Haddad & Moreaux

Dans [HM07], Haddad et Moreaux ont proposé un algorithme pour calculer une borne supérieure au sens de l'ordre "st" pour l'ensemble de DTMC défini par $\mathcal{P}_{\mathbf{L},\mathbf{U}}$. La configuration étudiée ne correspond pas initialement à une chaîne mal spécifiée, mais à une hypothèse sur la difficulté d'obtenir les probabilités exactes de transition dans une approche numérique généralisant l'algorithme de Muntz. Plutôt que de calculer exactement les probabilités (les formules sont connues), Haddad & Moreaux ont préféré les encadrer par des calculs plus rapides. Leur algorithme fournit cependant la réponse à notre problème si nous supposons que le modèle imprécis est spécifié par des encadrements de probabilités de transition (à savoir, les deux matrices positives $\mathbf{L}$ et $\mathbf{U}$ avec $\mathbf{L} \leq \mathbf{P} \leq \mathbf{U}$). Le point fort de cet algorithme est de calculer avec la même complexité que l'algorithme de Vincent une borne pour un ensemble infini de matrices. Cette approche fournit un algorithme qui permet de calculer la meilleure borne inférieure $\leq_{st}$-monotone pour la famille $\mathcal{P}_{\mathbf{L},\mathbf{U}}$ dans le cas particulier des chaînes de Markov avec un état absorbant.

Dans [HM07], Haddad & Moreaux traitent le cas particulier d'une DTMC $X = (\mathcal{S}, \mathbf{P})$ défini sur l'espace d'état $\mathcal{S} = \{1, \ldots, n+1\}$, où l'état $n+1$ est l'unique état absorbant de la chaîne. La matrice de transition sous-stochastique est définie sur l'ensemble $\{1, \ldots, n\}$ tel que *i)* la $(n+1)^{ième}$ ligne est le $(n+1)$ vecteur ligne $[0, \ldots, 0, 1]$, *ii)* toute matrice sous-stochastique $n \times n$ peut avoir une unique extension vers la matrice stochastique $(n+1) \times (n+1)$.
Soit $\mathcal{P}_{\mathbf{L},\mathbf{U},\Delta}$ l'ensemble des matrices de probabilités de transition associé à la chaîne de Markov X défini par la matrice sous-stochastique $n \times n$ $\mathbf{L}$, la matrice positive $n \times n$ $\mathbf{U}$ et un vecteur positif Δ de taille n vérifiant :

$$0 \leq \mathbf{L}[i,j] \leq \mathbf{U}[i,j] \ \wedge \sum_{z \in \mathcal{S}} \mathbf{L}[i,z] + \Delta[i] \leq 1.$$

La matrice sous-stochastique $n \times n$ $\mathbf{P}$ est dite appartenir à l'ensemble $\mathcal{P}_{\mathbf{L},\mathbf{U},\Delta}$ si $\forall\, i, j$

$$\mathbf{L}[i,j] \leq \mathbf{P}[i,j] \leq \mathbf{U}[i,j] \ \wedge \ \sum_{j \in \mathcal{S}} \mathbf{P}[i,j] \leq 1 - \Delta[i].$$

Théorème 3.5 *Soient* $\mathbf{L}$ *et* $\mathbf{U}$ *comme décrit ci-dessus. Il existe une matrice st-monotone* $\mathbf{P}^\star$ *telle que :*

$$\forall\, \mathbf{P} \in \mathcal{P}_{\mathbf{L},\mathbf{U}}, \quad \mathbf{P}^\star \leq_{st} \mathbf{P}.$$

La matrice $\mathbf{P}^\star$ est la plus grande matrice st-monotone et représente la matrice borne inférieure au sens de l'ordre "st" pour l'ensemble $\mathcal{P}_{\mathbf{L},\mathbf{U}}$.
La construction de $\mathbf{P}^\star$ s'effectue en deux étapes.

1. D'abord, nous définissons une borne st-inférieure $\mathbf{P}^\bullet \in \mathcal{P}_{\mathbf{L},\mathbf{U}}$ à partir des matrices $\mathbf{L}$ et $\mathbf{U}$ (voir algorithme 4).

Algorithme 4 : Construction de la plus grande borne inférieure $\mathbf{P}^{\bullet}$

ENTRÉES : $\mathbf{L}$, $\mathbf{U}$: matrices $n \times n$; Δ : vecteur de taille n et représente les probabilités de transition minimale pour atteindre l'état absorbant ;
SORTIES : $\mathbf{P}^{\bullet} \in \mathcal{P}_{\mathbf{L},\mathbf{U},\Delta}$;
$\mathbf{P}^{\bullet} \leq_{st} \forall\, \mathbf{P} \in \mathcal{P}_{\mathbf{L},\mathbf{U},\Delta}$;

Pour $i = 1$ **à** n **faire**
  Pour $j = 1$ **à** n **faire**
    $\mathbf{P}^{a}[i,\, j] = \min(\sum_{k=1}^{j} \mathbf{U}[i,\, k],\, 1 - \sum_{k=j+1}^{n} \mathbf{L}[i,\, k]) - \Delta[i]$;
    Si $(i \leq j)$ **et** $(\mathbf{P}^{a}[i,\, j] == 1)$ **alors**
      break ;
    Fin si
  Fin pour
  $\underline{\mathbf{P}}^{\bullet}[i,\, 1] = \mathbf{P}^{a}[i,\, 1]$;
  Pour $j = 2$ **à** n **faire**
    $\underline{\mathbf{P}}^{\bullet}[i,\, j] = \mathbf{P}^{a}[i,\, j] - \mathbf{P}^{a}[i,\, j-1]$;
  Fin pour
Fin pour

2. Ensuite, la matrice $\mathbf{P}^{\star}$ est construite à partir $\mathbf{P}^{\bullet}$ afin d'être st-monotone (voir l'algorithme de Vincent 5).

Partant d'une matrice donnée en paramètre d'entrée, l'algorithme 5 permet de déterminer la plus grande matrice monotone borne inférieure au sens de l'ordre $\leq_{st}$. De plus, la matrice $\mathbf{P}^{\star}$ est une matrice borne inférieure de toutes les matrices de transition prises dans l'ensemble. En d'autres termes, $\forall \mathbf{P} \in \mathcal{P}_{\mathbf{L},\mathbf{U}}$:

$$\forall i,\ \forall j,\ \sum_{k=1}^{j} \mathbf{P}^{\star}[i,\, k] \geq \sum_{k=1}^{j} \mathbf{P}[i,\, k].$$

Algorithme 5 : Construction de la borne inférieure $\leq_{st}$-monotone $\mathbf{P}^{\star}$ (algorithme de **Vincent**)

ENTRÉES : $\mathbf{P}^{\bullet}$, voir l'algorithme 4 ;
SORTIES : $\mathbf{P}^{\star} \leq_{st} \forall\, \mathbf{P} \in \mathcal{P}_{\mathbf{L},\mathbf{U}}$; $\mathbf{P}^{\star}$ est monotone ;

$\mathbf{P}^{\star}[n,\, \bullet] = \mathbf{P}^{\bullet}[n,\, \bullet]$;
Pour $i = n - 1$ **jusqu'à** 1 **faire**
  $x = 0$;
  Pour $j = 1$ **à** n **faire**
    $\mathbf{P}^{\star}[i,\, j] = \max(\sum_{k=1}^{j} \mathbf{P}^{\bullet}[i,\, k], \sum_{k=1}^{j} \mathbf{P}^{\star}[i+1,\, k]) - x$;
    $x = x + \mathbf{P}^{\star}[i,\, j]$;
  Fin pour
Fin pour

Nous remarquons que la matrice $\mathbf{P}^{\star}$ peut être construite à partir de $\mathbf{L}$ et $\mathbf{U}$ en un temps linéaire par rapport à leur taille (voir les algorithmes 4 et 5).

3.3.2 Méthode de Haddad & Moreaux - Généralisation Bušić

Une version légèrement modifiée de l'approche de Haddad & Moreaux a été développée par Bušić [Bus07] dans le cadre de sa thèse et consiste à étudier le cas général où la chaîne de Markov est décrite par une matrice ou une famille de matrices arbitraires, pas uniquement absorbante. Bušić dans [Bus07] a aussi étendu l'étude en déduisant un algorithme pour le calcul de la meilleure borne supérieure $\leq_{st}$-monotone. Les algorithmes de construction de la meilleure borne st-inférieure et st-supérieure de Haddad & Moreaux - Généralisation Bušić sont détaillés ci-après.

▸ **Construction de la meilleure borne st-inférieure $\underline{\mathbf{P}}_{H\&M,B}$**

Nous notons $\underline{\mathbf{P}}_{H\&M,B}$ la matrice de transition borne inférieure associée à la chaîne de Markov $\underline{X} = \{\underline{X}(n), n \geq 0\}$ définie par le 2-tuple $(\mathcal{S}, \underline{\mathbf{P}}_{H\&M,B})$.

Algorithme 6 : Construction de la plus grande borne inférieure $\underline{\mathbf{P}}^{\bullet}$

ENTRÉES : $\mathbf{L}, \mathbf{U}$: matrices $n \times n$.
SORTIES : $\underline{\mathbf{P}}^{\bullet} \in \mathcal{P}_{\mathbf{L},\mathbf{U}}$;
$\underline{\mathbf{P}}^{\bullet} \leq_{st} \forall\, \mathbf{P} \in \mathcal{P}_{\mathbf{L},\mathbf{U}}$;

Pour $i = 1$ **à** n **faire**
  Pour $j = 1$ **à** n **faire**
    $\mathbf{P}^{a}[i,\, j] = \min(\sum_{k=1}^{j} \mathbf{U}[i,\, k],\, 1 - \sum_{k=j+1}^{n} \mathbf{L}[i,\, k])$;
    Si $(i \leq j)$ **et** $(\mathbf{P}^{a}[i,\, j] == 1)$ **alors**
      halt ;
    Fin si
  Fin pour
  $\underline{\mathbf{P}}^{\bullet}[i,\, 1] = \mathbf{P}^{a}[i,\, 1]$;
  Pour $j = 2$ **à** n **faire**
    $\underline{\mathbf{P}}^{\bullet}[i,\, j] = \mathbf{P}^{a}[i,\, j] - \mathbf{P}^{a}[i,\, j-1]$;
  Fin pour
Fin pour

Partant de la matrice $\underline{\mathbf{P}}^{\bullet}$ donnée en entrée, l'algorithme 5 permet de déterminer la plus grande matrice borne inférieure $\leq_{st}$-monotone notée $\underline{\mathbf{P}}_{H\&M,B}$.

▸ **Construction de la meilleure borne st-supérieure $\overline{\mathbf{P}}_{H\&M,B}$**

De la même façon, on construit la matrice borne supérieure avec l'algorithme 7.

En utilisant l'algorithme 1, nous obtenons une matrice borne supérieure monotone $\overline{\mathbf{P}}_{H\&M,B}$ au sens de l'ordre "st". Nous notons par $\{\overline{X}(n), n \geq 0\}$ la chaîne de Markov décrit par le 2-tuple $(\mathcal{S},\ \overline{\mathbf{P}}_{H\&M,B})$.

Algorithme 7 : Bušić : Construction de la matrice borne supérieure $\overline{\mathbf{P}}^{\bullet}$

ENTRÉES : $\mathbf{L}$, $\mathbf{U}$: matrices $n \times n$.
SORTIES : $\overline{\mathbf{P}}^{\bullet} \in \mathcal{P}(\mathbf{L}, \mathbf{U})$;
$\forall\, \mathbf{P} \in \mathcal{P}_{\mathbf{L},\mathbf{U}} \leq_{st} \overline{\mathbf{P}}^{\bullet}$;

1: **Pour** $i = 1$ **à** n **faire**
2: **Pour** $j = n$ **jusqu'à** 2 **faire**
3: $\mathbf{P}^a[i, j] = \min(\sum_{k=j}^{n} \mathbf{U}[i, k], 1 - \sum_{k=1}^{j-1} \mathbf{L}[i, k])$;
4: **Fin pour**
5: $\mathbf{P}^a[i, 1] = 1, \overline{\mathbf{P}}^{\bullet}[i, n] = \mathbf{P}^a[i, n]$;
6: **Pour** $j = n - 1$ **jusqu'à** 1 **faire**
7: $\overline{\mathbf{P}}^{\bullet}[i, j] = \mathbf{P}^a[i, j] - \mathbf{P}^a[i, j + 1]$;
8: **Fin pour**
9: **Fin pour**

Exemple 3.3 *Soient les deux matrices sous-stochastique* $\boldsymbol{L}$ *et sur-stochastique* $\boldsymbol{U}$ *données précédemment :*

$$\boldsymbol{L} = \begin{bmatrix} 0.2744 & 0.2765 & 0.3779 & 0 \\ 0.1024 & 0 & 0.2651 & 0.1430 \\ 0.3844 & 0.3912 & 0 & 0.0488 \\ 0.0248 & 0.4571 & 0.2229 & 0.0001 \end{bmatrix}, \qquad \boldsymbol{U} = \begin{bmatrix} 0.3456 & 0.3477 & 0.4490 & 0 \\ 0.3067 & 0 & 0.7282 & 0.4282 \\ 0.5085 & 0.5152 & 0 & 0.1004 \\ 0.0544 & 0.7522 & 0.4885 & 0.0003 \end{bmatrix}$$

l'application des algorithmes 6 et 2 donne :

$$\underline{\boldsymbol{P}}^{\bullet} = \begin{bmatrix} 0.3456 & 0.2765 & 0.3779 & 0 \\ 0.3067 & 0 & 0.5503 & 0.1430 \\ 0.5085 & 0.4427 & 0 & 0.0488 \\ 0.0544 & 0.7165 & 0.2290 & 0.0001 \end{bmatrix} \text{ et } \quad \underline{\boldsymbol{P}}_{H\&M,B} = \begin{bmatrix} 0.5085 & 0.4427 & 0.0488 & 0 \\ 0.5085 & 0.4427 & 0.0487 & 0.0001 \\ 0.5085 & 0.4427 & 0.0487 & 0.0001 \\ 0.0544 & 0.7165 & 0.2290 & 0.0001 \end{bmatrix}$$

la borne inférieure et la borne inférieure monotone pour l'ordre "st".
De même, la borne supérieure et la borne supérieure $\leq_{st}$*-monotone sont déterminées respectivement par l'application de l'algorithme 7 et 1 :*

$$\overline{\boldsymbol{P}}^{\bullet} = \begin{bmatrix} 0.2744 & 0.2766 & 0.4490 & 0 \\ 0.1024 & 0 & 0.4694 & 0.4282 \\ 0.3844 & 0.5152 & 0 & 0.1004 \\ 0.0248 & 0.4864 & 0.4885 & 0.0003 \end{bmatrix} \text{ et } \quad \overline{\boldsymbol{P}}_{H\&M,B} = \begin{bmatrix} 0.2744 & 0.2766 & 0.4490 & 0 \\ 0.1024 & 0 & 0.4694 & 0.4282 \\ 0.1024 & 0 & 0.4694 & 0.4282 \\ 0.0248 & 0.0776 & 0.4694 & 0.4282 \end{bmatrix}$$

3.3.2.1 Complexité

Notons que la complexité liée à l'application des algorithmes 1 et 7 pour le calcul d'une borne supérieure st-monotone (resp. les algorithmes 2 et 6 pour le calcul d'une borne inférieure st-monotone) est au pire quadratique (dans le cas de matrices pleines) en taille de l'espace d'état.

3.4 Précision des bornes : Résultats théoriques

Nous nous sommes intéressés dans cette partie à l'étude de la précision des différentes bornes développées dans ce chapitre. En effet, nous avons établi certains résultats théoriques permettant de faire le lien entre les différents algorithmes étudiés. Pour ce faire, nous avons prouvé certaines relations théoriques sur les matrices, mais aussi sur les fonctions croissantes des distributions de probabilités stationnaires générées. De même, nous avons utilisé la théorie de comparaison stochastique afin de voir si l'ordre stochastique peut être défini entre les matrices de transition et donc entre les chaînes de Markov. On notera que seuls les résultats sur les bornes supérieures sont présentés. Les résultats sur les bornes inférieures peuvent être obtenus de manière analogue et sont donc omis.

Nous avons comme premier résultat le théorème suivant.

Théorème 3.6 *Les matrices de probabilité de transition* $\overline{\boldsymbol{P}}_{H\&M,\,B}$ *et* $\overline{\boldsymbol{P}}_{\mathrm{Buch}}$ *sont "st" comparable :*

$$\overline{\boldsymbol{P}}_{\mathrm{Buch}} \leq_{st} \overline{\boldsymbol{P}}_{H\&M,\,B}$$

PREUVE. L'algorithme de Haddad & Moreaux permet de construire la matrice borne supérieure pour toute matrice $\mathbf{P} \in \mathcal{P}_{\mathbf{L},\mathbf{U}}$. La matrice $\overline{\mathbf{P}}_{\mathrm{Buch}}$ (algorithme de Buchholz) correspond à une des matrices $\mathbf{P} \in \mathcal{P}_{\mathbf{L},\mathbf{U}}$, une déduction simple est alors établie et revient à dire que la matrice $\overline{\mathbf{P}}_{\mathrm{Buch}}$ est stochastiquement inférieure à $\overline{\mathbf{P}}_{H\&M,\,B}$. ■

Soient $\{\overline{\mathbf{X}}_{H\&M,\,B}(n),\ n \geq 0\}$ une chaîne de Markov finie avec matrice de transition monotone $\overline{\mathbf{P}}_{H\&M,\,B}$ tels que $\overline{\mathbf{P}}^{\bullet} \leq_{st} \overline{\mathbf{P}}_{H\&M,\,B}$ et $\{\overline{\mathbf{X}}_{\mathrm{Buch}}(n),\ n \geq 0\}$ une chaîne de Markov finie avec matrice de transition $\overline{\mathbf{P}}_{\mathrm{Buch}}$, nous avons établi les relations suivantes.

Propriété 3.1 *Les chaînes de Markov sont "st" comparable si les distributions initiales sont "st" comparable :*

$$\{\overline{\boldsymbol{X}}_{\mathrm{Buch}}(n), n \geq 0\} \leq_{st} \{\overline{\boldsymbol{X}}_{H\&M,\,B}(n), n \geq 0\}$$

PREUVE. Nous supposons que la relation est vraie pour $n = 0$. Vu que $\overline{\mathbf{P}}^{\bullet} \leq_{st} \overline{\mathbf{P}}_{H\&M,\,B}$, alors on déduit que $\overline{\mathbf{P}}_{\mathrm{Buch}} \leq_{st} \overline{\mathbf{P}}_{H\&M,\,B}$. De plus, comme la matrice $\overline{\mathbf{P}}_{H\&M,\,B}$ est monotone, la comparaison des chaînes de Markov correspondantes est établie (voir théorème 2.6). ■

Étant donné que la comparaison entre les chaînes de Markov induit la comparaison des distributions de probabilité, nous avons alors :

Propriété 3.2 *Si les distributions stationnaires existent, elles sont "st" comparable :*

$$\boldsymbol{\pi}_{\overline{\boldsymbol{P}}_{\mathrm{Buch}}} \leq_{st} \boldsymbol{\pi}_{\overline{\boldsymbol{P}}_{H\&M,\,B}}$$

Propriété 3.3 *Si les distributions initiales sont égales, les distributions transitoires sont "st" comparable à tout instant* t.

Pour des fonctions de récompenses croissantes sur les distributions de probabilités stationnaires, nous avons le résultat suivant :

Propriété 3.4 *Pour toute récompense positive non décroissante, on a :*

$$\overline{R}_{\text{Buch}} \leq \overline{R}_{H\&M,\,B}$$

Nous avons montré dans cette section, que les algorithmes donnés par Buchholz et Haddad & Moreaux généralisation Bušić sont comparables à partir de mesures décrites par des fonctions positives non décroissantes, sur les distributions de probabilités stationnaires. Notons que la propriété 3.4 est conforme à la preuve faite par Buchholz mentionnant que son algorithme fournit la meilleure borne pour une récompense donnée. Néanmoins, nous avons établi beaucoup plus de résultats. En effet, les propriétés 3.1,3.2 et 3.3 n'ont pas été établies avant et permettent de ce fait de comparer les deux matrices de transition ainsi que leurs distributions. Cependant, une évaluation de l'écart entre les bornes est impossible à définir explicitement, nous proposons donc de vérifier ces algorithmes par une approche numérique en considérant des modèles avec différentes tailles d'espace d'états.

3.5 Exemples et résultats numériques

Afin de comparer les différentes approches pour le calcul de borne, nous considérons deux exemples détaillés ci-après. Toutes les expériences ont été effectuées sur un PC ayant comme caractéristiques 4GB de mémoire vive et un processeur Intel Core i7 2.53GHz. L'environnement Matlab est utilisé pour l'implémentation des algorithmes.

Exemple 1

Ce premier exemple a pour objectif de comparer les matrices de transition obtenues par les différents algorithmes. On considère pour cela un système assez petit décrit comme suit : Le système est défini par un buffer de capacité 1 et d'un serveur, les pièces arrivent suivant un processus de Poisson de taux λ. Pendant le service, une panne peut subvenir sur le serveur et nécessite par conséquent d'être réparé. Le temps de panne et les délais de réparation sont distribués exponentiellement avec respectivement les taux μ, ω et ν. L'espace d'état peut être décrit par le tuple (x_1, x_2) où x_1 décrit le nombre de pièces en service (0 ou 1) et x_2 décrit l'état du serveur (0 panne, 1 ok). L'espace d'état global se compose donc des états suivants : $(0, 1)$, $(1, 1)$ et $(0, 0)$ et la matrice génératrice de la Chaîne de Markov à temps continu est donnée par :

$$\begin{bmatrix} -\lambda & \lambda & 0 \\ \mu & -(\mu+\omega) & \omega \\ \nu & 0 & -\nu \end{bmatrix}$$

Les différents taux appartiennent aux intervalles suivants : taux d'arrivée (λ) appartient à l'intervalle $[0.495, 0.505]$, taux de service (μ) est dans [0.99, 1.01], taux de défaillance (ω) dans [9.99 . 10^{-5}, 1.01 . 10^{-4}] et le taux de réparation (ν) est dans [9.99 . 10−4, 1.01 . 10^{-3}].

Comme mesure de récompense, nous prenons une récompense non décroissante sur le nombre d'états, c.-à-d., $\mathbf{r}[i] \leq \mathbf{r}[j]$ pour $i < j$ et $\mathbf{r} = [1, 2, 3]^t$. Tous les taux sont connus avec une précision de $\pm 1\%$. Pour q=1.010101 les matrices **L** et **U** sont données comme suit :

$$\mathbf{L} = \begin{bmatrix} 5.0005\text{e-}1 & 4.9005\text{e-}1 & 0 \\ 9.8010\text{e-}1 & 0 & 9.8901\text{e-}5 \\ 9.8901\text{e-}4 & 0 & 9.9900\text{e-}1 \end{bmatrix} \qquad \mathbf{U} = \begin{bmatrix} 5.0995\text{e-}1 & 4.9995\text{e-}1 & 0 \\ 9.9990\text{e-}1 & 1.9801\text{e-}2 & 9.9990\text{e-}5 \\ 9.9990\text{e-}4 & 0 & 9.9901\text{e-}1 \end{bmatrix}$$

Nous nous intéressons ici au calcul de la borne supérieure. L'application des algorithmes de Haddad & Moreaux, Coutois & Semal et Buchholz, nous permet d'obtenir les matrices de probabilités de transition suivantes :

$$\overline{\mathbf{P}}_{H\&M,\,B} = \begin{bmatrix} 5.0005e-1 & 4.9995e-1 & 0 \\ 5.0005e-1 & 4.9985e-1 & 9.9990e-5 \\ 9.8901e-4 & 0 & 9.9901e-1 \end{bmatrix}$$

$$\overline{\mathbf{P}}_{C\&S} = \begin{bmatrix} 5.0005e-1 & 4.9005e-1 & 9.9000e-3 \\ 9.8010e-1 & 0 & 1.9899e-2 \\ 9.8901e-4 & 0 & 9.9901e-1 \end{bmatrix}$$

$$\overline{\mathbf{P}}_{\mathrm{Buch}} = \begin{bmatrix} 5.0995e-1 & 4.9005e-1 & 0 \\ 9.9990e-1 & 0 & 9.9990e-5 \\ 9.8901e-4 & 0 & 9.9901e-1 \end{bmatrix}$$

Les distributions stationnaires ainsi que les récompenses moyennes associées sont :

$$\boldsymbol{\pi}_{\overline{\mathbf{P}}_{H\&M,\,B}} = [4.7604e-1,\ 4.7585e-1,\ \ 4.8109e-2],\ \mathrm{R}_{H\&M,\,B} = 1.5721.$$
$$\boldsymbol{\pi}_{\overline{\mathbf{P}}_{C\&S}} = [4.6817e-2,\ 2.2943e-2,\ \ 9.3024e-1],\ \mathrm{R}_{C\&S} = 2.8834.$$
$$\boldsymbol{\pi}_{\overline{\mathbf{P}}_{\mathrm{Buch}}} = [6.4952e-1,\ 3.1830e-1,\ \ 3.2180e-2],\ \mathrm{R}_{\mathrm{Buch}} = 1.3827.$$

Une première observation peut déjà être tirée et semble tout à fait justifiée. Les bornes basées sur la connaissance des deux matrices **L** et **U** sont nettement plus précises que celles basées sur la matrice **L** (resp. **U**) uniquement. De plus, nous pouvons voir que les bornes de Courtois & Semal ne sont pas vraiment précises vu que les probabilités de transition associées à la matrice $\overline{\mathbf{P}}_{C\&S}$ sont plus grandes que nécessaires. La raison de ce comportement vient du fait que seule la matrice **L** (ou **U**) est connue et par conséquent la contrainte sur la matrice $\overline{\mathbf{P}}_{C\&S} \leq \mathbf{U}$ (resp. $\overline{\mathbf{P}}_{C\&S} \geq \mathbf{L}$) est nullement visible, donc il n'est pas possible de calculer des bornes précises pour le modèle étudié.

Exemple 2

Dans cet exemple, pour nos expériences numériques nous avons utilisé un algorithme pour générer des matrices respectivement supérieures et inférieures par élément **L** et **U**. Ces matrices sont supposées pleines et l'algorithme est donné comme suit : a) générer aléatoirement une matrice stochastique irréductible **G** ayant comme taille d'espace d'état n tels que $\forall i, j, \mathbf{G}[i, j] > 0$. b) dériver la matrice sous-stochastique **L** (resp. matrice sur-stochastique

U) telle que la connaissance de **L** impose certaines restrictions sur les valeurs des éléments de **U** et vice versa. Nous supposons que les inégalités (3.2) et (3.3) sont satisfaites.

Nous avons effectué un grand nombre de tests sur des matrices avec diverses tailles d'espaces d'états. Nous présentons ici une sélection de ces résultats, afin d'expliquer nos conclusions générales. Dans le tableau 3.1, nous donnons quelques indices de performances pour les différentes approches associées aux valeurs bornes supérieures obtenues par les différents algorithmes. Nous supposons que la fonction de récompense croissante est donnée par $\mathbf{r}[i] = \frac{1}{2}i^2,\ \forall\ i \in \mathbb{N}\ et\ 1 \leq i \leq n$.

n Taille des Matrices (**L** et **U**)	Algorithmes	Resultats		
		Temps d'exécution (s)	R	$\mathbb{E}[\boldsymbol{\pi}]$
200	Buchholz	0.3421	5.4631 e+3	86.6566
	Courtois & Semal	0.0076	1.2122 e+4	140.9104
	Haddad & Moreaux - Bušić	0.0319	8.0419 e+3	141.7419
500	Buchholz	2.2921	3.3194 e+4	214.9855
	Courtois & Semal	0.0549	7.5515 e+4	351.200
	Haddad & Moreaux - Bušić	0.6836	6.3938 e+4	335.6318
700	Buchholz	5.8855	6.4710 e+4	300.9577
	Courtois & Semal	0.1141	1.4738 e+5	490.9176
	Haddad & Moreaux - Bušić	1.5993	1.2605 e+5	470.9664
1000	Buchholz	21.0880	1.3277 e+5	431.7384
	Courtois & Semal	0.2820	3.0612 e+5	708.9148
	Haddad & Moreaux - Bušić	7.1773	2.4996 e+5	660.4634
2000	Buchholz	102.4151	5.3003 e+5	863.6831
	Courtois & Semal	2.8089	1.2410 e+6	1.4306 e+3
	Haddad & Moreaux - Bušić	60.7278	9.9466 e+5	1.3206 e+3

TABLE 3.1 – Résultats pour les différentes tailles de l'espace d'états

Comme on pouvait s'y attendre, le temps d'exécution de la méthode de Buchholz est plus grand que celui de la méthode de Haddad & Moreaux et de Courtois & Semal, car la méthode repose sur le calcul de matrices inverses. Concernant le temps de calcul de la méthode Courtois & Semal, il demeure largement inférieure aux deux autres méthodes. Ceci s'explique par le fait que seule la matrice **L** est considérée lors de la résolution. Pour ce qui est de la qualité des différentes bornes, nous ne pouvons effectuer pas de comparaison directe entre les différentes mesures calculées, car ces méthodes ne reposent pas sur les mêmes critères d'optimalité. Cependant, nous pouvons émettre quelques lignes directrices aidant aux choix des méthodes appropriées.

3.6 Quelques lignes directrices pour le choix des algorithmes appropriés

Dans ce chapitre, nous avons prouvé ou constaté expérimentalement les assertions suivantes :

- La matrice définie par Buchholz est toujours plus petite que la matrice obtenue par Haddad et Moreaux au sens de l'ordre stochastique.
- La matrice construite par l'algorithme de Haddad & Moreaux peut être utilisée dans le but de dériver un plus grand nombre de résultats. En effet, nous pouvons borner aussi bien les distributions stationnaires que transitoires et toute fonction de récompenses calculée sur ces distributions. La distribution du temps de premier passage peut également être calculée à partir de la même matrice.
- La complexité liée à la construction des algorithmes Haddad & Moreaux et Haddad & Moreaux-Vincent (matrice monotone) est au plus quadratique avec une implémentation de matrices pleines, tandis que les approches de Buchholz et Courtois ont une complexité cubique.
- Même si nous n'avons pas de preuve formelle, les algorithmes Haddad & Moreaux et Haddad & Moreaux-Vincent semblent être numériquement stables. En effet, la plupart des calculs sont des opérations d'addition et de $\max$ sur des matrices positives. Ces opérations sont connues pour être stables.
- Pour différentes valeurs de la fonction de récompense, l'approche de Buchholz nécessite l'analyse du modèle pour chaque fonction de récompense définissant ainsi plusieurs matrices bornantes. Ce calcul est effectué de façon indépendante rendant ainsi le calcul très coûteux. Il n'existe pour le moment aucune méthode qui permet de construire de manière itérative une matrice bornante à partir d'une matrice précédemment calculée.
- Buchholz a récemment rapporté dans [Buc10] que son algorithme peut avoir quelques problèmes de précision. Ces fluctuations sont dues à une certaine instabilité numérique qui survient lors de l'opération d'inversion de la matrice.

Il est important de noter que ces remarques peuvent nous aider à trouver la solution la plus appropriée en fonction de l'usage que nous avons des bornes.

3.7 Conclusion

Nous avons présenté dans ce chapitre différentes méthodes de bornes utilisées pour des chaînes de Markov imprécises avec récompense. Pour ce faire, nous avons étudié deux approches distinctes ; une basée sur la théorie polyédrale et l'autre sur la comparaison stochastique. Nous avons remarqué que les méthodes associées aux deux approches reposent sur des hypothèses et des critères d'optimalité différents. Afin de mener notre étude comparative, nous avons considéré des hypothèses permettant de satisfaire les deux approches. Cette dernière permet de définir quelques lignes directrices pour le choix de la méthode à adopter.

Néanmoins, pour des chaînes de Markov à espace d'état très grand ces méthodes nécessitent

un temps et un nombre d'étapes de calcul importants. Par exemple, la théorie polyédrale développée par Courtois et Semal [CS84] et appliquée par Muntz dans [FM94] est basée sur la résolution de n chaînes de Markov (où n représente la taille de l'espace d'état) résultant ainsi sur une complexité globale de l'ordre de n^4. De même, Buchholz a donné dans [Buc05] et [Buc10] deux autres méthodes reposant sur la construction de bornes sur un ensemble de chaînes de Markov. Les deux algorithmes proposés exigent un grand nombre d'étapes de calcul comparé aux algorithmes que nous allons développer dans le chapitre suivant. De plus, le premier algorithme est signalé par l'auteur dans [Buc10] être numériquement instable et l'analyse de grandes chaînes n'est pas réalisable.

Dans l'optique de faire un compromis entre le temps de calcul et la précision des bornes sur le vecteur de distribution stationnaire, nous avons développé dans le chapitre suivant, de nouveaux algorithmes itératifs pour déterminer des bornes inférieures et supérieures par élément de la distribution stationnaire.

Chapitre 4

Nouvelles bornes par élément pour les chaînes de Markov

De nombreux résultats ont été proposés pour déterminer des bornes sur le vecteur de distribution stationnaire associé à une chaîne de Markov imprécise (théorie polyédrale développée par Courtois et Semal [CS84] et appliquée par Muntz et ses collègues dans [FM94] pour un problème de fiabilité et, plus récemment, par Buchholz [Buc05]). Dans ce chapitre, nous construisons de nouveaux algorithmes de bornes par élément sur le vecteur de distribution stationnaire de chaînes de Markov incomplètement spécifiées. Nous proposons de développer une nouvelle méthode combinant respectivement la théorie polyédrale de Muntz et la technique numérique développée par Bušić et Fourneau dans [BF11]. Ce nouvel algorithme plus rapide, permet de fournir à chaque itération des bornes supérieures et inférieures par élément de la distribution stationnaire (*i.e.,* nous obtenons des bornes à la première itération et à chaque itération ces bornes sont améliorées).

Nous nous intéressons également au cas des chaînes de Markov absorbantes et imprécises. Nous montrons dans ce cas, l'applicabilité de notre nouvel algorithme pour déterminer des bornes supérieures et inférieures sur les temps moyens d'absorption (*Mean Time To Failure*, noté MTTF) pour un ensemble de DTMC absorbantes. Nous présentons pour ce faire, un lien entre le MTTF d'une chaîne absorbante et le vecteur de distribution stationnaire d'une chaîne de Markov ergodique. Considérant la chaîne de Markov ergodique dérivée de la chaîne absorbante, nous proposons d'apporter quelques transformations au modèle initial afin d'en dériver des bornes sur le temps moyen d'absorption. Pour finir, nous abordons un cas particulier pour les chaînes de Markov finies en temps discret. Déjà étudié par Bušić et Fourneau dans [BF10], la problématique revient à considérer une chaîne de Markov avec matrice de probabilités de transition $\mathbf{P}$ telle que la quantité $\nabla_{\mathbf{P}}$ est égale au vecteur nul (la quantité ∇ est définie de manière précise dans la section 4.1). Bušić et Fourneau ont proposé plusieurs algorithmes pour tenter de trouver des solutions approchées du vecteur de distribution stationnaire de la chaîne de Markov. Cependant, ces algorithmes ne convergent que vers un multiple inconnu de la distribution stationnaire. Ainsi, nous proposons dans ce chapitre d'apporter quelques modifications voire améliorations aux algorithmes de Bušić et Fourneau en intégrant la notion de diamètre du graphe associé à la chaîne de Markov étudiée. Nous développons de nouveaux algorithmes pour définir des bornes inférieures (resp. supérieures)

sur la distribution stationnaire de la chaîne de Markov dont la convergence a été prouvée.

Ce chapitre suivra le plan suivant : dans les sections 4.1 et 4.2, nous décrivons respectivement les algorithmes $I\nabla L$ et $I\nabla U$ présentés par Bušić et Fourneau dans [BF11] et l'approche polyédrale de Muntz [FM94]. Ensuite, en nous reposant sur le théorème de Muntz fondé sur l'approche polyédrale de Courtois et Semal, nous montrons comment combiner ces résultats avec ceux de Bušić et Fourneau [BF11] pour dériver des bornes sur la distribution stationnaire de l'ensemble des chaînes de Markov. La section 4.4 est consacrée à l'étude de chaînes de Markov absorbantes et imprécises. Pour ce nouveau problème, nous présentons une nouvelle démarche permettant de définir des bornes sur le temps moyen d'absorption associée à la chaîne. Pour finir, nous étudions dans la section 4.5, le cas particulier des chaînes de Markov avec $\nabla_{\mathbf{P}} = \mathbf{0}$.

4.1 Algorithmes basés sur des suites monotones

Bušić et Fourneau [BF11] ont proposé de nouveaux algorithmes itératifs qui permettent de fournir des bornes par élément de la distribution de probabilités stationnaires d'une chaîne de Markov irréductible et apériodique. Fondée sur les propriétés $(\max, +)$ et $(\min, +)$, l'idée est de concevoir une nouvelle définition constructive qui améliore à chaque itération la distribution stationnaire d'une chaîne de Markov en temps discret. Cette méthode offre ainsi la possibilité de faire un compromis entre la précision des bornes et le temps de calcul. Nous allons présenter dans cette section le schéma itératif croissant (resp. décroissant) pour la construction d'une limite inférieure (resp. supérieure) de la distribution stationnaire associée à une chaîne de Markov.

Soit $\mathbf{P}$ la matrice de transition associée à une DTMC finie irréductible et apériodique avec vecteur de distribution stationnaire $\boldsymbol{\pi}$. Nous introduisons au préalable quelques quantités faciles à calculer ainsi que le lemme trivial 4.1 qui permet de définir le vecteur initial de la séquence à construire.

Définition 4.1 *Soient* $\nabla_{\mathbf{P}}[j] = \min_i \mathbf{P}[i, j]$ *et* $\Delta_{\mathbf{P}}[j] = \max_i \mathbf{P}[i, j]$. *Nous précisons que le vecteur* $\nabla_{\mathbf{P}}$ *peut être égal à* $\mathbf{0}$ *par contre le vecteur* $\Delta_{\mathbf{P}}$ *est toujours positif puisque la chaîne est irréductible.*

Lemme 4.1 *Sachant que* $\boldsymbol{\pi} = \boldsymbol{\pi}\,\mathbf{P}$*, et que pour tout* j*,* $\boldsymbol{\pi}[j]$ *est compris entre* 0 *et* 1*, alors nous avons :*

$$\nabla_{\mathbf{P}}[j] \leq \boldsymbol{\pi}[j] = \sum_i \boldsymbol{\pi}[i]\,\mathbf{P}[i, j] \leq \Delta_{\mathbf{P}}[j].$$

Les bornes du lemme 4.1 sont données de manière gloutonne et peuvent donc être améliorées et affinées afin de définir un meilleur encadrement de la solution exacte. Pour ce faire, Bušić et Fourneau [BF11] ont proposé deux algorithmes itératifs basés sur les propriétés $(\max, +)$ et $(\min,+)$ qui permettent de calculer des bornes inférieures par l'algorithme noté $I\nabla L$ et des bornes supérieures en utilisant l'algorithme $I\nabla U$. Les deux algorithmes présentés ci-après reposent sur une condition importante qui consiste à avoir $\|\nabla_{\mathbf{P}}\| > 0$.

Nous pouvons trouver dans [BF11] une preuve qui stipule que l'algorithme $I\nabla L$ (algorithme 8) fournit à chaque itération une nouvelle borne inférieure $\mathbf{x}^{(k)}$. Il est intéressant de

remarquer que nous pouvons initialiser l'algorithme avec $\mathbf{a} = \mathbf{b} = \nabla_{\mathbf{P}}$. Notons que l'opérateur "max" est appliqué composante par composante.

Algorithme 8 : Algorithme itératif pour le calcul d'une borne inférieure : I∇L
Iterate on ∇ Lower Bound

ENTRÉES : $0 \leq \mathbf{a} \leq \boldsymbol{\pi}$, $\mathbf{b} \leq \nabla_{\mathbf{P}}$ et $\mathbf{b} \neq \mathbf{0}$.
SORTIES : Valeurs successives de $\mathbf{x}^{(k)}$.

1: $\mathbf{x}^{(0)} = \mathbf{a}$.
2: **Répéter**
3: $\quad \mathbf{x}^{(k+1)} = \max\{\mathbf{x}^{(k)}, \mathbf{x}^{(k)}\mathbf{P} + \mathbf{b}(1 - \|\mathbf{x}^{(k)}\|)\}$.
4: **Jusqu'à** $1 - \|\mathbf{x}^{(k)}\| < \epsilon$.

De même, un algorithme légèrement différent pour le calcul d'une borne supérieure $\mathbf{y}^{(k)}$ du vecteur de distribution stationnaire (noté $I\nabla U$, algorithme 9) a été proposé. Les deux seules différences avec l'algorithme $I\nabla L$ résident dans l'étape d'initialisation et l'utilisation de l'opérateur "min". Le vecteur $\mathbf{y}^{(0)}$ est initialisé par $\mathbf{c} = \Delta_{\mathbf{P}}$.

Algorithme 9 : Algorithme itératif pour le calcul d'une borne supérieure : I∇ U
Iterate on ∇ Upper Bound

ENTRÉES : $\mathbf{c} \geq \boldsymbol{\pi}$, $\mathbf{b} \leq \nabla_{\mathbf{P}}$ et $\mathbf{b} \neq \mathbf{0}$.
SORTIES : Valeurs successives de $\mathbf{y}^{(k)}$.

1: $\mathbf{y}^{(0)} = \mathbf{c}$.
2: **Répéter**
3: $\quad \mathbf{y}^{(k+1)} = \min\{\mathbf{y}^{(k)}, \mathbf{y}^{(k)}\mathbf{P} + \mathbf{b}(1 - \|\mathbf{y}^{(k)}\|)\}$.
4: **Jusqu'à** $\|\mathbf{y}^{(k)}\| - 1 < \epsilon$.

Le théorème suivant présente les résultats de convergence établis par Bušić et Fourneau [BF11] pour la détermination des bornes par élément sur le vecteur de distribution stationnaire.

Théorème 4.1 *Soit* $\mathbf{P}$ *une matrice stochastique irréductible et apériodique avec distribution stationnaire* π. *Si* $\nabla_{\mathbf{P}} \neq \mathbf{0}$, *l'algorithme* $I\nabla L$ *fournit à chaque itération des bornes inférieures pour toutes les composantes de* $\boldsymbol{\pi}$ *et converge vers* $\boldsymbol{\pi}$ *pour toute valeur des paramètres* $\mathbf{a}$ *et* $\mathbf{b}$ *tel que* $\mathbf{a} \leq \boldsymbol{\pi}$, $\mathbf{b} \leq \nabla_{\mathbf{P}}$ *et* $\mathbf{b} \neq \boldsymbol{0}$. *De même, l'algorithme de* $I\nabla U$ *fournit à chaque itération des bornes supérieures pour toutes les composantes de* $\boldsymbol{\pi}$ *et converge vers* $\boldsymbol{\pi}$ *pour toute valeur des paramètres* $\mathbf{c}$ *et* $\mathbf{b}$ *tel que* $\boldsymbol{\pi} \leq \mathbf{c}$, $\mathbf{b} \leq \nabla_{\mathbf{P}}$ *et* $\mathbf{b} \neq \boldsymbol{0}$.

Le lecteur peut se référer à l'article [BF11] pour plus de détails et voir l'ensemble des preuves.

Exemple 4.1 *Soit une matrice stochastique* $\mathbf{P}$ *:*

$$\mathbf{P} = \begin{pmatrix} 0.6 & 0 & 0.2 & 0.2 & 0 \\ 0.4 & 0.2 & 0.1 & 0.2 & 0.1 \\ 0.2 & 0.1 & 0.2 & 0.3 & 0.2 \\ 0.2 & 0 & 0.2 & 0.3 & 0.3 \\ 0.1 & 0 & 0 & 0.4 & 0.5 \end{pmatrix}.$$

Nous avons $\nabla_{\mathbf{P}} = [0.1, 0, 0, 0.2, 0]$ *et* $\Delta_{\mathbf{P}} = [0.6, 0.2, 0.2, 0.4, 0.5]$. *Pour* $\epsilon = 10^{-5}$, *l'algorithme* $I\nabla L$ *avec* $\mathbf{a} = \mathbf{b} = \nabla_{\mathbf{P}}$ *fournit la séquence suivante de bornes inférieures sur les probabilités. La première colonne du tableau représente l'indice d'itération. La dernière colonne correspond au résidu (soit la masse de probabilité qui n'a pas été affectée à cette itération). Les autres colonnes donnent le vecteur de probabilité calculé.*

k	1	2	3	4	5	$1 - \|\|\mathbf{x}^{(k)}\|\|$
1	0.17	0	0.06	0.22	0.06	0.4900
3	0.2413	0.0102	0.1092	0.2546	0.1446	0.2401
7	0.2869	0.0169	0.1409	0.2826	0.2151	0.0576
11	0.2968	0.0183	0.1481	0.2897	0.2332	0.0139
21	0.2997	0.0188	0.1502	0.2920	0.2389	0.0004
31	0.2997	0.0188	0.1503	0.2921	0.2391	$1.1\ 10^{-5}$

Et l'algorithme $I\nabla U$ *initialisé avec* $\mathbf{b} = \nabla_{\mathbf{P}}$ *et* $\mathbf{c} = \Delta_{\mathbf{P}}$ *donne la séquence suivante de bornes supérieures sur les probabilités.*

k	1	2	3	4	5	$\|\|\mathbf{y}^{(k)}\|\| - 1$
1	0.5200	0.0600	0.2000	0.3600	0.4300	0.5700
3	0.3808	0.0264	0.1984	0.3280	0.3303	0.2639
7	0.3149	0.0210	0.1610	0.3021	0.2644	0.0634
11	0.3031	0.0193	0.1528	0.2946	0.2455	0.0153
31	0.2997	0.0188	0.1503	0.2921	0.2391	$1.2\ 10^{-5}$

Précisons que la combinaison des deux résultats nous permet d'obtenir une enveloppe prouvée pour toutes les composantes du vecteur $\boldsymbol{\pi}$. La norme de l'enveloppe converge vers zéro plus vite qu'une géométrique avec un taux $(1 - \|\mathbf{b}\|)$, voir [BF11]. De plus, ces algorithmes ont deux propriétés importantes. La première est que sous certaines conditions techniques, le fait d'avoir une borne par élément sur les matrices stochastiques permet de fournir une borne par élément sur le vecteur de distribution stationnaire. Deuxièmement, sous certaines contraintes, ces algorithmes peuvent être étendus et utilisés pour l'analyse de matrices infinies [FQ12].

4.2 Bornes polyédrales de Muntz

Dans cette section, nous nous intéressons aux résultats de Muntz et Franceschinis [FM94] (théorème 3.3). Fondée sur l'approche polyédrale de Courtois et Semal [CS84], cet article

Muntz permet d'exprimer des bornes sur le vecteur de distribution stationnaire associé au modèle décrit par une matrice sous-stochastique $\mathbf{L}$.

Nous reprenons ci-dessous le théorème de Muntz et al. de manière un peu plus détaillée. Nous notons par n le nombre d'états et m le nombre d'entrées non nulles de la matrice $\mathbf{L}$.

Théorème 4.2 *[FM94] Soit* $\mathbf{L}$ *une borne inférieure par élément d'une matrice de transition associée à une DTMC donnée. Nous pouvons déterminer des bornes sur le vecteur de probabilité stationnaire associé à cette DTMC noté* $\boldsymbol{\pi}$. *Soit* n *la taille de la matrice* $\mathbf{L}$, *pour tout* $i = 1, \ldots, n$, *nous déterminons la matrice de transition* $\mathbf{L}^i$ *associée à la* i*-ème DTMC, obtenue à partir de la matrice sous-stochastique* $\mathbf{L}$ *en augmentant les éléments de la colonne* i *afin de rendre* $\mathbf{L}^i$ *stochastique. La matrice* $\mathbf{L}^i$ *est supposée ergodique. Soit* $\boldsymbol{\pi}^i$ *le vecteur de probabilité stationnaire associé à la* i*-ème DTMC. La borne inférieure (resp. supérieure) du vecteur de distribution stationnaire associée à l'état* j *est calculée par* $\min_i \boldsymbol{\pi}^i[j]$ *(resp.* $\max_i \boldsymbol{\pi}^i[j]$*).*

$$\min_i \boldsymbol{\pi}^i[j] \leq \boldsymbol{\pi}[j] \leq \max_i \boldsymbol{\pi}^i[j] \tag{4.1}$$

Exemple 4.2 *Soit la matrice sous-stochastique* $\mathbf{L}$ *:*

$$\mathbf{L} = \begin{pmatrix} 0.5 & 0.4 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.8 & 0.0 \\ 0.1 & 0.0 & 0.0 & 0.6 \\ 0.3 & 0.0 & 0.0 & 0.6 \end{pmatrix}$$

La résolution des quatres DTMCs ($\mathbf{L}^i$, $i \in \{1, \ldots, 4\}$*) permet de définir les vecteurs de distribution stationnaire suivants :*

$\boldsymbol{\pi}^1 = [0.4545, 0.1818, 0.1455, 0.2182]$, $\boldsymbol{\pi}^2 = [0.2268, 0.2577, 0.2062, 0.3093]$,

$\boldsymbol{\pi}^3 = [0.2723, 0.1089, 0.2475, 0.3713]$ *et* $\boldsymbol{\pi}^4 = [0.3049, 0.1220, 0.0976, 0.4756]$,

Selon Muntz [FM94], le vecteur de distribution stationnaire associé à la matrice $\mathbf{L}$ *est borné respectivement par :*

$$[0.2268, 0.1089, 0.0976, 0.2182] \leq \boldsymbol{\pi} \leq [0.4545, 0.2577, 0.2475, 0.4756]$$

4.3 Calcul de bornes sur la distribution stationnaire d'une chaîne de Markov imprécise

Après avoir présenté les deux résultats importants sur lesquels repose notre approche, nous allons à présent démontrer comment nous pouvons combiner le théorème 4.2 et les algorithmes $I \triangledown L$ et $I \triangledown U$. Cette démarche a pour but de prouver une nouvelle méthode permettant de définir à chaque itération des bornes par élément sur la distribution stationnaire associée à la chaîne partiellement spécifiée.

Définition 4.2 *Soit* $\mathbf{L}$ *une matrice sous-stochastique définie sur* $\mathcal{F}$, *nous spécifions pour tout* $i \in \mathcal{F}$, *la quantité* $\boldsymbol{\alpha}^t[i] = 1 - \sum_j \mathbf{L}[i, j]$. *Nous supposons que* $\forall i \in \mathcal{F}$, $\boldsymbol{\alpha}\,[i] > 0$.

L'hypothèse de positivité de chaque entrée du vecteur $\boldsymbol{\alpha}$ est nécessaire pour notre approche, mais dans le cas où l'hypothèse n'est pas vérifiée il est suffisant de considérer une nouvelle matrice inférieure par élément à $\mathbf{L}$ tel que l'hypothèse sur le vecteur $\boldsymbol{\alpha}$ soit satisfaite.
Nous rappelons que pour tout $i \in \mathcal{F}$, $\mathbf{L}^i$ est une matrice construite à partir de $\mathbf{L}$ en augmentant la colonne i jusqu'à ce que la matrice devienne stochastique. Ainsi, $\mathbf{L}^i = \mathbf{L} + \boldsymbol{\alpha}^t \mathbf{e}_i$. Nous notons par $\boldsymbol{\pi}^i$ le vecteur propre de la matrice $\mathbf{L}^i$.

Propriété 4.1 *Pour tout $i \in \mathcal{F}$, nous avons $\|\nabla_{\mathbf{L}^i}\| > 0$.*

PREUVE. L'hypothèse sur $\boldsymbol{\alpha}$ implique que $\nabla_{\mathbf{L}^i}[i] > 0$ et par conséquent $\|\nabla_{\mathbf{L}^i}\| \neq \mathbf{0}$. ■

4.3.1 Algorithme de calcul de la distribution borne supérieure par élément

Soit $Y^{(k),i}$ une borne supérieure à l'itération k définie par l'algorithme itératif $I\nabla U$ pour la matrice $\mathbf{L}^i$. Suite aux résultats obtenus dans [BF11] ainsi qu'au théorème 4.2, nous établissons les lemmes suivants :

Lemme 4.2 *$\forall i \in \mathcal{F}$, nous avons $\boldsymbol{\pi}^i \leq Y^{(k),i}$, par conséquent $\max_i\{\boldsymbol{\pi}^i\} \leq \max_i\{Y^{(k),i}\}$.*

Lemme 4.3 *Pour tout $k \geq 0$, nous avons $\boldsymbol{\pi} \leq \max_i\{\boldsymbol{\pi}^i\} \leq \max_i\{Y^{(k),i}\}$.*

En se basant sur l'algorithme $I\nabla U$ et sur les résultats établis ci-dessous, nous dérivons un algorithme itératif pour calculer à chaque itération (indice k dans l'algorithme) une borne supérieure sur la distribution stationnaire de $\mathbf{L}$. L'idée principale de l'algorithme présenté ici consiste à calculer d'abord, pour tout $i \in \mathcal{F}$ une borne supérieure $Y^{(k),i}$ associée à la matrice $\mathbf{L}^i$ en utilisant l'algorithme $I\nabla U$. Ensuite, nous utilisons le résultat de Muntz [FM94] pour déduire une borne supérieure sur la distribution stationnaire $\boldsymbol{\pi}$. Ce procédé est répété jusqu'à ce que le critère d'arrêt soit atteint.

Il est prouvé dans [BF11] que chaque séquence $Y^{(k),i}$ converge plus rapidement qu'une géométrique de taux de $(1 - \|\nabla_{\mathbf{L}^i}\|)$. Ainsi, notre critère d'arrêt serait la somme résiduelle de toutes les séquences $Y^{(k),i}$. Une fois que toutes les séquences ont convergé, l'utilisation de l'opérateur $\min$ entre les distributions n'aura donc pas d'influence. Notons que dans le cas où toutes les séquences $Y^{(k),i}$ ont convergé, les bornes que nous définissons sont égales aux bornes fournies par Muntz [FM94].

Algorithme 10 : Algorithme itératif pour le calcul d'une borne supérieure
Iterate on ∇ Upper Bound on a Set : $I\nabla US$

ENTRÉES : $\forall\, i \in \mathcal{F},\ \boldsymbol{\alpha}[i] > 0$;
SORTIES : Valeurs successives de $Y^{(k)}$.

1: $\forall i \in \mathcal{F}$, $\mathbf{L}^i = \mathbf{L} + \boldsymbol{\alpha}^t\mathbf{e}_i$, $\mathbf{c}^i = \Delta_{\mathbf{L}^i}$, $\mathbf{b}^i = \nabla_{\mathbf{L}^i}$, $Y^{(0),i} = \mathbf{c}^i$.
2: **Répéter**
3: $\quad \forall$ i$\in \mathcal{F}$, $Y^{(k+1),i} = \min\{Y^{(k),i}, Y^{(k),i}\mathbf{L}^i + \mathbf{b}^i(1 - \|Y^{(k),i}\|)\}$.
4: $\quad Y^{(k+1)} = \max_i\{Y^{(k+1),i}\}$.
5: **Jusqu'à** $\sum_i(\|Y^{(k),i}\| - 1) < \epsilon$.

Théorème 4.3 *Soit* $\mathbf{L}$ *une matrice sous-stochastique irréductible, l'algorithme 10 fournit à chaque itération* k *une borne supérieure par élément de la distribution stationnaire et cela pour toute matrice ergodique dont les éléments en entrées sont supérieurs à la matrice* $\mathbf{L}$.

PREUVE. Conséquence directe du lemme 4.3. ■

Complexité

L'algorithme repose sur la multiplication d'un vecteur par une matrice, une addition de deux vecteurs et une comparaison par élément de n vecteurs. Ainsi, nous observons que la complexité de l'algorithme est dominée par le produit de la séquence $Y^{(k),i}$ par la matrice $\mathbf{L}^i$. Par contre, pour ce qui est de l'approche de Muntz, sa complexité est beaucoup plus grande. En effet, nous considérons n matrices et nous calculons leur distribution stationnaire en utilisant un algorithme précis comme la méthode GTH [GTH85]. La complexité finale est de $O(n^4)$ en considérant une représentation pleine de la matrice $\mathbf{L}$.

Propriété 4.2 *La complexité à chaque itération de l'algorithme* $I\nabla US$ *est de* $O(n.m)$ *si nous utilisons une représentation de matrice creuse avec* $m > n$.

PREUVE. Supposons que la matrice $\mathbf{L}$ a une représentation creuse avec n lignes et m éléments non nuls. La matrice augmentée $\mathbf{L}^i$, $i \in \mathcal{F}$ a tout au plus $m + n$ éléments différents de zéro. Par conséquent, la complexité du produit vecteur-matrice est de $O(n.m)$ si $m > n$. En outre, la complexité pour rechercher le maximum (resp. minimum) dans un vecteur non trié de longueur n est $O(n)$. ■

Nous illustrons cet algorithme sur un petit exemple.

Exemple 4.3 *Nous considérons la matrice sous-stochastique définie dans l'exemple 4.2. L'algorithme* $I\nabla US$ *permet de définir la séquence de bornes supérieures suivante :*

k	*1*	*2*	*3*	*4*	$\|\|Y^{(k)}\| - 1$
1	*0.6000*	*0.5000*	*0.6600*	*0.9000*	*1.6600*
3	*0.5888*	*0.4110*	*0.3624*	*0.7880*	*1.1502*
11	*0.4769*	*0.3245*	*0.2966*	*0.5853*	*0.6833*
21	*0.4570*	*0.2810*	*0.2646*	*0.5139*	*0.5165*
41	*0.4546*	*0.2606*	*0.2496*	*0.4803*	*0.4450*
61	*0.4545*	*0.2581*	*0.2478*	*0.4762*	*0.4366*
79	*0.4545*	*0.2577*	*0.2475*	*0.4756*	*0.4353*

4.3.2 Algorithme de calcul de la distribution borne inférieure par élément

Soit $X^{(k),i}$ une borne inférieure de la distribution stationnaire de la matrice stochastique L^i à l'itération k fournie par l'algorithme itératif $I\nabla L$.

Lemme 4.4 *Pour tout $k \geq 0$, nous avons : $\mathbf{0} \leq \min_i\{X^{(k),i}\} \leq \min_i\{\boldsymbol{\pi}^i\} \leq \boldsymbol{\pi}$.*

Nous notons que le vecteur $\min_i\{X^{(k),i}\}$ est non négatif. Nous cherchons à définir respectivement le plus petit indice k tel que la séquence $X^{(k),i}$ ait au moins une composante positive. Ainsi que l'indice le plus petit vérifiant que toutes les entrées du vecteur soient positives. Rappelons tout d'abord quelques notions de la théorie des graphes.

▸ **Quelques propriétés de graphe : distance, diamètre et excentricité**

Soit G le graphe orienté associé à la matrice de transition $\mathbf{M}$. Le graphe G est composé d'un ensemble de nœuds noté $\mathcal{V} = \{1, 2, \ldots, n\}$ et d'un ensemble d'arcs $\mathcal{E}$ associé aux entrées non nulles de $\mathbf{M}$. Nous supposons que la matrice $\mathbf{M}$ est irréductible, ce qui revient à dire que le graphe G est fortement connexe.

Définition 4.3 (Distance) *La* distance *d'un nœud x à un nœud y de G correspond au nombre d'arcs associé au plus court chemin de x à y dans G si ce dernier existe. Sinon, elle est égale à $+\infty$. Nous notons par $d^{\mathbf{M}}[x, y]$ la distance de x à y dans G.*

Définition 4.4 (Excentricité) *Pour un nœud donné x, l'*excentricité *sortante notée par $Ecc^{\mathbf{M}}[x]$ est définie comme étant la plus grande distance d'un nœud x à tout autre nœud de $\mathcal{V}$. À savoir :*

$$Ecc^{\mathbf{M}}[x] = \max_{y \in \mathcal{V}}\{d^{\mathbf{M}}[x, y]\}. \tag{4.2}$$

Définition 4.5 (Diamètre) *Le* diamètre *de G, noté $D^{\mathbf{M}}$, est la valeur associée à la plus grande excentricité sortante.*

$$D^{\mathbf{M}} = \max_{x \in \mathcal{V}} \max_{y \in \mathcal{V}}\{d^{\mathbf{M}}[x, y]\} = \max_{x \in \mathcal{V}}\{Ecc^{\mathbf{M}}[x]\}. \tag{4.3}$$

Après avoir défini ces quelques notions nécessaires à notre étude, nous pouvons à présent énoncer le lemme suivant.

Lemme 4.5 *Pour tout $i, j \in \mathcal{F}$: $d^{\mathbf{L}^i}[i, j] = d^{\mathbf{L}}[i, j]$ et $Ecc^{\mathbf{L}^i}[i] = Ecc^{\mathbf{L}}[i]$ (c.-à-d. l'augmentation de la colonne i de $\mathbf{L}$ ne change ni la distance de i à un autre nœud arbitraire j, ni l'excentricité du nœud i). De plus, $\max_i\{Ecc^{\mathbf{L}^i}[i]\} = \max_i\{Ecc^{\mathbf{L}}[i]\} = D^{\mathbf{L}}$.*

PREUVE. L'augmentation de la colonne i dans la matrice $\mathbf{L}$ implique que nous rajoutons des arcs entrants au nœud i dans le graphe orienté. Cependant, ces arcs n'interviennent nullement dans la recherche du plus court chemin entre le nœud i et les autres nœuds du graphe orienté. Donc, la distance calculée à partir de la matrice augmentée $\mathbf{L}^i$ correspond à la distance calculée sur $\mathbf{L}$ et n'est donc pas influencée par les éventuels changements effectués sur la colonne i. Ainsi, l'excentricité de i demeure également inchangée. ■

Propriété 4.3 *Si $k \geq d^{\mathbf{L}^i}[i, j]$, alors $X^{(k),i}[j] \neq 0$. De plus, si $k \geq Ecc^{\mathbf{L}^i}[i]$, alors $X^{(k),i}[j] \neq 0$ pour tout $j \in \mathcal{F}$.*

PREUVE. Pour tout $k \geq 0$, nous considérons l'ensemble des $j \in \mathcal{F}$ tel que $X^{(k),i}[j] \neq 0$. Soit $\mathcal{A}_k^i$ cet ensemble. Nous supposons que l'algorithme est initialisé avec $\mathbf{b} = \nabla_{\mathbf{L}^i}$. Nous rappelons que $\nabla_{\mathbf{L}} = \mathbf{0}$, donc la seule entrée positive de $\nabla_{\mathbf{L}^i}$, est celle associée à l'indice i. Ainsi, $\mathcal{A}_0^i = \{i\}$.

Soit $\mathcal{B}^i$ un ensemble arbitraire de nœuds, $\Gamma^+(\mathcal{B}^i)$ représente l'ensemble des nœuds j tel que $\mathbf{L}[i, j] > 0$. Le calcul de $\Gamma^+(\mathcal{B}^i)$ est proportionnel au nombre de transitions non nulles sortant de i. Dans [FQ12], il est prouvé qu'à l'itération k, les indices des entrées non nulles de $X^{(k),i}$ appartiennent à l'ensemble $\mathcal{A}_k^i$ donné par l'induction : $\mathcal{A}_k^i = \mathcal{A}_{k-1}^i \cup \Gamma^+(\mathcal{A}_{k-1}^i)$. Ainsi, nous pouvons déduire que pour une itération k supérieure ou égal à la distance entre le nœud i et j, le nœud j appartient sûrement à l'ensemble des indices $\mathcal{A}_k^i$. Par conséquent, si $k \geq d^{\mathbf{L}^i}[i, j]$, alors $j \in \mathcal{A}_k^i$.

La deuxième assertion de cette propriété est une généralisation du premier résultat. D'après l'équation 4.2, nous pouvons facilement déduire que si $k \geq Ecc^{\mathbf{L}^i}$, alors $j \in \mathcal{A}_k^i$. ■

Théorème 4.4 *Soit* $\mathbf{L}$ *une matrice sous-stochastique irréductible, l'algorithme 11 fournit à chaque itération* k *une borne inférieure sur la distribution stationnaire* $X^{(k)}$ *et pour tout* $k \geq D^{\mathbf{L}}$*, toutes les composantes de* $X^{(k)}$ *sont positives.*

PREUVE. La première assertion est une conséquence directe du lemme 4.4 et du théorème 4.2. De plus, la propriété 4.3 stipule que pour tout $k \geq Ecc^{\mathbf{L}^i}[i]$, nous avons $X^{(k),i}[j] \neq 0 \quad \forall\, j \in \mathcal{F}$. Suite au lemme 4.5, nous pouvons introduire la simplification suivante $Ecc^{\mathbf{L}^i}[i] = Ecc^{\mathbf{L}}[i]$. Par conséquent, après avoir considéré tous les indices i et pris en considération le maximum des excentricités, nous avons : $\forall\, k \geq \max_i(Ecc^{\mathbf{L}}[i]), \quad X^{(k)} \neq 0, \forall j \in \mathcal{F}$. Nous rappelons que le maximum des excentricités représente le diamètre. ■

Algorithme 11 : Algorithme itératif pour le calcul de la borne inférieure
Iterate on ∇ Lower Bound on a Set : $I\nabla LS$

ENTRÉES : $\forall\, i \in \mathcal{F},\ \boldsymbol{\alpha}[i] > 0$;
SORTIES : Valeurs successives de $X^{(k)}$.

1: $\forall i \in \mathcal{F}, \mathbf{L}^i = \mathbf{L} + \boldsymbol{\alpha}^t \mathbf{e}_i$, $\mathbf{b}^i = \nabla_{\mathbf{L}^i}$, $X^{(0),i} = \mathbf{b}^i$.
2: **Répéter**
3: $\quad \forall$ i$\in \mathcal{F}$, $X^{(k+1),i} = \max\left\{X^{(k),i}, X^{(k),i}\mathbf{L}^i + \mathbf{b}^i(1 - \|X^{(k),i}\|)\right\}$.
4: $\quad X^{(k+1)} = \min_i \left\{X^{(k+1),i}\right\}$.
5: **Jusqu'à** $\sum_i (1 - \|X^{(k),i}\|) < \epsilon$.

De même, une conséquence directe de la propriété 4.3 revient à dire que si $k > \max_i(d^{\mathbf{L}}[i, j])$, alors $X^{(k)}[j] > 0$. De plus, si $k > \min_j \max_i(d^{\mathbf{L}}[i, j])$, alors $X^{(k)} \neq \mathbf{0}$.

Complexité

La complexité de l'algorithme $I\nabla LS$ à chaque itération est pratiquement identique à celle de l'algorithme $I\nabla US$, elle est donnée par $O(m.n.D^{\mathbf{L}})$ qui permet ainsi d'atteindre la première borne inférieure significative sur chaque composante de la distribution stationnaire.

Exemple 4.4 *Nous considérons la matrice* $\mathbf{L}$ *présentée dans l'exemple 4.3. Le vecteur des excentricités pour chaque état de* $\mathcal{F}$ *est* $Ecc^{\mathbf{L}} = [2, 1, 1, 2]$ *donc, le diamètre de la matrice* $\mathbf{L}$ *est* $D^{\mathbf{L}} = 2$*. L'algorithme* $I\nabla LS$ *fournit les bornes inférieures suivantes :*

k	*1*	*2*	*3*	*4*	$1 - \|\|X^{(k)}\|\|$
0	*0.0000*	*0.0000*	*0.0000*	*0.0000*	*1.0000*
2	*0.0272*	*0.0140*	*0.0096*	*0.0384*	*0.9108*
10	*0.1427*	*0.0692*	*0.0609*	*0.1887*	*0.5386*
20	*0.1975*	*0.0951*	*0.0848*	*0.2150*	*0.4077*
40	*0.2232*	*0.1072*	*0.0960*	*0.2181*	*0.3554*
60	*0.2264*	*0.1087*	*0.0974*	*0.2182*	*0.3494*
78	*0.2267*	*0.1089*	*0.0975*	*0.2182*	*0.3487*

4.3.3 Comparaison avec l'approche de Muntz [FM94]

Notre approche présente un avantage considérable. En effet, fondée sur la théorie polyédrale nos algorithmes demeurent beaucoup plus rapide que l'approche de Muntz. Nous rappelons que les algorithmes proposés par Muntz nécessitent le calcul de la distribution stationnaire de n chaînes de Markov. La solution la plus précise est obtenue en utilisant l'algorithme de GTH. En effet, nous n'avons aucun test de convergence prouvé pour un algorithme itératif à part celui défini par Gauss Seidel ou SOR (voir le livre de Stewart pour les détails [Ste95]).

Reposant sur une nouvelle technique de résolution stationnaire, notre algorithme possède plusieurs caractéristiques intéressantes. Tout d'abord, c'est un algorithme itératif qui fournit à chaque itération des bornes supérieures (resp. inférieures) sur le vecteur de distribution stationnaire associé à la matrice $\mathbf{L}^i$. Le coût de chaque itération est de $O(n\,m)$. Pour ce qui est de la borne inférieure, un certain nombre d'itérations, au moins plus grand que le diamètre, est nécessaire pour obtenir une borne inférieure significative.

Nous reportons dans ce tableau, les temps d'exécution en secondes ainsi que la norme des bornes pour l'algorithme de Muntz et notre algorithme itératif $I\nabla LS$. Nous rappelons que la norme (norme 1) d'une borne représente la somme des éléments du vecteur associé. Ces algorithmes sont appliqués respectivement sur des chaînes de Markov générées aléatoirement en variant la taille de leur espace d'état n entre 50 et 500. Le nombre de transitions dans chaque ligne de la matrice est fixé à 4. L'exécution est effectuée sur l'environnement de programmation Matlab et les expériences ont été exécutées sur un processeur 2.8 GHz quad-core Xeon. Les expériences sont répétées 50 fois pour chaque taille de matrice. Pour $n = 100$ par exemple, l'écart type (noté *std*) associé aux les différentes approches sont respectivement données par : *std*(Muntz)=0.0350, *std*(I∇ ,K = 10) = $3.8056\,10^{-4}$ et *std* (I∇ ,K = 20) = $9.7118\,10^{-4}$.

n Taille de la matrice $\mathbf{L}$	Temps d'exécution(s)		Norme 1 de la borne inférieure	
	Muntz	$I\nabla LS$, k = 100	Muntz	$I\nabla LS$, k = 100
50	0.19	0.53	0.76	0.72
100	2.48	1.98	0.73	0.68
200	36.45	8.35	0.70	0.65
500	1.37 10^3	103.88	0.67	0.61

TABLE 4.1 – Temps d'exécution des algorithmes pour une génération aléatoire d'exemples.

Nous remarquons que lorsque la taille de la matrice sous-stochastique $\mathbf{L}$ devient importante, notre algorithme $I\nabla LS$ fournit une solution proche de celle de Muntz en terme de norme, et ceci, en un temps relativement court. Il semble donc clair que notre algorithme est plus rapide que celui de Muntz et permet de fournir des bornes par élément du vecteur de distribution stationnaire à chaque étape de calcul.

Remarque 2 *Nous notons que nos algorithmes peuvent également être utilisés dans d'autres cas, autres ceux mentionnés ici. En effet, en supposant que la matrice stochastique* $\mathbf{P}$ *soit connue, qu'elle soit ergodique et que* $\nabla_{\mathbf{P}} = \mathbf{0}$*, les algorithmes proposés par Bušić et Fourneau* $I\nabla L$ *et* $I\nabla U$ *ne sont pas applicables du fait que les hypothèses ne sont pas remplies. De plus, les autres algorithmes développés pour ce cas et présentés dans [BF10] ne fournissent pas de preuve de convergence vers la distribution stationnaire. Nous pouvons donc suggérer de déterminer ou de construire une matrice* $\mathbf{L} \leq \mathbf{P}$ *tel que* $\nabla_{\mathbf{L}^i} \neq \mathbf{0}$ *pour tout* i*, et de fournir des bornes pour les distributions stationnaires de* $\mathbf{L}$ *et* $\mathbf{P}$ *en utilisant nos algorithmes. Ce cas sera traité plus en détail dans la section 4.5.*

4.4 Calcul de bornes MTTF pour une chaîne de Markov absorbante partiellement connue

Suite aux travaux présentés dans la section précédente sur les distributions stationnaires d'une chaîne de Markov ergodique finie avec probabilités de transition définies dans un intervalle, cette section est consacrée à l'étude d'un nouveau problème. Nous nous sommes intéressés à l'analyse de chaînes de Markov absorbantes avec probabilités de transition partiellement connues. Plus formellement, nous supposons que la chaîne appartient à un ensemble de chaînes de Markov absorbantes, ayant toutes le même ensemble d'états absorbants, $\mathscr{A}$. Nous supposons également que la matrice de transition $\mathbf{M}$ satisfait les conditions $\mathbf{L} \leq \mathbf{M} \leq \mathbf{U}$.

Nous nous sommes intéressés dans ce travail à la définition de bornes sur le temps moyen d'absorption. Dans les problèmes de fiabilité, le temps moyen d'absorption est désigné comme le MTTF (*Mean Time To Failure*, *i.e.,* temps moyen avant panne) pour la chaîne de Markov $\mathbf{M}$. Nous montrons tout d'abord la relation entre le MTTF d'une chaîne absorbante et la DTMC (ergodique) dérivée à partir de cette chaîne absorbante. Nous utilisons par la suite la théorie

polyèdrale pour construire des bornes sur le MTTF. Nous utilisons la nouvelle technique numérique donnée dans [BF11] pour résoudre la distribution stationnaire de chaque matrice considérée dans l'approche polyèdrale et déjà proposée dans [AFP12] (section 4.1) pour l'étude des DTMCs partiellement définies.

Cet algorithme que nous appliquons sur un modèle absorbant et imprécis, fournit à chaque itération des bornes supérieures et inférieures sur le MTTF. Par conséquent, nous obtenons des bornes à la première itération et à chaque itération ces bornes sont améliorées. Notre algorithme est également numériquement stable, car il est seulement fondé sur le produit de vecteurs et de matrices non négatifs. Dans ce qui suit, nous décrivons la procédure permettant d'associer le calcul du MTTF d'une DTMC absorbante avec celui de la distribution stationnaire de la DTMC ergodique.

4.4.1 Chaîne de Markov absorbante

Rappelons qu'une chaîne est dite absorbante si et seulement s'il y a au moins un état absorbant, et de tout état non absorbant nous pouvons atteindre un état absorbant. Soit $\mathbf{P}$ la matrice de transition associée à la chaîne initiale $X = \{X(n), n \geq 0\}$ contenant des états absorbants. L'espace d'état est organisé comme suit : les états absorbants (ensemble $\mathscr{A}$) sont situés avant les états transitoires (ensemble $\mathscr{T}$), forme canonique :

$$\mathbf{M} = \begin{array}{c} \\ \mathscr{A} \\ \mathscr{T} \end{array} \begin{array}{c} \begin{array}{cc} \mathscr{A} & \mathscr{T} \end{array} \\ \left[\begin{array}{c|c} \mathbf{Id} & \mathbf{0} \\ \hline \mathbf{R} & \mathbf{Q} \end{array}\right] \end{array} \tag{4.4}$$

Nous supposons que tous les états de $\mathbf{Q}$ sont transients (*i.e.* pas d'états absorbants et pas de classes récurrentes). La matrice $\mathbf{F} = (\mathbf{Id} - \mathbf{Q})^{-1}$ est appelée matrice fondamentale [Tri02] de la chaîne X. La probabilité d'absorption par un état absorbant donné est la probabilité stationnaire d'atteindre cet état. Nous rappelons cette probabilité par la définition suivante.

Définition 4.6 (Probabilités d'absorption) *Soit* $\mathbf{Z}$ *la matrice définie par* $\mathbf{Z} = \mathbf{F}\,\mathbf{R}$. *L'élément* $\mathbf{Z}[i, j]$ *de la matrice* $\mathbf{Z}$ *correspond à la probabilité d'être absorbé par l'état* $j \in \mathscr{T}$ *sachant que l'état initial est* $i \in \mathscr{A}$.

Définition 4.7 (Temps moyen d'absorption) *Le temps moyen avant l'absorption sachant que l'état initial est* i *est égal à la somme des termes de la* i*-ème ligne de la matrice fondamentale* $\mathbf{F}$*, donné par* $(\mathbf{F}.\mathbf{e}^t)[i]$.

4.4.2 Transformation en un modèle ergodique

Nous considérons une DTMC absorbante avec matrice de transition $\mathbf{M}$. Nous supposons avoir plusieurs états absorbants et aucune classe récurrente. Et nous voulons calculer le temps moyen pour atteindre un état absorbant. Pour ce faire, nous montons comment transformer ce problème initial en construisant une DTMC ergodique et en analysant sa distribution stationnaire. Plus formellement, nous supposons que nous utilisons la décomposition de matrice comme illustrée dans l'équation 4.4.

Nous supposons qu'il n'y a aucune classe récurrente et donc la matrice fondamentale $\mathbf{F} = (\mathbf{Id} - \mathbf{Q})^{-1}$ existe.

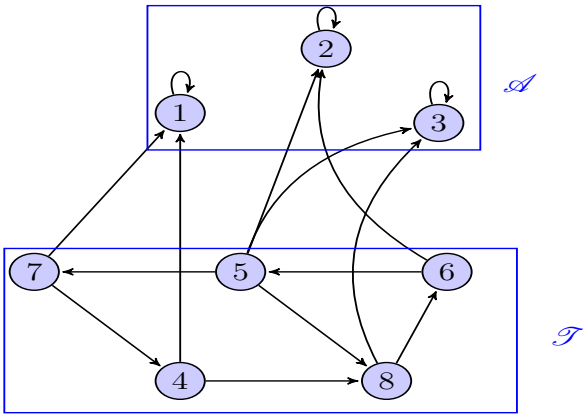


FIGURE 4.1 – Graphe de transition d'une chaîne de Markov absorbante.

Soit i un état arbitraire non absorbant. Nous construisons une nouvelle matrice comme suit :

1. Premièrement, nous agrégeons tous les états absorbants dans un seul et unique état qui sera considéré comme le premier dans l'espace d'état. La matrice $\mathbf{R}$ est sommée et exprimée par le vecteur $\mathbf{r}^t$.
2. Deuxièmement, nous ajoutons une boucle à l'état 1 avec une probabilité de $1/2$.
3. Troisièmement, nous modifions la première ligne : nous ajoutons un vecteur noté p_i dont les entrées sont toutes égales à zéro, excepté l'entrée $i - 1$ qui sera égale à $1/2$. Cette entrée non nulle correspond à la transition $M_i[1, i]$.
4. Enfin, la nouvelle matrice $\mathbf{M}_i$ est donnée comme suit :

$$\mathbf{M}_i = \left[\begin{array}{c|c} 1/2 & p_i \\ \hline r^t & \mathbf{Q} \end{array}\right]. \tag{4.5}$$

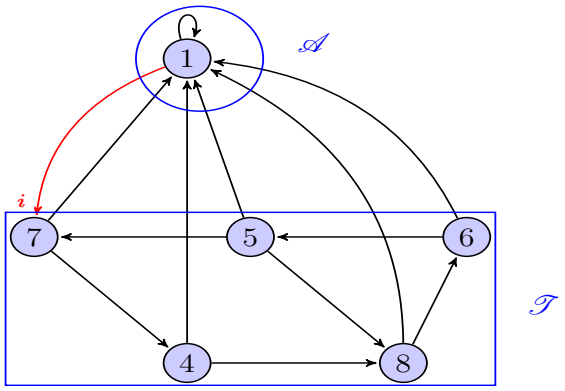


FIGURE 4.2 – Graphe de transition associé à la nouvelle chaîne de Markov permettant d'obtenir le temps avant absorption, sachant que l'état initial est 7.

Dans ce qui suit, l'ensemble $\mathcal{F}$ dénotera l'espace d'état modifié pour définir la matrice M_i. Cet ensemble contient l'ensemble $\mathscr{T}$ ainsi que l'état représentant l'agrégation des états absorbants de M.

Propriété 4.4 *La matrice M_i est ergodique. De plus, la distribution stationnaire de M_i (notée $\boldsymbol{\pi}_i$) existe.*

PREUVE. La matrice M_i est associée à une chaîne de Markov finie. Nous supposons que le graphe de transition associé aux états de $\mathscr{T}$ est fortement connexe. On note qu'il existe un arc de l'état 1 vers l'état i, et que tout les états de $\mathscr{T}$ ont un arc entrant vers l'état 1, car le vecteur $\mathbf{r}$ est non nul par hypothèse. Il en résulte donc que le graphe associé à l'ensemble des états de la chaîne est fortement connexe. Enfin, comme l'état 1 admet une boucle, la chaîne est apériodique. Par conséquent, la chaîne est ergodique et la distribution stationnaire existe. ■

Pour une chaîne de Markov avec matrice de transition M, l'idée consiste à trouver une relation entre le MTTF de la chaîne et le vecteur de distribution stationnaire $\boldsymbol{\pi}_i$ associé à la matrice M_i. Pour ce faire, nous énonçons la proposition suivante.

Proposition 4.1 *Soient M une chaîne absorbante et $E[T_i]$ le temps moyen d'absorption sachant que l'état initial est i. Le MTTF est donné par :*

$$E[T_i] = \frac{2}{\boldsymbol{\pi}_i[1]} - 2. \tag{4.6}$$

PREUVE. Pour une DTMC ergodique M_i, nous savons que $\frac{1}{\boldsymbol{\pi}_i[1]}$ représente le temps moyen entre deux passages par l'état 1. Le fait de conditionner sur la première transition sortante de l'état 1, nous permet d'avoir deux transitions possibles : une boucle sur l'état 1 avec une probabilité de $1/2$, tel que le temps entre deux visites est de 1 et une transition sur l'état i avec une probabilité $1/2$, et un temps entre deux visites qui est de $(1 + E[T_i])$. Par conséquent :

$$\frac{1}{\boldsymbol{\pi}_i[1]} = \frac{(1 + E[T_i])}{2} + \frac{1}{2}.$$

Finalement, nous obtenons $E[T_i]$:

$$E[T_i] = \frac{2}{\boldsymbol{\pi}_i[1]} - 2.$$

■

À présent, nous devons trouver des bornes sur le vecteur $\boldsymbol{\pi}_i$ lorsque la chaîne est partiellement connue c'est-à-dire, lorsque la chaîne est spécifiée par les matrices $\mathbf{L}$ et $\mathbf{U}$. Pour ce faire, nous considérons l'algorithme itératif développé dans la section précédente et présenté dans [AFP12] pour les chaînes imprécises. Les bornes sur $\boldsymbol{\pi}_i[1]$ fourniront alors des bornes sur le MTTF en utilisant la proposition 4.1.

4.4.3 Bornes sur la MTTF

Dans la section 4.3, nous avons montré la démarche à suivre pour combiner le théorème de Muntz (théorème 4.2) et les algorithmes de Bušić et Fourneau [BF10]. Nous avons prouvé de nouveaux algorithmes pour déterminer à chaque itération des bornes (supérieures et inférieures) par élément sur la distribution stationnaire d'une chaîne de Markov imprécise. Nous rappelons que l'algorithme borne supérieure $I\nabla US$ consiste à calculer d'abord, pour tout $s \in \mathcal{F}$ une borne supérieure $Y^{(k),s}$ associée à la matrice $\mathbf{L}^s$ avec l'algorithme $I\nabla U$. Puis, d'appliquer le résultat de Muntz pour déduire une borne supérieure sur la distribution stationnaire de $\boldsymbol{\pi}$. Ce procédé est réitéré jusqu'à ce que le critère d'arrêt soit atteint.

Les algorithmes $I\nabla LS$ et $I\nabla US$ (algorithmes 8 et 9) peuvent être combinés et exprimés à travers l'algorithme 12. Cet algorithme permet de regrouper le calcul des deux bornes inférieure et supérieure de la chaîne de Markov imprécise.

Tout d'abord, nous considérons un ensemble de matrices stochastiques $\mathscr{P} = \{\mathbf{L}^s \,|\, \mathbf{L}^s$ est une matrice stochastique irréductible et $\mathbf{L}^s \geq \mathbf{L}\}$. L'indice s correspond à un indice colonne. La matrice $\mathbf{L}^s$ est construite à partir de la matrice $\mathbf{L}$ en augmentant suffisamment les éléments de la colonne s afin de rendre la matrice stochastique. Nous ajoutons d'abord l'hypothèse suivante : **dans ce qui suit, nous supposons que pour tout** k**,** $\mathbf{r}[k] > 0$**.** Par conséquent, la condition sur $\nabla(\mathbf{L}^s)$ est toujours vérifiée ($\nabla(\mathbf{L}^s) > 0$), rendant ainsi l'application de nos algorithmes possible.
Pour chaque matrice $\mathbf{M} \in \mathscr{P}$, nous définissons la matrice de transition $\mathbf{M}_{i,s}$ conformément à l'équation 4.5. En utilisant l'algorithme 12, le vecteur de distribution stationnaire à l'itération n est borné par :

$$\min_s\{X^{(n),s}\} \leq \boldsymbol{\pi}_i \leq \max_s\{Y^{(n),s}\}.$$

Algorithme 12 : Algorithme de bornes Itératif ∇ pour une chaîne de Markov imprécise

ENTRÉES : $\forall\, s \in \mathcal{F},\ \boldsymbol{\alpha}[s] > 0$.
SORTIES : Valeurs successives de $Y^{(k)}$ et $X^{(k)}$.

1: $\forall s \in \mathcal{F}, \mathbf{L}^s = \mathbf{L} + \boldsymbol{\alpha}^t e_s$, $\mathbf{c}^s = \triangle_{\mathbf{L}^s}$, $\mathbf{b}^s = \nabla_{\mathbf{L}^s}$, $Y^{(0),s} = \mathbf{c}^s$, $X^{(0),s} = \mathbf{b}^s$.
2: **Répéter**
3: $\quad \forall\ s \in \mathcal{F}, Y^{(k+1),s} = \min\{Y^{(k),s}, Y^{(k),s}\mathbf{L}^s + \mathbf{b}^s(1 - \|Y^{(k),s}\|)\}$.
4: $\quad Y^{(k+1)} = \max_s\{Y^{(k+1),s}\}$.
5: $\quad \forall\ s \in \mathcal{F}, X^{(k+1),s} = \max\{X^{(k),s}, X^{(k),s}\mathbf{L}^s + b^s(1 - \|X^{(k),s}\|)\}$.
6: $\quad X^{(k+1)} = \min_s\{X^{(k+1),s}\}$.
7: **Jusqu'à** $\sum_s(\|Y^{(k),s}\| - 1) < \epsilon$ et $\sum_s(1 - \|X^{(k),s}\|) < \epsilon$.

Théorème 4.5 *Soit* $\mathbf{L}$ *une matrice sous-stochastique irréductible. L'algorithme 12 fournit à chaque itération* k *une borne supérieure et inférieure par élément sur la distribution stationnaire de toute matrice ergodique dont les entrées sont supérieures par élément à* $\mathbf{L}$*.*

La combinaison de tous ces résultats nous permet d'établir des bornes sur le MTTF. Nous considérons une matrice absorbante non négative $\mathbf{M}$ définie par $\mathbf{L} \leq \mathbf{M} \leq \mathbf{U}$, tel que la somme de chaque ligne est inférieure ou égale à 1. Les bornes sur le temps moyen $E[T_i]$ sont :

$$E[\underline{T_i}] = \frac{2}{\max_s\{Y^{(n),s}\}[1]} - 2 \leq E[T_i] \leq \frac{2}{\min_s\{X^{(n),s}\}[1]} - 2 = E[\overline{T_i}].$$

Théorème 4.6 *Soit* $\mathbf{L}$ *une matrice sous-stochastique, l'algorithme 12 fournit à chaque itération* k *une borne inférieure et une borne supérieure du MTTF pour toute matrice absorbante dont les entrées sont supérieures par élément à* $\mathbf{L}$ *et définie sur le même ensemble d'états transitoires* $\mathscr{T}$.

Nous présentons ci-après, un petit exemple pour illustrer ce résultat.

Exemple 4.5 *Soit la matrice sous-stochastique absorbante suivante :*

$$\mathbf{L} = \left(\begin{array}{cc|cccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ \hline 0.30 & 0.1 & 0.2 & 0.1 & 0.1 & 0.2 \\ 0.20 & 0.3 & 0.2 & 0.1 & 0 & 0.1 \\ 0.25 & 0.2 & 0.1 & 0.2 & 0.2 & 0 \\ 0.10 & 0 & 0.2 & 0.1 & 0.2 & 0.1 \end{array}\right).$$

Nous voulons calculer le temps moyen avant absorption sachant qu'on commence de l'état $i = 5$. *Tout d'abord, nous agrégeons les états* 1 *et* 2, *qui représentent les états absorbants de la matrice* $\mathbf{L}$. *Nous modifions la transition entre le premier état et l'état* $i = 5$, *nous notons que l'état* 5 *est maintenant le quatrième état en raison de l'agrégation de l'état* 1 *et* 2.

La matrice résultante est :
$$\left(\begin{array}{c|cccc} \boxed{0.5} & 0 & 0 & \boxed{0.5} & 0 \\ \hline 0.4 & 0.2 & 0.1 & 0.1 & 0.2 \\ 0.5 & 0.2 & 0.1 & 0 & 0.1 \\ 0.45 & 0.1 & 0.2 & 0.2 & 0 \\ 0.1 & 0.2 & 0.1 & 0.2 & 0.1 \end{array}\right).$$

Notons que $\mathbf{r}[k] > 0, \forall\ k \in \mathcal{F}$. *Ainsi, nous pouvons utiliser les algorithmes* ∇ *vue que toutes les entrées de la première colonne de la matrice sont positives. Nous dérivons ensuite les cinq matrices appartenant à l'ensemble* $\mathscr{P}$ *:*

$$M_{5,1} = \left(\begin{array}{c|cccc} 0.5 & 0 & 0 & 0.5 & 0 \\ \hline 0.4 & 0.2 & 0.1 & 0.1 & 0.2 \\ 0.6 & 0.2 & 0.1 & 0 & 0.1 \\ 0.5 & 0.1 & 0.2 & 0.2 & 0 \\ 0.4 & 0.2 & 0.1 & 0.2 & 0.1 \end{array}\right), M_{5,2} = \left(\begin{array}{c|cccc} 0.5 & 0 & 0 & 0.5 & 0 \\ \hline 0.4 & 0.2 & 0.1 & 0.1 & 0.2 \\ 0.5 & 0.3 & 0.1 & 0 & 0.1 \\ 0.45 & 0.15 & 0.2 & 0.2 & 0 \\ 0.1 & 0.5 & 0.1 & 0.2 & 0.1 \end{array}\right),$$

$$M_{5,3} = \left(\begin{array}{c|cccc} 0.5 & 0 & 0 & 0.5 & 0 \\ \hline 0.4 & 0.2 & 0.1 & 0.1 & 0.2 \\ 0.5 & 0.2 & 0.2 & 0 & 0.1 \\ 0.45 & 0.1 & 0.25 & 0.2 & 0 \\ 0.1 & 0.2 & 0.4 & 0.2 & 0.1 \end{array}\right), M_{5,4} = \left(\begin{array}{c|cccc} 0.5 & 0 & 0 & 0.5 & 0 \\ \hline 0.4 & 0.2 & 0.1 & 0.1 & 0.2 \\ 0.5 & 0.2 & 0.1 & 0.1 & 0.1 \\ 0.45 & 0.1 & 0.2 & 0.25 & 0 \\ 0.1 & 0.2 & 0.1 & 0.5 & 0.1 \end{array}\right) \text{ et}$$

$$M_{5,5} = \left(\begin{array}{c|cccc} 0.5 & 0 & 0 & 0.5 & 0 \\ \hline 0.4 & 0.2 & 0.1 & 0.1 & 0.2 \\ 0.5 & 0.2 & 0.1 & 0 & 0.2 \\ 0.45 & 0.1 & 0.2 & 0.2 & 0.05 \\ 0.1 & 0.2 & 0.1 & 0.2 & 0.4 \end{array}\right).$$

Le fait de combiner les résultats de Muntz et les algorithmes ∇ permet de fournir des bornes à chaque itération. À l'itération $n = D^{\mathbf{L}} = 3$, nous obtenons la première borne inférieure significative pour chaque composante de la distribution stationnaire. D'après le tableau ci-dessous, le temps moyen d'absorption sachant que l'état initial est l'état 5 est borné par :

$$2.0072 \leq E[T_5] \leq 2.5043.$$

n	$E[\underline{T_5}]$	$E[\overline{T_5}]$	$\|\max_s\{Y^{(n),s}\}\|-1$	$1-\|\min_s\{X^{(n),s}\}\|$
3	*1.8469*	*7.8571*	*0.4733*	*0.6623*
4	*1.9104*	*6.7711*	*0.3723*	*0.6009*
13	*2.0062*	*3.5435*	*0.1934*	*0.2671*
34	*2.0072*	*2.5982*	*0.1642*	*0.0910*
64	*2.0072*	*2.5078*	*0.1613*	*0.0745*
84	*2.0072*	*2.5043*	*0.1612*	*0.0739*

4.5 Calcul de bornes de la distribution stationnaire pour une chaîne de Markov avec $\nabla_{\mathbf{P}} = 0$

Évoluant toujours dans le cadre de la détermination de bornes de la distribution stationnaire par un algorithme itératif. Nous nous sommes intéressés ici au cas particulier d'une DTMC avec matrice de probabilité stochastique $\mathbf{P}$ où $\nabla_{\mathbf{P}} = \mathbf{0}$. Ce cas a été étudié par Bušić et Fourneau dans [BF10]. Les deux auteurs ont proposé différents algorithmes itératifs pour le calcul de bornes de la distribution stationnaire associée à la matrice $\mathbf{P}$. Les algorithmes proposés ne présentent cependant pas de test de convergence prouvé. En effet, les algorithmes développés convergent vers un multiple inconnu du vecteur $\boldsymbol{\pi}$. Nous proposons donc dans cette section d'arborer une démarche un peu différente. En effet, nous avons apporté quelques modifications voir améliorations aux algorithmes de Bušić et Fourneau (algorithmes SLB et SUB). Nous avons par ailleurs, proposé quelques résultats théoriques et quelques algorithmes pour calculer une borne inférieure et supérieure de la distribution stationnaire dont la convergence vers $\boldsymbol{\pi}$ est prouvée.

4.5.1 Algorithmes de Bušić et Fourneau

Soient $\mathbf{P}$ une matrice de transition stochastique irréductible et apériodique et $\boldsymbol{\pi}$ le vecteur de distribution stationnaire associé. Dans le cas où $\nabla_{\mathbf{P}} = \mathbf{0}$, Bušić et Fourneau [BF10] ont proposé plusieurs heuristiques pour le calcul itératif de bornes inférieures et supérieures.

L'algorithme 13 considère que l'on a déterminé au préalable un vecteur non trivial $\mathbf{d}$ ($\neq \mathbf{0}$) tel que $\mathbf{0} \leq \mathbf{d} \leq \boldsymbol{\pi}$ (voir la section 3.3 dans [BF10] pour le choix de la borne inférieure initiale $\mathbf{d}$).

Algorithme 13 : Algorithme Simple pour le calcul de la Borne Inférieure : SLB
Simple Lower Bound Algorithm

ENTRÉES : $\mathbf{0} \leq \mathbf{d} \leq \boldsymbol{\pi}$; $\mathbf{d} \neq \mathbf{0}$.
SORTIES : Valeurs successives de $\mathbf{x}^{(k)}$.

$\mathbf{x}^{(0)} = \mathbf{d}$.
Répéter
$\mathbf{x}^{(k+1)} = \max\{\mathbf{x}^{(k)}, \mathbf{x}^{(k)}\mathbf{P}\}$.
Jusqu'à $\|\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}\| < \epsilon$.

Par une démarche similaire, le calcul de la borne supérieure est donné par l'algorithme 14.

Algorithme 14 : Algorithme Simple pour le calcul de la Borne Inférieure : SUB
Simple Upper Bound Algorithm

ENTRÉES : $\mathbf{c} \geq \boldsymbol{\pi}$.
SORTIES : Valeurs successives de $\mathbf{y}^{(k)}$.

$\mathbf{y}^{(0)} = \mathbf{c}$.
Répéter
$\mathbf{y}^{(k+1)} = \min(\mathbf{y}^{(k)}, \mathbf{y}^{(k)}\mathbf{P})$.
Jusqu'à $\|\mathbf{y}^{(k)} - \mathbf{y}^{(k+1)}\| < \epsilon$.

Remarque 3 *Les algorithmes SLB et SUB ne représentent que des approximations, car le critère d'arrêt fondé sur le calcul de la distance entre deux itérations successives n'implique pas la convergence.*

Avant de passer à la présentation de nos algorithmes, nous allons tout d'abord mener une petite étude comparative. En effet, nous voulons savoir si les algorithmes développés dans la section 4.3 (algorithmes I∇ LS et I∇ US) pour les chaînes de Markov partiellement connues peuvent être exploitées dans le cas d'une matrice de transition $\mathbf{P}$ avec $\|\nabla_{\mathbf{P}}\| = 0$. Et surtout, connaître si les bornes définies par ces algorithmes fournissent de meilleures bornes que celles de Bušić et Fourneau (algorithmes SLB et SUB).

4.5.2 Comparaison des algorithmes : I∇ LS vs. SLB (resp. I∇ US vs. SUB)

Faisant suite à la remarque 2, nous proposons de comparer les algorithmes SLB (resp. SUB) avec nos algorithmes $I\nabla LS$ et $I\nabla US$ pour calculer respectivement la séquence bornes inférieures et supérieures sur le vecteur de distribution stationnaire.

Afin d'appliquer notre approche, nous proposons de suivre la démarche suivante : pour une matrice stochastique $\mathbf{P}$ avec $\nabla_{\mathbf{P}} = \mathbf{0}$, nous définissons une matrice $\mathbf{L} \leq \mathbf{P}$ tel que $\nabla_{\mathbf{L}^i} \neq \mathbf{0}, \ \forall i$.

Et nous appliquons respectivement les algorithmes $I\nabla LS$ et $I\nabla US$ (algorithmes 11 et 10) pour calculer les bornes inférieures est supérieures de la distribution stationnaire. Nous proposons d'effectuer cette comparaison à travers l'exemple suivant.

Exemple 4.6 *Nous considérons la matrice stochastique* $\mathbf{P}$ *:*

$$\mathbf{P} = \begin{pmatrix} 0.2 & 0.3 & 0.2 & 0.3 \\ 0.5 & 0.1 & 0.0 & 0.4 \\ 0.0 & 0.6 & 0.4 & 0.0 \\ 0.5 & 0.0 & 0.2 & 0.3 \end{pmatrix}.$$

Nous remarquons que $\nabla_{\mathbf{P}} = [0.0, 0.0, 0.0, 0.0]$ *et* $\Delta_{\mathbf{P}} = [0.5, 0.6, 0.4, 0.4]$*. La borne inférieure significative pour* $\boldsymbol{\pi}_{\mathbf{P}}$ *est égale à* $\mathbf{d} = [0.0, 0.0, 0.15, 0.0]$ *(voir la section 3.3 de [BF10] pour le choix de la borne inférieure initiale). Les algorithmes SLB et SUB (initialisé avec* $\mathbf{d}$ *et* $\mathbf{c} = \Delta_{\mathbf{P}}$*) fournissent les bornes inférieures (premier tableau) et bornes supérieures (second tableau) suivantes :*

k	1	2	3	4	$1 - \|\mathbf{x}^{(k)}\|$
1	0.000000	0.090000	0.150000	0.000000	0.760000
5	0.126675	0.133614	0.150000	0.107145	0.482566
10	0.195324	0.161590	0.150000	0.166461	0.326625
20	0.234766	0.177670	0.150000	0.200543	0.237021
40	0.242429	0.180793	0.150000	0.207164	0.219614
60	0.242641	0.180880	0.150000	0.207348	0.219131
80	0.242647	0.180882	0.150000	0.207353	0.219118

k	1	2	3	4	$1 - \|\mathbf{y}^{(k)}\|$
1	0.500000	0.450000	0.340000	0.400000	-0.690000
5	0.483750	0.365535	0.298914	0.400000	-0.548199
10	0.471310	0.352347	0.291323	0.400000	-0.514980
20	0.468221	0.349080	0.289445	0.400000	-0.506746
40	0.468085	0.348936	0.289362	0.400000	-0.506384
60	0.468085	0.348936	0.289362	0.400000	-0.506383
80	0.468085	0.348936	0.289362	0.400000	-0.506383

Pour l'application de nos algorithmes, nous considérons la matrice $\mathbf{L} = \mathbf{P} - \rho\mathbf{Id}$*,* $\mathbf{L} \leq \mathbf{P}$ *tel que* $\|\nabla_{\mathbf{L}}\| = 0$*. Pour* $\rho = 0.05$*, la matrice* $\mathbf{L}$ *est donnée par :*

$$\mathbf{L} = \begin{pmatrix} 0.15 & 0.3 & 0.2 & 0.3 \\ 0.5 & 0.05 & 0.0 & 0.4 \\ 0.0 & 0.6 & 0.35 & 0.0 \\ 0.5 & 0.0 & 0.2 & 0.25 \end{pmatrix}.$$

Le diamètre de la matrice est $\mathbf{D^L} = 2$. *Les bornes inférieures et supérieures obtenues via les algorithmes* $I\nabla LS$ *et*$I\nabla US$ *sont présentées respectivement dans le premier tableau et le second tableau suivants :*

k	1	2	3	4	$1 - \|\|\mathbf{x}^{(k)}\|\|$
2	0.0150	0.0135	0.0090	0.0120	0.9505
5	0.0539	0.0420	0.0331	0.0454	0.82562
10	0.1055	0.0806	0.0650	0.0895	0.65932
20	0.1765	0.1334	0.1089	0.1501	0.43108
40	0.2444	0.1841	0.1508	0.2081	0.21261
60	0.2687	0.2022	0.1659	0.2289	0.13428
80	0.2774	0.2087	0.1713	0.2364	0.10621

k	1	2	3	4	$1 - \|\|\mathbf{y}^{(k)}\|\|$
2	0.5350	0.4036	0.3595	0.4500	-0.74812
5	0.4993	0.3728	0.3424	0.4367	-0.65116
10	0.4627	0.3476	0.3187	0.4058	-0.5349
20	0.4134	0.3122	0.2862	0.3636	-0.37538
40	0.3661	0.2782	0.2552	0.3232	-0.22269
60	0.3491	0.2661	0.2440	0.3087	-0.16795
80	0.3431	0.2617	0.2400	0.3035	-0.14833

Il est facile de constater que nos algorithmes fournissent de meilleures bornes sur le vecteur de distribution stationnaire $\boldsymbol{\pi}$.

4.5.3 Nouveaux algorithmes de calcul de bornes de la distribution stationnaire

Fondée sur les algorithmes développés par Bušić et Fourneau (algorithmes SLB et SUB), notre démarche consiste à apporter quelques modifications à ces algorithmes en intégrant la notion de diamètre. En effet, les propriétés du diamètre associés à la chaîne de Markov étudiée offrent des conditions simples de convergence.

Nous commençons par énoncer un résultat très important pour la suite de notre étude. Ce résultat a été étudié par Shen dans [She95] sous forme de conjecture puis dans [She96] et par Schneider dans [Sch02]. Le lecteur peut donc se référer à ces articles pour voir le détail des preuves et la démarche suivie.

Lemme 4.6 *Soit* $\mathbf{P}$ *une matrice de probabilité de transition. Si* $\mathbf{P}$ *est irréductible et apériodique alors,*

$$\exists \text{ un entier positif } m \mid \mathbf{P}^m > 0 \text{ et } \nabla_{\mathbf{P}^m} > \mathbf{0}.$$

Lemme 4.7 *Si* D *est le diamètre associé à la matrice* $\mathbf{P}$ *alors,* $\left(\frac{\mathbf{P}}{2} + \frac{\mathbf{Id}}{2}\right)^D > 0$.

PREUVE. Étant donné une matrice de transition stochastique irréductible et apériodique $\mathbf{P}$ et G le graphe de transition associé. Le graphe de transition associé à la matrice $(\frac{\mathbf{P}}{2} + \frac{\mathbf{Id}}{2})$ correspond au graphe G auquel des boucles sont ajoutées au niveau de chaque nœud. On note que ce changement n'affecte nullement l'excentricité sortante de chaque nœud du graphe (voir le calcul de l'excentricité définition 4.4). De même pour la valeur du diamètre associé au graphe qui reste inchangée. Ainsi, le diamètre du graphe associé à la matrice $\mathbf{P}$ est identique au diamètre de la matrice $\left(\frac{\mathbf{P}}{2} + \frac{\mathbf{Id}}{2}\right)$. Par conséquent, d'après le lemme 4.6, nous déduisons que $\left(\frac{\mathbf{P}}{2} + \frac{\mathbf{Id}}{2}\right)^D > 0$. ■

Pour une matrice de transition stochastique $\mathbf{P}$ d'une chaîne de Markov à temps discret finie irréductible et apériodique avec vecteur de probabilité stationnaire $\boldsymbol{\pi}$, nous considérons une nouvelle matrice $\mathbf{R}$ telle que $\mathbf{R} = \left(\frac{\mathbf{P}+\mathbf{Id}}{2}\right)$. Cette transformation linéaire sur la matrice $\mathbf{P}$ n'entraîne pas de modification sur le vecteur de distribution stationnaire (*i.e.,* $\boldsymbol{\pi}_{\mathbf{P}} \equiv \boldsymbol{\pi}_{\mathbf{R}}$). Ainsi, en intégrant la notion de diamètre de la matrice aux résultats présentés dans [BF11], nous avons développé plusieurs algorithmes, qui sont des améliorations des algorithmes précédents et qui permettent de définir des bornes par élément sur le vecteur de distribution stationnaire $\boldsymbol{\pi}$.

Algorithmes itératifs améliorés pour le calcul de bornes

Les calculs itératifs présentés ci-après ont toujours la distribution $\boldsymbol{\pi}$ comme point fixe, et ces algorithmes convergent bien vers $\boldsymbol{\pi}$. Le premier algorithme développé est retranscrit ci-après.

Algorithme 15 : $A_1\nabla$L : Amélioration de la borne inférieure

1: Calculer la matrice $\mathbf{R}$;

2: Appliquer l'algorithme I∇L sur la matrice $\mathbf{R}$ avec $\mathbf{0} \leq \mathbf{b} \leq \nabla_{\mathbf{R}^D}$.

Théorème 4.7 *Soient* $\mathbf{P}$ *une matrice stochastique irréductible et apériodique avec vecteur de distribution stationnaire* $\boldsymbol{\pi}$ *et de diamètre* D*. Si* $\nabla_{\mathbf{P}} = \mathbf{0}$*, nous considérons la matrice* $\mathbf{R} = \left(\frac{\mathbf{P}}{2} + \frac{\mathbf{Id}}{2}\right)$ *tel que l'algorithme* $A_1\nabla L$ *fournit à chaque itération des bornes inférieures pour toutes les composantes de* $\boldsymbol{\pi}$ *et converge vers* $\boldsymbol{\pi}$ *pour toute valeur du paramètre* $\mathbf{b}$ *tel que* $\mathbf{b} \leq \nabla_{\mathbf{R}^D}$ *et* $\mathbf{b} \neq 0$*.*

PREUVE. Conséquence de l'algorithme I∇L. ■

Nous présentons un petit exemple pour illustrer la convergence de cet algorithme.

Exemple 4.7 *Nous considérons la matrice stochastique définie dans l'exemple 4.6 avec* $\nabla_{\mathbf{P}} = \mathbf{0}$*. Le diamètre de la matrice est* $D = 2$*. Pour une précision égale à* $\epsilon = 10^{-5}$*, l'algorithme* $A_1\nabla L$ *permet d'obtenir la séquence bornes inférieures suivante :*

k	*1*	*2*	*3*	*4*	$1 - \|\|X^{(k)}\|\|$
1	*0.07500*	*0.06750*	*0.04500*	*0.06000*	*0.75250*
2	*0.13331*	*0.11267*	*0.07886*	*0.10890*	*0.56626*
5	*0.23661*	*0.18221*	*0.14083*	*0.19905*	*0.24129*
10	*0.29324*	*0.21933*	*0.17926*	*0.24987*	*0.05822*
15	*0.30655*	*0.22863*	*0.18898*	*0.26177*	*0.01405*
20	*0.30972*	*0.23091*	*0.19134*	*0.26463*	*0.003389*
30	*0.31067*	*0.23159*	*0.19205*	*0.26548*	*0.0001973*
40	*0.31073*	*0.23164*	*0.19208*	*0.26553*	$1.148\ 10^{-5}$

Remarque 4 *Le théorème 4.7 fournit un critère de convergence prouvé pour les algorithmes itératifs. Puisque, il repose essentiellement sur l'algorithme $I\nabla L$ dont la convergence a été prouvée.*

Nous
montrons ensuite un autre résultat qui repose également sur la construction de séquences croissantes de la distribution de probabilité et qui convergent vers la distribution stationnaire $\boldsymbol{\pi}$.

Algorithme 16 : $A_2\nabla$ L : Amélioration de la borne inférieure

ENTRÉES : $\mathrm{b} \leq \nabla_{\mathbf{R}^D}$ et $\mathrm{b} \neq \mathbf{0}$.
SORTIES : Valeurs successives de $X^{(k,1)}$.

1: $X^{(0,1)} = \mathbf{0}$;
2: **Répéter**
3: **Pour** $j = 2$ **à** D **faire**
4: $X^{(k,j)} = X^{(k,j-1)}\left(\frac{\mathbf{P}+\mathbf{Id}}{2}\right)$;
5: **Fin pour**
6: $X^{(k+1,1)} = \max\left\{X^{(k,1)}, X^{(k,D)}\mathbf{R} + \mathbf{b}\left(1 - \|X^{(k,1)}\|\right)\right\}$.
7: **Jusqu'à** $1 - \|X^{(k,1)}\| < \epsilon$.

Une légère modification de l'algorithme $A_2\nabla$ L au niveau de l'instruction 3 nous permet de dériver l'algorithme $A_3\nabla$ L présenté ci-après.

Algorithme 17 : $A_3\nabla$ L : Amélioration de la borne inférieure

ENTRÉES : $\mathrm{b} \leq \nabla_{\mathbf{R}^D}$ et $\mathrm{b} \neq \mathbf{0}$.
SORTIES : Valeurs successives de $X^{(k)}$.

1: $X^{(0,1)} = \mathbf{0}$;
2: **Répéter**
3: **Pour** $j = 2$ **à** D **faire**
4: $X^{(k,j)} = \max\left(X^{(k,j-1)}, X^{(k,j-1)}\left(\frac{\mathbf{P}+\mathbf{Id}}{2}\right)\right)$;
5: **Fin pour**
6: $X^{(k+1,1)} = \max\left\{X^{(k,1)}, X^{(k,D)}\mathbf{R} + \mathbf{b}\left(1 - \|X^{(k,1)}\|\right)\right\}$.
7: **Jusqu'à** $1 - \|X^{(k,1)}\| < \epsilon$.

Théorème 4.8 *Pour* $\mathbf{b}$ *tel que* $\mathbf{b} \leq \nabla_{\mathbf{R}^D}$ *et* $\mathbf{b} \neq \mathbf{0}$*, l'algorithme* $A_2\nabla$ *L (resp.* $A_3\nabla$ *L) converge vers* $\boldsymbol{\pi}$.

Remarque 5 *L'algorithme* $A_3\nabla$ *L converge plus rapidement que les algorithmes* $A_1\nabla$ *L et* $A_2\nabla$ *L.*

Exemple 4.8 *Soit la matrice* $\mathbf{P}$ *utilisée précédemment (exemple 4.7). Le diamètre de la matrice est toujours de* 2*. L'utilisation des algorithmes* $A_2\nabla$ *L et* $A_3\nabla$ *L permet de définir les séquences suivantes pour* $\epsilon = 10^{-5}$ *respectivement :*

k	*1*	*2*	*3*	*4*	$1 - \|\|X^{(k)}\|\|$
1	*0.07500*	*0.0675*	*0.04500*	*0.06000*	*0.75250*
2	*0.13397*	*0.10985*	*0.79425*	*0.11049*	*0.56626*
5	*0.23605*	*0.17946*	*0.14330*	*0.19989*	*0.241286*
10	*0.29272*	*0.21903*	*0.18046*	*0.25003*	*0.05822*
15	*0.30639*	*0.22859*	*0.18925*	*0.26172*	*0.01405*
20	*0.30968*	*0.23090*	*0.19140*	*0.26484*	*0.003389*
30	*0.31067*	*0.23159*	*0.19205*	*0.26548*	*0.0001973*
40	*0.31073*	*0.23163*	*0.19208*	*0.26553*	*1.148 10*$^{-5}$

k	*1*	*2*	*3*	*4*	$1 - \|\|X^{(k)}\|\|$
1	*0.07500*	*0.0675*	*0.04500*	*0.06000*	*0.75250*
2	*0.13537*	*0.11295*	*0.79425*	*0.11162*	*0.56063*
5	*0.24103*	*0.18417*	*0.14452*	*0.20412*	*0.22615*
10	*0.29553*	*0.22117*	*0.18165*	*0.25215*	*0.04950*
15	*0.30742*	*0.22934*	*0.18986*	*0.26266*	*0.010863*
20	*0.31000*	*0.23113*	*0.19158*	*0.26489*	*0.00238*
30	*0.31070*	*0.23161*	*0.19207*	*0.26550*	*0.000115*
38	*0.31073*	*0.23164*	*0.19208*	*0.26553*	*1.013 10*$^{-5}$

Nous constatons que l'algorithme $A_3\nabla$ *L converge en un nombre d'itérations un peu plus petit que les algorithmes* $A_1\nabla$ *L et* $A_2\nabla$ *L. Nous allons essayer de confirmer ce comportement à travers les expérimentations effectuées dans l'exemple ci-après.*

Exemple numérique

Nous allons vérifier l'efficacité des algorithmes sur de grandes matrices stochastiques générées aléatoirement. Les matrices ont les propriétés suivantes. L'ordre de la matrice est de n. Le nombre de transition non nulles par ligne est égal à f, la matrice générée est irréductible (*i.e.,* pour tous les états i, il existe une transition vers $i + 1$ et la transition de l'état n vers l'état 0 est strictement positive). Nous faisons varier n et f et nous présentons dans le tableau suivant le temps de calcul (en secondes) pour atteindre une précision de $\epsilon = 10^{-10}$ pour les algorithmes

$A_1\nabla$L, $A_2\nabla$L et $A_3\nabla$L. Nous comparons également les différents algorithmes avec la méthode GTH en terme de temps de calcul. Les calculs ont été effectués sur un bi-processeur à 2.8 GHz Quad Core Intel Xeon avec 4Gbytes de mémoire. Les matrices aléatoires de différentes tailles et différente densité ont été générées. Il est clair que pour de grandes matrices avec un grand nombre d'éléments non nuls, nos algorithmes permettent de définir des bornes en un temps très rapide.

		$A_1\nabla$ L		$A_2\nabla$ L		$A_3\nabla$ L		GTH
	n	Nbr. iteration	Temps	Nbr. iteration	Temps	Nbr. iteration	Temps	Temps
f = 10	100	144	0.0032	144	0.0825	112	0.0841	0.0482
	500	289	0.0397	289	0.4902	216	0.7123	3.4067
	1000	503	0.3761	503	2.6824	383	3.0782	31.2658
f = 100	500	252	0.0292	252	0.4757	232	0.4835	3.4377
	1000	175	0.1953	175	0.7379	149	0.8519	32.5422

TABLE 4.2 – Nombre d'itération et temps de calcul pour les différentes méthodes de convergence.

Pour chaque taille de matrice, les expériences sont répétées 100 fois. Pour $n = 100$, l'écart type des temps associé aux différentes méthodes sont respectivement donnés par : *std*($A_1\nabla$ L)=0.0207, *std*($A_2\nabla$ L) = 0.0334, *std* ($A_3\nabla$ L) = 0.0107 et *std*(GTH)=0.0067.

4.6 Conclusion

Dans ce chapitre, nous avons présenté dans un premier temps une technique numérique pour dériver des bornes par élément sur la distribution stationnaire de l'ensemble des matrices du polyèdre associé à une chaîne de Markov partiellement spécifiée. En se basant sur la nouvelle méthode numérique développée par Bušić et Fourneau [BF10] et l'approche polyédrale de Muntz [MDG89], nous avons construit de nouveaux algorithmes itératifs. Ces algorithmes permettent de définir à chaque itération une borne inférieure (resp. supérieure) par élément. Par la suite, nous avons considéré le cas des chaînes de Markov absorbantes et imprécises. Nous avons présenté un algorithme pour déterminer certaines bornes inférieures et supérieures du temps moyen d'absorption noté MTTF. Enfin, nous nous sommes intéressés au cas de chaînes de Markov avec matrice de transition $\mathbf{P}$ tel que la quantité $\nabla_{\mathbf{P}} = \mathbf{0}$. Pour ce faire, nous avons proposé de nouveaux algorithmes de calcul de bornes de la distribution stationnaire. Ces algorithmes définissent des bornes par élément dont la convergence vers le vecteur stationnaire $\boldsymbol{\pi}$ est prouvée.

Nous avons donc développé dans ce chapitre, des algorithmes de bornes qui offrent la possibilité d'avoir un compromis entre le nombre d'étapes de calcul et la précision des résultats numériques. De plus, sous certaines conditions techniques, le fait d'appliquer nos bornes sur

une matrice bornante inférieure par élément, nous permet d'obtenir des bornes prouvées sur le vecteur de distribution stationnaire. Ainsi, même si toutes les entrées de la matrice ne sont pas complètement calculées, nous sommes en mesure d'obtenir des garanties sur la distribution stationnaire ou sur la récompense moyenne.

Deuxième partie

Simplification des distributions discrètes par l'approche des bornes stochastiques

Chapitre 5

Spécification des variables aléatoires discrètes et leur réduction

L'évolution permanente des systèmes et réseaux informatiques et des télécommunications laisse entrevoir des systèmes de plus en plus complexes et met en évidence le besoin croissant d'outils facilitant leurs analyses. L'évaluation des performances passe par la définition de modèles représentant d'une manière aussi fidèle que possible l'état du système. Souvent, l'utilisation de modèles mathématiques basés sur des lois de probabilité précises est la solution adoptée pour le dimensionnement du système. Des lois connues comme la loi de Poisson ont été souvent utilisées pour représenter l'arrivée et/ou service des paquets. D'autres lois telles que les *Markov Modulated Poisson Process* (MMPP) ou *Switched Batch Bernoulli Process* (SBBP), plus complexes, permettent de représenter une certaine variabilité du trafic, ce qui permet de modéliser du trafic vidéo par exemple. L'intérêt de ces lois d'arrivées est de pouvoir utiliser des modèles probabilistes tels que les chaînes de Markov afin de calculer des indices de performance (probabilités de perte, temps de réponse...). Malheureusement, ces lois ne sont pas forcément une représentation fidèle du comportement du réseau. De plus, se pose le problème de la paramétrisation. Une MMPP ou une loi de type Phase nécessite un nombre important de paramètres. Certes il existe des algorithmes d'approximation par une loi (*fitting*) qui calculent les paramètres pour une famille de lois à partir d'un ensemble de mesures. Mais il s'agit fondamentalement d'une cause d'erreurs, car on cherche la loi la plus proche des mesures au sens d'un certain nombre d'estimateurs.

Autre approche qui a retenu toute notre attention est l'utilisation de modèles décrits par des *histogrammes*. Afin d'améliorer les résultats des analyses déterministes des systèmes de calcul et de réseau en temps réel, l'utilisation d'une description probabiliste du trafic de données permet de décrire de manière plus précise la variabilité du trafic, comparée à celles définies par des modèles de files d'attente déterministes ou trop simples. Cependant, cette caractérisation pose un problème majeur lié à la taille des systèmes étudiés. En effet, lorsque l'espace d'état du système devient très grand, son analyse exacte devient à son tour très complexe. Dans le domaine des télécoms par exemple, les mesures des réseaux internet deviennent de plus en plus nombreuses et beaucoup de traces sont disponibles pour tester différentes hypothèses sur le trafic. Par contre, ces traces sont trop volumineuses pour servir directement à l'analyse de performance. Elles sont utilisées typiquement pour trouver une

distribution théorique approchée (par exemple, les distributions de type phase PH [Kle76]) ou pour être injectées dans un simulateur. Dans le domaine de la recherche opérationnelle, la résolution numérique de certains problèmes d'optimisation ayant une description par variables aléatoires peut s'avérer également très difficile, voire impossible à réaliser de façon exacte. L'augmentation de la taille de la distribution à chaque étape de calcul entraîne une explosion de l'espace d'état. Par exemple, pour les modèles $\max$-plus, la taille maximale de la distribution après ces opérations correspond au produit des tailles des distributions en entrée.

Ainsi, afin de pallier les problèmes de taille des modèles étudiés, une idée naturelle consisterait à réduire la taille de ces histogrammes et essayer de trouver un compromis entre la précision et la complexité. Nous proposons donc de nous intéresser à cette démarche. Cependant, avant d'exposer notre travail, nous allons d'abord dédier ce chapitre à la présentation de l'approche par histogramme, la mise en évidence des travaux déjà menés dans ce sens et la clarification de certains points nécessaires à la réalisation de notre étude. Nous commençons dans la section 5.1 par présenter le principe d'une approche par fitting. Dans la section 5.2, nous décrivons brièvement les modèles par histogramme et présentons l'essentiel des travaux reposant sur cette approche trouvés dans la littérature. Puis, nous allons nous intéresser aux trafics issus de traces réelles. À cet effet, nous donnons dans la section 5.3 une caractérisation du trafic internet utilisée dans ce travail. Nous mettons l'accent sur l'importance du choix de la période d'échantillonnage pour avoir une bonne approximation et une description fiable du comportement de la trace traitée. Cette phase aura pour but de décrire la trace de trafic en entrée considérée lors des différentes études menées dans les chapitres qui suivent. La section 5.4 mettra en évidence les méthodes de réduction de tailles des supports associées aux histogrammes décrivant le modèle. La méthode d'Hernàndez et *al.* [HOVC10] et l'approche de Tancrez et *al.* [TSC09] sont respectivement présentées.

5.1 Méthode de fitting

Le fitting est une technique qui permet d'obtenir une modélisation réaliste de données brutes (par exemple une courbe, si l'on considère que les données se présentent sous la forme d'un ensemble de points). L'utilisation d'algorithmes dits de fitting, qui s'apparente à de l'approximation, permet d'approcher des distributions probabilistes réelles par des distributions de type phase (PH) par exemple afin de pouvoir ensuite modéliser le comportement du système par une chaîne de Markov (souvent continue). Un modèle dépend d'un certain nombre de paramètres (tailles de messages, vitesses des machines, etc.). Aussi, afin de conserver son réalisme suivant ces différents paramètres, on peut appliquer ces méthodes de fitting sur différents choix de paramètres pour ensuite générer une CMTC paramétrée modélisant le système dans ses différentes configurations possibles.

L'approximation de distributions générales pour la modélisation de nombreux phénomènes réels, qui peuvent être partiellement spécifiées sous forme de mesures de données ou de moments, par des distributions PH a été largement étudiée dans la littérature. Les techniques correspondantes peuvent être regroupées en deux catégories : les méthodes numériques sophistiquées (qui font usage de la programmation non linéaire [JT89], de procédures statistiques comme le maximum de vraisemblance [AA92] ou d'estimation de la distance

minimale [PS80]) et les méthodes analytiques simples (également appelé méthode des moments, par exemple, Whitt [Whi81], Johnson & Taaffe [JT88] ou Bobbio et *al.* [AAM05]) plus souvent utilisées.

Bien que cette approche puisse donner de bons résultats, elle souffre cependant du problème suivant : les modèles obtenus par des algorithmes de fitting sont non exploitables par un utilisateur, ils ne fournissent pas de bornes ou de preuves de précision. Même si la connaissance est bien présente dans le modèle, il est difficile, voire impossible, d'analyser cette connaissance et d'en extraire certaines informations. Il est donc possible d'obtenir des modèles réalistes de systèmes complexes, mais il est difficile de manipuler de tels modèles, de les transformer aisément et de les réutiliser pour d'autres situations. Nous proposons donc de nous intéresser aux modèles qui peuvent être décrits par des variables aléatoires discrètes, présentés ci-après

5.2 Modèles par histogramme

La caractérisation par histogramme de modèles décrivant des systèmes et réseaux informatiques ou autres permet de décrire les tâches d'un graphe de tâches ou le trafic réseau comme une fonction discrète de masse de probabilités statistiques. Plusieurs travaux reposant sur cette approche ont été développés dans la littérature. Dans le domaine des systèmes en temps réel, l'un des premiers travaux faisant référence à des modèles par histogrammes a été présenté par Tia et *al.* [TDS+95]. Cela consiste à modéliser les temps d'exécution à partir de variables statistiques pour calculer les interférences des tâches, en utilisant le concept de convolution statistique. Dans [MEP01], Manolache et *al.* ont analysé les performances d'ensembles de tâches, où le temps d'exécution d'une tâche est spécifié comme une distribution de probabilité continue. Nous citons également les travaux de Díaz et *al.* [DGK+02] qui consistent à faire une analyse complète de l'algorithme RM (Rate Monotonic) avec des temps d'exécution des tâches caractérisées par des variables aléatoires.

Dans le domaine du Network Calculus, les modèles par histogramme ont été utilisés par Skelly et *al.* dans [SSD93] pour prédire les distributions de la longueur des tampons et les taux de pertes lors du multiplexage. Ils ont également été utilisés par Kweon et Shin [KS01] pour proposer une implémentation des canaux statistiques en temps réel dans l'ATM et étudier un ordonnanceur équitable. Ces travaux reposent sur l'analyse d'un système de file d'attente M/D/1/N, où les sources de trafic sont estimées par une distribution de Poisson de taux λ modélisée par une variable aléatoire discrète. Viennent ensuite les méthodes de simplification et de réduction d'histogrammes. Nous précisons que cette démarche nous intéresse tout particulièrement et fait l'objet de notre travail. Parmi les publications existantes dans ce sens, nous citons les travaux d'Hernández et ses co-auteurs [HOVC07a, HOVC07b, HOVC09, HOVC10] qui traitent des processus HBSP (Histogram Based Stochastic Process). L'idée de la méthode HBSP consiste à réduire la taille des histogrammes par une agrégation de taille constante et d'injecter ces histogrammes dans des files FIFO simples pour calculer numériquement la distribution stationnaire de la longueur des files. Une autre approche légèrement différente a été donnée dans [TSC09]. Il s'agit de construire une distribution discrète, borne supérieure d'une distribution continue qui représente la durée d'un service dans une station d'une ligne de production.

Les deux méthodes dernièrement citées vont être présentées dans la section 5.4 et vont représenter un élément de comparaison intéressant avec la méthode de réduction basée sur les bornes stochastiques que nous allons développer dans le chapitre suivant. Nous notons que tout au long de notre étude, nous utilisons indifféremment les termes histogramme et distribution de probabilité discrète, car de nombreux articles publiés ont utilisé cette terminologie [HOVC07a, HOVC10, SSD93].

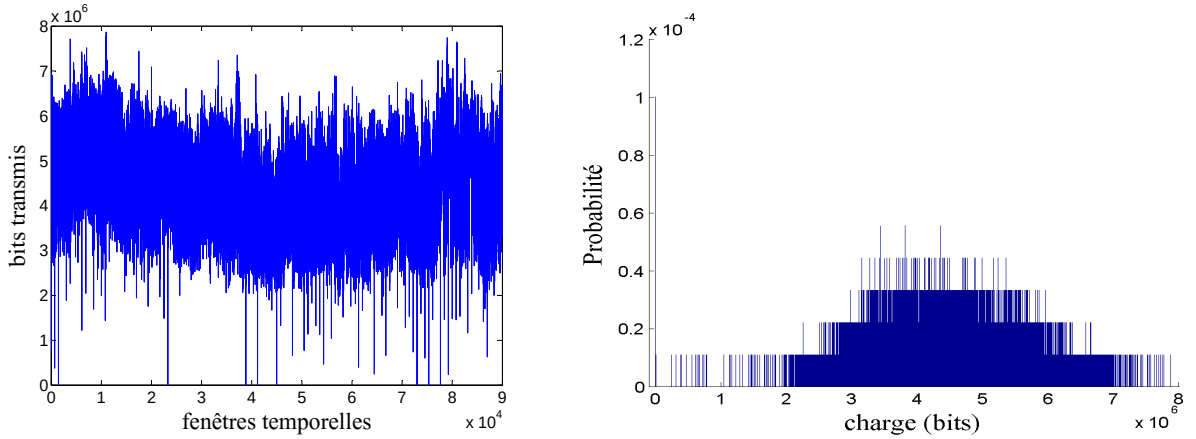
5.3 Représentation par histogramme des traces de trafic réelles

Lorsqu'on essaye de trouver des modèles de trafic d'entrée appropriés, on a généralement un ensemble de points de données (appelé trace) observé depuis un système réel à partir duquel on veut déterminer un modèle ou une description du processus qui s'approche ou ressemble aux caractéristiques de la trace réelle. Une trace est définie comme une séquence de points l chronologiquement ordonnée dans le temps $t_j > 0$ ($j = 1, \ldots, l$). Habituellement, t_j décrit le temps d'arrivée du j-ième événement, mais une trace pourrait également contenir les temps de service, la taille des paquets ou autres données.

Dans ce travail, nous utilisons la distribution empirique comme critère de représentation et comme caractéristique comportementale de la trace de trafic. En effet, l'histogramme empirique garde toute l'information statistique si le nombre d'états considéré est suffisamment grand (proche de l'exact). Nous considérons à titre d'exemple, la trace de trafic MAWI (Measurement and Analysis on the WIDE Internet) [SC00] qui correspond à une heure de mesure de trafic IP à 150 Mb/s sur une ligne transpacifique, réalisée le 9 janvier 2007 entre 12 h et 13 h. La trace est caractérisée par les instants d'arrivés des données (en microsecondes) noté $I(n)$, $n = 1, 2, \ldots$ et le nombre de données arrivées à l'instant $I(n)$ (en bits). Cette trace a un taux moyen de 109Mb/s. En échantillonnant avec une période de 40 ms (ce qui correspond à 25 échantillons par seconde), nous obtenons $90\,000$ échantillons (appelés également slots) de moyenne 4.37 Mb comme illustrée dans la figure 5.1. L'histogramme correspondant à cet échantillonnage est représenté dans la figure 5.1.b.

Il est important de noter que pour l'analyse d'un système d'attente, la stationnarité ou la non-stationnarité du trafic en entrée représente une hypothèse très importante pour la modélisation du système. Le fait de supposer la stationnarité du trafic en entrée induit une perte des corrélations temporelles. Dans cette thèse, nous nous sommes plus intéressés au cas de la stationnarité de la trace associée au trafic d'arrivée, hypothèse utilisée dans les chapitres 7, 8 et 9. L'hypothèse de la non-stationnarité est utilisée dans le chapitre 10, dans le but de montrer que notre contribution est également assurée dans ce cas.

Le trafic d'entrée est donc calculé à partir de l'échantillonnage d'une trace avec une période T fixée. Nous allons noter par $A(k)$ le nombre d'arrivées pendant la période k. Nous supposons que le trafic est stationnaire et indépendant. Donc, toutes les variables aléatoires $A(k)$ suivent la même distribution $\mathcal{A}$.



(a) Trace MAWI (b) Histogramme correspondant au trafic MAWI

FIGURE 5.1 – Trafic MAWI pour une période de 40 ms.

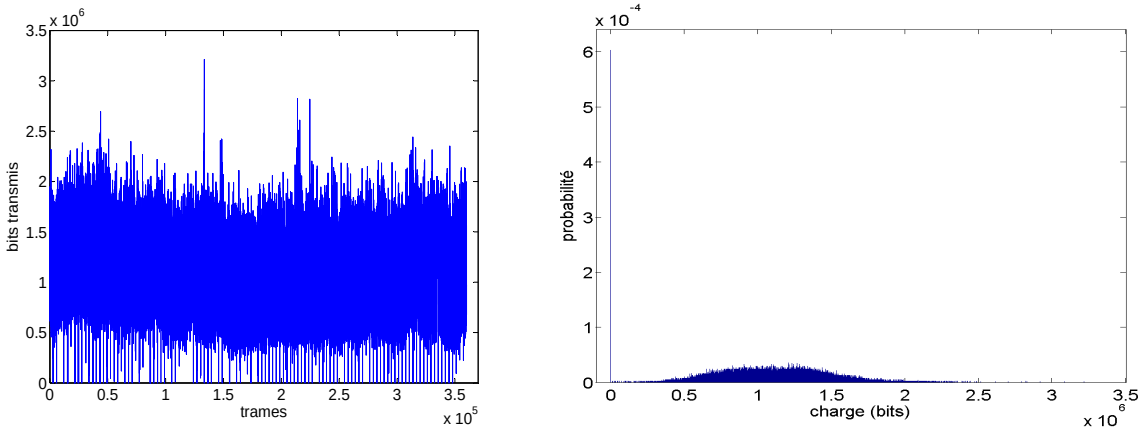
Le problème est bien entendu que la taille de l'histogramme est très grande et que cela impacte négativement les algorithmes d'analyse des files d'attente.

5.3.1 Modélisation du trafic

Les travaux de modélisation du trafic Internet se sont principalement intéressés à la modélisation stochastique des séries temporelles matérialisant les arrivées des données et des flux d'informations échangés sur le réseau mondial. Les motivations de telles études résident dans l'usage qui peut être fait a posteriori de ces modélisations pour assurer le bon fonctionnement du réseau et optimiser son développement (topologie, dimensionnement, etc.).

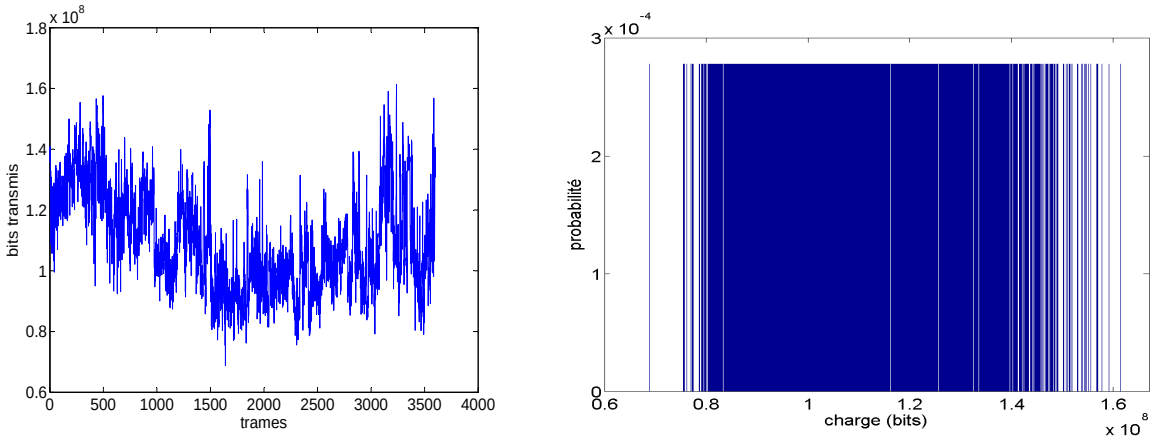
Les charges de travail du réseau seront caractérisées par le nombre d'unités d'émission produites par une source de trafic pendant une période de temps préétablie appelée la période d'échantillonnage ou slot. Le modèle proposé caractérise la charge de travail variable non comme une fonction du temps, mais comme une distribution statistique discrète (voir figure 5.1.b). Le choix de la période d'échantillonnage est une question importante. Dans le cas d'une charge périodique, la période d'échantillonnage correspond généralement à la période « naturelle ». Par exemple, dans la transmission de vidéo, il correspond habituellement à la période de la trame vidéo. Pour une trace de trafic générale, nous proposons d'observer l'impact du choix de la période d'échantillonnage sur l'allure et la description de trace de trafic en entrée. Pour ce faire, nous faisons varier la période d'échantillonnage et nous analysons les traces de trafic réelles résultantes. En considérant la trace de trafic MAWI présenté précédemment, nous étudions les périodes d'échantillonnages suivantes : $T = 10$ ms, $T = 40$ ms et $T = 1$ s. Les traces obtenues ainsi que leurs histogrammes sont représentés dans les figures 5.2, 5.1 et 5.3 respectivement.

Nous pouvons constater que l'utilisation d'une grande période d'échantillonnage ($T = 1$s) donne une description assez grossière du trafic et entraîne une perte d'informations assez importante, masquant ainsi le comportement des données (voir figure 5.3.b). Par contre, une période d'échantillonnage trop petite ($T = 10$ ms) offre une description trop fine et engendre un nombre d'états trop grand (voir figure 5.2.b) rendant ainsi l'analyse du modèle trop lente



(a) Trace MAWI (b) Histogramme correspondant au trafic MAWI

FIGURE 5.2 – Trafic MAWI pour une période de 10 ms.



(a) Trace MAWI (b) Histogramme correspondant au trafic MAWI

FIGURE 5.3 – Trafic MAWI pour une période de 1 s.

et coûteuse. De plus, une telle description n'est pas nécessaire à l'étude de performance du système. Enfin, pour une période d'échantillonnage de $T = 40$ ms (figure 5.1), nous pouvons dire que le comportement du trafic est bien conservé et que l'histogramme associé décrit de manière assez fiable le comportement des données de la trace.

Dans le cadre de notre étude, une première étape pour la modélisation du trafic consiste à approximer la trace réelle en utilisant une période d'échantillonnage appropriée, permettant une description fidèle du comportement du trafic. Pour la trace de trafic MAWI utilisée tout au long de nos expérimentations, nous considérons la période d'échantillonnage $T = 40$ ms. La seconde étape consiste à introduire une unité de données qui rassemble un nombre constant de bits appelé unité de données (notée D) afin de réduire la taille, et par conséquent la complexité des modèles étudiés. En effet, les mesures du trafic réseau sont faites en bits par seconde. Mais nous n'avons pas besoin d'une description aussi détaillée de la quantité de données. Pour nos calculs numériques (calcul de bornes), nous avons pris à titre d'exemple, une unité de données

égale à $1\,Kb$ ($1\,Kb = 10^3\,bits$). De plus, nous supposons que les arrivées et les services se produisent par groupe d'unités de données qui sont stationnaires.

5.4 Méthodes de réduction des tailles des histogrammes

Pour un modèle décrit par une distribution de variables aléatoires, nous distinguons respectivement deux méthodes de réduction de la taille du support initial : le modèle HBSP proposé par d'Hernàndez et *al.* et la méthode de borne stochastique de Tancrez et *al.*. Reposant sur deux approches complètement distinctes, ces méthodes ont pour but commun d'accélérer le temps d'analyse du système tout en conservant la précision.

5.4.1 Modèle HBSP proposé par d'Hernàndez et *al.*

Considérant des traces de trafic réelles, Hernández-Orallo et Vila-Carbó ont proposé dans [HOVC07a, HOVC07b, HOVC09, HOVC10] d'utiliser l'approche par histogramme pour caractériser le trafic réseau et le processus stochastique dans le but d'analyser les performances du modèle. Pour ce faire, le processus HBSP : *Histogram Buffer Stochastic Process* proposé permet de définir un histogramme sur l'occupation du tampon (distribution de la longueur du tampon) en utilisant un modèle de file d'attente finie. D'autres mesures de performance peuvent également être déduites.

Cependant, les traces réelles étant définies sur des espaces d'état très grands, l'analyse s'avère difficile et complexe à réaliser. Les auteurs ont donc suggéré de réduire la taille des distributions pour simplifier les calculs numériques effectués directement sur la distribution initiale. En effet, la méthode proposée ne nécessite ni une approximation du trafic par une distribution de Poisson ni une résolution de modèles de file d'attente. Nous allons détailler ci-après le modèle HBSP en présentant respectivement la méthode de réduction de la taille des distributions et le processus par histogramme proposé.

a) Réduction de la taille de la trace initiale

Partant d'une trace de trafic réelle $\{A'(t)_{t\geq 0}\}$ définie sur l'intervalle $I' = [0, N-1]$, Hernàndez et *al.* ont proposé de réduire la taille de la distribution à $K << N$ états en divisant le support de la distribution empirique issu de l'échantillonnage de la trace en K intervalles (appelés également classes ou bins) de même taille. Plus formellement, sachant que le support de la trace initiale est $I' = [0, N-1]$, la largeur des sous-intervalles sera de $l_{\mathbf{A}} = N/K$. Hernández et ses co-auteurs définissent alors la distribution approchante comme suit : le support de la distribution discrète est constitué des points médians de chaque intervalle et la masse de probabilités de ce point est celle de l'intervalle dans la distribution initiale. Cette description par intervalle permet de définir le processus $\{A(t)_{t\geq 0}\}$ à espace d'état réduit $I = \{0, \ldots, K-1\}$.

Une valeur a de I' est associé à l'état i dans I avec $i = \lfloor \frac{a}{l_A} \rfloor$, également notée par $i = class_A(a)$. Inversement, une valeur $i \in I$ correspond au point médian de l'intervalle I : $a = l_A \,.\, i + l_A/2$, $a \in I'$.

Le trafic est supposé stationnaire. La dépendance au temps pour $A(t)$ est supprimée et remplacée par une variable aléatoire discrète notée $\mathcal{A}$ (c'est-à-dire, $\mathcal{A} = A(t), \forall t$) définie

par un couple d'attributs (E^A, A). Chaque attribut est un vecteur de taille K, le premier vecteur représente les points médians des intervalles tandis que le second donne les probabilités correspondantes :

$$\mathcal{A} = (E^A, A) \begin{cases} E^A &= \{a_i \in \mathbb{N}, i = 0 \dots K-1\} \\ A &= [A[i] = \text{Prob}(\mathcal{A} = a_i) \mid i = 0 \dots K-1] \end{cases}$$

Exemple 5.1 *Nous présentons dans le tableau ci-après, un exemple de charge de travail $\mathcal{A}'$ analysé en utilisant une période d'échantillonnage de $T = 0.1\,s$. Les unités de transmission mesurées au cours de cette période d'échantillonnage sont définies sur l'intervalle $[0, 60[\,kb$. Pour une réduction à $K = 6$ classes, la méthode d'Hernàndez consiste à définir 6 sous-intervalles de même largeur avec $l_A = 10\,kb$. La classe 0 correspond à l'intervalle $[0, 10[\,kb$ dont le point médian est $a_0 = 5$. La probabilité que la source de trafic produise un nombre d'unités de transmission dans cet intervalle est $A[0] = 0$. De même la classe 1 correspond à l'intervalle $[10, 20[\,kb$ avec une probabilité de $A[1] = 0.2$, ainsi de suite. L'histogramme résultant est illustré dans la figure 5.4 et a une moyenne de $\mathbb{E}[\mathcal{A}] = 34\,kb$.*

Classe i	Intervalle	Point median	Probabilité
0	[0, 10[	5	0.00
1	[10, 20[	15	0.20
2	[20, 30[	25	0.30
3	[30, 40[	35	0.15
4	[40, 50[	45	0.10
5	[50, 60[	55	0.25

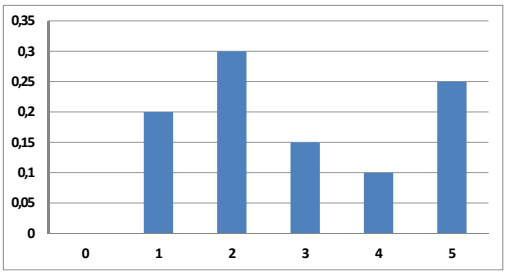


FIGURE 5.4 – Construction d'histogramme par la méthode HBSP.

Nous décrivons maintenant le processus stochastique proposé par Hernàndez et *al.* noté HBSP qui permet de définir la distribution de la longueur du tampon. L'étude de la file d'attente peut être effectuée soit par l'analyse des équations d'évolutions soit à travers une description de chaîne de Markov à temps discret. Nous présentons ci-après les deux démarches.

5.4.1.1 Processus stochastique du modèle HBSP décrit par les équations d'évolutions

Le modèle considère une file d'attente simple définie par un trafic d'entrée par histogramme noté HD (Histogram deterministic interarrival Distribution) et un tampon de capacité finie notée B. Le service noté S est de discipline FIFO (premier arrivée premier servi) avec une distribution déterministe D. Selon la notation de Kendall, la méthode revient à résoudre le modèle de file d'attente $HD/D/1/\text{B}$.

Le processus stochastique en temps discret de l'évolution de la longueur du tampon par le modèle HBSP est basé sur la relation de récurrence suivante :

$$Q(k) = \Phi_{\hat{s}}^{\hat{b}}(Q(k-1) \otimes A), \quad k = 0, 1, 2, \ldots \tag{5.1}$$

où $\hat{s} = class_A(\mathrm{S})$, $\hat{b} = class_A(\mathrm{B})$, $\otimes$: opérateur de convolution et l'opérateur Φ permet de limiter la taille du tampon de sorte qu'elle ne puissent ni être négative ni déborder de la classe associée à la longueur du tampon. Cet opérateur est défini comme suit :

$$\Phi_a^b(x) = \begin{cases} 0 & , \quad \text{pour } x < a \\ x & , \quad \text{pour } a \leq x \leq b + a \\ b & , \quad \text{pour } x \geq b + a \end{cases} \tag{5.2}$$

Cette approche numérique consiste à itérer le calcul de la distribution de Q pour chaque instant k jusqu'à obtenir deux distributions successives assez proches au sens d'une norme donnée. Cette distribution finale va représenter la distribution stationnaire de la longueur du tampon.

Exemple 5.2 *En reprenant l'exemple précédent, nous allons définir la distribution du tampon donnée par la méthode HBSP. Pour ce faire, nous considérons un taux de service de* $400\,kb/s$ *(c'est-à-dire,* $\mathrm{S} = 37\,kb$ *par période d'échantillonnage) et une capacité de tampon de* $\mathrm{B} = 55\,kb$*. En terme de pmf, ces valeurs correspondent à* $\hat{s} = class_A(\mathrm{S}) = 3$ *et* $\hat{b} = class_A(\mathrm{B}) = 5$*.*

La première étape consiste à additionner les classes 0 *à* 3 *de* $\mathcal{A}$ *en regroupant leurs masses de probabilité vers la gauche :*

$$Q(1) = \Phi_3(\mathcal{A}) = [0.65, 0.10, 0.25, 0, 0, 0].$$

Nous notons que Q(0)=0.

La probabilité d'avoir un tampon de longueur 1 *et* 2 *est de* 0.10 *et* 0.25 *respectivement. Étant donné que la longueur du tampon est* $\hat{b} = 5$*, il n'y a donc pas de probabilité dont l'état dépasse la capacité du tampon après la première itération. Mais, en général, l'opérateur de borne* Φ *permet d'établir une limite supérieure sur le nombre de données en attente dans la file vu que la longueur du tampon est finie :*

$$Q(1) = \Phi_3^5(\mathcal{A}) = [0.65, 0.10, 0.25]$$

Dans la seconde itération (période d'échantillonnage suivante), le tampon contient au préalable les données en attente $Q(1)$ *et, en additionne, les nouvelles données qui arrivent* $\mathcal{A}$*. L'histogramme des données cumulées dans le tampon après cette itération, est la convolution des histogrammes suivants :*

$$G(2) = Q(1) \otimes \mathcal{A} = [0, 0.1300, 0.2150, 0.1775, 0.1550, 0.2100, 0.0500, 0.0625]$$

Maintenant, le fait d'avoir un tampon fini ($\mathrm{B} = 5$ *classes) va entraîner un blocage (resp. une perte) dans le cas où il existe une probabilité telle que la longueur du tampon soit supérieure à*

5. *Dans ce cas, les données qui dépassent la classe de la capacité du tampon sont cumulées et exprimées dans la classe* B. *La deuxième itération de* Q *est donnée par :*

$$Q(2) = \Phi_3^5(G(2)) = [0.5225, 0.1550, 0.2100, 0.0500, 0.0625]$$

Pour une précision de $\epsilon = 10^{-4}$ *la méthode converge au bout de* 32 *itérations vers la solution suivante :*

$$Q = [0.2826, 0.1276, 0.1755, 0.1377, 0.1223, 0.1544];$$

$$et\ G = [0, 0.0565, 0.1103, 0.1158, 0.1276, 0.1755, 0.1377, 0.1223, 0.0698, 0.0460, 0.0386]$$

Nous notons que les distributions Q *et* G *sont définies respectivement sur les états :*

$$E^Q = \{5, 15, 25, 35, 45, 55\}\, kb;$$

$$et\ E^G = \{5, 15, 25, 35, 45, 55, 65, 75, 85, 95, 105\}\, kb.$$

5.4.1.2 Analyse du tampon par une Chaîne de Markov à temps discret

Dans cette partie, nous présentons la description du processus stochastique $\{Q(n)\}$ par une chaîne de Markov à temps discret (DTMC). Les probabilités de transition sont facilement calculables pour la construction de la matrice de transition $\mathbf{P}$. En utilisant cette matrice de probabilités, nous pouvons obtenir les valeurs de la distribution $Q(n)$. Cependant, pour $n \to \infty$ la résolution de la matrice nécessite un temps de calcul assez important. Pour cette raison, les auteurs ont préféré utiliser le procédé itératif décrit dans la partie précédente pour obtenir les distributions sur la longueur du tampon.

Rappelons qu'une chaîne de Markov à temps discret est un processus stochastique dont les probabilités des distributions dans l'état j ne dépendent que de l'état précédent i, et non de la façon dont le processus est arrivé à l'état i. Il est facile donc de prouver que $\{Q(n)\}$ est une DTMC. Pendant la période k, la probabilité que le tampon prenne la valeur j peut être exprimée en utilisant les probabilités du tampon de la période $k-1$, comme suit :

$$\mathrm{Prob}[Q(k) = j] = \sum_i \mathrm{Prob}[Q(k-1) = i]\, \mathrm{Prob}[Q(k) = j \,|\, Q(k-1) = i]$$

Les composantes de cette matrice de transitions sont faciles à obtenir à l'aide de la définition du processus stochastique $\{Q(n)\}$. En effet, pour obtenir la $i$ème ligne de $\mathbf{P}$ nous appliquons une itération du processus stochastique en utilisant une charge initiale d'une unité sur l'état j. Par exemple, la première ligne est obtenue par $\Phi_{\hat{s}}^{\hat{b}}([1, 0, 0, \ldots] \otimes \mathcal{A})$.
En utilisant la matrice $\mathbf{P}$, nous pouvons obtenir la *pmf* correspondante à $Q(k)$ comme suit :

$$Q(k) = Q(1)\, \mathbf{P}^k$$

La détermination de la distribution stationnaire notée $\boldsymbol{\pi}$ est obtenue en résolvant le système suivant :

$$\boldsymbol{\pi} = \boldsymbol{\pi}\, \mathbf{P} \qquad \boldsymbol{\pi}[i] \geq 0, \quad \sum_j \boldsymbol{\pi}[j] = 1.$$

Nous précisons que pour une capacité de tampon finie, la matrice $\mathbf{P}$ est de dimension $(\hat{b}+1) \times (\hat{b}+1)$. La résolution de cette matrice peut être faite par l'une des méthodes présentées dans la partie 2.1.1 du chapitre 2.

Exemple 5.3 *Nous considérons le même exemple que précédemment, à savoir* $A = [0, 0.2, 0.3, 0.15, 0.10, 0.25]$ *avec* $\hat{s} = 3$ *et* $\hat{b} = 5$*. Nous obtenons la matrice de probabilité de transition suivante :*

$$\mathbf{P} = \begin{pmatrix} 0.6500 & 0.1000 & 0.2500 & 0 & 0 & 0 \\ 0.5000 & 0.1500 & 0.1000 & 0.2500 & 0 & 0 \\ 0.2000 & 0.3000 & 0.1500 & 0.1000 & 0.2500 & 0 \\ 0 & 0.2000 & 0.3000 & 0.1500 & 0.1000 & 0.2500 \\ 0 & 0 & 0.2000 & 0.3000 & 0.1500 & 0.3500 \\ 0 & 0 & 0 & 0.2000 & 0.3000 & 0.5000 \end{pmatrix}$$

La distribution de la longueur du tampon à l'instant $k = 2$ *par exemple, est donnée par* $Q(2) = Q(1).\mathbf{P} = [0.65, 0.10, 0.25].\mathbf{P} = [0.5225, 0.1550, 0.2100, 0.0500, 0.0625, 0]$*. La distribution stationnaire obtenue par la méthode GTH est* $\boldsymbol{\pi} = [0.2826, 0.1276, 0.1755, 0.1377, 0.1223, 0.1544]$ *et elle correspond à la distribution trouvée en utilisant le procédé itératif de la méthode HBSP.*

5.4.2 Méthode de bornes stochastiques de Tancrez et *al.*

Une approche similaire a été proposée par Tancrez, Semal et Chevalier dans [TSC09] avec un contexte et une approche légèrement différente. En effet, les auteurs se sont intéressés à la construction de bornes de performances pour des chaînes de production. La méthode consiste essentiellement à discrétiser les distributions des temps de travail par des groupements en fin d'intervalles (*i.e.,* les masses de probabilités sur des intervalles réguliers sont cumulées en fin de ces intervalles). Le fonctionnement détaillé de la méthode est présenté ci-après.

La construction de la borne supérieure est effectuée comme suit. Partant d'une distribution continue qui modélise la durée de service dans une ligne de production, le support est partitionné en K sous-intervalles de même longueur (la distribution est d'abord discrétisée par conservation des masses de probabilités). Une fois la largeur des intervalles de discrétisation choisie, la masse de probabilité répartie sur chacun de ces intervalles est associée à un point de la distribution discrète. Pour obtenir une borne supérieure, le regroupement à la fin de l'intervalle est considéré et la masse de probabilité associée à ce point est donnée par l'intégrale de la masse sur l'intervalle. Une telle représentation des points et de la masse de probabilité fournit une borne supérieure au sens de l'ordre st. Des distributions discrètes des temps de travail sont ainsi obtenues, et elles peuvent aisément se traduire en terme de distributions de type phase. De plus, comme les systèmes considérés sont des variantes des réseaux de Petri "graphes d'événements", les travaux de Baccelli et *al.* [BCOQ92] montrent que la borne sur le temps de service permet de calculer une borne sur le délai moyen de bout en bout. Chacune des files peut être analysée par une méthode numérique adaptée aux chaînes de Markov structurées.

Nous illustrons la méthode par un petit exemple présenté dans la figure 5.5 (pour une longueur d'intervalle égale à 4 et $K = 6$).

Cependant, cette approche permet d'obtenir une borne qui n'est pas optimale, mais qui peut s'avérer plus régulière. Par un raisonnement similaire, la construction de la borne inférieure peut également être envisagée en mettant la masse de probabilité au début des intervalles.

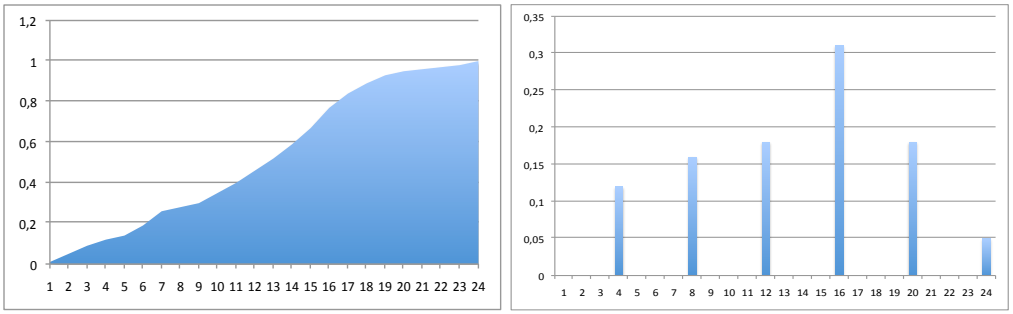


FIGURE 5.5 – Discrétisation d'une distribution continue (à gauche) et représentation sous forme de distribution de PH (à droite).

5.5 Rapport entre la taille du support de l'histogramme et la précision

Un des problèmes majeurs de la méthode de réduction de la taille des histogrammes est la précision. La question serait donc de déterminer le nombre d'états (bins) de l'histogramme nécessaire pour assurer l'obtention de résultats pertinents. C'est en général un compromis entre la complexité et la précision : avec trop d'intervalles, l'analyse sera lente et coûteuse, puisque la complexité des algorithmes dépend surtout du nombre de bins. Mais, d'autre part, trop peu d'intervalles peuvent entraîner une perte d'information sur la distribution et masquer le comportement des données. De plus, la considération d'un histogramme avec un faible nombre de bins produit des erreurs de précision importantes. Il est paradoxal que ces erreurs se produisent, même si le choix d'un nombre faible de bins est jugé suffisant pour décrire correctement le modèle de donnée (sans perdre une grande quantité d'informations). La raison de ces imprécisions semble être l'effet de l'utilisation d'un nombre entier de bins (arrondis de nombres réels) pour l'utilisation des algorithmes itératifs (équations d'évolution). Un nombre faible d'états provoque des erreurs de précision de plus en plus importantes, et elles sont cumulées à chaque itération.

Nous tenterons d'étudier dans les chapitres qui suivent, l'impact du nombre de bins sur la précision des résultats à travers quelques exemples numériques.

5.6 Conclusion

Nous venons d'introduire l'approche par histogramme et de mettre l'accent sur les problèmes de tailles qui s'avèrent être souvent un facteur de complexité pour la résolution et l'analyse des systèmes à grand espace d'état. Ces problèmes laissent ainsi envisager l'utilisation de méthodes de réduction de taille pour lesquels nous citons certains travaux récents effectués dans ce sens.

Nous avons abordé ces points, car la méthode que nous voulons proposer dans cette thèse repose sur l'étude de modèles ayant une description par histogramme. Nous développons une nouvelle approche de réduction fondée sur les bornes stochastiques. L'attrait principal de cette méthode consiste à générer des bornes stochastiques optimales. Nous allons donc chercher à

montrer son intérêt et son impact en faisant une comparaison avec les méthodes existantes.

Chapitre 6

Complexité ou précision : une approche liée à la comparaison stochastique

Nous présentons dans ce chapitre une nouvelle approche algorithmique qui permettra de déterminer des bornes stochastiques plutôt que des approximations pour une variable aléatoire discrète (distribution) correspondant à des mesures réelles. Ainsi, pour une distribution discrète, qui peut être un paramètre d'entrée du modèle, définie sur un ensemble d'états de taille N nous visons à réduire la taille de cette distribution puisque, dans les étapes de résolution du modèle, la taille de la distribution stationnaire pour calculer la mesure de performance peut s'avérer très grande. À cet effet, nous construisons une distribution stochastique bornante (inférieure et/ou supérieure) prenant ces valeurs sur un ensemble de taille $K << N$. De plus, pour une fonction de récompense donnée, la distribution bornante est optimale. Plutôt que de poser des hypothèses sur les distributions d'entrées ou de services de certains problèmes d'évaluation de performance des réseaux ou de la recherche opérationnelle, nous proposons d'utiliser l'approche par histogramme pour calculer des bornes stochastiques inférieure et supérieure sur une distribution discrète prenant leurs valeurs dans un espace de taille réduit. La méthode n'est pas nouvelle, par contre, l'utilisation des bornes stochastiques dans ce contexte nous semble originale.

Nous débutons ce chapitre par une description détaillée du problème étudié. Puis, nous démontrons quelques résultats théoriques sur les bornes stochastiques. La section 6.3 est consacrée à la présentation les différents algorithmes développés pour la construction de bornes stochastiques (algorithme glouton et algorithme optimal) pour une récompense positive croissante donnée. Enfin, nous illustrons notre approche par l'étude de deux problèmes bien connus de l'évaluation des performances : la distribution de la durée d'exécution dans un graphe de tâches stochastique et l'analyse des temps d'attente dans une file FIFO par l'équation de Loynes.

Dans notre étude, nous nous sommes intéressés particulièrement à l'ordre stochastique $\leq_{st}$, car il permet de borner les récompenses croissantes et cette propriété est souvent vraie pour les récompenses utilisées en évaluation de performances.

6.1 Description du problème

Soit $\boldsymbol{d}$ une distribution de probabilité définie sur un espace d'état discret, fini et totalement ordonné $\mathcal{H}$. L'ensemble $\mathcal{H}$ est un ensemble d'entiers de cardinalité N (noté $|\mathcal{H}| = N$). Nous utilisons le terme *distribution* dans la suite, mais nous rappelons que cette dernière correspond à la fonction densité d'une variable aléatoire discrète notée *pmf*. Nous notons que lors de la phase d'implémentation et dans le but d'avoir un gain de mémoire et d'accélérer le calcul, la distribution $\boldsymbol{d}$ est telle que $\boldsymbol{d}[i] > 0$ pour tout $i \in \mathcal{H}$ (aucun élément de $\mathcal{H}$ n'a de probabilité nulle). Nous considérons également une fonction de récompense positive croissante notée $\boldsymbol{r}$ et définissons par $R_{\boldsymbol{d}} = \sum \boldsymbol{r}[i]\boldsymbol{d}[i]$ l'espérance des récompenses associée à la distribution $\boldsymbol{d}$. Dans ce travail, nous supposons que la récompense est positive et croissante, car ses propriétés sont conformes à l'ordre stochastique sur les distributions (voir le chapitre 2). De nombreuses récompenses utilisées dans la modélisation de performance sont positives et croissantes : par exemple la moyenne et autres moments.

Notre objectif dans ce chapitre consiste à déterminer deux distributions $\boldsymbol{d1}$ et $\boldsymbol{d2}$ représentant la borne supérieure et la borne inférieure de la distribution initiale $\boldsymbol{d}$ au sens de l'ordre stochastique fort. Ces deux distributions correspondent aux meilleures approximations (bornes) pour la fonction de récompense $\boldsymbol{r}$, ce qui revient à dire que leurs espérances de récompenses $R_{\boldsymbol{d1}}$ et $R_{\boldsymbol{d2}}$ sont les plus proches de $R_{\boldsymbol{d}}$. La distribution $\boldsymbol{d1}$ (resp. $\boldsymbol{d2}$) est définie sur un support noté $\mathcal{F}_1$ (resp. $\mathcal{F}_2$) de cardinalité K, avec $K << N$. Afin d'établir une comparaison stochastique entre les différentes distributions, nous avons besoin de définir un ensemble plus large. Soient les ensembles $\mathcal{G}_1$ et $\mathcal{G}_2$ tels que $\mathcal{G}_1 = \mathcal{H} \cup \mathcal{F}_1$ et $\mathcal{G}_2 = \mathcal{H} \cup \mathcal{F}_2$, supposés totalement ordonnés. Pour une distribution $\boldsymbol{d}$ définie sur $\mathcal{H}$, $\boldsymbol{d}$ peut être représentée sur l'ensemble $\mathcal{G}_1$ (ou $\mathcal{G}_2$) en intégrant les probabilités nulles $\boldsymbol{d}[i] = 0$ pour tout i dans $\mathcal{G}_1 - \mathcal{H}$ (ou $\mathcal{G}_2 - \mathcal{H}$). De même, une distribution prenant ses valeurs sur $\mathcal{F}_1$ (ou $\mathcal{F}_2$) peut également être étendue à l'ensemble $\mathcal{G}_1$ (ou $\mathcal{G}_2$).

Plus formellement, pour une distribution $\boldsymbol{d}$ définie sur l'espace d'état $\mathcal{H}$ ($|\mathcal{H}| = N$), nous voulons calculer les distributions bornantes $\boldsymbol{d1}$ et $\boldsymbol{d2}$ tel que :

1. $\boldsymbol{d2} \leq_{st} \boldsymbol{d} \leq_{st} \boldsymbol{d1}$;
2. La distribution $\boldsymbol{d2}$ est définie sur un support $\mathcal{F}_2$ de taille K ;
3. Pour l'ensemble des distributions avec K états, la borne stochastique inférieure de $\boldsymbol{d}$ vérifie que $\sum_{i \in \mathcal{G}_2} \boldsymbol{r}[i]\boldsymbol{d}[i] - \sum_{i \in \mathcal{G}_2} \boldsymbol{r}[i]\boldsymbol{d2}[i]$ est minimale ;
4. La distribution $\boldsymbol{d1}$ est définie sur un support $\mathcal{F}_1$ de taille K ;
5. Pour l'ensemble des distributions en K états, la borne stochastique supérieure de $\boldsymbol{d}$ vérifie que $\sum_{i \in \mathcal{G}_1} \boldsymbol{r}[i]\boldsymbol{d1}[i] - \sum_{i \in \mathcal{G}_1} \boldsymbol{r}[i]\boldsymbol{d}[i]$ est minimale ;

L'intérêt d'avoir une distribution bornante définie sur un espace d'état de taille plus petite est de permettre de diminuer la complexité des opérations sur les distributions et rendre le calcul des mesures de performances plus rapide. De plus, si les opérations du problème étudié sont monotones par rapport à l'ordre stochastique, nous pouvons garantir également l'obtention de bornes sur les résultats finaux. Par exemple, le calcul de la distribution de la durée d'exécution d'un graphe de tâches où les durées sont données par des variables aléatoires nécessite deux opérations : l'addition et le maximum de variables aléatoires. Ces dernières correspondent respectivement à la convolution et aux produits des *pmf* respectives lorsque les

variables aléatoires sont mutuellement indépendantes. Ces deux opérations sont monotones par rapport à l'ordre stochastique fort. Ainsi, si les durées des tâches considérées sont des bornes stochastiques de tailles réduites, nous pouvons déduire des bornes sur le temps d'exécution du graphe de tâches.

Les distributions bornantes ***d1*** et ***d2*** représentent les *meilleures approximations* de ***d*** pour une récompense donnée ***r***, nous utilisons à cet effet la notion d'optimalité de la solution. Une définition formelle de l'optimalité est présentée ci-après. Nous présentons et développons dans ce chapitre différentes approches et algorithmes heuristiques et optimaux. Ceux-ci nous permettent de décrire et de distinguer entre une borne stochastique et la meilleure borne stochastique.

Définition 6.1 *Soient* ***d*** *une distribution de probabilité discrète définie sur* $\mathcal{H}$, $|\mathcal{H}| = N$ *et* ***r*** *une fonction de récompense positive croissante. La distribution* ***d2*** *définie sur* K *états est dite* ***distribution borne inférieure optimale*** *de* ***d*** *(****d2*** $\leq_{st}$ ***d****) pour la récompense* ***r*** *si et seulement si :*

$\nexists$ *de distribution* ***d'*** *définie sur K états tel que* ***d2*** $\leq_{st}$ ***d'*** $\leq_{st}$ ***d*** *et* $R_{\mathbf{d2}} \leq R_{\mathbf{d'}} < R_{\mathbf{d}}$.

De même, nous pouvons énoncer une définition analogue pour le cas d'une distribution borne supérieure optimale.

Définition 6.2 *Soient* ***d*** *une distribution de probabilité discrète définie sur* $\mathcal{H}$, $|\mathcal{H}| = N$ *et* ***r*** *une fonction de récompense positive croissante. La distribution* ***d1*** *définie sur* K *états est dite* ***distribution borne supérieure optimale*** *de* ***d*** *(****d*** $\leq_{st}$ ***d1****) pour la récompense* ***r*** *si et seulement si :*

$\nexists$ *de distribution* ***d'*** *définie sur K états tel que* ***d*** $\leq_{st}$ ***d'*** $\leq_{st}$ ***d1*** *et* $R_{\mathbf{d}} \leq R_{\mathbf{d'}} < R_{\mathbf{d1}}$.

6.2 Résultats sur la caractérisation des meilleures bornes stochastiques

Nous débutons cette section par énoncer quelques résultats théoriques sur les distributions bornantes. La première proposition formulée représente une implication directe de la comparaison stochastique (théorème 2.5).

Proposition 6.1 *Comme* ***r*** *est une fonction positive croissante et que* ***d2*** $\leq_{st}$ ***d****, nous avons :*

$$\sum_{i \in \mathcal{G}_2} \mathbf{r}[i]\mathbf{d}[i] - \sum_{i \in \mathcal{G}_2} \mathbf{r}[i]\mathbf{d2}[i] \geq 0.$$

De même, comme ***d*** $\leq_{st}$ ***d1*** *nous avons* $\sum_{i \in \mathcal{G}_1} \mathbf{r}[i]\mathbf{d1}[i] - \sum_{i \in \mathcal{G}_1} \mathbf{r}[i]\mathbf{d}[i]$ *est positif.*

Puisque les états de $\mathcal{G}_2$ (resp. $\mathcal{G}_1$) sont totalement ordonnés et finis, nous pouvons distinguer un état maximal et un état minimal noté respectivement *ÉtatMin* et *ÉtatMax*. Nous donnons les propositions suivantes pour caractériser les distributions bornantes ***d1*** et ***d2***.

Proposition 6.2 (Borne inférieure) *Si nous calculons la meilleure borne inférieure, alors l'état ÉtatMin est dans $\mathcal{F}_2 \cap \mathcal{H}$ et l'état ÉtatMax est dans $\mathcal{H}$ (c.-à-d. $\boldsymbol{d2}[\acute{E}tatMin] > 0$, $\boldsymbol{d}[\acute{E}tatMin] > 0$ et $\boldsymbol{d}[\acute{E}tatMax] > 0$).*

PREUVE. Premièrement, afin d'assurer que l'état *ÉtatMin* (état minimal de l'ensemble $\mathcal{G}_2 = \mathcal{F}_2 \cup \mathcal{H}$) est dans $\mathcal{F}_2 \cap \mathcal{H}$ nous devons prouver que $\boldsymbol{d2}[\acute{E}tatMin] \neq 0$ et que $\boldsymbol{d}[\acute{E}tatMin] \neq 0$. Nous avons $\boldsymbol{d2} \leq_{st} \boldsymbol{d}$ ce qui implique que $\boldsymbol{d2}[\acute{E}tatMin] \geq \boldsymbol{d}[\acute{E}tatMin]$ (définition 2.16).
Deux cas peuvent être observés pour $\boldsymbol{d}[\acute{E}tatMin]$:

- Si $\boldsymbol{d}[\acute{E}tatMin] > 0$, alors $\boldsymbol{d2}[\acute{E}tatMin] > 0$. Par conséquent, l'état *ÉtatMin* est à la fois dans $\mathcal{H}$ et dans $\mathcal{F}_2$.
- Si $\boldsymbol{d}[\acute{E}tatMin] = 0$, alors la probabilité $\boldsymbol{d2}[\acute{E}tatMin]$ doit être strictement positive car *ÉtatMin* est un état de $\mathcal{G}_2$ (les états de $\mathcal{G}_2$ ont une probabilité strictement positive pour $\boldsymbol{d}$ ou $\boldsymbol{d2}$). Ce cas représente une contradiction avec l'optimalité de $\boldsymbol{d2}$, car on peut facilement construire une distribution borne inférieure meilleure que $\boldsymbol{d2}$.
  Nous pouvons illustrer cette contradiction à travers un petit exemple. Soit une distribution discrète $\boldsymbol{d} = [0.3, 0.5, 0.1, 0.1]$ définie sur l'ensemble $\mathcal{H} = \{2, 3, 4, 5\}$ et soit $\boldsymbol{d2}$ sa meilleure borne stochastique inférieure sur 3 états ($3 < 4$) telle que $\boldsymbol{d2} = [0.3, 0.5, 0.2]$ définie sur $\mathcal{F}_2 = \{1, 3, 4\}$. Pour une fonction de récompense $\boldsymbol{r}[i] = \mathcal{H}[i]$, pour tout $i = 0, \ldots 3$, les espérances de récompenses associées aux distributions $\boldsymbol{d}$ et $\boldsymbol{d2}$ sont données par $R_{\boldsymbol{d}} = 3$ et $R_{\boldsymbol{d2}} = 2.6$ respectivement.
  Si l'on considère la distribution discrète $\boldsymbol{d2a} = [0.3, 0.5, 0.2]$ qui prend ces valeurs sur $\mathcal{F}_{2a} = \{2, 3, 4\}$ avec $R_{\boldsymbol{d2a}} = 2.9$, on peut facilement remarquer que $\boldsymbol{d2} \leq_{st} \boldsymbol{d2a} \leq_{st} \boldsymbol{d}$ et que $\boldsymbol{d2a}$ est meilleure au sens de l'espérance de récompense que $\boldsymbol{d2}$. La distribution $\boldsymbol{d2a}$ est construite à partir de $\boldsymbol{d2}$ en déplaçant la masse de probabilités des états inférieure à $\mathcal{H}[0]$ dans $\mathcal{F}_2$ et en les regroupant sur l'état $\mathcal{H}[0]$. Nous avons donc une contradiction avec l'optimalité de la distribution $\boldsymbol{d2}$.

Nous prouvons maintenant la deuxième partie de la proposition. Nous supposons que l'état *ÉtatMax* n'appartient pas à l'ensemble $\mathcal{H}$. Nous rappelons que pour $\boldsymbol{d2} \leq_{st} \boldsymbol{d}$ nous avons $\boldsymbol{d2}[\acute{E}tatMax] \leq \boldsymbol{d}[\acute{E}tatMax]$ (définition 2.16). Par conséquent, si $\boldsymbol{d}[\acute{E}tatMax] = 0$, alors $\boldsymbol{d2}[\acute{E}tatMax] = 0$ et l'état *ÉtatMax* n'appartient ni à l'ensemble $\mathcal{F}_2$ ni à l'ensemble $\mathcal{H}$ à cause de l'hypothèse sur les distributions. Ce qui revient à dire que l'état *ÉtatMax* n'est pas un état de $\mathcal{G}_2$. Il y a donc une contradiction. ■

Un résultat similaire peut également être énoncé pour la distribution borne supérieure $\boldsymbol{d1}$ en inversant les états *ÉtatMin* et *ÉtatMax*.

Proposition 6.3 (Borne supérieure) *Si nous calculons la meilleure borne supérieure, alors l'état ÉtatMax est dans $\mathcal{F}_1 \cap \mathcal{H}$ et l'état ÉtatMin est dans $\mathcal{H}$ (c.-à-d. $\boldsymbol{d1}[\acute{E}tatMax] > 0$, $\boldsymbol{d}[\acute{E}tatMax] > 0$ et $\boldsymbol{d}[\acute{E}tatMin] > 0$).*

PREUVE. La preuve est analogue à celle de la proposition 6.2. ■

Nous allons à présent établir et prouver le résultat le plus important sur les ensembles. Pour ce faire, nous définissons au préalable les notions suivantes. Soit $\mathcal{G}$ un ensemble non vide, pour

un état arbitraire k, $\Gamma^-_{\mathcal{G}}(k)$ dénote le plus grand état de l'ensemble $\mathcal{G}$ plus petit que k. De même, $\Gamma^+_{\mathcal{G}}(k)$ correspond au plus petit état de $\mathcal{G}$ plus grand que k.

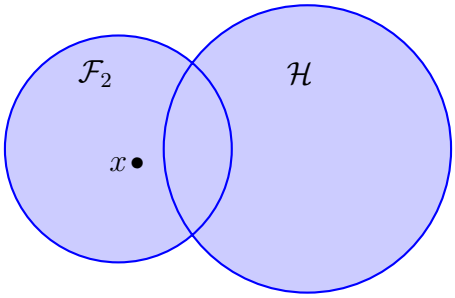
Lemme 6.1 *Soit **d2** la distribution borne inférieure optimale. Nous supposons que le support de **d** est $\mathcal{H}$ et que le support de **d2** est $\mathcal{F}_2$, alors $\mathcal{F}_2 \subset \mathcal{H}$. Par conséquent, $\mathcal{G}_2 = \mathcal{H}$.*

PREUVE. Soit x un état arbitraire de $\mathcal{G}_2$. nous allons prouver que l'état x est dans $\mathcal{H}$.

▸ Premièrement, nous supposons que $x = \textit{ÉtatMax}$. D'après la proposition 6.2, nous avons $\boldsymbol{d}[\textit{ÉtatMax}] > 0$ ce qui veut dire que *ÉtatMax* est dans $\mathcal{H}$.

▸ Deuxièmement, si nous supposons que $x < \textit{ÉtatMax}$ et que $\boldsymbol{d}[x] > 0$, l'état x est dans $\mathcal{H}$ et il est clair que la preuve est ainsi terminée.

▸ Maintenant, si nous supposons que $x < \textit{ÉtatMax}$ et que $\boldsymbol{d}[x] = 0$, nous avons alors $\boldsymbol{d2}[x] > 0$, c'est-à-dire que $\mathcal{G}_2 \neq \mathcal{H}$.



$\mathcal{G}_2 = \mathcal{H} \cup \mathcal{F}_2$

Nous allons montrer une contradiction en définissant une nouvelle distribution ***d3*** définie également sur K états tel que $\boldsymbol{d2} \leq_{st} \boldsymbol{d3} \leq_{st} \boldsymbol{d}$ et dont l'espérance des récompenses est meilleure que l'espérance des récompenses de ***d2***.

Soient h le plus petit état de l'ensemble $\mathcal{G}_2$ plus grand que x ($h = \Gamma^+_{\mathcal{G}_2}(x)$) tel que $\boldsymbol{d}[h] > 0$ et $y = \Gamma^-_{\mathcal{G}_2}(x)$. L'état h existe, car $\boldsymbol{d}[\textit{ÉtatMax}] > 0$ (proposition 6.2) et les états sont totalement ordonnés. Nous avons deux cas possibles, selon la valeur de $\boldsymbol{d2}[x]$:

1. $\boldsymbol{d2}[x] > 0$ ($\mathbf{x} \in \mathcal{F}_2$) : nous définissons ***d3*** comme suit :

$$\begin{cases} \boldsymbol{d3}[i] = \boldsymbol{d2}[i] & \forall i \leq y, \\ \boldsymbol{d3}[x] = \boldsymbol{d2}[x]/2, & \\ \boldsymbol{d3}[h] = \boldsymbol{d2}[h] + \boldsymbol{d2}[x]/2, & \\ \boldsymbol{d3}[j] = \boldsymbol{d2}[j] & \forall j > h. \end{cases}$$

Notant que la distribution ***d3*** est définie sur le même ensemble que ***d2*** : $\mathcal{F}_2$. Ce cas est représenté dans la figure 6.1.

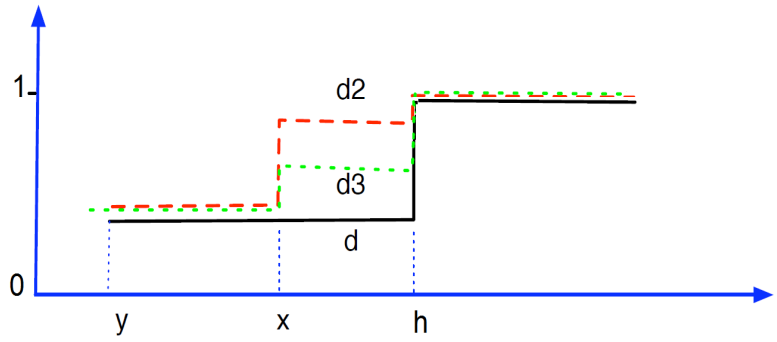


FIGURE 6.1 – Les distributions cumulées associées au premier cas.

2. $\boldsymbol{d2}[x] = 0$ ($\mathbf{x} \notin \mathcal{F}_2$) : la distribution $\boldsymbol{d}3$ est définie comme suit :

$$\begin{cases} \boldsymbol{d3}[i] = \boldsymbol{d2}[i] & \forall i \leq y, \\ \boldsymbol{d3}[x] = 0, & \\ \boldsymbol{d3}[h] = \boldsymbol{d2}[x], & \\ \boldsymbol{d3}[j] = \boldsymbol{d2}[j] & \forall j > h. \end{cases}$$

La distribution $\boldsymbol{d}3$ n'est pas définie sur le même ensemble que ***d2***, mais les deux distributions sont de même taille.

D'après ces deux cas, il est clair que nous avons les points suivants :

- La distribution ***d3*** a le même nombre d'états que la distribution ***d2***, pas forcement les mêmes. De plus, les probabilités associées sont toutes positives.
- ***d2*** $\leq_{st}$ ***d3***. Lors de la construction de ***d3*** à partir de la distribution ***d2*** nous déplaçons la masse de probabilité de l'état x vers l'état h qui est plus grand.
- ***d3*** $\leq_{st}$ ***d***. Cette propriété peut être vérifiée par les étapes de calcul suivantes.
  Tout d'abord, nous savons que ***d2*** $\leq_{st}$ ***d***, alors

$$\sum_{j \leq h} \boldsymbol{d2}[j] \leq \sum_{j \leq h} \boldsymbol{d}[j]$$

et

$$\sum_{j \leq x} \boldsymbol{d2}[j] \leq \sum_{j \leq x} \boldsymbol{d}[j]$$

Par hypothèse, $\boldsymbol{d}[x] = 0$ et h est l'indice le plus petit qui est plus grand que x tel que $\boldsymbol{d}[h] > 0$. Alors,

$$\sum_{j \leq x} \boldsymbol{d}[j] \leq \sum_{j \leq h} \boldsymbol{d}[j]$$

En prenant en considération la définition de ***d3***, nous obtenons pour tout $\ell > k$:

$$\sum_{j \leq \ell} \boldsymbol{d3}[j] = \sum_{j \leq \ell} \boldsymbol{d2}[j] \leq \sum_{j \leq \ell} \boldsymbol{d}[j],$$

et

$$\sum_{j \leq h} \boldsymbol{d3}[j] = \sum_{j \leq x} \boldsymbol{d2}[j] \leq \sum_{j \leq x} \boldsymbol{d}[j] = \sum_{j \leq h} \boldsymbol{d}[j]$$

et pour tout $\ell \leq x$

$$\sum_{j\leq\ell} \boldsymbol{d3}[j] = \sum_{j\leq\ell} \boldsymbol{d2}[j] \leq \sum_{j\leq\ell} \boldsymbol{d}[j].$$

Par conséquent, $\boldsymbol{d3} \leq_{st} \boldsymbol{d}$.

- Comme $\boldsymbol{d2} \leq_{st} \boldsymbol{d3}$ et $\boldsymbol{r}$ est croissante, alors l'espérance des récompenses de ***d3*** est plus grande que l'espérance des récompenses de ***d2*** ($R_{\boldsymbol{d3}} > R_{\boldsymbol{d2}}$). Ainsi, la distribution ***d3*** est la meilleure borne inférieure sur K états. Nous arrivons donc à une contradiction avec l'hypothèse de départ stipulant que la distribution ***d2*** est la meilleure borne. Par conséquent, $\boldsymbol{d}[x] = 0$ n'est pas correct. L'état x doit donc appartenir à l'ensemble $\mathcal{H}$.

■

Lemme 6.2 *Soit **d1** la distribution borne supérieure optimale. Nous supposons que le support de **d** est $\mathcal{H}$ et que le support de **d1** est $\mathcal{F}_1$, alors $\mathcal{F}_1 \subset \mathcal{H}$. Par conséquent $\mathcal{G}_1 = \mathcal{H}$.*

PREUVE. La preuve peut être faite de façon similaire à celle du lemme 6.1 en considérant dans ce cas l'état $y = \Gamma^{-}_{\mathcal{G}_1}(x)$. ■

Une synthèse des lemmes 6.1 et 6.2, nous permet d'énoncer le résultat suivant :

Lemme 6.3 *Soit une distribution discrète **d** définie sur l'ensemble $\mathcal{H}$ avec N états, et soient **d2** (resp. **d1**) sa meilleure borne inférieure (resp. supérieure) définie sur l'ensemble $\mathcal{F}_2$ (resp. $\mathcal{F}_1$) avec K états, alors $\mathcal{G}_2 = \mathcal{G}_1 = \mathcal{H}$, $\mathcal{F}_2 \subset \mathcal{H}$ et $\mathcal{F}_1 \subset \mathcal{H}$.*

6.2.1 Bornes stochastiques optimales sur $K = N - 1$ états

D'après les résultats prouvés jusqu'à présent, nous pouvons déjà dire que nous sommes aptes à définir une distribution borne inférieure (resp. supérieure) optimale définie sur $K = N-1$ états. Cette assertion est énoncée à travers le lemme 6.4 (resp. lemme 6.5) et est retranscrite dans l'algorithme 18 (resp. algorithme 19).

Lemme 6.4 *Soient une distribution discrète **d** définie sur N états et **d2** sa meilleure borne inférieure définie sur $N-1$ états pour une récompense positive **r**. Les distributions **d** et **d2** ont les mêmes valeurs sur les $N-2$ états de $\mathcal{H}$.*

PREUVE. La distribution ***d2*** est obtenue en minimisant la différence des espérances des récompenses $\sum_{i\in\mathcal{H}} \boldsymbol{r}[i]\boldsymbol{d}[i] - \sum_{i\in\mathcal{H}} \boldsymbol{r}[i]\boldsymbol{d2}[i]$. Nous savons que les supports de ***d2*** et ***d*** ne diffèrent que par un seul état, noté k qui est retiré de $\mathcal{H}$. Soit $l = \Gamma^{-}_{\mathcal{H}}(k)$. L'état l existe car la proposition 6.2 stipule que l'état *ÉtatMin* est à la fois dans $\mathcal{H}$ et $\mathcal{F}_2$. Donc, $k \neq$ *ÉtatMin* et $\Gamma^{-}_{\mathcal{H}}(k)$ existe.
Nous avons :

$$\begin{cases} \boldsymbol{d2}[i] = \boldsymbol{d}[i] & \forall i < l, \\ \boldsymbol{d2}[l] = \boldsymbol{d}[l] + \boldsymbol{d}[k], & \\ \boldsymbol{d2}[j] = \boldsymbol{d}[j] & \forall j > k. \end{cases}$$

La différence entre les espérances des récompenses est donnée par :

$$\begin{aligned}
\sum_{i\in\mathcal{H}} \boldsymbol{r}[i]\,\boldsymbol{d}[i] - \sum_{i\in\mathcal{H}} \boldsymbol{r}[i]\,\boldsymbol{d2}[i] &= \left(\sum_{i<l} \boldsymbol{r}[i]\,\boldsymbol{d}[i] + \boldsymbol{r}[l]\,\boldsymbol{d}[l] + \boldsymbol{r}[k]\,\boldsymbol{d}[k] + \sum_{i>k} \boldsymbol{r}[i]\,\boldsymbol{d}[i]\right) \\
&- \left(\sum_{i<l} \boldsymbol{r}[i]\,\boldsymbol{d}[i] + \boldsymbol{r}[l]\,(\boldsymbol{d}[l] + \boldsymbol{d}[k]) + 0 + \sum_{i>k} \boldsymbol{r}[i]\,\boldsymbol{d}[i]\right); \\
&= \boldsymbol{r}[l]\,\boldsymbol{d}[l] + \boldsymbol{r}[k]\,\boldsymbol{d}[k] - \boldsymbol{r}[l]\,\boldsymbol{d}[l] - \boldsymbol{r}[l]\,\boldsymbol{d}[k]; \\
&= \boldsymbol{d}[k](\boldsymbol{r}[k] - \boldsymbol{r}[l]).
\end{aligned}$$

Ainsi, la méthode dans ce cas consiste à calculer les valeurs de $\boldsymbol{d}[i](\boldsymbol{r}[i] - \boldsymbol{r}[\Gamma^{-}{}_{\mathcal{H}}(i)])$ pour tout état i dans $\mathcal{H}$ et à supprimer l'état qui minimise cette différence de récompenses cumulées. ■

Algorithme 18 : Suppression d'un seul état : algorithme optimal borne inférieure

1 Pour tout i dans $\mathcal{H}$ excepté l'état *ÉtatMin*, calculer $\mathbf{z}[i] = \boldsymbol{d}[i](\mathbf{r}[i] - \mathbf{r}[\Gamma^{-}{}_{\mathcal{H}}(i)])$.

2 Soit $\mathcal{F}_2 = \mathcal{H}$.

3 Trouver l'indice k qui minimise la quantité calculée.

4 Additionner $\boldsymbol{d}[k]$ à $\boldsymbol{d}[\Gamma^{-}{}_{\mathcal{F}_2}(k)]$.

5 Supprimer l'état k de $\mathcal{F}_2$.

Une démarche similaire est également entreprise pour déterminer la borne supérieure sur $K = N - 1$ états.

Lemme 6.5 *Soient une distribution discrète* $\boldsymbol{d}$ *définie sur* N *états et* $\boldsymbol{d1}$ *sa meilleure borne supérieure définie sur* $N-1$ *états pour une récompense positive* $\boldsymbol{r}$*. Les distributions* $\boldsymbol{d}$ *et* $\boldsymbol{d1}$ *ont les mêmes valeurs sur les* $N-2$ *états de* $\mathcal{H}$*.*

PREUVE. La distribution $\boldsymbol{d1}$ est également obtenue en minimisant la différence des espérances des récompenses $\sum_{i\in\mathcal{H}} \boldsymbol{r}[i]\,\boldsymbol{d1}[i] - \sum_{i\in\mathcal{H}} \boldsymbol{r}[i]\,\boldsymbol{d}[i]$. Les supports de $\boldsymbol{d1}$ et $\boldsymbol{d}$ ne diffèrent que par un seul état, noté k qui est retiré de $\mathcal{H}$. Soit $h = \Gamma^{+}{}_{\mathcal{H}}(k)$. D'après la proposition 6.3, l'état h existe (l'état *ÉtatMax* $\in \mathcal{H} \cap \mathcal{F}_1$).
Nous avons :

$$\left[\begin{array}{lcll}
\boldsymbol{d1}[i] & = & \boldsymbol{d}[i] & \forall i < k, \\
\boldsymbol{d1}[h] & = & \boldsymbol{d}[h] + \boldsymbol{d}[k], & \\
\boldsymbol{d1}[j] & = & \boldsymbol{d}[j] & \forall j > h.
\end{array}\right.$$

La différence entre les espérances des récompenses est donnée par :

$$\begin{aligned}\sum_{i\in\mathcal{H}} \boldsymbol{r}[i]\,\boldsymbol{dI}[i] - \sum_{i\in\mathcal{H}} \boldsymbol{r}[i]\,\boldsymbol{d}[i] &= \left(\sum_{i<k} \boldsymbol{r}[i]\,\boldsymbol{d}[i] + 0 + \boldsymbol{r}[h]\,(\boldsymbol{d}[h]+\boldsymbol{d}[k]) + \sum_{i>h} \boldsymbol{r}[i]\,\boldsymbol{d}[i]\right)\\ &- \left(\sum_{i<k} \boldsymbol{r}[i]\,\boldsymbol{d}[i] + \boldsymbol{r}[k]\,\boldsymbol{d}[k] + \boldsymbol{r}[h]\,\boldsymbol{d}[h] + \sum_{i>h} \boldsymbol{r}[i]\,\boldsymbol{d}[i]\right);\\ &= \boldsymbol{r}[h]\,\boldsymbol{d}[h] + \boldsymbol{r}[h]\,\boldsymbol{d}[k] - \boldsymbol{r}[k]\,\boldsymbol{d}[k] - \boldsymbol{r}[h]\,\boldsymbol{d}[h];\\ &= \boldsymbol{d}[k](\boldsymbol{r}[h] - \boldsymbol{r}[k]).\end{aligned}$$

La méthode consiste donc à calculer les valeurs de $\boldsymbol{d}[i](\boldsymbol{r}[\Gamma^{+}{}_{\mathcal{H}}(i)] - \boldsymbol{r}[i])$ pour tout état i dans $\mathcal{H}$ et à supprimer l'état qui minimise cette différence de récompenses cumulées. ■

Algorithme 19 : Algorithme borne supérieure pour la suppression d'un état

1 Pour tout i dans $\mathcal{H}$ excepté l'état *ÉtatMax*, calculer $\mathbf{z}[i] = \boldsymbol{d}[i](\mathbf{r}[\Gamma^{+}{}_{\mathcal{H}}(i)] - \mathbf{r}[i])$.

2 Soit $\mathcal{F}_1 = \mathcal{H}$.

3 Trouver l'indice k qui minimise la quantité calculée.

4 Additionner $\boldsymbol{d}[k]$ à $\boldsymbol{d}[\Gamma^{+}{}_{\mathcal{F}_1}(k)]$.

5 Supprimer l'état k de $\mathcal{F}_1$.

Propriété 6.1 *La complexité de l'algorithme 18 (resp. algorithme 19) est linéaire.*

PREUVE. Il s'agit de parcourir un ensemble de taille N ($|\mathcal{H}| = N$). ■

6.3 Algorithmes de construction de bornes stochastiques

Nous présentons dans cette section, les différents algorithmes développés pour la construction de bornes stochastiques pour une distribution discrète initiale $\boldsymbol{d}$.

6.3.1 Algorithmes gloutons

Une première idée consisterait à exploiter le résultat du lemme 6.4 (resp. lemme 6.5) et l'étendre au cas général à savoir une réduction sur $K \leq N - 1$ états. Ceci revient donc à trier les valeurs correspondant aux différences de récompenses et à sélectionner les $N - K$ premières valeurs. Nous présentons ci-après l'algorithme glouton pour la détermination d'une borne inférieure et l'algorithme glouton pour la détermination d'une borne supérieure.

a) Borne inférieure :

La détermination d'une borne inférieure pour $\boldsymbol{d}$ par une approche intuitive consiste à trier les valeurs correspondant à la différence de récompense : $\boldsymbol{d}[i](\boldsymbol{r}[i] - \boldsymbol{r}[\Gamma^{-}{}_{\mathcal{H}}(i)])$ pour tout $i \in \mathcal{H} \setminus \{\textit{ÉtatMin}\}$ et de sélectionner les $N - K$ premières valeurs. Cette démarche est reportée dans l'algorithme suivant et représente une forme itérée de l'algorithme 18.

Algorithme 20 : Algorithme borne inférieure avec suppression itérée d'un état

1 Soit $\mathcal{F}_2 = \mathcal{H}$;
2 Pour tout i dans $\mathcal{H}$ excepté *ÉtatMin*, calculer $\boldsymbol{z}[i] = \boldsymbol{d}[i](\boldsymbol{r}[i] - \boldsymbol{r}(\Gamma^{-}{}_{\mathcal{F}}[i]))$;
3 **repeat**
3.a Trouver l'indice k qui minimise $\boldsymbol{z}$;
3.b Additionner la probabilité $\boldsymbol{d}[k]$ à $\boldsymbol{d}[\Gamma^{-}{}_{\mathcal{F}}[k]]$;
3.c Supprimer l'état k de $\mathcal{F}_2$;
3.d Calculer les nouvelles valeurs de $\boldsymbol{z}[\Gamma^{+}{}_{\mathcal{F}_2}(k)]$ et $\boldsymbol{z}[\Gamma^{-}{}_{\mathcal{F}_2}(k)]$.
until $|\mathcal{F}_2| = K$;

Propriété 6.2 *Lorsque l'instruction* 3.a *de l'algorithme 20 donne plus d'un indice, il est difficile de déterminer la stratégie à aborder pour choisir le point à enlever afin de garantir la meilleure solution. Selon le choix du point à supprimer, la solution finale peut ne pas être la même.*

Nous allons illustrer ce cas à travers un petit exemple.

Exemple 6.1 *Soit une distribution discrète* $\boldsymbol{d} = [0.2, 0.1, 0.1, 0.2, 0.4]$ *définie sur* $\mathcal{H} = \{1, 2, 3, 4, 5\}$. *Soit* $\boldsymbol{r}[i] = \mathcal{H}[i]$ *une fonction de récompense croissante, l'espérance des récompenses de* $\boldsymbol{d}$ *est* $R_{\boldsymbol{d}} = 3.5$. *La construction de la meilleure borne inférieure sur deux états* ($K = 2$) *par l'algorithme 20 passe par les étapes de calculs suivantes :*

À la première itération, $\mathcal{F}_2 = \mathcal{H}$ *et le vecteur* $\mathbf{z}$ *est égal à* $[0.2, 0.1, 0.1, 0.2, 0.4]$, *nous avons deux états possible à éliminer : l'état* 2 *et l'état* 3. *Nous allons examiner les deux alternatives :*

- *Retirer l'état* 2 *: l'ensemble* $\mathcal{F}_2$ *devient* $\{1, 3, 4, 5\}$ *et la distribution* $\boldsymbol{d2}$ *est égale à* $[0.3, 0.1, 0.2, 0.4]$. *Le nouveau vecteur* $\mathbf{z}$ *est égal à* $[0.3, 0.2, 0.2, 0.4]$ *et la nouvelle espérance des récompenses est* 3.4. *Nous avons toujours deux choix pour le prochain état à supprimer : l'état* 3 *et l'état* 4. *Nous examinons de nouveau les deux cas :*
  - *Retirer l'état* 3 *:* $\mathcal{F}_2 = \{1, 4, 5\}$, $\boldsymbol{d2} = [0.4, 0.2, 0.4]$, $\boldsymbol{z} = [0.4, 0.6, 0.4]$ *et l'espérance des récompenses est* 3.2. *Lors de la dernière étape, nous supprimons l'état* 5 *(il n'est pas possible de supprimer l'état* 1 = *ÉtatMin). La solution finale est définie sur* $\mathcal{F}_2 = \{1, 4\}$ *avec une distribution* $[0.4, 0.6]$ *et* $R1a = 2.8$.
  - *Retirer l'état* 4 *:* $\mathcal{F}_2 = \{1, 3, 5\}$, $\boldsymbol{d2} = [0.3, 0.3, 0.4]$, $\boldsymbol{z} = [0.3, 0.6, 0.8]$ *et l'espérance des récompenses est* 3.2. *Enfin, nous supprimons l'état* 3. *La solution finale est définie sur* $\mathcal{F}_2 = \{1, 5\}$ *avec une distribution* $[0.6, 0.4]$ *et* $R1b = 2.6$.
- *Retirer l'état* 3 *: l'ensemble* $\mathcal{F}_2$ *est maintenant* $\{1, 2, 4, 5\}$ *et la distribution* $\boldsymbol{d2} = [0.2, 0.2, 0.2, 0.4]$. *Le vecteur* $\boldsymbol{z}$ *est égal à* $[0.2, 0.2, 0.6, 0.4]$ *et la nouvelle valeur de l'espérance des récompenses est* 3.4. *L'étape suivante, nous supprimons l'état* 2 *car nous ne pouvons pas supprimer l'état ÉtatMin* = 1. *À présent,* $\mathcal{F}_2 = \{1, 4, 5\}$,

d2 = $[0.4, 0.2, 0.4]$ *et* $\boldsymbol{z} = [0.4, 0.6, 0.4]$. *L'espérance des récompenses décroît à* 3.2. *Puis, nous supprimons l'état* 5. *Le résultat final est* $\mathcal{F}_2 = \{1, 4\}$, *une distribution de* $[0.4, 0.6]$ *et* $R1c = 2.8$.

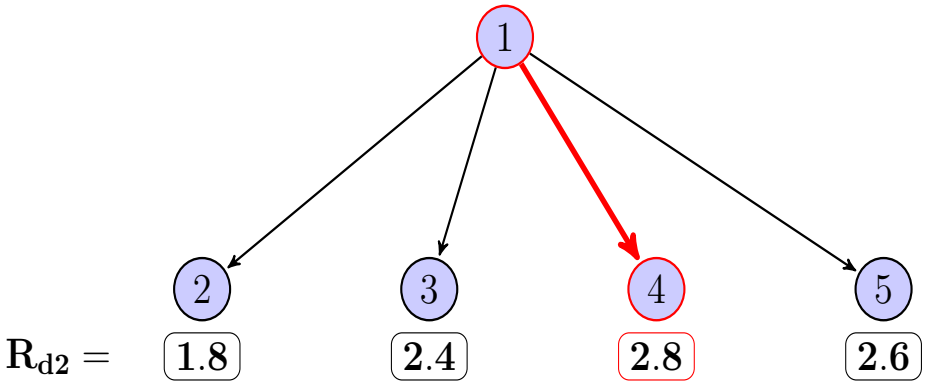


FIGURE 6.2 – Ensembles des états pouvant composer la distribution borne inférieure ***d2*** telle que $|\boldsymbol{d2}| = 2$.

Nous constatons que la meilleure solution est $\mathcal{F}_2 = \{1, 4\}$ *avec une distribution de* $[0.4, 0.6]$ *et une espérance des récompenses égale à* 2.8 *(l'espérance de récompenses* 2.8 *est la plus proche de l'espérance de récompenses initiale* 3.5*). Cette solution donne une différence minimale avec la récompense de* ***d*** *et vérifie la condition d'optimalité. Malheureusement, l'une des solutions trouvées par l'algorithme n'est pas optimale. Ainsi, lorsque le choix de l'indice qui minimise le vecteur* ***z*** *est ambigu, la considération d'une solution arbitraire entre les valeurs renvoyées par l'opération argmin n'est pas optimal.*

Autre point important qui peut également entraîner la non-optimalité de l'algorithme 20 consiste à supprimer deux états voisins (adjacent). Nous allons démontrer ce point à travers le lemme suivant.

Lemme 6.6 *La suppression de deux nœuds adjacents peut générer une différence de récompense plus grande que la somme de deux suppressions indépendantes.*

PREUVE. Supposons que nous supprimons les nœuds x et y de $\mathcal{H}$ pour construire ***d2***. Nous distinguons deux cas possibles :

1. x et y sont adjacents : nous supposons sans perte de généralité que $x = \Gamma^{-}{}_{\mathcal{H}}(y)$ et $v = \Gamma^{-}{}_{\mathcal{H}}(x)$. La différence des espérance de récompenses devient alors :
   $R_{\boldsymbol{d}} - (\boldsymbol{d}[x](\boldsymbol{r}[x] - \boldsymbol{r}[v]) + \boldsymbol{d}[y](\boldsymbol{r}[y] - \boldsymbol{r}[v]))$.
2. La suppression seule de x soustrait $\boldsymbol{d}[x](\boldsymbol{r}[x] - \boldsymbol{r}[v])$ à la récompense de ***d***. De même, la suppression de y seul soustrait $\boldsymbol{d}[y](\boldsymbol{r}[y] - \boldsymbol{r}[x])$ à la récompense de ***d***.

En comparant les résultats, nous pouvons voir que par la suppression de deux nœuds adjacents une quantité supplémentaire de $\boldsymbol{d}[y](\boldsymbol{r}[x] - \boldsymbol{r}[v])$ est retirée de la récompense $R_{\boldsymbol{d}}$. ■

Pour savoir si notre méthode de construction de bornes fournit bien des bornes optimales nous avons développé ci-après un algorithme glouton qui permet de renvoyer si la solution calculée est optimale ou pas et de déterminer une borne stochastique inférieure pour $\boldsymbol{d}$.

Algorithme 21 : Borne inférieure gloutonne (parfois optimale)

1: Soient $\boldsymbol{d2} = \boldsymbol{d}$ et $\boldsymbol{z}[\textit{ÉtatMin}] = \infty$.
2: Pour tous i dans $\mathcal{H}$, sauf *ÉtatMin*, calculer la valeur de $\boldsymbol{d}[i](\boldsymbol{r}[i] - \boldsymbol{r}[\Gamma^{-}{}_{\mathcal{H}}(i)])$ et la stocker dans le vecteur $\boldsymbol{z}$.
3: Soit $\mathcal{F}_2 = \mathcal{H}$ et *SelectionOptimale*= Vrai.
4: Trier le vecteur $\boldsymbol{z}$ dans l'ordre croissant.
5: Initialiser un vecteur ***Mark*** indexé par $\mathcal{F}_2$ à Faux.
6: **Pour** $l = 1$ à $N - K$ **faire**
7: Soit j l'état d'indice l dans $\boldsymbol{z}$.
8: Soient $k = \Gamma^{-}{}_{\mathcal{H}}(j)$ et $m = \Gamma^{-}{}_{\mathcal{F}_2}(j)$.
9: **Si** ***Mark***[j] ou ***Mark***[k] **alors**
10: *SelectionOptimale*=Faux
11: **Fin si**
12: Soit ***Mark***[j]=Vrai ; ***Mark***[k]=Vrai.
13: $\boldsymbol{d2}[m] = \boldsymbol{d2}[m] + \boldsymbol{d2}[j]$.
14: Supprimer l'état j de $\mathcal{F}_2$.
15: **Fin pour**

Nous avons utilisé pour l'algorithme 21 un tableau de booléens qui permet de garder en mémoire les états modifiés au cours des itérations. À chaque étape, deux états sont modifiés : le point retiré et le point qui le précède. En effet, l'état prédécesseur reçoit la probabilité du nœud supprimé et l'additionne à sa précédente masse. La probabilité du point supprimé est quant à elle mise à zéro.

Théorème 6.1 *L'algorithme 21 fournit une distribution avec support $\mathcal{F}_2$ et valeurs **d2**, qui représente une borne inférieure de **d** au sens de l'ordre stochastique fort. La distribution est constituée de K états et est optimale si la valeur de la variable "*SelectionOptimale*" à la fin de l'algorithme, est Vrai. Par ailleurs, la complexité de l'algorithme glouton est $O(NlogN)$ en raison de l'opération de tri.*

PREUVE. La première partie du théorème découle des instructions 8 et 13, où l'on déplace certaines masses de probabilité vers l'état le plus petit. Cette opération permet de définir une distribution borne inférieure selon l'ordre stochastique fort. L'assertion sur la taille est trivial vu que l'on supprime $N - K$ états de N. Montrons maintenant l'optimalité lorsque la condition "*SelectionOptimale*" est vérifiée à la fin de l'algorithme.

Supposons que le booléen "*SelectionOptimale*" est Vrai à la fin de l'algorithme. Cela implique que nous ne supprimons pas de points adjacents dans $\mathcal{H}$. Nous rappelons que le vecteur $\boldsymbol{z}$ est trié selon l'ordre croissant. Ainsi, $R_{\boldsymbol{d2}} = R_{\boldsymbol{d}} - \sum_{l=1}^{N-K} \boldsymbol{z}[l]$. En effet, si "*SelectionOptimale*" est Vrai, alors, à chaque étape de l'algorithme, nous avons $m = k$ et la modification de ***d2*** effectuée par l'instruction 13 implique que nous soustrayons $\boldsymbol{d2}[j](\boldsymbol{r}[j] - \boldsymbol{r}[m])$ de la récompense cumulée $R_{\boldsymbol{d2}}$. Par conséquent, la solution est optimale. ■

Malheureusement, l'algorithme glouton ne donne pas la meilleure borne lorsque le critère d'optimalité de l'algorithme n'est pas vérifié, comme nous pouvons le constater dans l'exemple suivant.

Exemple 6.2 *Soient* $\mathcal{H} = \{1, 2, 3, 4, 5, 6\}$, $\boldsymbol{d} = [0.3, 0.1, 0.1, 0.1, 0.2, 0.2]$ *et une fonction de récompense* $\boldsymbol{r}$ *tel que* $\boldsymbol{r}[i] = \mathcal{H}[i]$ *pour tout* $i \in \{1, \ldots, 6\}$. *La récompense cumulée associée à la distribution* $\boldsymbol{d}$ *est* $R_{\boldsymbol{d}} = 3.4$. *Nous voulons calculer l'approximation la plus proche de* $\boldsymbol{d}$ *définie sur trois états. Le vecteur* $\boldsymbol{z}$ *est égal à* $[\infty, 0.1, 0.1, 0.1, 0.2, 0.2]$. *L'algorithme glouton supprime les états* 2, 3 *et* 4. *Nous obtenons donc* $\mathcal{F}_2 = \{1, 5, 6\}$, $\boldsymbol{d2a} = [0.6, 0.2, 0.2]$, *et* $R_{\boldsymbol{d2a}} = 2.8$. *Cependant, nous pouvons facilement trouver une distribution borne inférieure* $\boldsymbol{d2b} = [0.5, 0.1, 0.4]$ *définie sur* $\mathcal{F}_2 = \{1, 4, 5\}$ *qui est une meilleure borne de* $\boldsymbol{d}$ *et dont la récompense cumulée est égale à* 2.9. *Ainsi, la solution apportée par l'algorithme glouton n'est pas optimale.*

b) Borne supérieure :

La construction de la borne supérieure par la méthode gloutonne est également définie et est retranscrite à travers l'algorithme 22.

Algorithme 22 : Borne supérieure gloutonne (parfois optimale)

1: Soient $\boldsymbol{d1} = \boldsymbol{d}$ et $\boldsymbol{z}[\textit{ÉtatMax}] = \infty$.
2: Pour tous i dans $\mathcal{H}$, sauf *ÉtatMax*, calculer la valeur de $\boldsymbol{d}[i](\boldsymbol{r}[\Gamma^{+}{}_{\mathcal{H}}(i)] - \boldsymbol{r}[i])$ et la stocker dans le vecteur $\boldsymbol{z}$.
3: Soit $\mathcal{F}_1 = \mathcal{H}$ et *SelectionOptimale* = Vrai.
4: Trier le vecteur $\boldsymbol{z}$ dans l'ordre croissant.
5: Initialiser un vecteur ***Mark*** indexé par $\mathcal{F}_1$ à Faux.
6: **Pour** $l = 1$ à $N - K$ **faire**
7: Soit j l'état d'indice l dans $\boldsymbol{z}$.
8: Soient $k = \Gamma^{+}{}_{\mathcal{H}}(j)$ et $m = \Gamma^{+}{}_{\mathcal{F}_1}(j)$.
9: **Si** ***Mark***[j] ou ***Mark***[k] **alors**
10: *SelectionOptimale*=Faux.
11: **Fin si**
12: Soit ***Mark***[j]=Vrai ; ***Mark***[k]=Vrai.
13: $\boldsymbol{d1}[m] = \boldsymbol{d1}[m] + \boldsymbol{d1}[j]$.
14: Supprimer l'état j de $\mathcal{F}_1$.
15: **Fin pour**

Théorème 6.2 *Pour une distribution discrète* $\boldsymbol{d}$, *l'algorithme 22 fournit une distribution borne stochastique supérieure de* $\boldsymbol{d}$ *définie sur* $\mathcal{F}_1$ *et de valeurs* $\boldsymbol{d1}$. *Cette distribution est constituée de* K *états et est optimale si la valeur de la variable "*SelectionOptimale*" est Vrai. Par ailleurs, la complexité de l'algorithme glouton est* $O(NlogN)$.

Exemple 6.3 *En reprenant les données de l'exemple 6.2, nous voulons déterminer la distribution borne supérieure avec* $K = 3$ *états en utilisant l'algorithme 22. Le vecteur* $\boldsymbol{z}$

est égale à $[0.3, 0.1, 0.1, 0.1, 0.2, \infty]$. *Nous retirons donc les états* 2, 3 *et* 4 *de l'ensemble* $\mathcal{F}_1$. *Le nouvel espace d'état est* $\mathcal{F}_1 = \{1, 5, 6\}$, *la distribution borne supérieure associée est* $\boldsymbol{d1} = [0.3, 0.5, 0.2]$ *avec une espérance de récompense de* $R_{\boldsymbol{d1}} = 4$. *Cependant, nous constatons que la variable "SelectionOptimale" renvoyée à la fin de l'algorithme est à Faux, ce qui revient à dire que la solution trouvée n'est pas optimale.*

Comme on a pu le constater, les algorithmes gloutons présentés dans cette partie ne garantissent pas l'optimalité de la borne calculée. Pour ce faire, nous allons présenter dans la section suivante une autre démarche permettant de construire une borne inférieure (resp. supérieure) optimale reposant cette fois-ci sur la programmation dynamique.

6.3.2 Algorithmes optimaux fondés sur la programmation dynamique

Afin de construire des bornes stochastiques optimales pour une distribution initiale $\boldsymbol{d}$, nous avons proposé de transformer notre problème décrit par une distribution discrète en un problème de théorie des graphes (voir figure 6.3). À cet effet, pour une fonction de récompense positive croissante $\boldsymbol{r}$, nous considérons les deux graphes pondérés suivants :

▸ **Calcul de la borne inférieure :** Soit le graphe pondéré $G_2 = (V, E)$ tel que :

- V est l'ensemble des sommets tels que $V = \mathcal{H}$
- E est l'ensemble des arcs tel que $(u, v) \in E$ si et seulement si $u < v \ \forall u, v \in \mathcal{H}$. Le poids de l'arc $e = (u, v)$, noté $\boldsymbol{w}(e)$, est défini comme suit :

$$\boldsymbol{w}(e) = \sum_{j \in \mathcal{H}: u < j} \boldsymbol{d}[j] \left(\boldsymbol{r}[j] - \boldsymbol{r}[u]\right) \qquad \forall\, u \in \mathcal{H}.$$

▸ **Calcul de la borne supérieure :** Soit le graphe pondéré $G_1 = (V, E)$ tel que :

- V est l'ensemble des sommets tels que $V = \mathcal{H}$
- E est l'ensemble des arcs tel que $(u, v) \in E$ si et seulement si $u < v \ \forall u, v \in \mathcal{H}$. Le poids de l'arc $e = (u, v)$, noté $\boldsymbol{w}(e)$, est défini quant à lui par :

$$\boldsymbol{w}(e) = \sum_{j \in \mathcal{H}: j < v} \boldsymbol{d}[j] \left(\boldsymbol{r}[v] - \boldsymbol{r}[j]\right) \qquad \forall\, v \in \mathcal{H}.$$

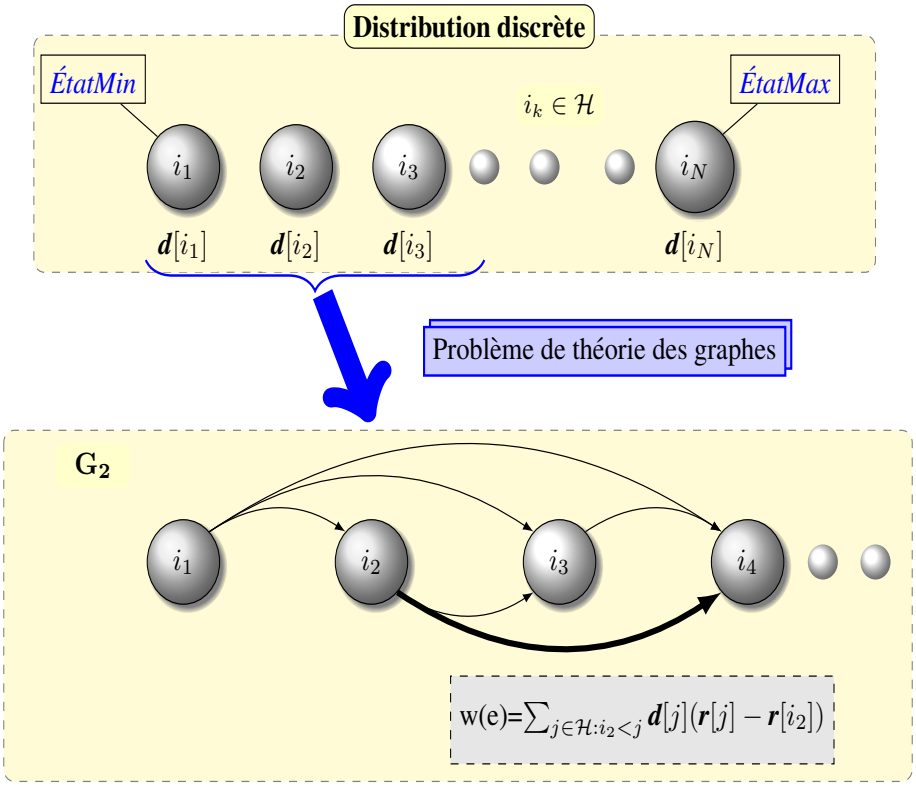


FIGURE 6.3 – Description du problème de détermination d'une borne inférieure à travers un problème de théorie des graphes.

Pour la détermination de la borne inférieure, nous nous concentrons sur certains chemins de l'état *ÉtatMin* à l'état *ÉtatFin* (tel que *ÉtatFin* $\in \mathcal{H}$ et *ÉtatFin* $\leq$ *ÉtatMax*) dans le graphe G_2 (voir figure 6.4).

Lemme 6.7 *Soit $\boldsymbol{d}_{P2}$ une distribution définie sur les états de l'ensemble $\mathcal{F}_2$ tel que $\boldsymbol{d}_{P2} \leq_{st} \boldsymbol{d}$. Le chemin $P2$ de l'état ÉtatMin à l'état ÉtatFin a un poids de :*

$$\sum_{i\in\mathcal{H}} \boldsymbol{r}[i]\,\boldsymbol{d}[i] - \sum_{i\in\mathcal{F}_2} \boldsymbol{r}[i]\,\boldsymbol{d}[i].$$

PREUVE. Soit $\mathcal{F}_2 = \{f_0 = \acute{E}tatMin, f_1, \ldots, f_{|\mathcal{F}_2|} = \acute{E}tatFin\}$. Par définition, le chemin $P2$ contient des arcs $(\acute{E}tatMin, f_1)$, (f_i, f_{i+1}), $(f_{|\mathcal{F}_2|-1}, \acute{E}tatFin)$, pour $i = 1, \ldots, |\mathcal{F}_2| - 2$. Le poids de $P2$ est la somme des poids de tous ces arcs et est égal à :

$$\sum_{i=0,\ldots,|\mathcal{F}_2|-1} \sum_{j\in\mathcal{H}:f_i<j<f_{i+1}} \boldsymbol{d}[j](\boldsymbol{r}[j]-\boldsymbol{r}[f_i]) + \sum_{j\in\mathcal{H}:f_{|\mathcal{F}_2|}<j\leq f_{\acute{E}tatMax}} \boldsymbol{d}[j](\boldsymbol{r}[j]-\boldsymbol{r}[f_i])$$

Après quelques calculs élémentaires, la somme précédente est égale à :

$$\sum_{i\in\mathcal{H}} \boldsymbol{r}[i]\boldsymbol{d}[i] - \sum_{i\in\mathcal{F}_2} \boldsymbol{r}[i]\boldsymbol{d}[i].$$

■

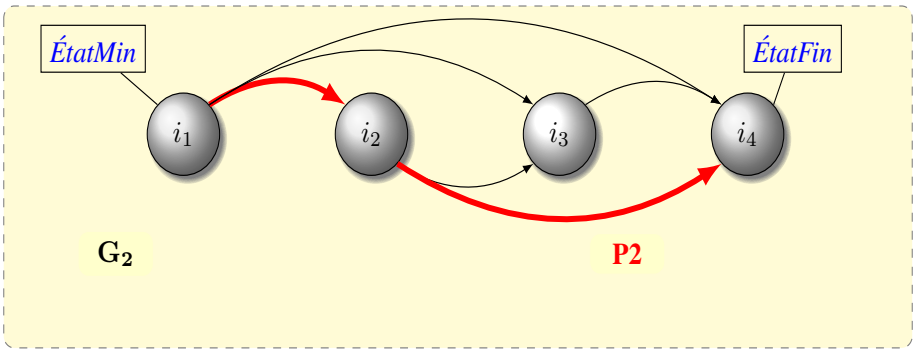


FIGURE 6.4 – Exemple de chemin de l'état *ÉtatMin* à l'état *ÉtatFin* correspondant à une distribution borne inférieure.

À partir de ce lemme, il est facile de remarquer que tout chemin $P2$ de l'état *ÉtatMin* à l'état *ÉtatFin* correspond à une distribution $\boldsymbol{d}_{P2}$. Ainsi, nous pouvons énoncer le lemme suivant qui représente une conséquence directe de ce dernier.

Lemme 6.8 *Soit $P2$ le chemin de l'état ÉtatMin à l'état ÉtatFin à travers tous les éléments de $\mathcal{F}_2$ dans G_2. Il existe une distribution $\boldsymbol{d}_{P2}$ sur les états de ensemble $\mathcal{F}_2$ tel que $\boldsymbol{d}_{P2} \leq_{st} \boldsymbol{d}$ avec un poids :*

$$\sum_{i \in \mathcal{H}} \boldsymbol{r}[i]\boldsymbol{d}[i] - \sum_{i \in \mathcal{F}_2} \boldsymbol{r}[i]\boldsymbol{d}[i].$$

Pour le calcul de la borne supérieure, nous nous intéressons à certains chemins de l'état *ÉtatDébut* (*ÉtatDébut* $\in \mathcal{H}$ et *ÉtatDébut* $\geq$ *ÉtatMin*) à l'état *ÉtatMax* dans le graphe G_1.

Lemme 6.9 *Soit $\boldsymbol{d}_{P1}$ une distribution définie sur les états de l'ensemble $\mathcal{F}_1$ tel que $\boldsymbol{d} \leq \boldsymbol{d}_{P1}$. Le chemin $P1$ de l'état ÉtatDébut à l'état ÉtatMax à travers tous les éléments de $\mathcal{F}_1$ a un poids de :*

$$\sum_{i \in \mathcal{F}_1} \boldsymbol{r}[i]\boldsymbol{d}[i] - \sum_{i \in \mathcal{H}} \boldsymbol{r}[i]\boldsymbol{d}[i].$$

Nous remarquons que la construction d'une borne stochastique optimale équivaut à la détermination du chemin le moins cher avec un nombre d'états égal à K. Cette propriété nous laisse donc entrevoir une démarche intéressante pour déterminer des bornes sur la distribution exacte. Une recherche bibliographique nous amène à énoncer le résultat important suivant :

Théorème 6.3 *[Gérin et Orda] Dans un graphe de taille N (nombre de nœuds), l'algorithme développé par Guérin et Orda dans [GO02] permet de rechercher le chemin de coût minimal pour un nombre de sauts donné K ($K < N$). Cet algorithme repose sur la programmation dynamique et a une complexité de $O(N^2\,K)$.*

Ainsi, en tenant compte de cet algorithme et des résultats théoriques prouvés jusqu'à présent, nous avons développé deux nouveaux algorithmes pour la construction de distributions optimales bornes inférieures et bornes supérieures pour une fonction de récompense positive et croissante $\boldsymbol{r}$.

Plus formellement, pour une distribution initiale $\boldsymbol{d}$ définie sur $\mathcal{H}$ avec $|\mathcal{H}| = N$ états, une récompense positive croissante $\boldsymbol{r}$ et une réduction donnée $K << N$, l'algorithme 23 (resp. l'algorithme 24) permet de fournir une distribution bornante optimale inférieure $\boldsymbol{d2}$ (resp. supérieure $\boldsymbol{d1}$) définie sur le support $\mathcal{F}_2$ ($|\mathcal{F}_2| = K$) tel que $\boldsymbol{d2} \leq_{st} \boldsymbol{d}$ (resp. $\mathcal{F}_1$ ($|\mathcal{F}_1| = K$), tel que $\boldsymbol{d} \leq_{st} \boldsymbol{d1}$).

a) Algorithme de construction de la borne inférieure optimale

La détermination du chemin de coût minimal dans G_2 de l'état *ÉtatMin* à l'état *ÉtatFin* avec K états est effectuée à travers l'algorithme 23.

Algorithme 23 : Algorithme de construction de la borne inférieure optimale de $\boldsymbol{d}$

Entrées : $\boldsymbol{d}$: Distribution de probabilités discrète définie sur l'espace d'état $\mathcal{H}$, $|\mathcal{H}| = N$;
$\boldsymbol{r}$: Récompense positive croissante ;
K : Taille du support $\mathcal{F}_2$ associée à la distribution bornante.

Sorties : $\boldsymbol{d2}$: Borne inférieure optimale de $\boldsymbol{d}$ définie sur $\mathcal{F}_2$, $|\mathcal{F}_2| = K$, ($\boldsymbol{d2} \leq_{st} \boldsymbol{d}$) ;
$R_{\boldsymbol{d2}}$: Espérance des récompenses de $\boldsymbol{d2}$.

1 $R_{\boldsymbol{d}} = \sum_i \boldsymbol{r}[i]\boldsymbol{d}[i]$; /* Calcul de l'espérance des récompense de $\boldsymbol{d}$ */,
2 $\mathbf{X}[i, j] = \mathbf{X}[i, j-1] + \boldsymbol{d}[j](\boldsymbol{r}[j] - \boldsymbol{r}[i])$, $\forall i \in \{2, \ldots, N\}$, $\forall j \in \{i+1, \ldots, N\}$;
3 $\mathbf{T}[i, j] = \infty$, $\forall i, j \in \{1, \ldots, N\}$;
$\mathbf{Save}[i, j] = 0$ et $\mathbf{Save}[i, 2] =$ *ÉtatMin*, $\forall i \in \{1 \cdots N\}, j \in \{1, \cdots N\}$;
4 $\mathbf{T}[i, 2] = X[1, i-1]$, $\forall i \in \{2 \cdots N\}$;
5 $\forall l \in \{3, \ldots, K\}$, $\forall i \in \{2, \ldots, N\}$ et $\forall j \in \{2, \ldots, i-1\}$
if $\mathbf{T}[j, l-1] + \mathbf{X}[j, i-1] < \mathbf{T}[i, l]$ **then**
  $\mathbf{T}[i, l] = \mathbf{T}[j, l-1] + \mathbf{X}[j, i-1]$;
  $\mathbf{Save}[i, l] = f_j$;
6 $\mathbf{T}[i, K] = \mathbf{T}[i, K] + \mathbf{X}[i, N]$, $\forall i \in \{1, \ldots, N-1\}$;
7 mi=argmin$_s\{T[s, K]\}$;/* Indice de l'état ayant le résidu minimal */
8 $\mathcal{F}_2$=$\{f_{mi}\}$; /* Construction de $\mathcal{F}_2$ associé à la distribution **d2** */
9 **for** $l = K$ **to** 2 **do**
  $\mathcal{F}_2 = \mathcal{F}_2 \cup \mathbf{Save}[mi, l]$;
  $mi = \mathbf{Save}[mi, l]$;
10 $\boldsymbol{d2}[m] = \sum_{s \in \{\Gamma^-_{\mathcal{H}}(\mathcal{F}_2[m+1]) \cap \Gamma^+_{\mathcal{H}}(\mathcal{F}_2[m])\} \cup \mathcal{F}_2[m]} \boldsymbol{d}[s]$, $\forall m \in \{1, \ldots, K-1\}$ et
$\boldsymbol{d2}[K] = \sum_{s \in \{\Gamma^+_{\mathcal{H}}(\mathcal{F}_2[K]) \cup \mathcal{F}_2[K]\}} \boldsymbol{d}[s]$;
11 $R_{\boldsymbol{d2}} = R_{\boldsymbol{d}} - \min_s T[s, K]$.

Avec :

- $\mathbf{X}[i, j]$ correspond à la différence de récompenses engendrée par la suppression de l'état f_j sachant que l'état f_i est conservé.
- $\mathbf{T}[i, l]$ correspond au résidu minimal associé au chemin de longueur $l-1$ dans la séquence $\{f_1, \ldots, f_i\}$ tel que l'état f_i est conservé.
- $\mathbf{Save}[i, l]$ correspond au dernier état conservé dans la séquence $\{f_1, \ldots, f_i\}$, pour un chemin de longueur $l-1$.

Notons qu'après avoir défini la matrice des résidus $\mathbf{T}$, la construction de l'espace d'état $\mathcal{F}_2$ associé à la distribution bornante ***d2*** s'effectue à travers l'instruction 8 et 9 de l'algorithme 23. Cette procédure permet de récolter les états gardés durant le parcours jusqu'à l'obtention du chemin ayant un nombre d'états égal à K. La détermination de la distribution de probabilités bornante ***d2*** consiste à regrouper les masses de probabilités des états non conservés sur leurs prédécesseurs immédiats dans $\mathcal{F}_2$.

Théorème 6.4 *L'algorithme 23 construit la distribution borne inférieure optimale avec une complexité $O(N^2K)$, qui est donc cubique quand K est de même ordre que N.*

PREUVE. Conséquence directe l'algorithme de Gérin et Orda (théorème 6.3). ■

Exemple 6.4 *Nous considérons la distribution discrète définie dans l'exemple précédent (exemple 6.2) à savoir* $\boldsymbol{d} = [0.3, 0.1, 0.1, 0.1, 0.2, 0.2]$ *définie sur le support* $\mathcal{H} = \{1, 2, 3, 4, 5, 6\}$ *avec fonction de récompense* $\boldsymbol{r}[i] = \mathcal{H}[i]$ *pour tout* $i \in \{1, \ldots, 6\}$ *et une récompense cumulée* $R_{\boldsymbol{d}} = 3.4$. *La détermination d'une borne inférieure optimale* ***d2*** *de* ***d*** *définie sur* $|\mathcal{F}_2| = 4$ *états par l'algorithme 23 revient à suivre les étapes de calcul suivantes. Les instructions* 1 *à* 6 *de l'algorithme permettent de calculer les matrices suivantes :*

$$\mathbf{X} = \begin{pmatrix} 0 & 0.1000 & 0.3000 & 0.6000 & 1.4000 & 2.4000 \\ 0 & 0 & 0.1000 & 0.3000 & 0.9000 & 1.7000 \\ 0 & 0 & 0 & 0.1000 & 0.5000 & 1.1000 \\ 0 & 0 & 0 & 0 & 0.2000 & 0.6000 \\ 0 & 0 & 0 & 0 & 0 & 0.2000 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix},$$

$$\mathbf{T} = \begin{array}{c} \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} \begin{array}{c} \begin{array}{cccccc} & 2 & 3 & 4 & & \end{array} \\ \begin{pmatrix} \text{Inf} & \text{Inf} & \text{Inf} & \text{Inf} & \text{Inf} & \text{Inf} \\ \text{Inf} & 1.7000 & \text{Inf} & \text{Inf} & \text{Inf} & \text{Inf} \\ \text{Inf} & 1.2000 & 1.1000 & 1.1000 & \text{Inf} & \text{Inf} \\ \text{Inf} & 0.9000 & 0.7000 & 0.6000 & \text{Inf} & \text{Inf} \\ \text{Inf} & 0.8000 & 0.4000 & 0.3000 & \text{Inf} & \text{Inf} \\ \text{Inf} & 1.4000 & 0.5000 & 0.2000 & \text{Inf} & \text{Inf} \end{pmatrix} \end{array} \quad \text{et} \quad \mathbf{Save} = \begin{array}{c} \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} \begin{array}{c} \begin{array}{cccc} 2 & 3 & 4 & \end{array} \\ \begin{pmatrix} / & / & / & / \\ / & 1 & / & / \\ / & 1 & 2 & / \\ / & 1 & 3 & 3 \\ / & 1 & 3 & 3 \\ / & 1 & 4 & 5 \end{pmatrix} \end{array}.$$

Nous rappelons que la matrice $\mathbf{T}$ *(resp.* $\mathbf{Save}$*) représente le résidu engendré (resp. les états composant le chemin) lors du calcul de la borne inférieure définie sur un nombre d'états* $K << N$. *Les colonnes de la matrice* $\mathbf{T}$ *et de la* $\mathbf{Save}$ *correspondent aux différentes longueurs de chemin* $l \in \{1, \ldots, K\}$ *et les lignes définissent les états (nœuds) qui composent le graphe* G. *Après avoir déterminé ces matrices, les instructions* 7 *et* 8 *de l'algorithme 23 vont nous permettre de construire le support* $\mathcal{F}_2$ *associé à la distribution bornante* ***d2***.

Pour un chemin composé de $K = 4$ *états, nous pouvons voir à travers la matrice* $\mathbf{T}$ *que le chemin dont l'état final est* $mi = 6$ *correspond au chemin qui engendre le résidu le plus faible (*$\mathbf{T}[6, 4]$*). Connaissant cet état final, la matrice* $\mathbf{Save}$ *va nous permettre de récolter de proche en proche les autres états qui composent le chemin optimal. Ceci revient*

à déterminer l'extrémité initiale de la transition qui permet le passage vers l'état 6 *(quatrième état composant le chemin), cette dernière est donnée par* $\mathbf{Save}[6, 4] = 5$. *Nous réitérons la démarche en cherchant maintenant l'extrémité initiale de la transition qui permet le passage vers l'état* 5 *(troisième état composant le chemin), et qui est donnée par* $\mathbf{Save}[5, 3] = 3$. *De même,* $\mathbf{Save}[3, 2] = 1$ *permet de récupérer l'état initial de notre chemin. Ainsi, le support associé à* ***d2*** *est composé des états* $\mathcal{F}_2 = \{1, 3, 5, 6\}$.

Enfin, l'instruction 10 *permet de regrouper la masse de probabilités des états supprimés vers leur prédécesseurs immédiats donnant la distribution de probabilités* ***d2*** $= [0.4, 0.2, 0.2, 0.2]$ *avec une espérance de récompenses de* $R_{\boldsymbol{d2}} = 3.2$.

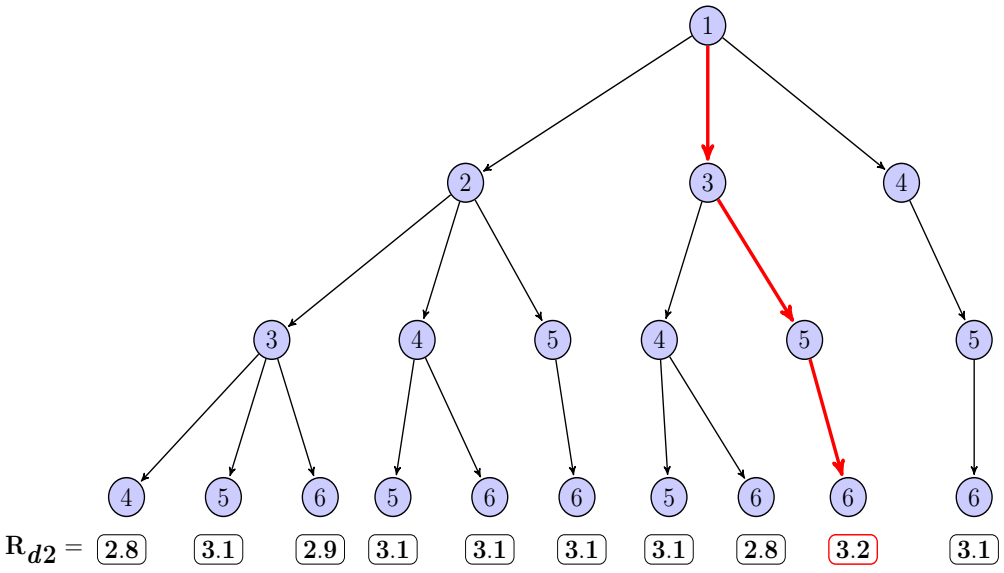


FIGURE 6.5 – Borne inférieure optimale de ***d*** pour $K = 4$ états.

b) Algorithme de construction de la borne supérieure optimale

La détermination du chemin de coût minimal dans G_2 de l'état *ÉtatDébut* à l'état *ÉtatMax* avec K états est effectuée via l'algorithme 24.

Algorithme 24 : Algorithme de construction de la borne supérieure optimale de $\boldsymbol{d}$

Entrées : $\boldsymbol{d}$: Distribution de probabilités discrète définie sur l'espace d'état $\mathcal{H}$, $|\mathcal{H}| = N$; $\boldsymbol{r}$: Récompense positive croissante ;
K : Taille du support $\mathcal{F}_1$ associée à la distribution bornante.

Sorties : $\boldsymbol{d1}$: Borne supérieure optimale de $\boldsymbol{d}$ définie sur $\mathcal{F}_1$, $|\mathcal{F}_1| = K$;
$R_{\boldsymbol{d1}}$: Espérance des récompense de $\boldsymbol{d1}$.

1 $R_{\boldsymbol{d}} = \sum_i \boldsymbol{r}[i]\,\boldsymbol{d}[i]$; /* Calcul de l'espérance des récompenses de **d** */,
2 **for** $i = N$ **to** 1 **do**
  for $j = i - 1$ **to** 1 **do**
    $\mathbf{X}[i, j] = \mathbf{X}[i, j + 1] + \boldsymbol{d}[j]\,(\boldsymbol{r}[i] - \boldsymbol{r}[j])$;
3 $\mathbf{T}[i, j] = \infty, \quad \forall\, i, j \in \{1 \cdots N\}$;
  $\mathbf{Save}[i, j] = 0$ et $\mathbf{Save}[i, 2] =$ *ÉtatMax*, $\forall\, i \in \{1 \cdots N\}, j \in \{1, \cdots N\}$;
4 $\mathbf{T}[i, 2] = X[N, i + 1], \quad \forall i \in \{1 \cdots N - 1\}$;
5 $\forall\, l \in \{3, \ldots, K\}$, $\forall\, i \in \{N - 1, \ldots, 1\}$ et $\forall\, j \in \{N - 1, \ldots, i + 1\}$
  if $\mathbf{T}[j, l - 1] + \mathbf{X}[j, i + 1] < \mathbf{T}[i, l]$ **then**
    $\mathbf{T}[i, l] = \mathbf{T}[j, l - 1] + \mathbf{X}[j, i + 1]$;
    $\mathbf{Save}[i, l] = f_j$;
6 $\mathbf{T}[i, K] = \mathbf{T}[i, K] + \mathbf{X}[i, 1], \quad \forall\, i \in \{2, \ldots N\}$;
7 ma=$\operatorname{argmin}_s\{T[s, K]\}$; /* Indice de l'état ayant le résidu minimal */
8 $\mathcal{F}_1$=$\{f_{ma}\}$; /* Construction de l'espace d'état $\mathcal{F}_1$ associé à la distribution **d1** */
9 **for** $l = K$ **down to** 2 **do**
    $\mathcal{F}_1 = \mathcal{F}_1 \cup \mathbf{Save}[ma, l]$;
    $ma = \mathbf{Save}[ma, l]$;
    $\boldsymbol{d1}[l] = \sum_{s \in \{\Gamma^-_{\mathcal{H}}(\mathcal{F}_1[l]) \cap \Gamma^+_{\mathcal{H}}(\mathcal{F}_1[l-1])\} \cup \mathcal{F}_1[l]} \boldsymbol{d}[s]$.
  $\boldsymbol{d1}[1] = \sum_{s \in \{\Gamma^-_{\mathcal{H}}(\mathcal{F}_1[1]) \cup \mathcal{F}_1[1]\}} \boldsymbol{d}[s]$;
10 $R_{\boldsymbol{d1}} = R_{\boldsymbol{d}} + \mathbf{T}[ma, K]$.

Théorème 6.5 *L'algorithme 24 fournit la distribution supérieure optimale et sa complexité est* $O(N^2K)$.

Exemple 6.5 *En reprenant les données de l'exemple 6.4, nous voulons déterminer la distribution borne supérieure optimale pour une réduction sur un nombre d'états égal à* $K = 4$. *L'application de l'algorithme 24 permet d'obtenir les résultats suivants :*

$$\mathbf{X} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0.3000 & 0 & 0 & 0 & 0 & 0 \\ 0.7000 & 0.1000 & 0 & 0 & 0 & 0 \\ 1.2000 & 0.3000 & 0.1000 & 0 & 0 & 0 \\ 1.8000 & 0.6000 & 0.3000 & 0.1000 & 0 & 0 \\ 2.6000 & 1.1000 & 0.7000 & 0.4000 & 0.2000 & 0 \end{pmatrix},$$

$$\mathbf{T} = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{cccccc} & 2 & 3 & 4 & & \end{array} \\ \begin{pmatrix} \text{Inf} & 1.1000 & 0.5000 & 0.2000 & \text{Inf} & \text{Inf} \\ \text{Inf} & 1.0000 & 0.6000 & 0.4000 & \text{Inf} & \text{Inf} \\ \text{Inf} & 1.1000 & 0.8000 & 0.7000 & \text{Inf} & \text{Inf} \\ \text{Inf} & 1.4000 & 1.2000 & \text{Inf} & \text{Inf} & \text{Inf} \\ \text{Inf} & 1.8000 & \text{Inf} & \text{Inf} & \text{Inf} & \text{Inf} \\ \text{Inf} & \text{Inf} & \text{Inf} & \text{Inf} & \text{Inf} & \text{Inf} \end{pmatrix} \end{array} \quad \text{et} \quad \mathbf{Save} = \begin{pmatrix} / & 6 & 4 & 3 \\ / & 6 & 5 & 4 \\ / & 6 & 5 & 4 \\ / & 6 & 5 & / \\ / & 6 & / & / \\ / & / & / & / \end{pmatrix}.$$

D'après la matrice $\mathbf{T}$*, le chemin de longueur* $K = 4$ *qui minimise la différence des espérances de récompenses a comme point de départ l'état* 1 *(*$mi = \mathbf{T}[1, 4]$*). Ainsi, partant de cet état nous récoltons de proche en proche grâce à la matrice* **Save** *les états qui composent le chemin associé à la distribution borne supérieure. La distribution de probabilités obtenue est* $\boldsymbol{d1} = [0.3, 0.2, 0.3, 0.2]$ *et est définie sur* $\mathcal{F}_1 = \{1, 3, 5, 6\}$ *avec une récompense cumulée* $R_{\boldsymbol{d1}} = 3.6$.

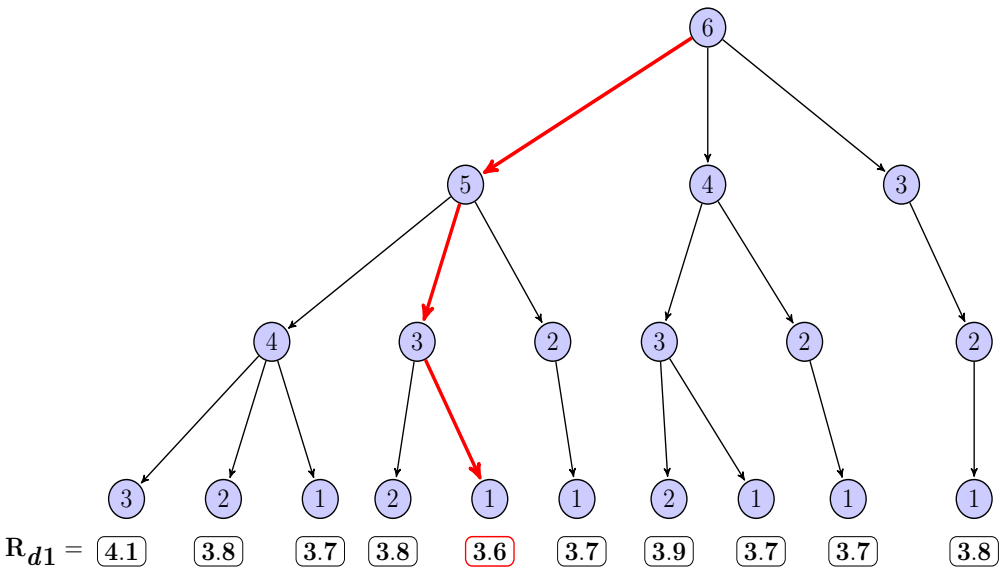


FIGURE 6.6 – Borne supérieure optimale de $\boldsymbol{d}$ pour $K = 4$ états.

Nous avons donné ici différents algorithmes pour construire des distributions discrètes bornantes au sens de l'ordre stochastique fort "*st*". Ces distributions ont un support de taille réduite et peuvent donc représenter un atout non négligeable dans la résolution de certains problèmes difficiles décrit par des variables aléatoires. Ainsi, au lieu d'utiliser les distributions initiales dont la taille s'avère être souvent très grande, nous pouvons utiliser nos distributions bornantes comme paramètres d'entrées et résoudre le système. La figure suivante illustre le procédé utilisé durant l'étape de résolution.

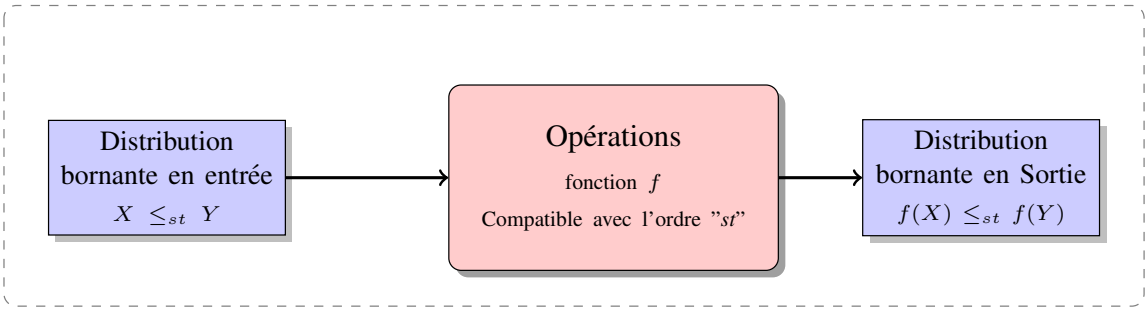


FIGURE 6.7 – Procédure de bornes.

Cette démarche peut être reproduite à chaque étape de calcul pour accélérer le temps de résolution global du système et permet d'assurer et de garantir l'obtention d'une borne en sortie. De plus, le fait d'avoir développé des bornes inférieures et supérieures, va nous permettre d'encadrer de manière pertinente le résultat final. Nous allons montrer ce point via quelques exemples d'application.

6.4 Exemples d'application

Dans le but d'illustrer notre approche, nous proposons d'étudier deux problèmes bien connus de l'évaluation de performance. Le premier problème relève du domaine de la recherche opérationnelle, et consiste à calculer la distribution de la durée d'exécution d'un graphe de tâche stochastique. Nous précisons que notre but ici consiste à présenter notre méthodologie et son application à un exemple bien connu plutôt qu'à une comparaison approfondie des résultats sur le calcul des bornes et des résultats approximatifs pour ce problème particulier. Nous nous référons à [O'C06] et [BJmL93] pour cette comparaison. Le deuxième problème abordé revient à déterminer des bornes stochastiques sur les temps d'attente dans une file FIFO spécifiés par l'équation de Loynes.

6.4.1 Distribution de la durée d'exécution dans un graphe de tâches stochastique

Nous modélisons un programme parallèle décrit par un graphe acyclique et orienté, appelé *graphe de tâches* [PV91]. Les nœuds du graphe représentent les différentes tâches, les durées des tâches sont décrites par des variables aléatoires positives notées $S_i, 1 \leq i \leq n$ mutuellement indépendantes. Les arcs définissent les contraintes de précédence (synchronisation) entre les tâches : il existe un arc de la tâche i à la tâche j si la tâche i doit précéder la tâche j. De plus, une tâche i commence son exécution dès lors que tous ses prédécesseurs immédiats terminent leurs exécutions. Nous supposons que le nœud 1 (nœud source) n'a aucun prédécesseur et que le nœud n (nœud puits) n'a pas de successeur. Soient T_i la date de fin d'exécution de la tâche i et Γ_i^- l'ensemble des prédécesseurs immédiats de cette tâche. Nous savons que :

$$T_i = \max_{j \in \Gamma_i^-} \{T_j\} + S_i \tag{6.1}$$

Comme le graphe est acyclique, les dates de fin d'exécution des tâches peuvent être obtenues de manière consécutive. On note par T_n le temps d'exécution global du graphe. Le calcul de la distribution du temps d'exécution du graphe de tâches avec un nombre infini (suffisamment grand) de processeurs est équivalant au calcul de la distribution des temps de réalisation d'un réseau PERT stochastique qui appartient à la classe de problème $\#P$-complet (voir [O'C06]). La solution exacte ne peut être obtenue que par une énumération exhaustive. Par conséquent, l'analyse précise de ces graphes n'est possible que lorsque la taille du graphe est faible ou lorsque le réseau a une topologie particulière. Ici, nous considérons des graphes de tâches série-parallèle. Pour un tel modèle, les dates de fin d'exécution de tâches peuvent être calculées en appliquant les opérations $(\max, +)$ sur des variables aléatoires indépendantes.

Nous rappelons que pour une variable aléatoire discrète X (resp. Y) prenant ses valeurs dans un ensemble $\mathcal{G}_\mathcal{X}$ (resp. $\mathcal{G}_\mathcal{Y}$) de taille $l_X > 1$ (resp. $l_Y > 1$). La distribution cumulée est notée $F_X(a) = \mathrm{Prob}(X \leq a)$.

Propriété 6.3 *Le calcul de la distribution maximum de deux variables aléatoires indépendantes nécessite $O(l_X + l_Y)$ opérations et au plus $l_X + l_Y - 1$ états pour la distribution résultante. La convolution de deux variables aléatoires indépendantes nécessite $O(l_X \times l_Y)$ operations et au plus $l_X \times l_Y$ états en utilisant une approche naïve et $O((l_X + l_Y)log(l_X + l_Y))$ en utilisant la transformé de Fourier rapide (FFT) [Rob92].*

La première assertion provient d'une propriété bien connue : pour tout $i \in \mathcal{G}_\mathcal{X} \cup \mathcal{G}_\mathcal{Y}$, $F_{\max\{X,Y\}}[i] = F_X[i]\,F_Y[i]$. En outre, l'addition de deux variables aléatoires est associée à la convolution des distributions :

$$F_{X+Y}[i] = \sum_{j \in \mathcal{G}_\mathcal{X}} \mathrm{Prob}(X = j) \sum_{c \in \mathcal{G}_\mathcal{Y} | \ell + j = i} \mathrm{Prob}(Y = \ell).$$

Si nous considérons par exemple un graphe série-parallèle avec trois branches et m tâches dans chaque branche (voir figure 6.8), si le temps d'exécution de chaque tâche est une distribution discrète de taille p, alors la distribution de la durée d'exécution d'une branche peut être au pire cas de taille p^m. La distribution des durées d'exécution dans ce cas est calculée comme suit :

$$\max_{1 \leq j \leq 3} \sum_{i=1}^{m} S_i^j.$$

où la distribution S_i^j correspond à la durée de la tâche i se trouvant à la j-ième branche.

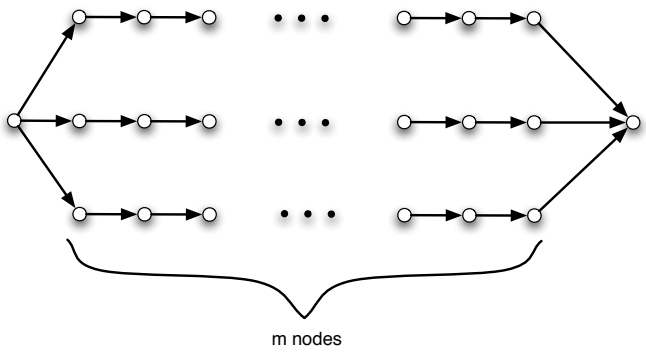


FIGURE 6.8 – Exemple de graphe de tâche.

Nous pouvons conclure que la taille de la distribution peut augmenter de façon exponentielle avec le nombre de nœuds. Ainsi, afin de pallier ce problème, nous proposons d'utiliser les différents algorithmes de bornes développés dans ce chapitre et réduire la taille des distributions à chaque étape de calcul. Dans cette optique, nous devons d'abord montrer que le fait de remplacer les paramètres d'entrée des opérations $(\max, +)$ par des distributions bornantes ayant des supports de tailles plus petites permet de garantir l'obtention de bornes sur les résultats finaux.

Nous notons que les graphes de tâches constituent un cas particulier de graphes d'événements temporisés stochastiques qui ont été largement étudiées dans le chapitre 8 de [BCOQ92]. Nous nous référons donc à ce dernier pour énoncer le résultat de monotonie des opérations $(\max, +)$ suivant :

Propriété 6.4 *Dans l'équation 6.1, si* $\forall j \in \Gamma_i^-, \bar{T}_j \leq_{st} T_j$ *et* $\bar{S}_i \leq_{st} S_i$*, alors*

$$\bar{T}_i = \max_{j \in \Gamma_i^-} \{\bar{T}_j\} + \bar{S}_i \leq_{st} T_i.$$

Ainsi, l'utilisation des bornes stochastiques après chaque opération nous permet de garantir l'obtention de bornes sur les distributions et le temps d'exécution global du graphe de tâches. Nous proposons donc d'appliquer les algorithmes de bornes stochastiques développés dans ce chapitre pour définir des bornes sur le temps d'exécution global du graphe et contrôler la taille du support des distributions. Après chaque opération $(\max, +)$ nous appliquons soit l'algorithme glouton donné dans la section 6.3.1 ou l'algorithme optimal présenté à la section 6.3.2 pour borner la taille de la distribution résultante pour un seuil K donné. Cette démarche est résumée dans l'algorithme suivant.

Algorithme 25 : Bornes stochastique sur les opérations $\max\{X, Y\}$ ou $X + Y$

ENTRÉES : X et Y : deux distributions discrètes ; K : la taille de la réduction ;

1: **Variables :** $\text{op} \in \{\max, +\}$ et $\text{Alg} \in \{\text{glouton}, \text{optimal}\}$: application d'un algorithme de borne.
2: Calculer $Z = \text{op}(X, Y)$;
3: **Si** $|Z| > K$ **alors**
4: Appliquer Alg pour construire la distribution bornante de taille K.
5: **Fin si**

Nous considérons un graphe de tâches série-parallèle composé de b branches avec m tâches dans chaque branche (figure 6.8). La durée d'exécution d'une tâche est une variable aléatoire discrète de taille 4 et ces valeurs sont choisies aléatoirement dans l'intervalle des réels $[0, 10[$. Le temps d'exécution global du graphe de tâches peut être calculé en appliquant les opérations $(\max, +)$ sur des variables aléatoires indépendantes comme suit : $S_1 + \max_{1 \leq j \leq b}\{L_j\} + S_{(j-1).m+2}$, où le temps d'exécution de la j-ième branche est $L_j = \sum_{i=(j-1)m+2}^{jm+1} S_i$.

m	L	T	R_d
4	12160	0.7383	37.1455
5	46256	7.8542	43.3317
6	188416	415.1603	46.3308
7	785504	8.3653 10^3	46.5201
8	2974896	2.4244 10^5	56.1796

TABLE 6.1 – Résultats exacts

Dans le tableau 6.1, nous donnons les résultats exacts pour des graphes de tâches avec $b = 3$ branches et m varié. La colonne L représente la taille (c.-à-d. le nombre de valeurs) de la distribution finale, la colonne T contient le temps de calcul en secondes, et la colonne R_d contient la récompense calculée : la valeur moyenne de la durée d'exécution globale des tâches.

Le tableau 6.2 contient les valeurs bornes inférieures correspondantes pour $K = 25$ et $K = 50$. Nous présentons les résultats de l'algorithme glouton et l'algorithme de programmation dynamique.

		Glouton		Optimal-(local)	
m	K	T	R_{d2}	T	R_{d2}
4	25	0.1125	35.6090	0.5781	36.3648
	50	0.1705	36.5403	3.7996	36.8294
5	25	0.1412	41.4151	0.8191	42.2156
	50	0.2484	42.5091	6.0513	42.8496
6	25	0.1793	43.6972	1.0872	45.0021
	50	0.3083	45.0599	8.1150	45.7225
7	25	0.2134	42.9925	1.3683	44.7492
	50	0.3697	45.0117	10.1021	45.7387
8	25	0.2552	52.2219	1.4004	53.9880
	50	10.5801	54.1708	13.4566	55.0026
	100	1.1274	55.0394	94.0466	55.5080

TABLE 6.2 – Bornes inférieures

De toute évidence, les algorithmes de bornes sont beaucoup plus rapides que le calcul exact. Le temps de calcul associé à l'algorithme glouton n'augmente pas sensiblement avec la taille du graphe tandis qu'il augmente avec l'algorithme optimal local, mais il reste largement plus petit que le temps exact de calcul. La précision des bornes est améliorée pour les grandes valeurs de K, et pour $K = 50$ la qualité des bornes (glouton et optimal local) sont comparables et ils sont tous les deux précis.

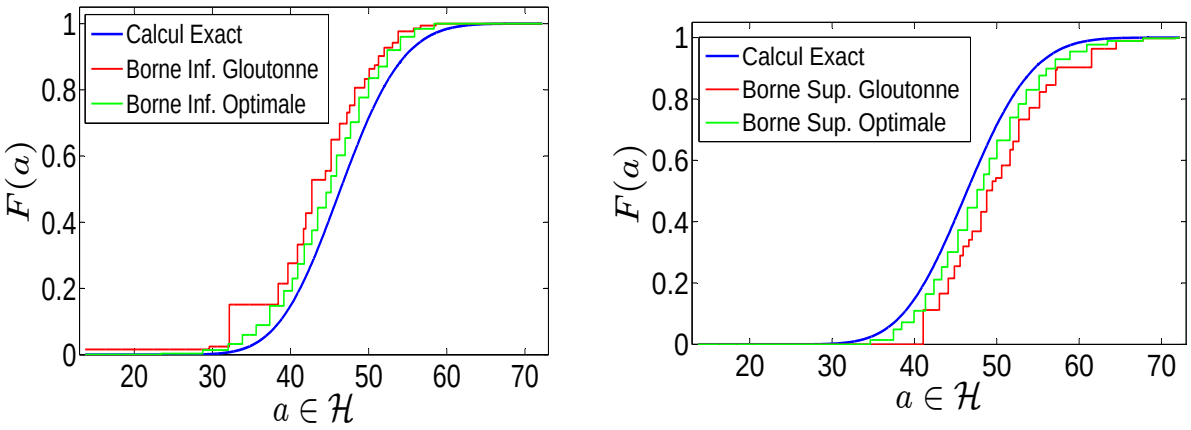


FIGURE 6.9 – Bornes inférieures (à gauche) et supérieures (à droite) des distributions cumulées pour $m = 7$ et $K = 25$

Nous tenons à souligner ici que l'algorithme de programmation dynamique fournit une borne optimale pour chaque résultat intermédiaire, et nous n'avons pas de preuve que l'utilisation itérée de cette méthode est suffisante pour obtenir la meilleure borne sur K états de la distribution exacte. C'est la raison pour laquelle nous désignons cette méthode comme une borne localement optimale.

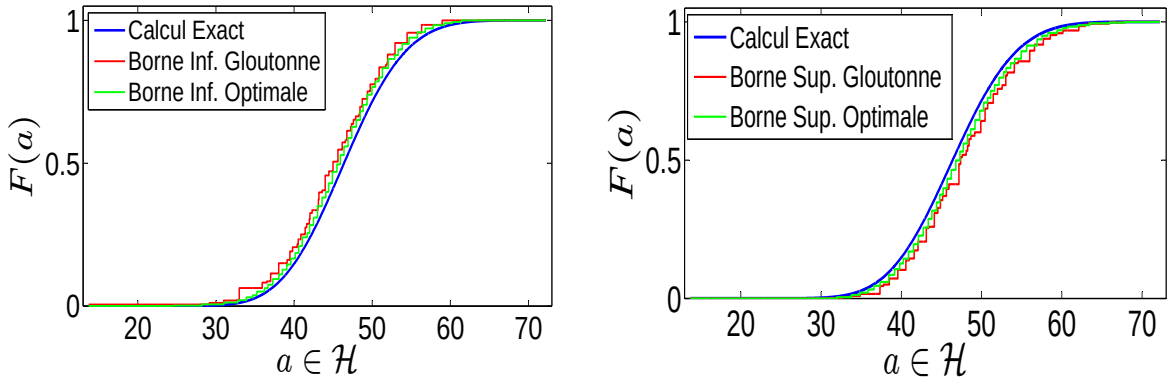


FIGURE 6.10 – Bornes inférieures (à gauche) et supérieures (à droite) des distributions cumulées pour $m = 7$ et $K = 50$

6.4.2 Bornes sur les histogrammes et équation de Loynes pour l'analyse rapide d'une file d'attente FIFO

Nous considérons une file d'attente à un serveur gérée selon la discipline de service FIFO (premier arrivé premier servi) avec des temps d'inter-arrivée $T(n)$ et les durées de service $S(n)$ qui sont des variables aléatoires indépendantes. Les arrivées des clients dans le système sont supposées stationnaires. La variable aléatoire $W(n)$ désigne le temps d'attente du n-ième

client arrivé dans le système, qui satisfait l'équation de Loynes [Loy62] :

$$W(n+1) = \max\left(0,\, W(n) + (S(n) - T(n))\right); \tag{6.2}$$

tel que $T(n)$ correspond à la durée entre la n-ième et la $(n+1)$-ième arrivées et S_n représente le temps de service du n-ième client.

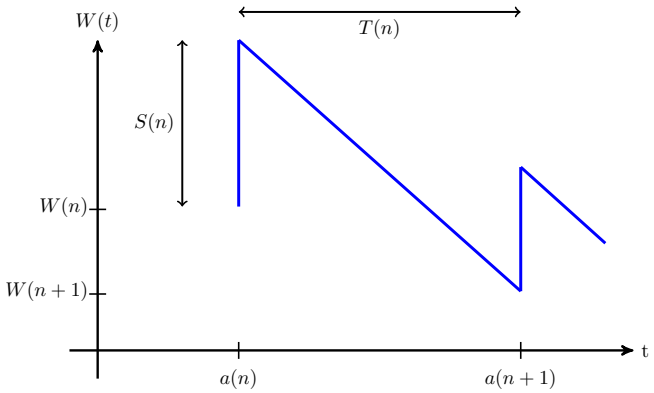


FIGURE 6.11 – Temps d'attente - Cas FFIO -.

Comme le service est effectué dans l'ordre d'arrivée des clients, nous notons que le client arrivant à l'instant "1" commence directement son service et n'a par conséquent pas besoin d'attendre, $W(1) = 0$. Cependant, les clients suivants devront attendre dans le cas où ils arrivent avant que le client précédent n'ait été servi.

Nous voulons montrer à travers cet exemple comment accélérer l'analyse d'une file d'attente FIFO dont les paramètres proviendraient de traces de trafic réelles. Pour ce faire, nous considérons les trois applications suivantes :

1. **Application 1** : Sous l'hypothèse de stationnarité, nous réduisons les distributions empiriques des interarrivées et des services, pour les transitoires et le stationnaire,
2. **Application 2** : Sans utiliser l'hypothèse de stationnarité, nous utilisons la même réduction, pour les transitoires,
3. **Application 3** : Sans utiliser l'hypothèse de stationnarité, nous réduisons également la taille des distributions transitoires.

L'équation de Loynes 6.2 montre que la distribution des attentes s'obtient à partir de la convolée des distributions des interarrivées et des services. Ainsi, pour faciliter l'analyse, nous simplifions les distributions empiriques en garantissant que nous construisons des bornes stochastiques sur ces distributions.

Avant de présenter les trois applications investiguées, nous établissons d'abord la notion de monotonie des opérateurs utilisés dans l'équation de Loynes (opérateurs $(\max, +)$) pour garantir l'obtention de bornes stochastiques sur la distribution des temps d'attente à partir des bornes sur les processus empiriques.

Propriété 6.5 (Monotonie stochastique) *Si* $T(n) \leq_{st} \overline{T}(n)$ *et* $\underline{S}(n) \leq_{st} S(n)$, *alors* $W(n+1) \leq_{st} \overline{W}(n+1)$.

PREUVE. Par récurrence sur n, le fait d'avoir l'opérateur $\max$ qui est une fonction croissante, nous permet d'exploiter le théorème 4.3.9 de Stoyan [MS02]. Ainsi, si on remplace la distribution $T(n)$ par une autre v.a. plus grande au sens de l'ordre "st" et la distribution $S(n)$ par une v.a. plus petite au sens de l'ordre st, alors la distribution du temps d'attente $W(n+1)$ va augmenter stochastiquement. ■

Symétriquement, nous pouvons définir une borne inférieure sur les temps d'attente.

Propriété 6.6 (Monotonie stochastique) *Si* $\underline{T}(n) \leq_{st} T(n)$ *et* $S(n) \leq_{st} \overline{S}(n)$, *alors* $\underline{W}(n+1) \leq_{st} W(n+1)$.

PREUVE. Identique à celle de la propriété 6.5. ■

Afin d'illustrer notre approche, nous avons utilisé une trace de trafic réelle [SC00]. Plus précisément, nous avons pris la trace de trafic MAWI correspond à une heure de mesure de trafic IP réalisée le 14 mai 1999 entre 11 h et 12 h d'un lien américano-japonais. Cette trace de trafic a un nombre de paquets de 6.016.846 d'une taille globale de $3,6$ Gbytes et un taux moyen de $8,43$ Mb/s. Nous précisons que ces traces de trafic MAWI sont données en format brut tcpdump (soit des traces de près de 9 Gbytes), nous avons considéré le format distillé correspondant à un fichier simple qui contient les instants d'arrivées (en millisecondes) et la taille (en bytes) de tous les paquets transmis pendant cette heure.

Pour l'étude numérique, nous avons besoin d'utiliser les mesures réelles des durées de service des paquets. Cependant, à partir de la trace de trafic MAWI seule les données relatives à la distribution de la taille des paquets sont données. En effet, dans [SC00] les données sur la taille de paquet sont données en bytes, nous devons donc convertir la taille des paquets et les exprimer par les temps de service correspondants. Ici, nous supposons que la capacité des liens est de 128×9 Kbps [Lie] pour un $\Delta_t = 10^{-6}$ s, ce qui revient à dire que les liens de 128 bytes sont desservis à chaque slot de longueur Δ_t. Dans ce cas, la charge du système est de 0.86. La distribution des temps d'inter-Arrivée et des durées de service sont données dans la figure 6.12.

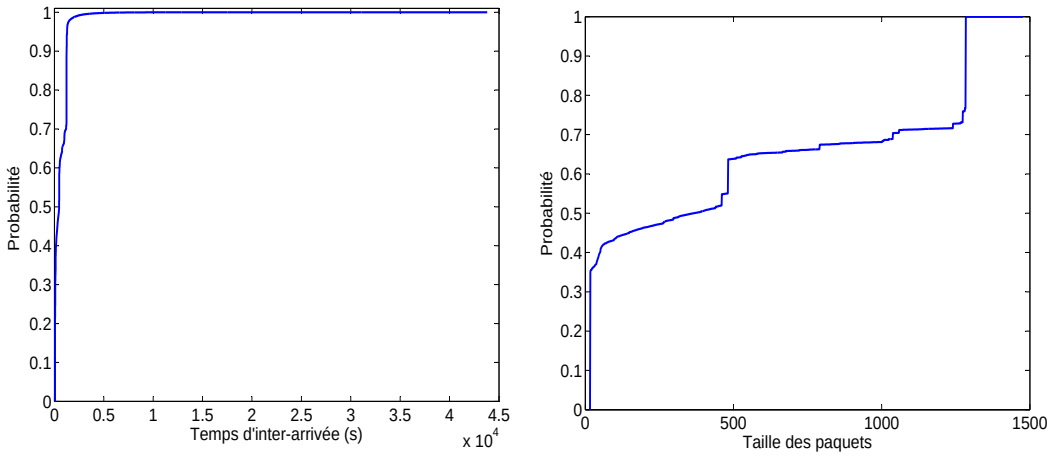


FIGURE 6.12 – Distributions cumulées des temps d'interarrivées (à gauche) et des temps de service(à droite) associée à la trace MAWI.

Nous notons que les tailles des distributions des temps d'interarrivées et de service sont respectivement égales à 9871 et 1273 états.

Dans le but d'accélérer le temps de calcul, nous proposons d'utiliser nos algorithmes de bornes et construire les distributions stochastiques bornes inférieures et bornes supérieures de la distribution des temps d'attente. Pour ce faire, nous proposons de décrire les trois applications possibles citées précédemment.

a) Application 1 :

Sous l'hypothèse de stationnarité des distributions des inter-arrivées et des temps de services, nous calculons initialement la distribution borne stochastique supérieure $\overline{T}$ et la distribution borne stochastique inférieure $\underline{S}$ tel que $T(n) \leq_{st} \overline{T}$ et $\underline{S} \leq_{st} S(n)$ pour tout n.

À chaque date n nous avons :

$$\overline{W}(n+1) = \max\left(0, \overline{W}(n) + \overline{T} - \underline{S}\right). \tag{6.3}$$

Le fait de réduire la taille du support de T et de S va nous permettre de simplifier les calculs et accélérer les temps d'execution.

De manière symétrique, nous pouvons également obtenir la borne inférieure des temps d'attente.

$$\underline{W}(n+1) = \max\left(0, W(n) + \underline{T} - \overline{S}\right). \tag{6.4}$$

Nous pouvons également tester l'existence d'un équilibre stationnaire pour la borne supérieure.

En reprenant l'exemple cité précédemment, nous proposons de traiter ce cas et calculer les distributions des temps d'attente. Nous considérons à cet effet les deux réductions suivantes : 20 et 100 états appliquées au processus d'inter-arrivée et au processus de service. Les distributions résultantes sont illustrées dans les figures suivantes.

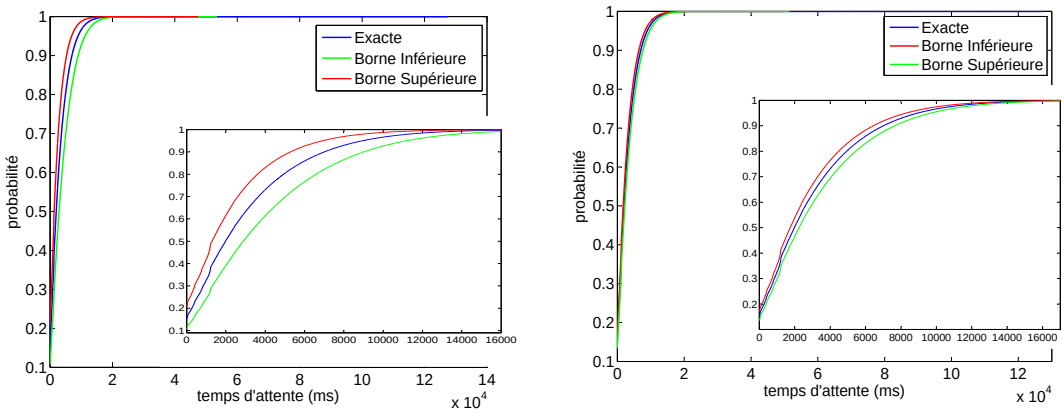


FIGURE 6.13 – Distribution cumulée des temps d'attente associée au 100-ième client pour un nombre de bins égal à 20 (à gauche) et bins égal à 100 (à droite).

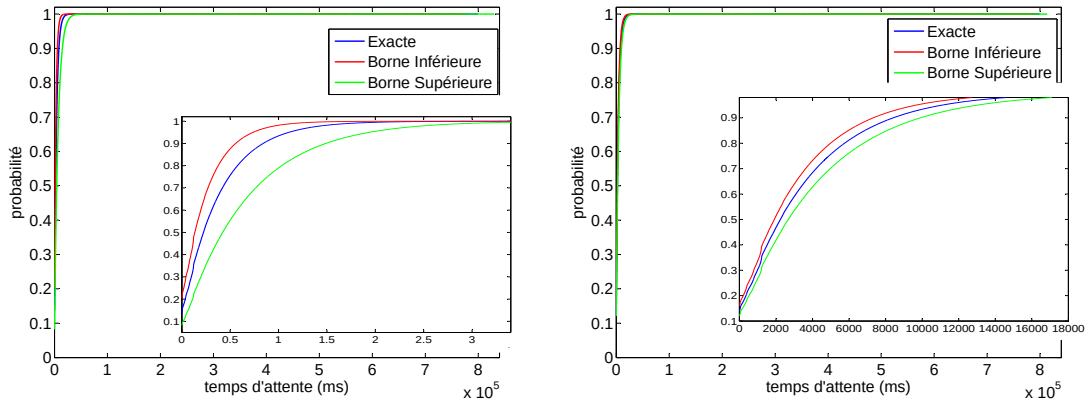


FIGURE 6.14 – Distribution cumulée des temps d'attente associée au 1000-ième client pour un nombre de bins égal à 20 (à gauche) et bins égal à 100 (à droite).

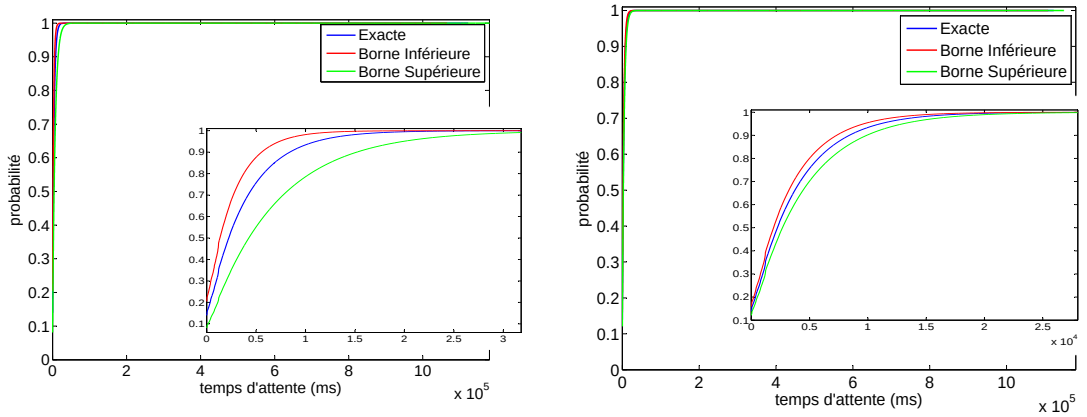


FIGURE 6.15 – Distribution cumulée des temps d'attente associée au 2000-ième client pour un nombre de bins égal à 20 (à gauche) et bins égal à 100 (à droite).

		bins	20		100	
		Exact	Borne inf.	Borne sup.	Borne inf.	Borne sup.
n-ième client	100	69	8.2	8.0	16.6	15.9
	1000	7847	17.9	18	66.4	63.4
	2000	19140	38.5	38.6	170	165.9

TABLE 6.3 – Temps d'exécution (en secondes) nécessaire au calcul de la distribution des temps d'attente.

Nous remarquons que pour une réduction sur 20 bins, notre méthode permet de construire des bornes assez pertinentes qui deviennent encore meilleures quand on augmente le nombre

d'états. Pour un nombre de bins égal à 100, nos résultats sont très proches de la distribution exacte et montrent de manière claire l'intérêt de notre approche. De plus, d'après le tableau 6.3, il est aisé de constater que notre méthode accélère de manière assez sensible le calcul de mesures de performance d'un modèle.

b) Application 2 :

Dans ce cas, il n'est pas nécessaire que les distributions $T(n)$ et $S(n)$ soient stationnaires pour faire le calcul itératif pour toutes les dates. L'itération numérique est effectuée avec des bornes différentes à chaque étape de calcul.

Ainsi, si nous construisons une distribution $\overline{T}(n)$ et une distribution $\underline{S}(n)$ tel que $T(n) \leq_{st} \overline{T}(n)$ et $\underline{S}(n) \leq_{st} S(n)$ pour tout n, nous avons :

$$\overline{W}(n+1) = \max(0, \overline{W}(n) + \overline{T}(n) - \underline{S}(n)). \tag{6.5}$$

De même, si nous définissons une distribution $\underline{T}(n)$ et une distribution $\overline{S}(n)$ tel que $\underline{T}(n) \leq_{st} T(n)$ et $S(n) \leq_{st} \overline{S}(n)$ pour tout n, nous avons :

$$\underline{W}(n+1) = \max(0, \underline{W}(n) + \underline{T}(n) - \overline{S}(n)). \tag{6.6}$$

c) Application 3 :

En n'utilisant toujours pas l'hypothèse de stationnarité des processus $T(n)$ et $S(n)$, nous proposons ici de réduire la taille de la distribution $W(n)$ après chaque calcul. Ceci va nous permettre de limiter la complexité et accélérer le temps de calcul. Les équations suivantes résument le procédé utilisé pour calculer la borne supérieure et la borne inférieure des temps d'attente.

$$\overline{W}(n+1) = \max\left(0, \mathrm{Reduction}(\overline{W}(n)) + \overline{T}(n) - \underline{S}(n)\right). \tag{6.7}$$

$$\underline{W}(n+1) = \max\left(0, \mathrm{Reduction}(\underline{W}(n)) + \underline{T}(n) - \overline{S}(n)\right). \tag{6.8}$$

Où $\mathrm{Reduction}$ est une des opérations de contraction des distributions citées précédemment.

Afin de déterminer les temps d'attente définis dans ce cas, nous allons utiliser deux algorithmes distincts pour calculer les bornes sur les distributions. La première réduction est effectuée sur les distributions d'inter-arrivée et de service en utilisant les algorithmes optimaux. La seconde réduction intervient après chaque calcul de la distribution d'attente $\overline{W}(n)$ (resp. $\underline{W}(n)$) en utilisant cette fois-ci l'algorithme glouton. Sachant que l'algorithme optimal est de complexité cubique, le fait d'utiliser cet algorithme à chaque étape de calcul peut engendrer un temps de résolution global important, ce qui n'est pas le but recherché. Notre choix se dirige donc vers l'utilisation de l'algorithme glouton dont la complexité est moindre $O(NlogN)$. Nous notons cependant que pour obtenir des bornes pertinentes avec l'algorithme glouton, nous devons considérer une réduction sur un nombre d'états plus important ($K > 100$). En effet, l'algorithme glouton donne lieu à de mauvaises bornes quand la réduction est faible. Nous avons pour ce faire considéré une réduction sur un nombre d'états égal à 2000 (jugée satisfaisante).

Dans cet exemple, nous proposons d'effectuer une réduction de 100 bins sur les distribution $T(n)$ et $S(n)$. À chaque itération si la taille du support de $\overline{W}(n)$ (resp. $\underline{W}(n)$) est supérieure à un seuil de 2000 états, nous appliquons l'algorithme glouton sur un nombre de bins égal à ce même seuil (2000 bins). La figure suivante illustre les résultats obtenus.

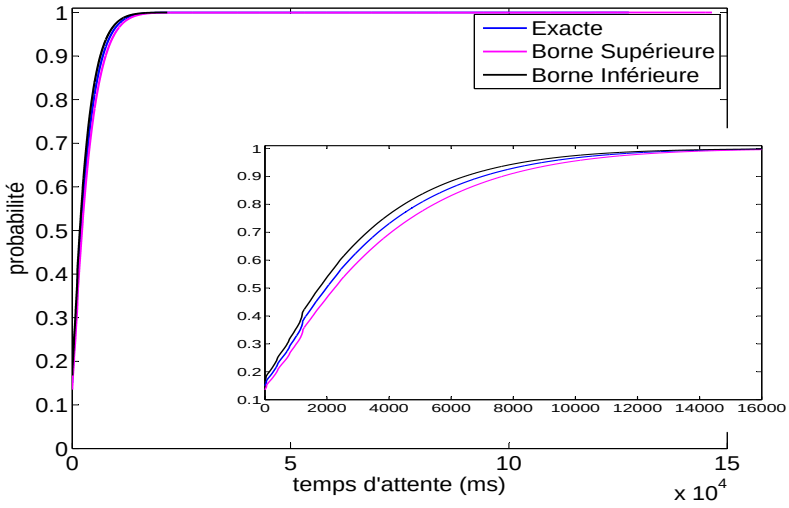


FIGURE 6.16 – Distribution cumulée des temps d'attente associée au 100-ième client.

Le calcul de la distribution borne supérieure et de la distribution borne inférieure du temps d'attente du client 100 a nécessité respectivement un temps de calcul de 30 s et de 19 s. Comme nous pouvons le voir dans la figure 6.16, notre méthode donnent lieu à des bornes très précises avec des temps de calcul qui demeurent nettement inférieurs au temps de calcul exact (69s).

6.5 Conclusion

L'approche de bornes stochastiques développée dans ce chapitre montre que notre méthode offre un compromis intéressant entre la précision et la vitesse de calcul. En effet, selon le choix de la réduction apporté à la taille des distributions, nous pouvons déterminer des bornes pertinentes et un encadrement très fin du résultat exact en des temps relativement rapides. De plus, l'utilisation de la théorie des bornes stochastiques nous permet d'assurer que les résultats calculés représentent bien des bornes de la distribution exacte. Nous avons illustré notre approche par l'étude de deux exemples bien distincts qui ont permis de montrer et d'observer tout l'impact et l'intérêt de notre approche. De manière générale, les opérations de contraction monotones des distributions discrètes constituent un moyen d'obtenir des bornes stochastiques à moindre coût pour des problèmes stochastiquement monotones.

Nous émettons également une remarque concernant l'utilisation de l'algorithme glouton. En effet, malgré que cet algorithme ne garantisse pas l'optimalité sur les bornes calculées pour une fonction de récompense donnée, le fait qu'il soit plus rapide (de complexité $O(NlogN)$) que l'algorithme optimal (de complexité $O(N^2K)$) représente un avantage non négligeable. Il serait donc tout à fait logique de vouloir s'orienter vers l'utilisation de ce dernier. Cependant,

nous avons noté que pour une distribution initiale définie sur un support très grand, l'algorithme glouton ne permet de fournir des bornes intéressantes que dans le cas où la réduction n'est pas faible (≥ 1000 par exemple). Sinon, il donne lieu à de très mauvaises bornes. Ce constat n'est contrairement pas observé lors de l'utilisation de l'algorithme optimal. En effet, une réduction à un nombre d'états de 10 voir 100 permet largement à l'algorithme optimal de déterminer des bornes relativement pertinentes. Ce point important nous a donc amenés à considérer cet algorithme dans la suite de notre étude. Cependant, comme perspective, il serait intéressant d'étudier le comportement de l'algorithme glouton et voir quelles modifications apporter pour assurer l'obtention de bornes précises sans augmenter pour autant sa complexité.

Nous allons nous intéresser à présent à l'application de notre nouvelle méthode de bornes à l'étude et l'analyse de problèmes d'évaluation des performances des files d'attente et des réseaux. Fondées sur la discrétisation des mesures réelles du processus en entrée (ou de service), nous voulons appliquer nos algorithmes de bornes stochastiques aux processus d'arrivée et/ou de service et tenter de dériver des bornes sur certaines mesures de performance du système. Cette étude fera l'objet des chapitres qui suivent.

Chapitre 7

Analyse d'une file d'attente simple

Dans ce chapitre, nous étendons l'application de notre approche basée sur des histogrammes bornants à l'étude de systèmes de files d'attente dont les entrées correspondent à des trafics généraux provenant de traces réelles. Nous considérons une file d'attente simple et nous proposons d'étudier l'influence et la pertinence de notre approche sur l'analyse de performances de ce système.

L'objectif est de déterminer des histogrammes bornants de l'histogramme initial obtenu à partir des traces de trafics réelles. De plus, intuitivement, le fait de paramétrer la taille des histogrammes bornants va impacter à la fois la complexité et la qualité des calculs. Ainsi, en posant comme hypothèse la stationnarité du trafic en entrée, nous allons montrer à travers des résultats théoriques et des analyses expérimentales la véracité de cette assertion, en mettant en évidence le compromis intéressant qu'offre notre méthode entre la précision des résultats et les temps de calcul.

Notre démarche consiste à construire, grâce aux algorithmes optimaux développés dans le chapitre précédent, deux distributions discrètes de support réduit représentant les bornes stochastiques inférieure et supérieure de la distribution en entrée. Puis, de dériver par la théorie de la comparaison stochastique des bornes inférieure et supérieure sur les mesures de performance (longueur moyenne du tampon, probabilité de blocage...). Un autre point important également traité dans ce chapitre, consiste à développer un nouveau test de convergence que nous avons prouvé et qui est nécessaire à la résolution numérique. En effet, le calcul itéré de la distribution stationnaire jusqu'à obtention de deux distributions successives assez proches au sens d'une norme, comme utilisé par Hernández et *al.* dans [HOVC07a, HOVC10] pour le processus HBSP, ne représente pas une preuve de convergence (on peut se référer à l'ouvrage de Stewart [Ste95] pour une discussion détaillée sur la convergence des itérations pour les chaînes de Markov). De plus, par rapport aux travaux développés par Hernàndez et *al.* dans [HOVC07a, HOVC07b, HOVC09, HOVC10] sur leur méthode de réduction d'histogrammes pour des systèmes de files d'attente prenant en entrée des traces de trafic réelles, nous proposons de mener une étude comparative des travaux en question avec notre approche de bornes stochastiques. Nous rappelons qu'il y a deux sources d'approximation dans le processus HBSP proposé par Hernàndez : la première lorsqu'on agrège à l'aveugle la trace, la seconde quand on effectue les calculs numériques.

Nous commençons ce chapitre par décrire l'analyse d'une file d'attente isolée à temps

discret recevant un trafic modélisé par un histogramme. Nous donnons des conditions pour que la chaîne soit ergodique puis nous démontrons le test de convergence. Dans la section 7.2, nous montrons que la chaîne de Markov modélisant l'élément de réseau FIFO avec une capacité limitée du tampon, est stochastiquement monotone, ce qui permet de calculer et de garantir des bornes sur la longueur de la file d'attente et sur le processus de sortie. De plus, l'application des algorithmes de bornes développés dans le chapitre précédent, pour définir des distributions bornantes optimales (de tailles réduites) sur l'histogramme du trafic en entrée nous permet de contrôler la précision et la complexité de l'analyse en modifiant la taille des histogrammes dans ces algorithmes. Enfin, pour montrer l'intérêt de notre approche et l'influence du nombre d'états considérés, nous présentons dans la section 7.3 quelques exemples numériques. Nous comparons notamment nos résultats avec ceux obtenus par le calcul exact, l'approximation HBSP et la méthode de borne stochastique supérieure de Tancrez.

7.1 Analyse de files d'attente en temps discret

Nous considérons les modèles de files d'attente à temps discret, ce qui revient à dire que les événements sont uniquement observés à des moments qui sont des multiples entiers de l'intervalle de temps. L'intervalle de temps entre deux instants successifs est appelé *slot* et est supposé constant. Dans le but de caractériser au mieux le système étudié et garder une cohérence avec l'évolution du modèle à décrire, nous précisons au préalable les moments où les différents événements sont observés. Pour ce faire, nous avons posé les hypothèses suivantes : les arrivées (notées A) ont lieu avant les services et un groupe de données reste au minimum un intervalle temporel dans la file avant de sortir (le départ est noté par D). La figure 7.1 retranscrit les instants où le système est amené à changer. Nous précisons que cette description est compatible avec celle développée par [HOVC07a, HOVC10, Kle76].

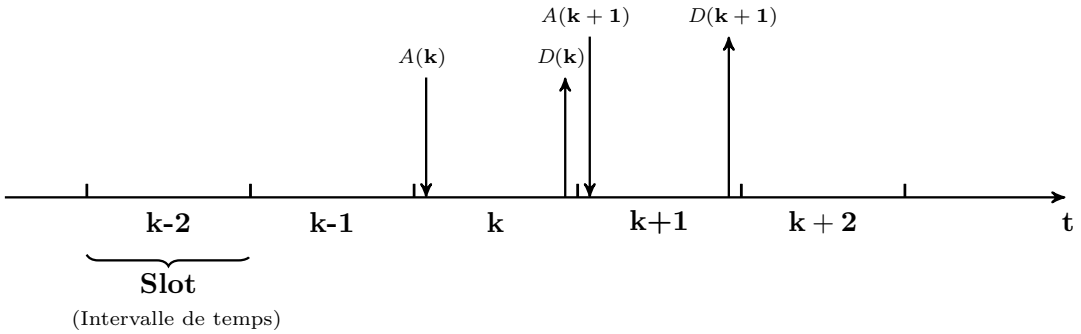


FIGURE 7.1 – Diagramme temporel d'un processus à temps discret.

7.1.1 Modèle d'une file FIFO

Nous considérons une file d'attente simple qui est composée d'un tampon et d'un serveur comme illustré dans la figure 7.2. La variable aléatoire $A(k)$ représente le trafic entrant dans la file pendant le k-ième slot (intervalle de temps). Les variables $Q(k)$ et $D(k)$ représentent respectivement la longueur du tampon et le nombre de sorties durant ce même intervalle

temporel. Le service est supposé déterministe de capacité S et la taille du tampon est donnée par B.

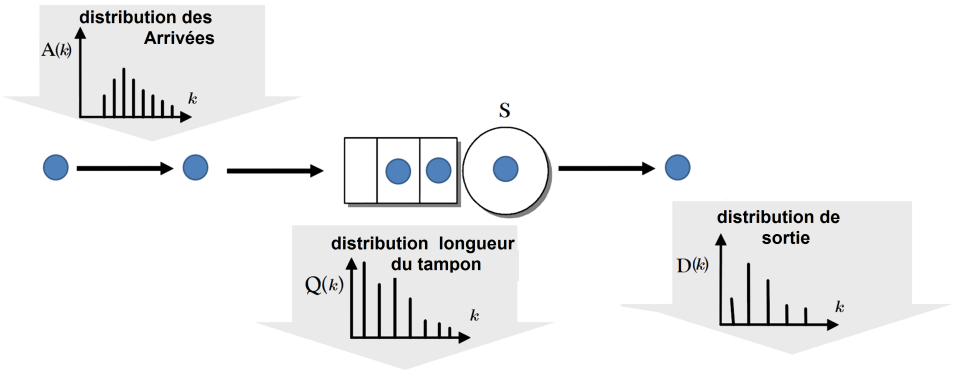


FIGURE 7.2 – Paramètres d'entrée et de sorties d'un modèle de file d'attente

L'équation de récurrence sur la longueur du tampon dans ce modèle est bien connue [Kle76] :

$$Q(k) = \mathtt{min}(\mathrm{B},\ (Q(k-1) + A(k) - S)^+), \tag{7.1}$$

où $(X)^+ = \mathtt{max}(X, 0)$. Nous obtenons également la distribution de sortie grâce à cette récurrence :

$$D(k) = \mathtt{min}(S,\ Q(k-1) + A(k)) \qquad \forall\, k \tag{7.2}$$

Le taux d'utilisation de la file est défini par $\rho = \frac{\mathbb{E}[A]}{S}$, où $\mathbb{E}[A]$ représente le trafic moyen. Dans le cas où la capacité du tampon est infinie, nous devons assurer la stabilité du système. En vérifiant la condition suivante : $\mathbb{E}[A] < S$. L'équation 7.1 définit une chaîne de Markov en temps discret si les arrivées sont indépendantes et le processus $A(k)$ est stationnaire. Comme cette chaîne est de taille finie, il suffit de vérifier que le processus des arrivées rend la chaîne irréductible et apériodique pour qu'elle soit ergodique. Nous donnons ci-après quelques conditions suffisantes et simples pour garantir ces deux propriétés.

Proposition 7.1 *Si les conditions :*

1. *il existe $i < S$ dans le support de A tel que $p[i] > 0$,*
2. *il existe $j > S$ dans le support de A tel que $p[j] > 0$,*
3. *$j = S + 1$ ou $i = S - 1$*

sont satisfaites, alors la chaîne est ergodique.

PREUVE. La première propriété implique que partant de l'état 0 on y retourne avec une probabilité supérieure ou égale à $p[i]$. En effet, si i clients arrivent dans le système, la taille de la file avant les services est de i, comme $i < S$, on retourne donc à l'état 0 après la fin de service. Ainsi, l'état 0 est apériodique. De plus, la première propriété implique que la longueur de la file peut diminuer jusqu'à 0 par une suite de transitions d'un groupe d'arrivées de taille $i < S$. Donc, l'état 0 est atteignable depuis tout état de la chaîne. La condition 2 implique que l'on peut atteindre l'état B depuis l'état 0. Enfin, la dernière condition implique que l'on peut rejoindre tous les états depuis 0 ou B par des sauts d'amplitude 1 qui sont possibles d'après les hypothèses 1 et 2. ■

Nous pouvons remarquer que les conditions 1 et 2 sont nécessaires. Par contre, la condition 3 n'est qu'une condition suffisante. Nous pouvons facilement en trouver d'autres. L'idée consiste simplement à éviter les cas pathologiques où S et B seraient pairs, de même pour tous les éléments du support de A, ce qui rendrait les états impairs inatteignables. La chaîne ne serait pas alors irréductible.
Nous présentons dans l'exemple suivant, un cas illustrant la non-irréductibilité d'une file d'attente finie lorsque la condition 3 n'est pas assurée.

Exemple 7.1 *Nous considérons un modèle de file d'attente simple. Les données sont supposées arriver par groupes de taille* 12 *ou* 0 *et le service est considéré déterministe de capacité* $S = 5$*. Le modèle admet donc des transitions d'une ampleur de* $-\mathbf{5}$ *ou de* $+\mathbf{7}$*.*
Nous distinguons respectivement deux cas : un tampon de capacité infinie et un tampon de capacité finie.

▸ *Pour le modèle de capacité infinie, l'irréductibilité de la chaîne est toujours garantie. En effet, si l'on considère par exemple l'arrivée de deux groupes de données et le départ de trois groupes, nous obtenons une baisse de* 1 *de la taille de la file. Ainsi, nous avons une transition de* 0 *à* i *en faisant* $\lceil \frac{i}{7} \rceil$ *arrivées d'une ampleur de* $+\mathbf{7}$ *puis deux slots avec une transition de* **+7** *et trois slots avec une transition de* **-5** $(2 \times (+\mathbf{7}) + 3 \times (-\mathbf{5}))$ *autant que nécessaire pour arriver en* i*, rendant ainsi tous les états atteignables.*

▸ *Dans le cas où la capacité du tampon est finie, l'irréductibilité de la chaîne n'est pas toujours assurée. Pour une capacité de tampon égale à* 10*, la figure 7.3 illustre le graphe de transition associé au modèle. Nous notons par* d *et* a *les probabilités de transitions possibles, à savoir* $-\mathbf{5}$ *et* $+\mathbf{7}$ *respectivement.*

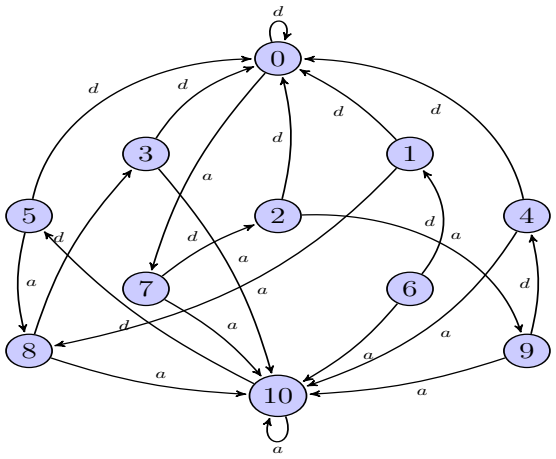


FIGURE 7.3 – Exemple de chaîne non-irréductible

Nous constatons aisément que la chaîne n'est pas fortement connexe (l'état 6 *est non atteignable) et que les conditions* 1 *et* 2 *de la proposition 7.1 ne sont pas suffisantes pour garantir l'irréductibilité de la chaîne.*

Nous passons à présent à la résolution de la chaîne de Markov. Nous notons que plusieurs méthodes existent pour résoudre numériquement le problème. Pour ce qui est du modèle HBSP

[HOVC07a], nous avons remarqué que la méthode utilise directement l'équation 7.1 pour itérer sur la distribution $Q(k)$ jusqu'à obtenir un point fixe. Ce calcul repose sur la version en distribution de l'équation 7.1 qui fait intervenir la convolution $\otimes$ des distributions. Cependant, le test de convergence utilisé par la méthode n'est pas prouvé et n'est donc pas suffisant pour garantir l'obtention de la distribution stationnaire de la chaîne. Nous proposons ci-après un algorithme prouvé pour la convergence du système.

7.1.2 Convergence de la méthode de borne

Pour calculer la distribution stationnaire, nous avons besoin d'un algorithme avec un test de convergence prouvé. Le calcul de la différence entre deux distributions successives comme dans [HOVC10] n'est pas un bon test de convergence (voir le livre de Stewart [Ste95]). Nous proposons l'algorithme suivant basé sur le calcul d'une enveloppe stochastique constituée de Q^L et Q^U afin de prouver la convergence.

Algorithme 26 : Algorithme d'évolution des états

1: $Q^U[0] = \delta_{\mathrm{B}}$, (Dirac à l'état B).
2: $Q^L[0] = \delta_0$, (Dirac à l'état 0).
3: $k = 0$.
4: **Répéter**
5: $\quad Q^U[k+1] = f(Q^U[k]) = \min(\mathrm{B}, (Q^U[k] + \mathcal{A} - S)^+)$.
6: $\quad Q^L[k+1] = f(Q^L[k]) = \min(\mathrm{B}, (Q^L[k] + \mathcal{A} - S)^+)$.
7: **Jusqu'à** $\|Q^U[k+1] - Q^L[k+1]\|_\infty < \epsilon$.

Théorème 7.1 *Supposons que la chaîne est ergodique et que la distribution stationnaire est* $\boldsymbol{\pi}$. *Nous avons :*

$$Q^L[k] \leq_{st} Q^L[k+1] \leq_{st} \boldsymbol{\pi} \leq_{st} Q^U[k+1] \leq_{st} Q^U[k].$$

De plus, la limite de $Q^L[k]$ *et* $Q^U[k]$ *est* $\boldsymbol{\pi}$.

PREUVE. Rappelons que, pour toute fonction croissante f, si $X \leq_{st} Y$ alors, $f(X) \leq_{st} f(Y)$ [MS02]. Nous notons que $\delta_0 \leq_{st} X$ est vrai pour toute distribution X définie sur $\{0..\mathrm{B}\}$. Par conséquent, $Q^L[0] \leq_{st} Q^L[1]$ et $f(Q^L[0]) \leq_{st} f(Q^L(1))$ car f est croissante. Alors, $Q^L[1] \leq_{st} Q^L[2]$. Par induction, nous avons $Q^L[k] \leq_{st} Q^L[k+1]$. La preuve pour $Q^U[k]$ est similaire, car $X \leq_{st} \delta_{\mathrm{B}}$ est vrai pour tout X.

Comme la séquence $Q^L[k] \leq_{st} \delta_{\mathrm{B}}$ est bornée et croissante, alors la limite existe. De même, la séquence de $Q^U[k]$ a une limite. Enfin, de par l'ergodicité de la chaîne, les deux limites sont égales. Les deux distributions Q^L, Q^U convergent. La vérification de la différence entre Q^L et Q^U fournit un test de convergence prouvé. ■

7.2 Monotonie des files d'attente FIFO

Nous présentons dans cette partie quelques résultats de monotonie dans la file d'attente. Plus précisément, cette monotonie signifie que si l'on prend deux processus d'entrée

stochastiquement comparables alors la longueur de la file et le processus de sortie seront également comparables. Nous nous intéressons dans ce chapitre aux files d'attente simples, les réseaux sont plus complexes et nécessitent une analyse plus poussée. Ces derniers seront étudiés dans les chapitres qui suivent. Pour un élément de réseau (file d'attente), nous prouvons ci-après, des résultats théoriques permettant de justifier notre approche.

Afin d'établir un résultat de comparaison stochastique, nous considérons une file à capacité finie et nous injectons les trafics bornants (inférieurs ou supérieurs). À cet effet, nous considérons le trafic d'entrée $\tilde{A}(k)$ décrivant le processus d'arrivée qui borne supérieurement ou inférieurement le trafic réel, nous notons par $\tilde{Q}(k)$ l'état de la chaîne de Markov associée à ce processus d'arrivée. Le nombre de sorties sous les mêmes hypothèses est noté par $\tilde{D}(k)$. Le théorème 4.3.9 de [MS02] permet de prouver la propriété suivante :

Proposition 7.2 *Si* $A(k) \leq_{st} \tilde{A}(k), \forall k \geq 0$ *alors* $Q(k) \leq_{st} \tilde{Q}(k), \forall k \geq 0$

PREUVE. Par récurrence sur k, nous utilisons le fait que les fonctions `min` et `max` sont des fonctions croissantes et nous appliquons le théorème 4.3.9 de Muller et Stoyan [MS02]. Ainsi, si nous avons $Q(k) \leq_{st} \tilde{Q}(k)$ et $A(k) \leq_{st} \tilde{A}(k)$, alors nous déduisons que :

$$Q(k+1) \leq_{st} \tilde{Q}(k+1)$$

■

Nous avons également le résultat pour les bornes inférieures.

Proposition 7.3 *Si* $\tilde{A}(k) \leq_{st} A(k)$ *pour tout* $k \geq 0$*, alors* $\tilde{Q}(k) \leq_{st} Q(k)$ *pour tout* $k \geq 0$*.*

D'après l'équation 7.2, nous déduisons que nous avons également des bornes sur le processus de sortie. Les preuves sont similaires à celle de la proposition 7.2 et sont donc omises.

Proposition 7.4 *Si* $A(k) \leq_{st} \tilde{A}(k)$ *pour tout* $k \geq 0$*, alors* $D(k) \leq_{st} \tilde{D}(k)$*,* $\forall k \geq 0$*. De même, si* $\tilde{A}(k) \leq_{st} A(k) \ \ \forall k \geq 0$*, alors,* $\tilde{D}(k) \leq_{st} D(k), \ \forall k \geq 0$*.*

Sachant que ces propositions sont vraies pour toutes les dates (slot) k, alors elles sont également vraies pour les versions stationnaires lorsque les chaînes sont ergodiques. En supposant que les arrivées $\tilde{A}(k)$ correspondent à des bornes stochastiques (par exemple celles développées dans le chapitre précèdent), nous énonçons le théorème important suivant. Nous donnons le cas des bornes supérieures, les bornes inférieures peuvent être obtenues de manière similaire.

Théorème 7.2 *Soient* $\mathcal{A}$ *(resp.* $\tilde{\mathcal{A}}$*) l'histogramme d'entrée associé à la distribution stationnaire exacte (resp. borne supérieure) et* $\mathcal{Q}, \mathcal{D}$ *(resp.* $\tilde{\mathcal{Q}}, \tilde{\mathcal{D}}$*) la distribution stationnaire associée à la longueur du tampon et au processus de sortie obtenus à partir de la distribution d'entrée exacte* $\mathcal{A}$*, (resp. borne supérieure* $\tilde{\mathcal{A}}$*). Nous avons les inégalités suivantes :*

$$\mathcal{Q} \leq_{st} \tilde{\mathcal{Q}} \ \ et \ \ \mathcal{D} \leq_{st} \tilde{\mathcal{D}}.$$

PREUVE. Par construction $\mathcal{A} \leq_{st} \tilde{\mathcal{A}}$, il découle des propositions 7.2 et 7.4 que nous avons des comparaisons pour tout k, donc également pour les processus stationnaires lorsque $k \to \infty$. Remarquons que par construction, $\mathcal{Q}$ et $\mathcal{D}$ existent (en raison de l'hypothèse d'ergodicité). ■

Après avoir montré les résultats de monotonie sur la file d'attente, nous allons à présent mettre en évidence notre méthode de bornes à travers des résultats numériques en les comparant avec d'autres méthodes.

7.3 Exemples numériques à partir de traces réelles

Cette section présente une évaluation pour valider notre approche et montrer son intérêt et son impact dans le calcul des performances d'une file d'attente simple. Les expériences ont été réalisées avec des traces réelles de trafic Internet, permettant l'analyse de modèles ayant une description par histogramme. Nous déterminons certaines mesures de performance telles que les probabilités de blocage, la longueur moyenne des tampons en fonction des temps d'exécution. Nous étudions la qualité de notre méthode, car l'utilisation de variables discrètes peut présenter des problèmes de précision importants. À cet effet, nous comparons les résultats obtenus par nos algorithmes de borne (borne stochastique inférieure et borne stochastique supérieure), avec ceux obtenus par le calcul exact (sans réduction de support), par la méthode HBSP et par la méthode de borne stochastique supérieure de Tancrez. Toutes les expériences présentées utilisent un facteur de précision de $\epsilon = 10^{-6}$. Notons également que toutes les analyses excepté la méthode HBSP utilisent l'algorithme 26 comme test de convergence prouvé. Les paramètres utilisés dans cette étude proviennent de [HOVC10, HOVC07a] et sont conservés pour faciliter les comparaisons des résultats.

Il est clair que les modèles de trafic par histogrammes peuvent représenter une description puissante et compacte s'ils garantissent une grande précision avec un faible nombre de bins (états). Ainsi, pour une file d'attente simple alimentée par le trafic d'une trace réelle, la question serait donc de savoir : combien de bins seront nécessaires pour obtenir une bonne précision ? Nous allons tenter de répondre à cette question en étudiant respectivement les points suivants : l'influence de la taille du support (après réduction) sur la précision des résultats, La relation entre la taille du tampon et quelques mesures de performance pour un certain nombre de bins et enfin l'interaction entre ces trois facteurs, à savoir : le nombre de bins, la taille du tampon et la qualité des résultats.

7.3.1 Influence de la taille du support sur la précision des résultats

Nous considérons que le trafic en entrée correspond à la trace MAWI (voir figure 5.1), le taux de service est pris égal à $S = 110\,Mb/s$ et la capacité de tampon est de $\mathrm{B} = 1Mb$.

Nous représentons dans la figure 7.4 les probabilités de blocage et la longueur moyenne du tampon pour des supports d'histogrammes variant de 10 à 200 bins.

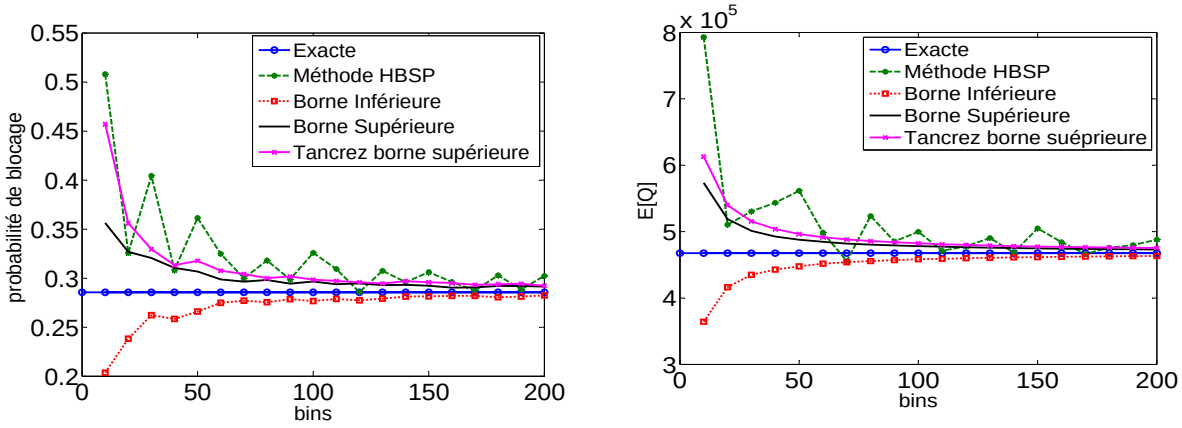


FIGURE 7.4 – Probabilité de blocage (à gauche) et longueur moyenne de la file (à droite) en fonction de la taille du support (nombre de bins).

D'après ces deux graphiques, nous observons que lorsque la taille du support augmente, les résultats obtenus par les différentes méthodes deviennent de plus en plus précis. Cependant, pour des tailles de support faibles, les résultats de la méthode d'Hernàndez (HBSP) sont bien plus mauvais que les nôtres. De plus, nous constatons un comportement oscillatoire des résultats de cette méthode qui ne représente pas une borne, ces résultats sont parfois inférieurs et souvent supérieurs aux valeurs exactes (voir la figure 7.4). Nous précisons que la méthode HBSP est une méthode approximative ce qui peut donc expliquer que ces résultats oscillent. Autre point également observé est que notre borne supérieure est meilleure que celle définie par la méthode de Tancrez. Ceci s'explique par le fait que notre méthode fournit des bornes stochastiques optimales et donc meilleures au sens de la fonction de récompense.

Afin d'avoir une idée sur l'allure des histogrammes en entrée après réduction, nous proposons d'illustrer les deux réductions suivantes : nombre de bins égale à 20 et le nombre de bins égale à 100.

Histogrammes bornants obtenus par les différentes méthodes

Nous proposons de visualiser les histogrammes bornants de l'histogramme MAWI (figure 5.1), obtenus par les différentes méthodes de réduction pour les tailles de support 20 et 100.

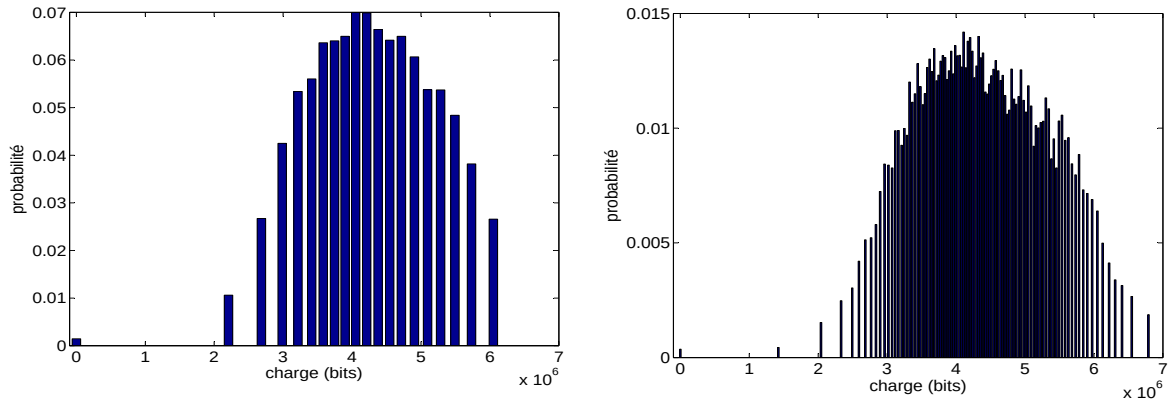


FIGURE 7.5 – Histogrammes bornes inférieures optimales de notre approche pour une réduction sur 20 bins (à gauche) et 100 bins (à droite).

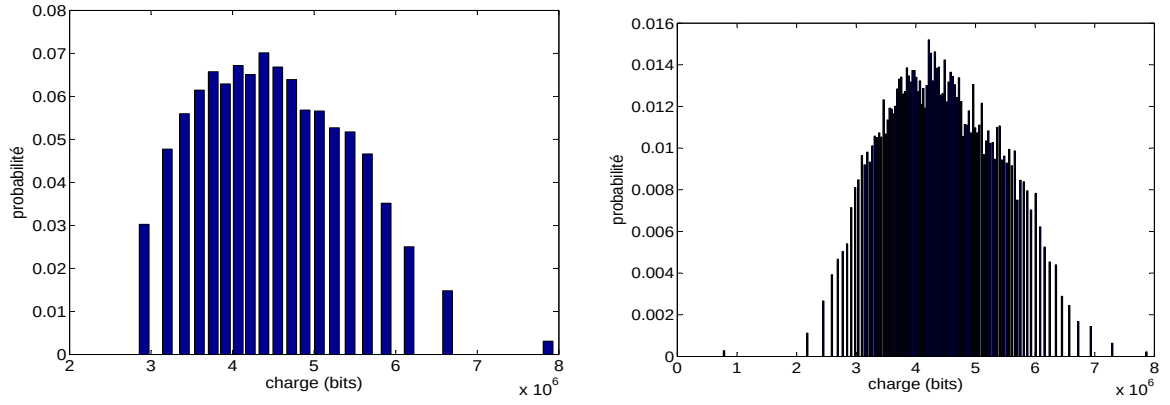


FIGURE 7.6 – Histogrammes bornes supérieures optimales de notre approche pour une réduction sur 20 bins (à gauche) et 100 bins (à droite).

Les figures 7.5, 7.6, 7.7 et 7.8 représentent les histogrammes simplifiés obtenus respectivement par les méthodes suivantes : borne stochastique inférieure optimale, borne stochastique supérieure optimale, méthode HBSP d'Hernàndez et la méthode de Tancrez pour le calcul d'une borne stochastique supérieure.

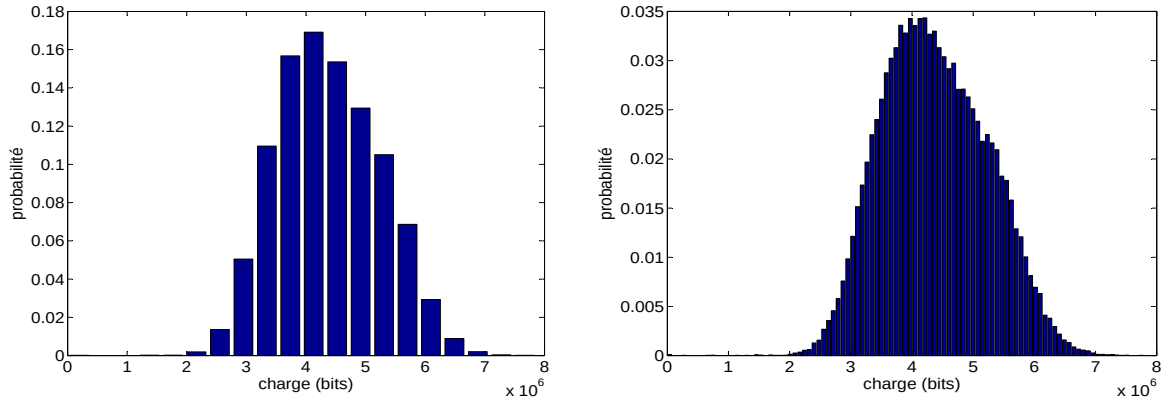


FIGURE 7.7 – Histogrammes HBSP pour une réduction sur 20 bins (à gauche) et 100 bins (à droite).

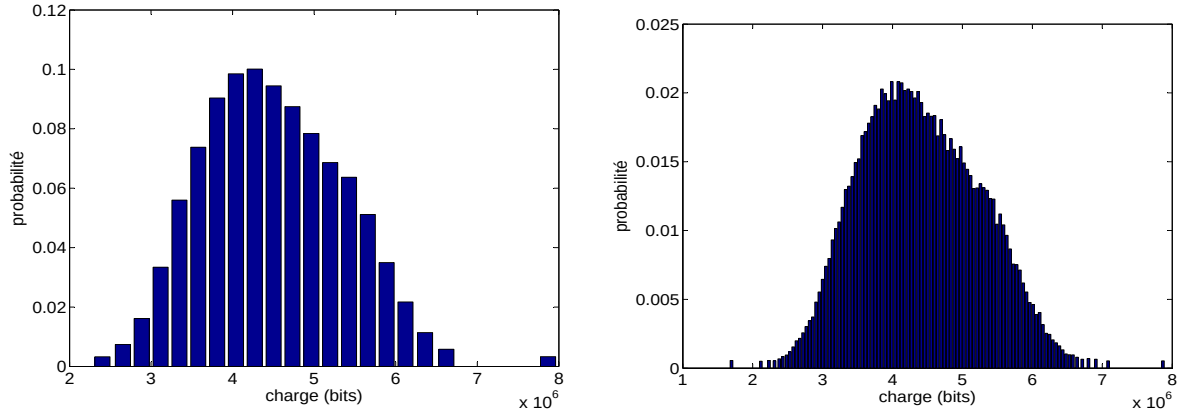


FIGURE 7.8 – Histogrammes de borne supérieure de Tancrez et *al.* pour une réduction sur 20 bins (à gauche) et 100 bins (à droite).

Il est aisé de constater que ces méthodes laissent entrevoir des distributions complètement distinctes définies sur des états différents. De plus, nous remarquons que pour des tailles de support qui demeurent encore bien plus petites comparées à la taille de la distribution exacte (80511 états), ces histogrammes fournissent des résultats intéressants voire très pertinents lorsque la taille du support est égale à 100 états (100 bins).

Étude des distributions obtenues par les différentes méthodes

Pour observer tout l'intérêt de la méthode de réduction et plus particulièrement notre méthode de bornes, nous proposons de tracer les distributions de probabilités cumulées associées à la longueur du tampon en considérant les deux réductions : 20 bins et 100 bins (voir la figure 7.9).

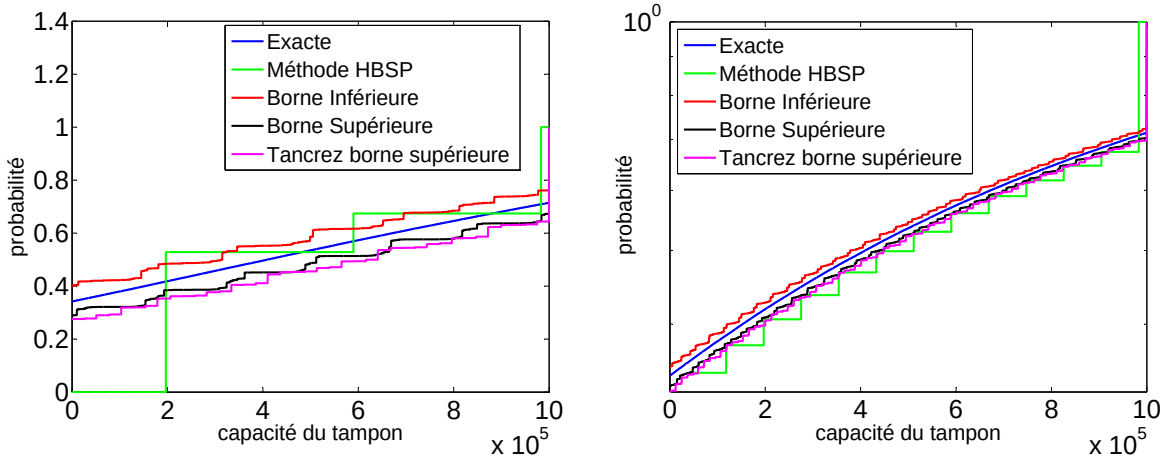


FIGURE 7.9 – Distributions de probabilités cumulées de la longueur du tampon pour la trace MAWI traffic, taille du support : 20 bins (à gauche), 100 bins (à droite).

Encore une fois, nous pouvons observer que la méthode HBSP ne représente pas une borne de la distribution exacte et ne fournit pas une bonne approximation de cette dernière pour des valeurs de bins faible (bins =20). Pour un nombre de bins égal à 100 toutes les méthodes donnent de meilleurs résultats, mais nos bornes restent les plus précises. En ce qui concerne les temps de résolutions, pour un nombre de bins égal à 100, le calcul exact est obtenu au bout de 1897 secondes (s), la méthode HBSP permet d'obtenir un résultat en 0.007 s, les bornes optimales inférieures et supérieures sont obtenues respectivement en 0.35 s et 0.33 s et la méthode de Tancrez prend quant à elle un temps de calcul de 0.012 s. Ainsi, nous remarquons que la méthode HBSP et la méthode de Tancrez sont les plus rapides, mais nos temps restent tout de même très intéressants comparés au temps de calcul exact.

7.3.2 Relation entre la taille du tampon et certaines mesures de performance

À partir de la trace réelle du trafic MAWI, nous évaluons et comparons certaines mesures de performance telles que la probabilité de blocage et la longueur moyenne du tampon en utilisant respectivement notre approche de bornes (borne inférieure notée $L.b$ et borne supérieure notée $L.U$), le calcul exact, la méthode HBSP et la méthode de Tancrez. Nous comparons également les résultats en fonction des temps de résolution. Nous considérons à cet effet, un service déterministe $S = 110\,Mb/s$, une unité de données de $D = 1\,Kb$ et une taille de tampon (B) qui varie de $5\,10^4$ $bits$ à $3\,10^6$ $bits$. Nous présentons dans le tableau 7.1 les résultats obtenus sur les probabilités de blocage pour un nombre de bins égale à 20 et un nombre de bins égal à 100. Les temps d'exécution associés à la plus grande capacité du tampon ($\mathbf{B} = 3\,Mb$) pour les différentes méthodes considérées sont également présentés. Les tableaux 7.2 et 7.3 résument quant à eux les longueurs moyennes du tampon pour un nombre de bins égal à 20 et 100.

			20				100			
		Exact	$L.b$	$U.b$	HBSP	Tancrez U. b	$L.b$	$U.b$	HBSP	Tancrez U. b
Capacité du tampon (10^5 bits)	0.5	0.456	0.411	0.469	/	0.491	0.451	0.460	/	0.471
	2	0.425	0.374	0.437	/	0.464	0.418	0.431	0.464	0.442
	3	0.405	0.371	0.437	/	0.460	0.398	0.412	0.448	0.415
	4	0.386	0.338	0.407	0.407	0.429	0.379	0.393	0.418	0.398
	5	0.367	0.333	0.380	0.407	0.429	0.361	0.374	0.403	0.379
	8	0.317	0.271	0.354	0.326	0.372	0.307	0.326	0.349	0.331
	9	0.301	0.241	0.330	0.326	0.349	0.294	0.309	0.338	0.317
	10	0.285	0.238	0.325	0.326	0.347	0.277	0.297	0.326	0.298
	20	0.182	0.124	0.235	0.155	0.258	0.171	0.194	0.215	0.197
	30	0.131	0.076	0.173	0.102	0.215	0.120	0.144	0.161	0.148

TABLE 7.1 – Probabilités de blocage ($\mathrm{Prob(B)}$) pour bins = 20 et bins = 100.

Nous notons que le temps d'exécution associé à la plus grande capacité du tampon (B =3Mb) est de 3378 secondes pour déterminer les mesures exactes. Les temps d'exécution des autres méthodes pour un nombre de bins égal à 20 sont respectivement de 4.5 secondes pour la borne optimale inférieure, 5.3 s pour la borne optimale supérieure, 0.01 s pour la méthode HBSP et 0.06 s pour la détermination de la borne supérieure de Tancrez. Pour un nombre de bins égal à 100, les temps sont de 19.5 secondes pour la borne optimale inférieure, 23 s pour la borne optimale supérieure, 0.03 s pour la méthode HBSP et 0.07 s pour la borne supérieure de Tancrez.

		Exact	$L.b$	$U.b$	HBSP	Tancrez U. b
Capacité du tampon (10^5 bits)	0.5	23301	20557	24960	/	25310
	2	92992	85096	100169	/	100830
	3	139407	128363	150760	/	150451
	4	185845	169664	201530	356563	202097
	5	232309	211768	252709	356563	253244
	8	372917	335126	409923	510300	410125
	9	420127	376429	464449	510300	464813
	10	467475	415772	518405	510300	518724
	20	938529	785393	1094650	923901	1137680
	30	1399670	1098820	1709590	1156000	1778910

TABLE 7.2 – Longueur moyenne du tampon ($\mathbb{E}[Q]$) pour 20 bins.

Capacité du tampon (10^5 bits)	**Exact**	$L.b$	$U.b$	HBSP	Tancrez U. b
0.5	23301	23017	23648	/	23827
2	92992	91711	94531	114894	95229
3	139407	137410	141771	152699	142842
4	185845	183102	189121	228551	190567
5	232309	228704	236561	266694	238432
8	372917	366337	380648	421122	384001
9	420127	412384	429214	460333	433114
10	467475	458388	478066	499711	482500
20	938529	910570	970608	1019260	980361
30	1399670	1343350	1463610	1544850	1479740

TABLE 7.3 – Longueur moyenne du tampon ($\mathbb{E}[Q]$) pour 100 bins.

Nous pouvons donc observer que les temps de résolution de notre méthode de bornes sont nettement inférieurs à ceux obtenus par le calcul exact, et que les méthodes HBSP et Tancrez demeurent les plus rapides. Cependant, notre approche nous permet de gagner en précision. En effet, nous constatons que nos bornes donnent d'assez bons résultats et deviennent plus précises et très proches de l'exact quand on augmente le nombre d'états (bins). Pour ce qui est de la méthode HBSP, nous remarquons que lorsque le nombre de bins est égale à 20, la méthode ne converge pas pour des tailles de tampons inférieures à $3\,10^5$. De plus, on observe dans le tableau 7.1 pour un nombre de bins égal à 100 que la méthode HBSP donne de moins bon résultats que Tancrez et que notre borne supérieure. Cette remarque est également observée dans le tableau 7.3. Dans le tableau 7.2 pour un nombre de bins égal à 20, on observe que la méthode HBSP fournit des valeurs supérieures à notre borne supérieure quand la capacité du tampon est inférieure à $9\,10^5$ et des valeurs inférieures dans les autres cas. Par ailleurs, les valeurs obtenues sont parfois supérieures aux valeurs exactes et inférieures dans d'autres cas, ce qui montre que c'est une approximation et non une méthode de borne.

Pour ce qui est de la méthode de Tancrez, nous voyons clairement que notre borne supérieure est meilleure ce qui est évident puisque notre borne est optimale. Nous montrons à travers ces tableaux l'intérêt de notre approche car nous calculons à la fois des bornes supérieures et inférieures permettant l'encadrement de la valeur exacte. Ajouté à cela, cet encadrement est plus ou moins précis selon le nombre de bins considéré.

La seconde expérience, que nous montrons ici, repose sur la trace de trafic CAIDA OC-48 (Cooperative Association for Internet Data Analysis) [CAI] collectée sur les deux directions d'un lien OC48 à AIX le 24 avril 2003. Le taux moyen est de $92\,Mb/s$. Nous avons échantillonné 5 mn de la trace avec une période $T = 10ms$ (100 échantillons par seconde). L'histogramme de la trace a un support de taille 30.000 bins et une espérance de $E[\mathcal{A}] = 1.2885\,10^5\,bits$. Nous avons calculé la probabilité de blocage et la longueur moyenne de la file en fonction de différentes capacités des liens (variant de $5.\,10^3\,bits$ à $10^5\,bits$). Pour une taille de support d'histogramme égale à 10 puis égal à 100 états, les résultats obtenus sont reportés dans le figure 7.10 et la figure 7.11 respectivement.

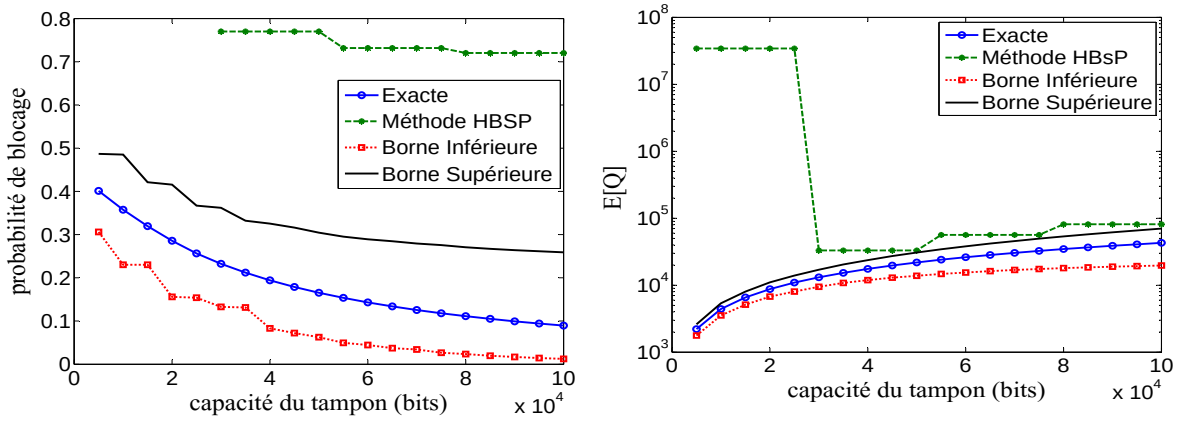


FIGURE 7.10 – Probabilité de blocage (à gauche) et longueur moyenne du tampon (à droite) avec une trace de trafic CAIDA OC-48 pour un histogramme de taille 10 bins.

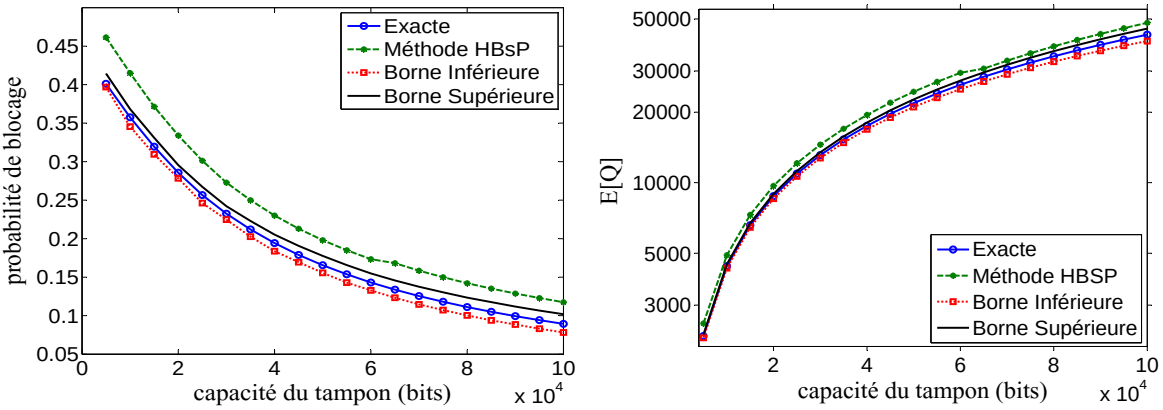


FIGURE 7.11 – Probabilité de blocage (à gauche) et longueur moyenne du tampon (à droite) avec une trace de trafic CAIDA OC-48 pour un histogramme de taille 100 bins.

Cette expérimentation nous conduit aux mêmes conclusions que précédemment. Dans la figure 7.10, la méthode HBSP ne converge pas pour des petites capacités de tampon (c'est pour cela que les points n'ont pas été représentés). Pour les autres points, on voit globalement que nos bornes sont de meilleures qualités que les résultats obtenus par la méthode HBSP. Ainsi, pour finir nos évaluations, nous proposons de faire une représentation 3D de la probabilité de blocage en fonction de la capacité du tampon et du support de la distribution de trafic pour les trois méthodes : calcul exact, méthode HBSP et notre méthode de borne. D'après la figure 7.13, on remarque bien que nos bornes fournissent un bon encadrement de la solution exacte et elles deviennent plus précises lorsqu'on augmente le nombre de bins. Pour ce qui est de la méthode HBSP, on observe clairement l'aspect approximation de la méthode, la partie en bas à droite de la figure montre qu'elle ne fournit pas de borne (par moment elle donne des résultats en dessous de l'exacte et d'autre fois en dessus).

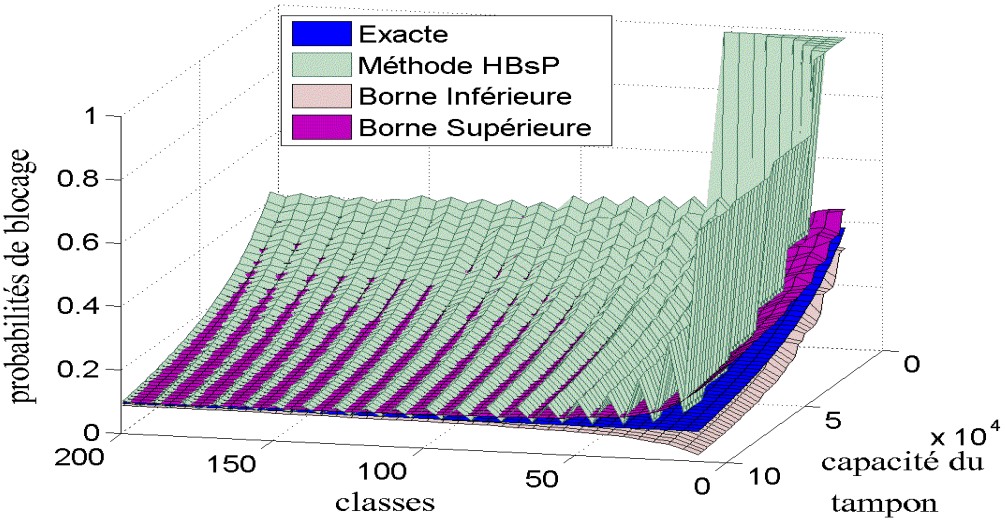


FIGURE 7.12 – Probabilité de blocage pour la trace CAIDA OC-48 : comparaison des trois méthodes.

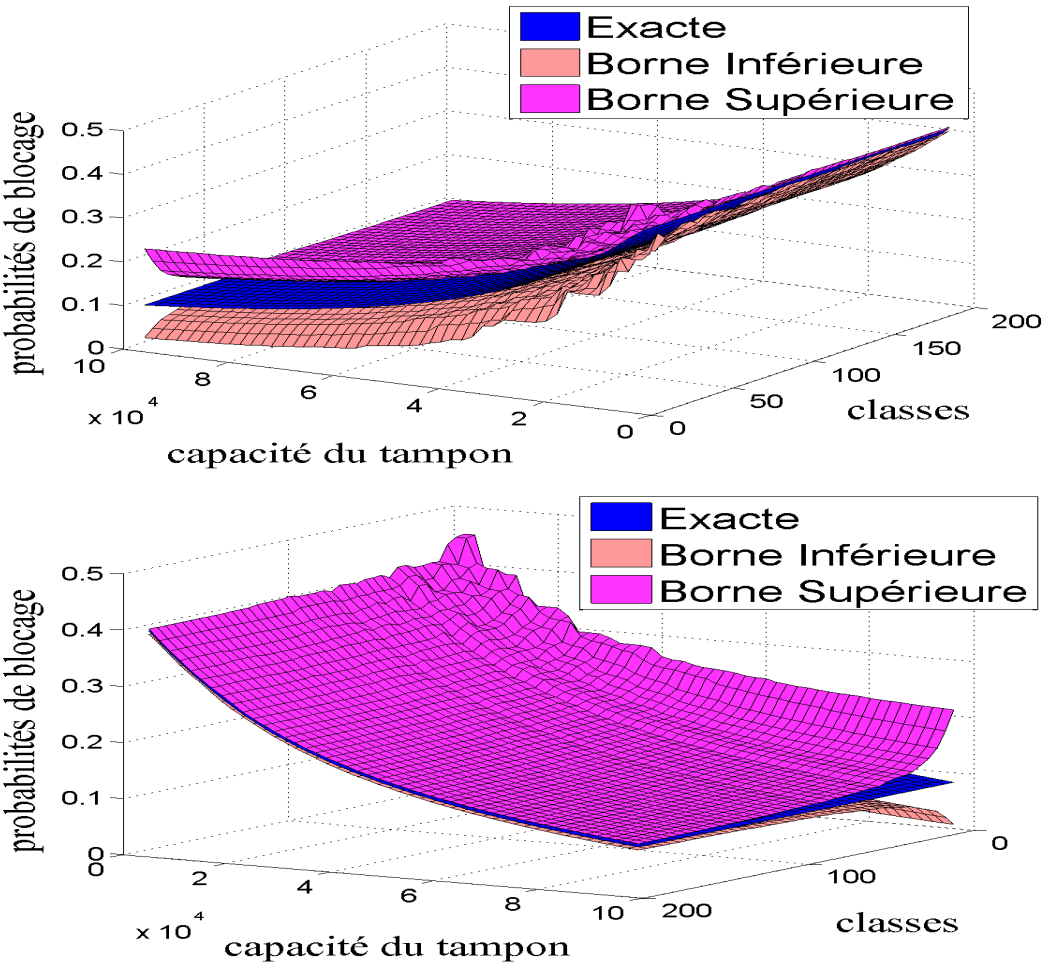


FIGURE 7.13 – Probabilité de blocage pour la trace CAIDA OC-48 : comparaison entre nos bornes et le résultat exact.

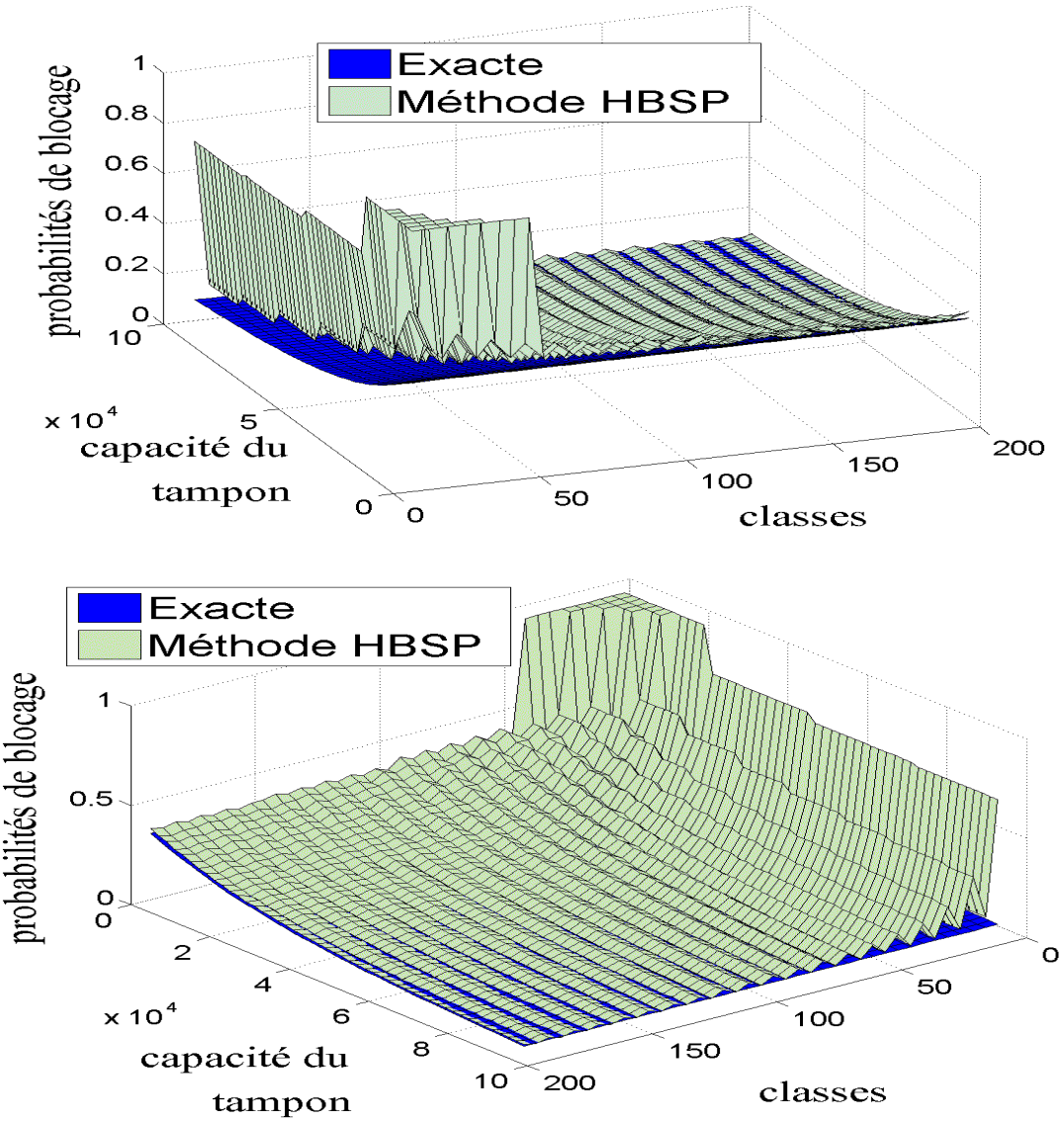


FIGURE 7.14 – Probabilité de blocage pour la trace CAIDA OC-48 : comparaison HSBP et méthode exacte.

7.4 Conclusion et perspective

Dans le but d'analyser les performances d'un système de file d'attente simple, nous avons proposé dans ce chapitre de développer un formalisme fondé sur les bornes stochastiques afin de réduire la taille de l'histogramme du trafic en entrée. Pour ce faire, nous avons appliqué un algorithme reposant sur la programmation dynamique pour définir des histogrammes bornants permettant de calculer des bornes sur les différentes mesures de performance : les probabilités de blocage, la longueur moyenne du tampon, etc. Nous avons effectué une comparaison entre les résultats obtenus par notre approche de borne et ceux déterminés par le calcul exact, l'approximation HBSP et la méthode de la borne supérieure de Tancrez. À travers nos expériences, nous avons montré clairement l'intérêt significatif de notre approche : nos

résultats sont plus précis et peuvent être obtenus dans des temps d'exécution très intéressants. De plus, elle offre un avantage important qui consiste à définir des bornes stochastiques permettant de fournir une garantie sur les performances du système pour des récompenses positives croissantes. Par exemple, si l'on veut déterminer la capacité du tampon garantissant une probabilité de blocage inférieure à une valeur seuil, nos bornes permettent de répondre à ce problème de dimensionnement. Ce que ne permet pas de faire une méthode approximative. De plus, vu la qualité de nos bornes, les résultats sont tout à fait pertinents.

Dans le chapitre suivant, nous allons nous pencher sur le cas où le service est également exprimé sous forme d'histogramme. N'utilisant plus l'approche par équations d'évolution, comme présentée dans ce chapitre, nous allons construire les matrices de probabilités de transition associées aux chaînes de Markov décrivant le système. Nous allons également présenter d'autres mesures de performance telles que les temps de réponse et les probabilités de pertes et montrer leurs monotonies. Enfin, dans le but d'étendre notre approche à l'analyse de réseaux de files d'attente, nous allons étudier la notion de monotonie des processus de routage permettant de séparer le trafic ou de le rassembler [Whi83, Sch07].

Chapitre 8

Modélisation des éléments de réseaux

Nous proposons dans ce chapitre de nous intéresser à l'analyse de performance d'une file d'attente simple étudiée sous un trafic général provenant de traces réelles dont le processus de service peut également être décrit sous forme de variables aléatoires (histogramme). Nous prouvons ici la notion de monotonie stochastique notée *H-monotonie* sur l'élément de réseau et prouvons que l'utilisation de notre approche de bornes stochastiques sur les processus d'entrées et/ou de services de la file nous permet de dériver, d'une manière très efficace, des bornes sur les indices de performance tels que les délais et les pertes. Ainsi, lorsque nous utilisons une borne stochastique supérieure (resp. inférieure) du processus d'entrée et/ou une borne inférieure (resp. supérieure) du processus de service, nous obtenons une borne stochastique supérieure (resp. inférieure) de la distribution de la longueur du tampon, du processus de départ, des temps de réponse ainsi que des probabilités de perte.

Nous présentons également deux extensions à cette étude : l'analyse des éléments de routage dans les réseaux gérant différents flux (élément de fusion et de division des flux) et l'étude de certains mécanismes de gestion active de file d'attente de type AQM (Active Queue Management). L'intérêt des mécanismes AQM est de gérer les problèmes de congestion dans les nœuds des réseaux [KAC04]. La première extension nous permettra d'envisager l'analyse de réseaux de files d'attente (réseaux en arbre ou réseaux feed-forward), tandis que la deuxième extension nous permet d'étudier les performances des réseaux actuels où les mécanises d'AQM sont implémentés.

Nous commençons ce chapitre par la description de manière assez détaillée du modèle de file d'attente étudié ainsi que les différentes terminologies utilisées. Puis, en utilisant la formulation par matrice de transitions, nous définissons et construisons la chaîne de Markov décrivant l'évolution de la longueur du tampon dans la file d'attente. La résolution de cette DTMC nous permet d'obtenir différents paramètres de sorties tels que la distribution de la longueur du tampon, la distribution du processus de sortie, la distribution des pertes et la distribution des temps de réponse. Dans la section 8.2, nous démontrons quelques propriétés importantes de monotonie stochastique pour le calcul de bornes dans une file d'attente simple. Dans la section 8.5, nous présentons les éléments de routage (division et fusion) ainsi que les résultats théoriques de leurs monotonies dans le réseau. Nous prouvons que certains éléments de division qui séparent un flux en plusieurs sous-flux acheminés vers des nœuds distincts sont monotones. De même, nous montrons que les éléments de fusion qui combinent plusieurs flux

en un flux global sont également monotones. Dans la section 8.4, nous proposons d'étudier certains mécanismes de gestion active de files d'attente et introduisons le mécanisme IRED (*Immediate Random Early Detection*). Enfin, une évaluation numérique d'une file d'attente isolée ayant pour un premier cas un service déterministe puis dans un second cas un service variable a été présentée afin d'observer non seulement la notion de *H-monotonie* des paramètres du système démontrée dans ce chapitre, mais également de voir l'impact et l'intérêt de notre approche en termes de complexité et de précision.

8.1 File d'attente isolée

Soit H une distribution de probabilité discrète, $H[i]$ désigne la probabilité de i et E^H représente l'ensemble des éléments tels que la probabilité $H[i]$ est positive (*i.e.,* $H[i] > 0$). La taille de la distribution H est, par définition, le cardinal de l'ensemble E^H (noté $|E^H|$). Il est important de préciser que tous les résultats théoriques sont obtenus sur les distributions à support dans les ensembles $\mathbb{N}$ ou $\mathbb{Z}$ (nous avons besoin des nombres négatifs lorsque nous combinons les arrivées et les départs), mais nos algorithmes ne prennent en compte que des éléments avec une probabilité positive.

La figure 8.1 illustre le modèle d'une file d'attente simple munie d'un seul serveur et d'une capacité de tampon finie notée B, tels que :

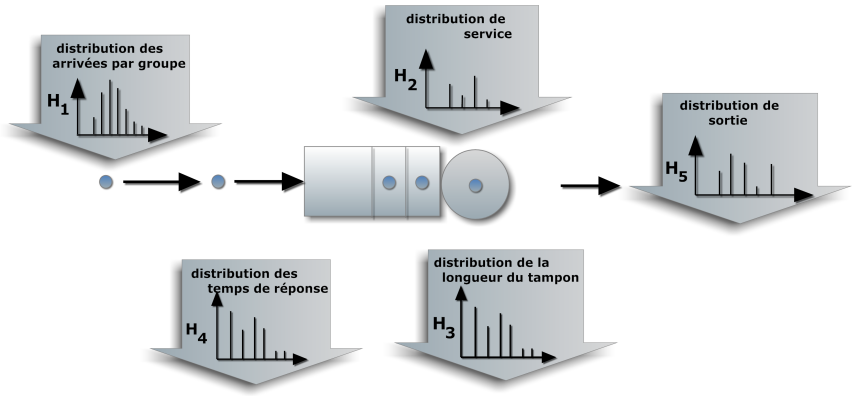


FIGURE 8.1 – Paramètres d'entrées et de sorties d'un modèle de file d'attente simple

Les arrivées (*le nombre d'unités de données arrivées pendant un slot*) : représentent une séquence i.i.d. de flux stationnaire définie par l'histogramme H_1 qui prend ses valeurs sur l'espace d'états $E^{H_1} \subset \{0, \ldots, \mathrm{N}\}$ ($\mathrm{N} < \mathrm{B}$). Sans perte de généralité, nous supposons que l'état 0 est dans E^{H_1}.

La discipline de service : est "Premier arrivée premier servi" (FIFO) et donc work-conserving, c'est-à-dire que le serveur ne peut pas être inactif s'il y a des unités de données dans le tampon.

La capacité de service (*le nombre d'unités de données qui peut être servi à chaque slot*) : est indépendant et identiquement distribué et spécifiée par l'histogramme H_2 défini sur $E^{H_2} \subset \{1, \ldots, \mathrm{C}\}$. Notons que l'état 0 n'est pas dans E^{H_2} en raison de la propriété de work-conserving. Nous notons que les histogrammes associés aux paramètres de sorties sont calculés

à l'état stationnaire.

La longueur du tampon (*le nombre d'unités de données dans le tampon*) : est représenté par l'histogramme H_3 défini sur l'ensemble $E^{H_3} \subset \{0, \ldots, \mathrm{B}\}$.

Le processus de sortie (*le nombre d'unités de données quittant la file d'attente après le service, à chaque slot*) : est décrit par l'histogramme H_5 défini sur l'ensemble $E^{H_5} \subset \{0, \ldots, \mathrm{C}\}$.

Le temps de réponse (*le nombre de slots entre la date d'arrivée et de départ d'une unité de données*) : également noté *Délai* est spécifié par l'histogramme H_4 défini sur l'espace d'état $E^{H_4} \subset \mathbb{N}$.

Notre analyse a pour but de calculer les distributions H_3, H_4 et H_5 sachant H_1 et H_2. Pour cela, nous allons construire la chaîne de Markov définie dans la partie suivante.

8.1.1 Construction de la chaîne de Markov

L'évolution de la longueur du tampon dans la file d'attente est donnée par une chaîne de Markov à temps discret (DTMC) homogène dans le temps $\{X(n), n \geq 0\}$ à valeurs dans un espace d'états totalement ordonné $\{0 \cdots \mathrm{B}\}$. Soit A (resp. S) une variable aléatoire i.i.d. représentant le nombre d'unités de données reçues (resp. la capacité de service) pendant un slot de temps dont la *pmf* est H_1 (resp. H_2). Nous supposons que les arrivées ont lieu au début d'un slot tandis que les départs ont lieu à la fin (voir figure 7.1). L'équation d'évolution de la longueur de la file d'attente est donnée par [Kle76] :

$$X(n+1) = \min(\mathrm{B}, (X(n) + A - S)^+), \tag{8.1}$$

où $(X)^+ = \max(0, X)$.

Il est clair, en raison des hypothèses sur le modèle, que $\{X(n), n \geq 0\}$ est une chaîne de Markov à temps discret. Afin de construire la matrice de probabilité de transition $\mathbf{P}$ associée à $\{X(n), n \geq 0\}$, nous considérons une famille de matrices $\mathcal{U} = \{\mathbf{U}_k\}_{k \in \mathcal{K}}$ qui représentent l'ensemble des modifications possibles depuis un état en fonction de la capacité du tampon, de la distribution d'arrivée et la distribution service. Les matrices $\{\mathbf{U}_k\}_{k \in \mathcal{K}}$ sont définies comme suit :

$$\mathcal{K} = \{x - y, \forall x \in E^{H_1}, \forall y \in E^{H_2}\}.$$

$$k \in \{\mathcal{K} | k \geq 0\}, \quad \mathbf{U}_k[i,j] = \begin{cases} 1 & \text{if } \; j = \min(\mathrm{B}, k+i), \\ 0 & \text{sinon.} \end{cases}$$

$$k \in \{\mathcal{K} | k < 0\}, \quad \mathbf{U}_k[i,j] = \begin{cases} 1 & \text{if } \; j = \max(0, k+i), \\ 0 & \text{sinon.} \end{cases}$$

Exemple 8.1 *Pour une capacité de tampon* $\mathrm{B} = 3$*, les matrices* $\mathbf{U}_1$ *et* $\mathbf{U}_{-2}$ *sont données par :*

$$\mathbf{U}_1 = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad \textit{et} \qquad \mathbf{U}_{-2} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

Proposition 8.1 (Construction de la matrice P) *La matrice de transition* $\mathbf{P}$ *est décrite par :*

$$\mathbf{P} = \sum_{k \in \mathcal{K}} \mathbf{U}_k \left(\sum_{i \in E^{H_1}} \sum_{j \in E^{H_2}} H_1[i]\, H_2[j]\, \mathbb{1}_{\{min(\mathrm{B}, i-j)=k\}} \right).$$

PREUVE. Pour chaque ligne (état) i, l'impact d'une éventuelle transition $k \in \mathcal{K}$ est définie par la matrice $\mathbf{U}_k$. Sa probabilité est décrite par la somme des probabilités à l'intérieur des parenthèses qui énumèrent toutes les combinaisons des arrivées et des départs qui conduisent à l'état k. Après sommation sur toutes les transitions (événements) $k \in \mathcal{K}$, chaque ligne i est un vecteur de probabilités, et $\mathbf{P}$ est une matrice de probabilités de transition de la chaîne de Markov. ■

Nous proposons d'illustrer cette proposition à travers un petit exemple.

Exemple 8.2 *Soient une distribution d'arrivée* $H_1 = [0.2, 0.5, 0.3]$ *définie sur* $E^{H_1} = \{0, 1, 2\}$*, un histogramme de service donné par* $H_2 = [0.5, 0.5]$ *avec* $E^{H_2} = \{1, 2\}$ *et un tampon de capacité* $\mathrm{B} = 3$*. La matrice de transition* $\mathbf{P}$ *de* $\{X(n),\, n \geq 0\}$ *est déterminée comme suit :*

$$\mathbf{P} = \sum_{k \in \mathcal{K}} \mathbf{L}_k,$$

où $\mathcal{K} = \{-2, -1, 0, 1\}$ *et* $\mathbf{L}_k = \mathbf{U}_k . \left(\sum_{i \in E^{H_1}} \sum_{j \in E^{H_2}} H_1[i]\, H_2[j]\, \mathbb{1}_{\{min(3, i-j)=k\}} \right)$,

• $\forall\ k \in \mathcal{K}$*, nous avons*

$$\mathbf{L}_{-2} = \begin{pmatrix} 0.1 & 0 & 0 & 0 \\ 0.1 & 0 & 0 & 0 \\ 0.1 & 0 & 0 & 0 \\ 0 & 0.1 & 0 & 0 \end{pmatrix}, \qquad \mathbf{L}_{-1} = \begin{pmatrix} 0.35 & 0 & 0 & 0 \\ 0.35 & 0 & 0 & 0 \\ 0 & 0.35 & 0 & 0 \\ 0 & 0 & 0.35 & 0 \end{pmatrix}$$

$$\mathbf{L}_0 = \begin{pmatrix} 0.4 & 0 & 0 & 0 \\ 0 & 0.4 & 0 & 0 \\ 0 & 0 & 0.4 & 0 \\ 0 & 0 & 0 & 0.4 \end{pmatrix} \quad \text{et} \qquad \mathbf{L}_1 = \begin{pmatrix} 0 & 0.15 & 0 & 0 \\ 0 & 0 & 0.15 & 0 \\ 0 & 0 & 0 & 0.15 \\ 0 & 0 & 0 & 0.15 \end{pmatrix}$$

• *Alors,*

$$\mathbf{P} = \begin{pmatrix} 0.85 & 0.15 & 0 & 0 \\ 0.45 & 0.4 & 0.15 & 0 \\ 0.1 & 0.35 & 0.4 & 0.15 \\ 0 & 0.1 & 0.35 & 0.55 \end{pmatrix}.$$

Nous allons prouver à présent, l'une des propriétés les plus importantes pour les éléments de réseau à savoir ***la monotonie selon l'ordre stochastique fort***. Une telle propriété implique que les histogrammes des performances des éléments du réseau soient stochastiquement

comparables selon l'ordre fort quand les histogrammes des paramètres d'entrée le sont également.

Propriété 8.1 *Pour tout $k \in \mathcal{K}$, la matrice positive*
$\mathbf{U}_k.\left(\sum_{i\in E^{H_1}} \sum_{j\in E^{H_2}} H_1[i]\ H_2[j]\ \mathbb{1}_{\{min(\mathrm{B},i-j)=k\}}\right)$ *est st-monotone.*

PREUVE. Tout d'abord nous avons par construction que les matrices $\mathbf{U}_k$ sont st-monotones. Ensuite, puisque la somme des probabilités entre parenthèses est une constante positive qui ne dépend que de k, alors les versions pondérées sont également st-monotone (voir propriété 2.3 chapitre 2). ■

Proposition 8.2 (st-Monotonie de la chaîne de Markov) *La chaîne de Markov à temps discret $\{X(n),\ n \geq 0\}$ est st-monotone.*

PREUVE. Puisque la somme convexe de matrices positives st-monotone est st-monotone (voir propriété 2.3 chapitre 2), il découle de la propriété 8.1 que la matrice de transition $\mathbf{P}$ construite à travers la proposition 8.1 est st-monotone. Par conséquent, la DTMC associée est st-monotone (voir le théorème 2.6 chapitre 2). ■

La propriété de monotonie est utilisée pour la preuve de convergence de notre méthode [ACFP13c] (voir la section 7.1.2).

Proposition 8.3 (Stationnarité de la chaîne de Markov) *Soit une DTMC $\{X(n), n \geq 0\}$ de matrice de transition $\mathbf{P}$ construite suivant la proposition 8.1. Nous supposons que la chaîne est irréductible. Alors elle admet un unique vecteur stationnaire $\boldsymbol{\pi}$ calculé comme suit :*

$$\boldsymbol{\pi}\,.\,\mathbf{P} = \boldsymbol{\pi},\ \ \boldsymbol{\pi}[i] \geq 0, \qquad \sum_{i=0}^{\mathrm{B}} \boldsymbol{\pi}[i] = 1. \tag{8.2}$$

PREUVE. Tout d'abord, nous pouvons vérifier que la chaîne est apériodique. Il est clair qu'à partir de l'état 0 nous avons une probabilité positive de rester en 0 après l'arrivée d'un groupe de données vide. Ce qui revient à dire qu'aucun service n'est effectué et $\mathbf{P}[0,\ 0] > 0$. La chaîne est donc apériodique. Il en résulte que la chaîne de Markov à états finis est ergodique, ce qui implique l'existence d'une distribution stationnaire unique $\boldsymbol{\pi}$. ■

Le calcul des probabilités stationnaires, en considérant la formulation de matrice, peut être effectué en utilisant plusieurs techniques numériques (voir section 2.1.1 chapitre 2) . Nous préconisons que, lorsque nous calculons des garanties pour la qualité de service, il faut fournir des algorithmes stables et précis avec un test de convergence prouvé. Nous avons constaté que la technique numérique utilisée dans [HOVC10, HOVC07a, HOVC07b] pouvait être numériquement instable. De plus, il n'existe aucune preuve de convergence pour cet algorithme, qui est fondé sur l'itération d'un point fixe sur la distribution. Beaucoup de références (voir à nouveau [Ste95] pour ce problème) ont fait remarquer que ce type de méthode peut conduire à des erreurs numériques importantes en raison d'un faux test de convergence. Ainsi, dans les expériences numériques, nous avons utilisé un nouvel algorithme (algorithme

26, page 133), qui prend en considération la monotonie de la matrice $\mathbf{P}$ pour prouver le test de convergence. Il a également la même complexité que l'approche présentée dans [HOVC07a]. Notez que nous avons deux sortes de problèmes numériques : l'analyse à l'état stationnaire et le calcul de la distribution du temps de réponse. Ce dernier calcul est basé sur la multiplication de vecteur-matrice comme nous le verrons dans la prochaine section. Nous avons développé un solveur dans Matlab® pour un banc d'essai simple.

8.1.2 Calcul des paramètres de sortie du modèle

Nous présentons ici les différents paramètres de sortie pour un système constitué d'une file d'attente simple. Pour une DTMC ergodique, nous décrivons respectivement les histogrammes associés à la longueur du tampon, au processus de sortie, aux probabilités de pertes ainsi qu'au temps de réponse.

a) Distribution de la longueur du tampon : H_3

La distribution de la longueur du tampon notée H_3 correspond à la distribution de l'état vu par le groupe de données en entrée.

Proposition 8.4 (Longueur du tampon, H_3) *Sachant que la file d'attente considère d'abord les arrivées puis le service, alors la distribution de la longueur du tampon H_3 avant l'instant d'arrivée correspond à la distribution stationnaire $\boldsymbol{\pi}$ de la DTMC.*

L'ajout d'un nouveau groupe de données représenté par la distribution (H_1) entraîne la modification de la distribution H_3. Ainsi, après les arrivées, la longueur du tampon est donnée par la distribution H_q :

$$H_q = H_3 \otimes H_1. \tag{8.3}$$

où l'opérateur $\otimes$ correspond à la convolution des distributions.

b) Distribution du processus de sortie de la file : H_5

Nous distinguons deux cas : service variable *i.e.,* service décrit sous forme d'histogramme et service constant.

Proposition 8.5 (Nombre d'unités de données en sortie pour un service variable) *L'histogramme de sortie H_5 est défini sur $\mathcal{S}$ tel que $\mathcal{S} = \{k \,|\, \forall\, i \in E^{H_q} \text{ et } \forall\, j \in E^{H_2},\ k = \min(i, j)\}$ et calculé à partir de H_q comme suit :*

$$H_5[w] = \sum_{i \in E^{H_q}} \sum_{j \in E^{H_2}} H_q[i]\, H_2[j]\, \mathbb{1}_{\{\min(i,j)=w\}}, \quad \forall\, w \in \mathcal{S} \tag{8.4}$$

Dans le cas où le service est constant, la proposition 8.5 peut être écrite de manière plus simple. En effet, pour une capacité de service notée C et sachant que les éléments du réseau sont work conserving, le nombre d'unités de données émises correspond au minimum entre la capacité du lien et la taille de la file d'attente avant le service.

Proposition 8.6 (Nombre d'unités de données en sortie pour un service constant) *L'histogramme de sortie H_5 est calculé comme suit :*

$$\begin{cases} H_5[w] = & H_q[w] & \text{si } w < \mathrm{C}, \\ H_5[\mathrm{C}] = & \sum_{w \geq \mathrm{C}} H_q[w] & \text{sinon.} \end{cases}$$

c) **Distribution des pertes :** H_L

Sous la politique Tail Drop, la distribution H_L représente le nombre d'unités de données perdues à l'entrée de la file d'attente finie. Nous rappelons que la politique *Tail Drop* consiste à mettre dans la file d'attente les unités de données qui arrivent si la longueur de la file est inférieure à la capacité du tampon, et sinon à les rejeter. Donc, le nombre de pertes à l'instant n est de $(X(n) + A - \mathrm{C} - \mathrm{B})^+$.

Dans le cas où le service considéré est variable, la distribution de perte est déterminée comme suit. Soit $(-H_2)$ la distribution définie par :

$$E^{-H_2} = \{x | -x \in E^{H_2}\} \quad \text{et} \quad -H_2(x) = H_2(-x).$$

Pour une description abstraite, nous calculons d'abord $H_n = H_3 \otimes H_1 \otimes (-H_2)$ puis nous réécrivons et tronquons la distribution. Notons que l'ensemble E^{H_n} peut contenir certains éléments négatifs.

Proposition 8.7 (Pertes avec service variable) *La distribution des pertes sous la politique Tail Drop avec un service variable est :*

$$\begin{cases} H_L[k - \mathrm{B}] &= H_n[k] & k > \mathrm{B} \\ H_L[0] &= \sum_{k \leq \mathrm{B}} H_n[k] \end{cases}$$

Proposition 8.8 (Pertes avec service déterministe) *La distribution des pertes sous la politique Tail Drop avec un service déterministe est :*

$$\begin{cases} H_L[k - \mathrm{B} - C] &= H_q[k] & k > \mathrm{B} + C \\ H_L[0] &= \sum_{k \leq \mathrm{B}+C} H_q[k] \end{cases}$$

Ainsi, la probabilité de perte notée P_L peut être définie comme suit :

$$P_L = \frac{\mathbb{E}[H_L]}{\mathbb{E}[H_1]}.$$

d) **Distribution des temps de réponse :** H_4

La détermination de l'histogramme associé au temps de réponse dépend de la politique de service. Nous considérons ici deux politiques : la politique *work conserving* générale et la politique FIFO. Pour les deux politiques, nous dérivons respectivement des bornes supérieures et inférieures. Pour FIFO, nous présentons également le résultat exact, mais dont la complexité de calcul est très élevée. Nous notons que pour la politique FIFO les bornes sont plus précises.

Ce point sera mis en évidence lors de la présentation des algorithmes qui permettent d'obtenir les différentes distributions.

Nous commençons par considérer l'arrivée d'une unité de données dans l'élément du réseau et nous modélisons son temps de réponse dans la file d'attente à travers une chaîne de Markov absorbante. Et nous calculons la distribution du temps avant absorption. Comme l'approche est plus simple avec FIFO, nous commençons avec cette politique avant d'étudier le cas général.

♦ **Politique de service FIFO :** Les distributions du temps de réponse (résultats exacts et bornes stochastiques) sont obtenues avec la même technique. Nous conditionnons sur l'état vu par un groupe de données en arrivées. Ensuite, nous devons prendre en compte la place qu'occupe l'unité de données dans le groupe. Nous développons d'abord la démarche suivie pour le calcul des bornes avant d'effectuer le calcul pour la distribution exacte.

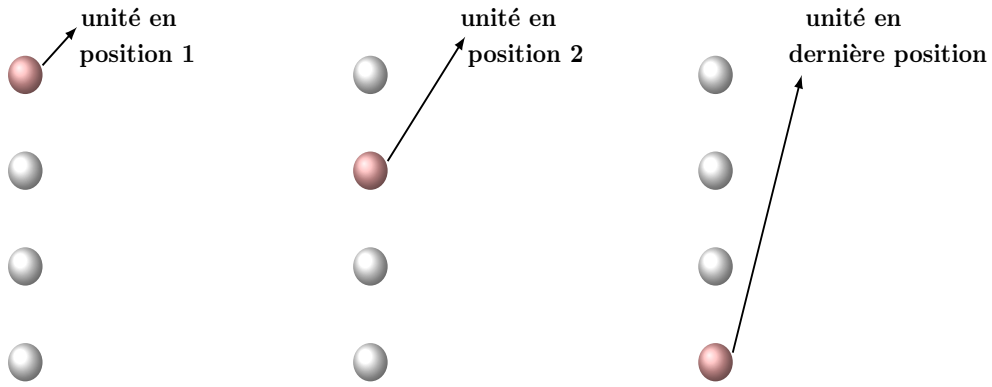


FIGURE 8.2 – Position d'une unité dans un groupe de données.

Pour les bornes supérieures, nous considérons un groupe avec une taille maximale et nous supposons que l'unité de données est la dernière du groupe. Pour la borne inférieure, nous supposons que l'unité de données est au début du groupe (*i.e.*, nous supposons que la taille du groupe est de 1).

En premier, nous construisons la matrice stochastique $\mathbf{R}$ comme suit : les transitions de $\mathbf{R}$ sont obtenues en laissant la distribution de H_1 être une distribution de Dirac à 0 (*i.e.,* $H_1[0] = 1$) et donc rendre l'état 0 absorbant. Intuitivement, la matrice $\mathbf{R}$ représente la chaîne lorsque les arrivées sont stoppées juste après l'arrivée de l'unité de données.

$$\mathbf{R} = \sum_{j \in E^{H_2}} \mathbf{U}_{-j}\, H_2[j].$$

La distribution de la borne inférieure du temps de réponse notée H_4^l est obtenue de manière algorithmique (voir algorithme 27). Pour une constante donnée T inférieure à B, nous calculons les temps de réponse de façon exacte entre 1 et T et nous concentrons la masse résiduelle de probabilité en $\mathrm{T}+1$ pour une borne inférieure. Cela permet d'assurer l'ordre stochastique avec la distribution exacte, qui elle est définie de 1 à B.

En premier lieu, nous définissons la distribution α_1 représentant la longueur de la file d'attente incluant la nouvelle unité de données. Ensuite, par un procédé itératif sur les instants de temps nous calculons la probabilité que l'unité de données quitte l'élément du réseau à cet

Algorithme 27 : Borne inférieure du temps de réponse sous la discipline FIFO

ENTRÉES : $\boldsymbol{\pi}$: vecteur $1 \times \mathrm{B} + 1$; $\mathbf{R}$: matrice $\mathrm{B} + 1 \times \mathrm{B} + 1$; T : constante.
SORTIES : Distribution borne inférieure du temp de réponse : H_4^l.

1: $\alpha_1[i] = \boldsymbol{\pi}[i-1], \quad 0 < i < \mathrm{B}$.
2: $\alpha_1[0] = 0$; $\alpha_1[\mathrm{B}] = \boldsymbol{\pi}[\mathrm{B}-1] + \boldsymbol{\pi}[\mathrm{B}]$.
3: **Pour** ($j = 0$ to B $\mid \alpha_1[j] > 0$) **faire**
4: $\quad \delta = e_j$.
5: $\quad$ **Pour** $t = 1$ to T **faire**
6: $\quad\quad \beta = \delta\ \mathbf{R}$.
7: $\quad\quad \eta_j[t] = \beta[0] - \delta[0]$; $\delta = \beta$.
8: $\quad$ **Fin pour**
9: $\quad \eta_j[\mathrm{T}+1] = 1 - \sum_{t=1}^{\mathrm{T}} \eta_j[t]$.
10: $\quad \eta_j[t] = 0, \quad \forall t > \mathrm{T}+1$.
11: **Fin pour**
12: $H_4^l[t] = \sum_j \alpha_1[j]\ \eta_j[t], \quad 0 < t \leq \mathrm{B}$.

instant précis. Clairement, pour t entre 1 et T, $H_4^l[t]$ correspond à la distribution exacte du temps de réponse de la première unité de données quand il reste moins de $\mathrm{T} + 1$ slots dans l'élément de réseau. La borne inférieure est obtenue en considérant le délai minimal plus grand que T slots pour la partie restante de la distribution. On précise que la valeur de T ($T \leq B$) est telle que plus elle est grande plus la distribution de la borne inférieure du temps de réponse est précise.

Algorithme 28 : Borne supérieure du temps de réponse sous la discipline FIFO

ENTRÉES : $\boldsymbol{\pi}$: vecteur $1 \times \mathrm{B} + 1$; $\mathbf{R}$: matrice $\mathrm{B} + 1 \times \mathrm{B} + 1$;
N : taille du groupe de données et T : constante.
SORTIES : Distribution borne supérieure du temps de réponse : H_4^u.

1: $\alpha_{\mathrm{N}}[i] = 0, \quad \forall\ 0 \leq i < \mathrm{N}$.
2: $\alpha_{\mathrm{N}}[i] = \pi(i - \mathrm{N}), \quad \forall\ \mathbf{N} \leq i < \mathbf{B}$.
3: $\alpha_{\mathrm{N}}[B] = 1 - \sum_{i=0}^{B-1} \alpha_{\mathrm{N}}[i]$;
4: **Pour** ($j = 0$ à B $\mid \alpha_{\mathrm{N}}[j] > 0$) **faire**
5: $\quad \delta = e_j$.
6: $\quad$ **Pour** $t = 1$ à T **faire**
7: $\quad\quad \beta = \delta\ \mathbf{R}$.
8: $\quad\quad \eta_j[t] = \beta[0] - \delta[0]$; $\delta = \beta$.
9: $\quad$ **Fin pour**
10: $\quad \eta_j[t] = 0, \quad \forall t = \mathrm{T}+1 \cdots \mathrm{B}-1$.
11: $\quad \eta_j[\mathrm{B}] = 1 - \sum_{t=1}^{\mathrm{T}} \eta_j[t]$.
12: **Fin pour**
13: $H_4^u[t] = \sum_j \alpha_{\mathrm{N}}[j]\ \eta_j[t], \quad 0 < t \leq \mathrm{B}$.

Nous utilisons une approche similaire pour la distribution de la borne supérieure H_4^u. Premièrement, nous supposons que pour une taille de groupe égale à N, l'unité de données se place en dernière position dans le groupe. Nous calculons la distribution de la taille de la file d'attente (appelée α_{N}) incluant le groupe de N unités de données. Nous notons que ce groupe de données est entré au début de l'intervalle de temps. Ensuite, nous calculons sur les instants jusqu'au temps T, la probabilité qu'une unité de données finisse son service. Le calcul de la borne supérieure du temps de réponse sous la discipline FIFO est décrit à travers l'algorithme 28. Le reste de la distribution est fixé à $H_4^u[\mathrm{B}]$, car l'élément est work conserving et de discipline FIFO. Le temps de réponse est donc inférieur à la capacité du tampon.

Nous décrivons maintenant le cas général :

Proposition 8.9 *Soit H_1 la distribution des arrivées par groupe. Nous considérons deux distributions discrètes ξ et φ tel que :*
• *ξ est la taille du groupe quand il n'est pas vide.*
• *φ modélise la position de l'unité de donnée dans le groupe de taille N. La variable φ est définie sur $E^\varphi = \{\forall i \in \mathbb{N}^* \mid i \leq N\}$, tel que*

$$\mathbf{p}^\xi[\ell] = \mathrm{Prob}(E^\xi = \ell) = \frac{H_1[\ell]}{1 - H_1[0]}, \quad \forall \ell \in E^\xi.$$

et une probabilité conditionnelle de $\mathrm{Prob}(E^\varphi = i | E^\xi = \ell) = \frac{1}{\ell}\, \mathbb{1}_{i \leq \ell}$.

Nous déduisons que pour tout $i \in E^\varphi$:

$$\mathbf{p}^\varphi[i] = Pr(E^\varphi = i) = \frac{1}{1 - H_1[0]} \sum_{\ell=1}^{N} \frac{1}{\ell}\, H_1[\ell]\, \mathbb{1}_{i \leq \ell}.$$

$\mathbf{p}^\varphi[i]$ est la probabilité que l'unité de donnée soit à la $i$ème position dans le groupe.

Algorithme 29 : Temps de réponse exact sous la discipline de service FIFO

ENTRÉES : $\boldsymbol{\pi}$: vecteur $1 \times \mathrm{B} + 1$; $\mathbf{R}$: matrice $\mathrm{B} + 1 \times \mathrm{B} + 1$; φ : distribution discrète.
SORTIES : Distribution exacte du temps de réponse : H_4.

1: **Pour** $i \in E^\varphi$ **faire**
2: $\quad \alpha_i[j] = \boldsymbol{\pi}[j - i], \text{pour } i \leq j < \mathrm{B}.$
3: $\quad \alpha_i[j] = 0, \text{pour } j < i\,;\ \alpha_i[\mathrm{B}] = \sum_{\mathrm{j}=1}^{\mathrm{B}-1} \alpha_{\mathrm{i}}[\mathrm{j}].$
4: **Fin pour**
5: $\gamma = \sum_{i \in E^\varphi} p^\varphi[i]\ \alpha_i.$
6: **Pour** $j = 0$ to $\mathrm{B} \mid \gamma[\mathrm{j}] > 0$ **faire**
7: $\quad \delta_j = e_j.$
8: $\quad$ **Pour** $t = 1$ to B **faire**
9: $\quad\quad \beta = \delta_j \,.\, \mathbf{R}.$
10: $\quad\quad \eta_j[t] = \beta[0] - \delta_j[0]\,;\ \delta_j = \beta.$
11: $\quad$ **Fin pour**
12: **Fin pour**
13: $H_4[t] = \sum_j \gamma[j]\ \eta_j[t], \quad 0 < t \leq \mathrm{B}.$

Pour construire la distribution exacte du temps de réponse, nous considérons la même matrice $\mathbf{R}$ utilisée précédemment. Nous définissons la distribution α_i correspondant à la longueur de la file d'attente incluant la $i$ème unité de donnée. Ensuite, nous calculons par itération sur les instants de temps la probabilité qu'une unité de donnée quitte le système (voir algorithme 29).

♦ **Politique de service work conserving :** La borne inférieure est simplement donnée par une unité de temps, car les unités de données peuvent quitter immédiatement l'élément de réseau. Nous rappelons que pour l'étude de file d'attente à temps discret, nous avons posé l'hypothèse que les départs ont lieu à la fin de l'intervalle de temps. Par conséquent, une unité de donnée sortant dans le même intervalle temporel que l'arrivée a un temps de réponse égal à 1.

Les bornes supérieures sont obtenues en supposant que l'unité de donnée sera la dernière à quitter la file d'attente. En effet, avec une politique work conserving, nous ne connaissons pas l'ordonnancement des unités de données à chaque intervalle de temps dans le futur. D'après [Cha00], la seule propriété que nous pouvons garantir, c'est que l'unité de donnée va être servie avant la fin de la période d'occupation. Pour ce faire, nous modifions la matrice $\mathbf{P}$ en rendant l'état 0 absorbant. Et nous calculons le temps avant d'être absorbé par l'état 0 après conditionnement de l'état initial vu par l'unité de donnée entrante et qui correspond à la distribution stationnaire. Soit $\mathbf{Q}$ la matrice modifiée afin de rendre l'état 0 absorbant.

Algorithme 30 : Borne supérieure du temps de réponse sous la discipline work conserving

ENTRÉES : $\boldsymbol{\pi}$: vecteur $1 \times \mathrm{B}+1$; $\mathbf{Q}$: matrice $\mathrm{B}+1 \times \mathrm{B}+1$; ϵ : facteur de précision.
SORTIES : Distribution borne supérieure du temps de réponse : H_4^u.

1: **Pour** ($j = 0$ à $\mathrm{B} \mid \boldsymbol{\pi}[\mathrm{j}] > 0$) **faire**
2: $\quad \delta_j = e_j$; $\mathrm{T} = 0$.
3: $\quad$ **Répéter**
4: $\quad\quad \mathrm{T} = \mathrm{T} + 1$.
5: $\quad\quad \beta = \delta_j \ \mathbf{Q}$.
6: $\quad\quad \eta_j[\mathrm{T}] = \beta[0] - \delta_j[0]$; $\delta_j = \beta$.
7: $\quad$ **Jusqu'à** $\sum_{t=1}^{\mathrm{T}} \eta_j[t] > 1 - \epsilon$.
8: **Fin pour**
9: $H_4^u[t] = \sum_j \boldsymbol{\pi}[j]\ \eta_j[t], \quad t > 0$.

Notons que cette distribution peut nécessiter un très grand nombre d'étapes de calcul jusqu'à atteindre la précision souhaitée.

Nous proposons d'illustrer à travers un petit exemple le calcul de la distribution du temps de réponse obtenue via les différents algorithmes énoncés.

Exemple 8.3 *Pour une file d'attente simple alimentée par une distribution d'arrivée* H_1 *munie d'un service déterministe noté* S *et une capacité de tampon finie* B*, nous considérons respectivement trois cas. Le premier exemple consiste à prendre la distribution d'arrivée suivante :* $H_1 = [0.1, 0.3, 0.2, 0.1, 0.3]$ *définie sur* $E^{H_1} = \{1, 2, 3, 4, 5\}$ *avec un service constant de taux* $\mathrm{S} = 2$ *et un tampon de capacité* $\mathrm{B} = 6$*. Pour le second cas, nous générons*

aléatoirement une distribution d'arrivée H_1 *telle que* $|H_1| = 100$ *états variant dans un interval de* $]0, 300]$ *avec* $\mathbb{E}[H_1] = 102.15$. *Le service est pris égal à* $\mathrm{S} = 110$ *et le tampon est considéré fini de capacité* $\mathrm{B} = 1000$. *Pour le dernier cas, nous considérons la même distribution d'arrivée et le même service que le deuxième cas mais avec une capacité de tampon égale à* $\mathrm{B} = 1500$.

Sous la politique de service FIFO respectivement work-conserving nous calculons la distribution du temps de réponse en utilisons les différents algorithmes développés ci-dessus. En utilisant l'algorithme 29 et les algorithmes 27 et 28 avec $\mathrm{T} = 100$, *nous déterminons respectivement la distribution exacte, la distribution borne inférieure et la distribution borne supérieure du temps d'attente sous la discipline de service FIFO. Pour un facteur de précision* $\epsilon = 10^{-6}$ *et sous la discipline de service work-conserving, l'algorithme 30 nous permet de définir la distribution borne supérieure du temps de réponse. Les résultats obtenus sont résumés dans le tableau ci-après (espérance des temps de réponse noté* $\mathbb{E}[H4]$ *et temps de calcul noté* Temps d'exc.*).*

		FIFO			**Work-conserving**
		Exact $\mathbb{E}[H_4]$	Borne inférieure $\mathbb{E}[H_4^l]$	Borne supérieure $\mathbb{E}[H_4^u]$	Borne supérieure $\mathbb{E}[H_4^u]$
Cas 1	$\mathbb{E}[H4]$	1.99845	1.9965	1.99992	162950
	Temps d'exc. (s)	0.0286	0.0111	0.0049	774.237
Cas 2	$\mathbb{E}[H4]$	1.7137	1.32764	2.12366	25.4081
	Temps d'exc. (s)	288.662	31.068	27.680	103.068
Cas 3	$\mathbb{E}[H4]$	1.79168	1.4042	2.20325	27.2967
	Temps d'exc. (s)	2158.29	185.929	173.862	744.932

TABLE 8.1 – Détermination des temps de réponse pour une file d'attente simple.

D'après les résultats présentés dans le tableau ci-dessus, nous constatons que sous la discipline FIFO, les algorithmes 27 et 28 fournissent un bon encadrement du temps de réponse exact. Les algorithmes de bornes permettent ainsi de trouver une solution très intéressante en terme de précision et de temps de calcul. Nous notons également que lorsque le support de la distribution d'arrivée est grand et que la capacité du tampon est importante, le calcul de la distribution exacte du temps de réponse devient très difficile voir impossible à déterminer. Par conséquent, l'utilisation des algorithmes 27 et 28 peut représenter un bon compromis entre la qualité des résultats et la complexité de calcul.

Nous remarquons également que le calcul de la borne supérieure sous la discipline work-conserving prend un temps de calcul plus important. Ce constat est conforme à la remarque énoncée lors de la présentation de l'algorithme 30.

8.2 Résultats de monotonie stochastique de la file d'attente simple

Nous prouvons maintenant certaines propriétés de monotonie pour une file d'attente. Nous voulons montrer que si les paramètres d'entrée (*i.e.,* les histogrammes d'arrivée et de service) sont comparables au sens de l'ordre $\leq_{st}$, alors les paramètres de sortie (*i.e.,* les histogrammes d'occupation du tampon, des flux de sortie et des temps de réponse) sont également comparables avec le même ordre. Cette propriété sera appelée la H-monotonie des histogrammes des paramètres de sortie pour les histogrammes d'arrivées H_1 et services H_2. Nous montrons tout d'abord que les chaînes de Markov sont comparables.

Proposition 8.10 (H-Monotonie de la DTMC) *Soit $\{X_i^a, i \geq 0\}$ une DTMC associée à l'histogramme d'arrivée (resp. service) H_1^a (resp. H_2^a), et soit $\{X_i^b, i \geq 0\}$ une DTMC associée à l'histogramme d'arrivée (resp. service) H_1^b (resp. H_2^b). Nous supposons que les conditions de stationnarité (proposition 8.3) sont satisfaites pour les deux chaînes.*

$$Si \;\; H_1^a \leq_{st} H_1^b \; et \; H_2^a \geq_{st} H_2^b, \;\; alors \;\; \{X_i^a, i \geq 0\} \leq_{st} \{X_i^b, i \geq 0\}.$$

PREUVE. Par souci de clarté, la preuve donnée ici est fondée sur l'équation 8.1. Les arrivées et services aléatoires A et S sont définis par les histogrammes H_1 et H_2. Soit A^a (resp. A^b) la variable aléatoire associée à l'histogramme H_1^a (resp. H_1^b), de même, la variable aléatoire S^a (resp. S^b) correspondant à l'histogramme H_2^a (resp. H_2^b). Puisque $H_1^a \leq_{st} H_1^b$, nous avons $A^a \leq_{st} A^b$. Nous avons également $-S^a \leq_{st} -S^b$, à partir de la propriété 2.4 et l'hypothèse que $H_2^a \geq_{st} H_2^b$. Nous considérons maintenant deux DTMCs $\{X^a\}$ et $\{X^b\}$ respectivement régies par les processus A^a, $-S^a$ et A^b, $-S^b$. Nous montrons par induction que $X_n^a \leq_{st} X_n^b, \; \forall n$.
Soit $X_0^a = X_0^b = 0$. Puisque, $\max(0, .)$ et $\min(\mathrm{B}, .)$ sont des fonctions croissantes, il résulte des propriétés de conservation de l'ordre $\leq_{st}$ (Property 2.4) que $X_1^a \leq_{st} X_1^b$. De même, si $X_n^a \leq_{st} X_n^b$, alors à partir de la propriété 2.4 nous avons : $X_{n+1}^a \leq_{st} X_{n+1}^b$. ■

Dans la suite, nous montrons la H-monotonie des distributions H_3 et H_5.

Théorème 8.1 (H-monotonie pour H_3 et H_5)

$$\text{Si } \; H_1^a \leq_{st} H_1^b \text{ et } H_2^a \geq_{st} H_2^b \Longrightarrow \left\{ \begin{array}{l} H_3^a \leq_{st} H_3^b. \\ H_5^a \leq_{st} H_5^b. \end{array} \right.$$

PREUVE. D'après la proposition 8.10, les DTMCs correspondantes sont également comparables et elles sont ergodiques par hypothèse. Ainsi, les distributions stationnaires sont comparables en raison du théorème 2.6 (*i.e.,* $\boldsymbol{\pi}^a \leq_{st} \boldsymbol{\pi}^b$) et $H_3^a \leq_{st} H_3^b$ d'après la proposition 8.4. La comparaison des processus de sortie (H_5) découle de la proposition 8.5. En effet, la convolution et la troncature des distributions conserve l'ordre stochastique. ■

Théorème 8.2 (H-Monotonie pour H_L)
Si $H_1^a \leq_{st} H_1^b$ et $H_2^b \leq_{st} H_2^a$ et que l'élément est work-conserving et utilisé sous la politique Tail Drop, alors les pertes sont monotones, i.e., $H_L^a \leq_{st} H_L^b$.

PREUVE. D'après le théorème 8.1, les hypothèses de la proposition impliquent que les distributions stationnaires sont également comparables : *i.e.,* $H_3^a \leq_{st} H_3^b$. Clairement, $H_2^b \leq_{st} H_2^a$ implique que $-(H_2^a) \leq_{st} -(H_2^b)$. De plus, d'après Stoyan [MS02] la convolution des distribuions discrètes conserve l'ordre stochastique fort. Par conséquent, $H_3^a \otimes H_1^a \otimes (-H_2^a) \leq_{st} H_3^b \otimes H_1^b \otimes (-H_2^b)$. De plus, elle ne change pas la relation d'ordre entre les distributions. ■

Théorème 8.3 (H-monotonie pour H_4)
Sous la discipline FIFO,

$$si\ H_1^a \leq_{st} H_1^b\ et\ H_2^a \geq_{st} H_2^b \Longrightarrow \begin{cases} H_4^{l,a} \leq_{st} H_4^{l,b}. \\ H_4^{u,a} \leq_{st} H_4^{u,b}. \end{cases}$$

PREUVE. Nous donnons la preuve pour la borne inférieure. La preuve de la borne supérieure est similaire et donc omise. Nous notons par α_1^a (resp. α_1^b) la longueur du tampon vue par le nouveau client (y compris lui même) avec les distributions d'arrivée et de service H_1^a, H_2^a (resp. H_1^b, H_2^b). Puisque α_1 est calculé par $\boldsymbol{\pi}$ (H_3), il résulte du théorème 8.1 et de l'algorithme 27, que $\alpha_1^a \leq_{st} \alpha_1^b$. De même, la matrice $\mathbf{R}$ est construite par l'arrêt des arrivées, les DTMCs absorbantes correspondantes sont comparables au sens de l'ordre $\leq_{st}$, ainsi il découle de la proposition 2.3 que $\forall j$ le temps d'absorption quand il y a j unités de données, η_j sont comparables : $\eta_j^a \leq_{st} \eta_j^b$. Et nous avons la monotonie : $\eta_j^a \leq_{st} \eta_{j+1}^a$. Puisque $\eta_i^a \leq_{st} \eta_i^b$, nous pouvons écrire : $\forall 0 < t \leq \mathrm{B} :\ \sum_t^{\mathrm{B}} H_4^{l,a}[t] = \sum_i \alpha_1^a[i] \sum_t^{\mathrm{B}} \eta_i^a[t] \leq \sum_i \alpha_1^a[i] \sum_t^{\mathrm{B}} \eta_i^b[t]$. La dernière expression peut être écrite comme suit

$\sum_i \alpha_1^b[i] \sum_t^{\mathrm{B}} \eta_i^b[t] + \sum_i (\alpha_1^a[i] - \alpha_1^b[i]) \sum_t^{\mathrm{B}} \eta_i^b[t]$.

À partir du lemme 2.1, la dernière expression est inférieure à $\sum_i \alpha_1^b[i] \sum_t^{\mathrm{B}} \eta_i^b[t] = \sum_t^{\mathrm{B}} H_4^{l,b}[t]$.

Ainsi, $H_4^{l,a} \leq_{st} H_4^{l,b}$. ■

Théorème 8.4 (H-monotonie pour H_4)
Sous la discipline work-conserving,

$$si\ H_1^a \leq_{st} H_1^b\ et\ H_2^a \geq_{st} H_2^b \Longrightarrow H_4^{u,a} <_{st} H_4^{u,b}.$$

La preuve peut être faite de manière similaire que le théorème précédent.

8.3 Exemples numériques

Dans le but d'évaluer les mesures de performance présentées dans ce chapitre et de montrer la notion de monotonie garantie par notre approche de bornes, nous proposons d'étudier une file d'attente simple alimentée par la trace de trafic Mawi (figure 5.1) en considérons respectivement un service déterministe et un service variable.

8.3.1 File d'attente simple avec service déterministe

Sous la discipline de service FIFO, nous considérons un service constant de taux $C = 110\,Mb/s$ et une unité de données égale à $D = 5\,Kb$. Nous rappelons que la notion d'unité de donnée (D) est considérée qu'au niveau du calcul de nos bornes pour accélérer le temps de calcul. Les résultats obtenus sont ensuite convertis en bits pour être comparés avec les résultats exacts. Nous faisons varier la taille du tampon (B) de $5\,10^6\ bits$ à $3\,10^7\ bits$. Pour un nombre de bins égal à 20 et 100, nous présentons les résultats pour : les probabilités de pertes (tableau 8.2), la longueur moyenne du tampon (tableau 8.3), le nombre moyen de données en sorties (tableau 8.4), les temps de réponse (tableau 8.6) et les temps d'exécution globaux en secondes nécessaires au calcul des différents paramètres (tableau 8.5). Nous précisons que le calcul des temps de réponse n'est effectué que pour notre méthode de bornes stochastiques, car le calcul exact de la distribution du temps de réponse via l'algorithme 29 requiert un temps de calcul trop important.

		bins	20		100	
		Exact	$L.b$	$U.b$	$L.b$	$U.b$
capacité du tampon (10^6 bits)	5	0.011659	0.004530	0.024995	0.010207	0.014282
	7	-	0.002154	0.022152	0.006779	0.010725
	10	-	0.000753	0.020169	0.004084	0.007824
	11	-	0.000535	0.019798	0.003510	0.007179
	13	-	0.000271	0.019289	0.0026389	0.006169
	15	-	0.000138	0.018978	0.002018	0.005411
	16	-	9.88967e-05	0.018871	0.001774	0.005099
	18	-	5.05533e-05	0.018718	0.001380	0.004574
	20	-	2.58626e-05	0.018621	0.001083	0.004149
	30	-	9.09447e-07	0.018472	0.000345	0.002849

TABLE 8.2 – Probabilités de pertes ($\mathbb{E}[H_L]$).

		bins	20		100	
		Exact	$L.b$	$U.b$	$L.b$	$U.b$
capacité du tampon (10^6 bits)	5	2.29103e+6	1.6084e+06	3.07486e+06	2.18055e+06	2.48001e+06
	7	-	1.95239e+06	4.55449e+06	2.93708e+06	3.48558e+06
	10	-	2.26231e+06	6.9763e+06	3.94833e+06	4.99816e+06
	11	-	2.32704e+06	7.82881e+06	4.25381e+06	5.50339e+06
	13	-	2.41796e+06	9.58909e+06	4.81967e+06	6.51534e+06
	15	-	2.47307e+06	1.14103e+07	5.32826e+06	7.52926e+06
	16	-	2.49158e+06	1.23394e+07	5.5622e+06	8.03694e+06
	18	-	2.51662e+06	1.42273e+07	5.99169e+06	9.05375e+06
	20	-	2.53112e+06	1.61465e+07	6.37322e+06	1.00725e+07
	30	-	2.54902e+06	2.5986e+07	7.69442e+6	1.51947e+07

TABLE 8.3 – Longueur moyenne du tampon ($\mathbb{E}[H_3]$).

		bins	20		100	
		Exact	$L.b$	$U.b$	$L.b$	$U.b$
capacité du tampon (10^6 bits)	5	4.32461e+06	4.25772e+06	4.37068e+06	4.3158e+06	4.3382e+06
	7	-	4.26788e+06	4.38343e+06	4.33075e+06	4.35385e+06
	10	-	4.27387e+06	4.39231e+06	4.3425e+06	4.36662e+06
	11	-	4.27481e+06	4.39398e+06	4.345e+06	4.36945e+06
	13	-	4.2765e+06	4.39626e+06	4.3488e+06	4.3739e+06
	15	-	4.27418e+06	4.39765e+06	4.35151e+06	4.37724e+06
	16	-	4.27667e+06	4.39813e+06	4.35258e+06	4.37861e+06
	18	-	4.27688e+06	4.39882e+06	4.35429e+06	4.38092e+06
	20	-	4.27699e+06	4.39925e+06	4.35559e+06	4.38279e+06
	30	-	4.27709e+06	4.39992e+06	4.35881e+06	4.38851e+06

TABLE 8.4 – Nombre moyen de données en sorties ($\mathbb{E}[H_5]$).

	bins	20		100	
		$L.b$	$U.b$	$L.b$	$U.b$
capacité du tampon (10^6 bits)	5	1.06978	2	1.13705	2
	7	1.13663	2	1.28942	2
	10	1.19593	2.96356	1.49748	2.87067
	11	1.20923	2.97143	1.56311	2.88165
	13	1.22534	2.98225	1.66242	2.89889
	15	1.23906	3.91087	1.79405	3.55112
	16	1.24284	3.92923	1.84255	3.57818
	18	1.24811	4.82574	1.93587	4.09036
	20	1.2513	4.88943	2.02227	4.17949
	30	1.25507	6.89557	2.29392	5.38591

TABLE 8.5 – Temps de réponse sous la discipline de service FIFO ($\mathbb{E}[H4]$).

En raison d'un temps de calcul trop élevé et une augmentation conséquente de l'espace d'état de la distribution exacte à chaque étape de calcul, le calcul exact des mesures de performance de la file d'attente n'est effectuée que pour une capacité de tampon égale à $\mathrm{B} = 5\,Mb$. Nous notons par "-" les valeurs non calculées. Nous pouvons remarquer que pour cette capacité de tampon, les résultats ont pu être déterminés au bout d'un temps d'environ 5 heures et 23 minutes, ce qui s'avère être trop important par rapport aux temps de calcul de nos bornes (3 secondes maximum). De plus, nos bornes sont de qualité surtout quand le nombre de bins est grand, par exemple dans le tableau 8.4, pour un nombre de bins égal à 100, nous avons une espérance des données en sorties de $4.3158e$+06 pour la borne inférieure et de

		bins	20		100	
		Exact	$L.b$	$U.b$	$L.b$	$U.b$
capacité du tampon (10^6 bits)	5	19383.1	2.71887	2.39153	3.12165	2.13099
	7	-	8.09863	2.52294	9.36486	3.12869
	10	-	21.8618	73.8284	22.8557	68.5851
	11	-	29.2602	128.004	27.2997	118.198
	13	-	45.4277	282.721	44.6385	267.21
	15	-	71.6268	535.6	64.9832	486.109
	16	-	86.636	657.672	77.5449	630.623
	18	-	122.282	1040.19	111.035	995.888
	20	-	175.361	1546.05	156.967	1461.22
	30	-	570.721	6246.19	498.978	6123.36

TABLE 8.6 – Temps d'execution (secondes).

$4.3382e$+06 pour la borne supérieure. Nous remarquons également que vu la qualité des bornes (la différence entre la borne supérieure et la borne inférieure est relativement faible) il n'est pas nécessaire de calculer les valeurs exactes.

Concernant les temps d'exécution, nous constatons que les temps de calcul d'une borne (inférieure ou supérieure) pour un nombre de bins égal à 20 et un nombre de bins égal 100 sont proches. De plus, pour une majorité des capacités de tampon, nous observons que le calcul de cette dernière pour bins=100 est plus rapide que pour bins=20. Ceci est dû expressément aux temps requis pour la détermination d'une borne sur les temps de réponse. Prenant à titre d'exemple le calcul de la borne inférieure pour une capacité de tampon de 1.3 Mb/s. Nous notons que le temps nécessaire au calcul des mesures de performance $\mathbb{E}[H_L]$, $\mathbb{E}[H_3]$ et $\mathbb{E}[H_5]$ requière un temps de 0.7529 secondes pour un nombre de bins égal à 20 et 1.9673 secondes pour un nombre de bins égal à 100. Cependant, le calcul de la borne inférieure des temps de réponse via l'algorithme 27, nécessite un temps de calcul de $44.6748\,s$ pour un nombre de bins égal à 20 et un temps de $42.6712\,s$ pour le nombre de bins égal à 100.

Pour ce qui est de la précision des résultats obtenus, nous remarquons que pour la première valeur de la capacité de tampon, notre approche nous permet de déterminer des bornes intéressantes sur la solution exacte, qui deviennent très proches lorsqu'on augmente le nombre de bins (états) considéré. Pour le reste, nous observons que l'augmentation du nombre de bins (états) permet de resserrer l'intervalle de l'erreur commise et offre un encadrement pertinent du résultat exact.

Pour conclure, nous distinguons que l'utilisation de nos bornes stochastiques nous permet d'assurer l'obtention de bornes sur les mesures de performance grâce à la notion de monotonie prouvée dans ce chapitre.

8.3.2 File d'attente simple avec service variable

Nous abordons ici le second cas, qui consiste à considérer un nœud simple étudié sous la trace de trafic Mawi (figure 5.1) avec un service variable. Le service est représenté sous forme d'histogramme dont la distribution est donnée comme suit : $H_2 = [0.3, 0.4, 0.3]$ définie sur $E^{H_2} = \{85, 125, 140\}\, Mb/s$ et l'unité de données est prise égale à $D = 5\, Kb$. Nous faisons varier la taille du tampon (B) de $5\,10^6\ bits$ à $3\,10^7\ bits$.

		bins	20		100	
		Exact	$L.b$	$U.b$	$L.b$	$U.b$
capacité du tampon (10^6 bits)	5	0.005815	0.003250	0.010283	0.005307	0.00668345
	7	-	0.001081	0.005488	0.002160	0.002991
	10	-	0.000212	0.002287	0.000582	0.000938
	11	-	0.000123	0.001724	0.000378	0.000641
	13	-	4.19071e-05	0.000987	0.000159	0.000300
	15	-	1.42301e-05	0.000569	6.73425e-05	0.000141
	16	-	8.29284e-06	0.000432	4.37849e-05	9.6732e-05
	18	-	2.81657e-06	0.000250	1.85147e-05	4.5509e-05
	20	-	9.56654e-07	0.000145	7.83049e-06	2.14189e-05
	30	-	4.32475e-09	9.5969e-06	1.06025e-07	4.95379e-07

TABLE 8.7 – Probabilités de pertes.

Pour un nombre de bins égal à 20 et 100, nous présentons les résultats sur les probabilités de pertes (tableau 8.7), la longueur moyenne du tampon (tableau 8.8), le nombre moyen de données en sorties (tableau 8.9), les temps de réponse (tableau 8.10) et les temps d'exécution en secondes (tableau 8.11) nécessaires au calcul des différents paramètres.

		bins	20		100	
		Exact	$L.b$	$U.b$	$L.b$	$U.b$
capacité du tampon (10^6 bits)	5	1.28131e+06	1.02577e+06	1.61216e+06	1.23791e+06	1.35486e+06
	7	-	1.14433e+06	2.01908e+06	1.44075e+06	1.61441e+06
	10	-	1.21168e+06	2.43249e+06	1.58829e+06	1.82665e+06
	11	-	1.22064e+06	2.52935e+06	1.61337e+06	1.86708e+06
	13	-	1.23002e+06	2.67718e+06	1.64404e+06	1.92041e+06
	15	-	1.23377e+06	2.77803e+06	1.65931e+06	1.95001e+06
	16	-	1.23467e+06	2.81528e+06	1.66371e+06	1.95929e+06
	18	-	1.23557e+06	2.87032e+06	1.66888e+06	1.97106e+06
	20	-	1.23592e+06	2.90644e+06	1.67134e+06	1.97731e+06
	30	-	1.23612e+06	2.96434e+06	1.67344e+06	1.98383e+06

TABLE 8.8 – Longueur moyenne du tampon.

	bins	20		100	
	Exact	$L.b$	$U.b$	$L.b$	$U.b$
capacité du tampon (10^6 bits) 5	4.35018e+06	4.2632e+06	4.43663e+06	4.33717e+06	4.37164e+06
7	-	4.27247e+06	4.45812e+06	4.35089e+06	4.38789e+06
10	-	4.27619e+06	4.47247e+06	4.35777e+06	4.39692e+06
11	-	4.27657e+06	4.475e+06	4.35866e+06	4.39823e+06
13	-	4.27692e+06	4.4783e+06	4.35961e+06	4.39973e+06
15	-	4.27704e+06	4.48018e+06	4.36002e+06	4.40043e+06
16	-	4.27706e+06	4.48079e+06	4.36012e+06	4.40063e+06
18	-	4.27708e+06	4.4816e+06	4.36023e+06	4.40085e+06
20	-	4.27709e+06	4.48208e+06	4.36028e+06	4.40096e+06
30	-	4.2771e+06	4.48268e+06	4.36031e+06	4.40105e+06

TABLE 8.9 – Nombre moyen de données en sorties.

	bins	20		100	
		$L.b$	$U.b$	$L.b$	$U.b$
capacité du tampon (10^6 bits) 5		1.02415	1.3	1.03383	1.3
7		1.05181	2.09	1.08129	2.09
10		1.06266	2.35481	1.10548	2.31925
11		1.06455	2.52542	1.11083	2.44543
13		1.0663	2.60637	1.11662	2.49609
15		1.06701	2.67837	1.11955	2.53237
16		1.06719	2.70857	1.12044	2.54528
18		1.06737	2.73689	1.12145	2.55541
20		1.06744	2.75744	1.12194	2.56131
30		1.06748	2.78522	1.12236	2.56646

TABLE 8.10 – Temps de réponse sous la discipline de service FIFO.

Dans le cas d'un trafic modélisé par une suite de variables aléatoires iid. (trafic stationnaire indépendant) avec un service également défini par une distribution discrète iid., nous observons que notre méthode permet de borner de manière assez pertinente l'erreur commise.

capacité du tampon (10^6 bits)	bins	20		100	
	Exact	$L.b$	$U.b$	$L.b$	$U.b$
5	22602.2	2.28731	1.24064	8.00238	2.60865
7	-	7.14619	2.23856	16.6102	3.90438
10	-	20.6269	67.603	40.5646	80.1019
11	-	26.1092	116.768	27.2133	125.072
13	-	33.4842	261.497	44.2973	279.313
15	-	52.6096	482.412	66.7719	497.71
16	-	77.1375	622.007	80.7921	634.48
18	-	109.955	1000.83	115.637	1013.54
20	-	154.249	1468.63	133.304	1481.41
30	-	506.262	6422.03	493.324	6065.42

TABLE 8.11 – Temps d'exécution (seconde).

8.4 Analyse de certains mécanismes AQM

Nous nous intéressons ici à l'étude de certains mécanismes de gestion active de files d'attente, plus communément appelés *AQM* (*Active Queue Management*), et nous définissons quelques conditions pour que ces derniers soient H-monotones. Dans ce chapitre, nous avons considéré une file d'attente étudiée sous la politique Tail Drop où les unités de données sont acceptées dans la file d'attente jusqu'à ce que le tampon soit plein. Nous notons que cette politique représente un cas particulier de gestion active de files d'attente et sa monotonie a été prouvée précédemment. Dans cette section, nous considérons un autre mécanisme de gestion active qui est le Random Early Detection noté RED [FJ93] appelée détection précoce aléatoire utilisé pour obtenir un haut débit en sortie et de faibles délais. Le mécanisme RED est un algorithme de gestion active de file d'attente qui consiste à éliminer de façon probabiliste les données en entrées. La probabilité d'élimination augmente avec la croissance de la longueur moyenne de la file d'attente. Nous nous sommes limités ici aux AQM dont les probabilités de rejet dépendent de la taille de la file d'attente juste avant l'insertion.
Nous donnons ci-après une brève définition de l'AQM et notons que cette dernière est une version restreinte des mécanismes de gestion active de files d'attente. Par conséquent, certains mécanismes comme la notification explicite de congestion qui permettent de signaler la congestion du réseau avant que la perte de données ne se produise n'ont pas été considérés.

Définition 8.1 *L'AQM est immédiate si elle fonctionne indépendamment et de manière séquentielle pour chaque unité de donnée dans le groupe, et si la probabilité de rejet prend en compte l'état de la file d'attente, juste avant l'insertion.*

Dans les mécanismes tels que le RED, le calcul de la probabilité d'acceptation est effectué en fonction d'une moyenne de la longueur de la file d'attente calculée à chaque arrivée et non en fonction de la longueur instantanée du tampon. Ainsi, notre définition est plutôt générale, elle

correspond au cas étudié par la suite, elle peut être également utilisée dans d'autres cas comme une limite ou une approximation.
Nous décrivons rapidement ce mécanisme. Nous définissons une acceptation dans un mécanisme AQM par une variable aléatoire binaire notée $q(X)$. Cette variable dépend de la taille du tampon X à l'instant d'acceptation. Par conséquent, $q(X) = 1$, si l'unité de données est acceptée et $q(X) = 0$ dans le cas contraire.
La fonction d'acceptation des données a la propriété suivante :

Propriété 8.2 *L'AQM est dite décroissante si la probabilité $q(X)$ est décroissante avec la longueur du tampon.*

Exemple 8.4 *La politique de Tail Drop utilisée jusqu'à présent est décrite par la fonction d'acceptation suivante :*

$$q(X) = \mathbb{1}_{\{X<B\}}.$$

La politique Tail Drop au niveau de l'unité de donnée est clairement immédiate (dépend de la longueur instantanée de la file) et décroissante.

Nous considérons une file d'attente simple munie d'un serveur dont C est le nombre d'unités de données servies pendant un slot, et dont B est la taille du tampon. Le tampon est géré par le mécanisme de gestion active *Random Early Detection Immédiate* noté IRED. Ce mécanisme est décrit ci-après.

Définition 8.2 *[IRED] La politique de Random Early Detection Immédiate est un exemple d'AQM. Nous supposons qu'il fonctionne au niveau de l'unité de données. Contrairement au Tail Drop, l'acceptation du IRED est donnée avec des probabilités. De nombreuses implémentations de RED sont basées sur des fonctions cubiques ou sur des fonctions de coefficients linéaires pour calculer les probabilités d'acceptation comme ici :*

- *Si* $X < \lfloor \frac{B+C}{2} \rfloor$ *:* $\mathrm{Prob}(q(X) = 1) = 1$ *;*
- *Si* $\lceil \frac{B+C}{2} \rceil \leq X < B + C$ *:* $\mathrm{Prob}(q(X) = 1) = \frac{2(B+C)-2X}{B+C}$ *;*
- *Si* $X \geq (B + C)$ *:* $\mathrm{Prob}(q(X) = 1) = 0$.

Ainsi, pour ces fonctions, la probabilité que $q(X) = 1$ diminue avec la longueur de la file, X (figure8.3).

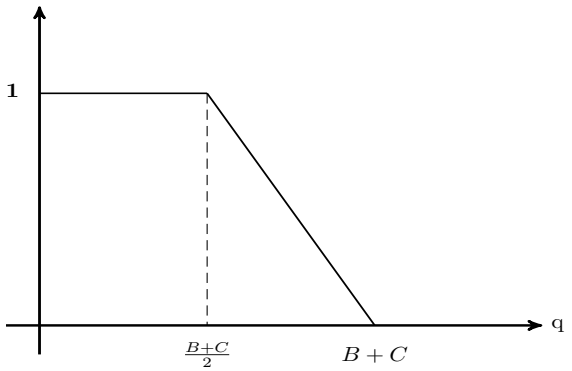


FIGURE 8.3 – Fonction d'acceptation pour l'AQM IRED.

Si nous considérons par exemple une file d'attente ayant comme histogramme d'arrivée $H1 = [P_0, P_1, P_2]$ représentant respectivement les probabilités d'arrivée d'aucune donnée, d'une seule donnée ou deux données, une capacité de tampon $\mathrm{B} = 4$ et un service de $C = 1$. En utilisant le mécanisme IRED, le tampon est modélisé par une chaîne de Markov de taille 5 dont le graphe de transition est présenté ci-après. Nous rappelons que l'équation d'évolution de la longueur de la file d'attente est donnée par l'équation 8.1.

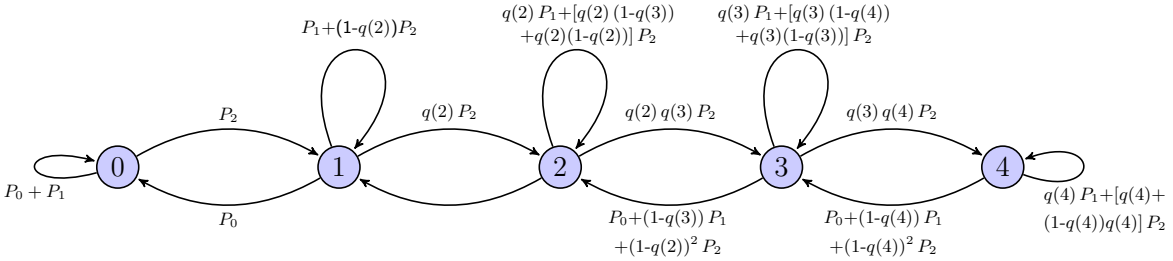


FIGURE 8.4 – DTMC du tampon IRED.

Nous étendons la définition de H-monotonie aux éléments de réseau avec AQM et nous prouvons certaines conditions sur l'AQM pour être H-monotone.

Définition 8.3 *L'AQM est H-monotone, ssi*

$$H_1^a \leq_{st} H_1^b \Rightarrow H_3^a \leq_{st} H_3^b \text{ et } H_L^a \leq_{st} H_L^b$$

Nous supposons que la file d'attente fonctionne avec une AQM immédiate (IRED) spécifiée par une fonction d'admission décroissante $q(X)$. Nous notons par X_n la longueur de la file d'attente au slot n et par $Y_{n,j}$ la longueur de la file d'attente au slot n après l'admission de j unités de données. Pour les paramètres de la file, nous considérons les mêmes hypothèses énoncées dans l'analyse d'une file d'attente simple (section 8.1.1). La taille maximale du groupe en entrée est notée par K. L'équation d'évolution de la longueur de la file d'attente peut être donnée comme suit, dans le cas où les arrivées sont considérées avant les services.

$$\begin{aligned} Y_{n+1,0} &= X_n \\ Y_{n+1,j+1} &= Y_{n+1,j} + \mathbb{1}_{\{A_n > j \text{ et } q(Y_{n+1,j})=1\}} \quad 0 \leq j \leq K-1 \\ X_{n+1} &= (Y_{n+1,K} - C)^+ \end{aligned} \tag{8.5}$$

Théorème 8.5 *Si l'AQM est immédiate et que la fonction d'acceptation est décroissante, alors l'AQM est H-monotone.*

PREUVE. La preuve est basée sur la propriété simple de l'ordre stochastique fort [MS02]. Nous montrons par induction que l'existence des réalisations des variables aléatoires associées à l'évolution de la longueur de la file d'attente (équation 8.5) est telle que :

$$x_n^a \leq x_n^b, \quad \forall n$$

Nous supposons que les longueurs des files d'attente sont les mêmes pour le slot 0 et nous donnons le cas général pour l'induction avec n. Supposant que $x_n^a \leq x_n^b$ alors, par définition $y_{n+1,0}^a \leq y_{n+1,0}^b$. Nous allons monter que si $y_{n+1,j}^a \leq y_{n+1,j}^b$ alors $y_{n+1,j+1}^a \leq y_{n+1,j+1}^b$. Il y a deux cas :

1. si $y_{n+1,j}^a < y_{n+1,j}^b$, puisque l'augmentation s'effectue un par un, nous sommes sûrs que : $y_{n+1,j+1}^a \leq y_{n+1,j+1}^b$.
2. si $y_{n+1,j}^a = y_{n+1,j}^b$, alors $q(y_{n+1,j}^a) = q(y_{n+1,j}^b)$, et il résulte de l'hypothèse que $H_1^a \leq_{st} H_1^b$ alors $\mathbb{1}_{A_n^a > j} \leq \mathbb{1}_{A_n^b > j}$. Ainsi, $y_{n+1,j+1}^a \leq y_{n+1,j+1}^b$.

Donc, nous déduisons que :

$$x_{n+1}^a = y_{n+1,K}^a \leq y_{n+1,K}^b = x_{n+1}^b$$

Par conséquent, nous avons la comparaison stochastique des évolutions de la longueur de la file d'attente : $X_n^a \leq_{st} X_n^b, \ \ \forall n$ et pour le processus stationnaire : $H_3^a \leq_{st} H_3^b$. ■

Le nombre d'unités de données perdues pendant le slot $n+1$ est donné par :

$$\sum_{j=1}^{K} \mathbb{1}_{\{A_n > j \ et \ q(Y_{n+1,j})=0\}}$$

Il résulte de la preuve précédente que $Y_{n,j}^a \leq_{st} Y_{n,j}^b$. Comme les fonctions d'acceptation q sont des fonctions décroissantes et $H_1^a \leq_{st} H_1^b$, si la fonction indicatrice précédente est 1 sous H_1^a alors elle est également 1 sous H_1^b. Ainsi, le nombre d'unités de données perdues dans chaque slot et dans la limite sera comparable :

$$H_L^a \leq_{st} H_L^b.$$

8.4.0.1 Exemple de file d'attente avec mécanisme IRED

Nous donnons un exemple simple d'une file d'attente isolée pour illustrer l'impact de notre méthode sur un seul nœud avec mécanisme IRED. Nous considérons l'histogramme d'entrée $H_1 = [0.10, 0.05, 0.10, 0.10, 0.15, 0.15, 0.10, 0.10, 0.05, 0.10]$ définie sur l'espace d'état $E^{H_1} = \{1, \ldots, 10\}$ et un service déterministe $C = 2$.

Les mesures de performance (probabilités de blocage, espérance de la longueur de la file d'attente et le temps d'exécution) sont calculées selon la fonction d'acceptation définie dans la définition 8.2 et nous faisons varier la longueur du tampon de 4 à 30 unités de données. Dans les figures 8.5 et 8.6, nous présentons les mesures de performance en utilisant le calcul exact et la borne optimale inférieure pour pour un nombre de bins égal à 3 et 5.

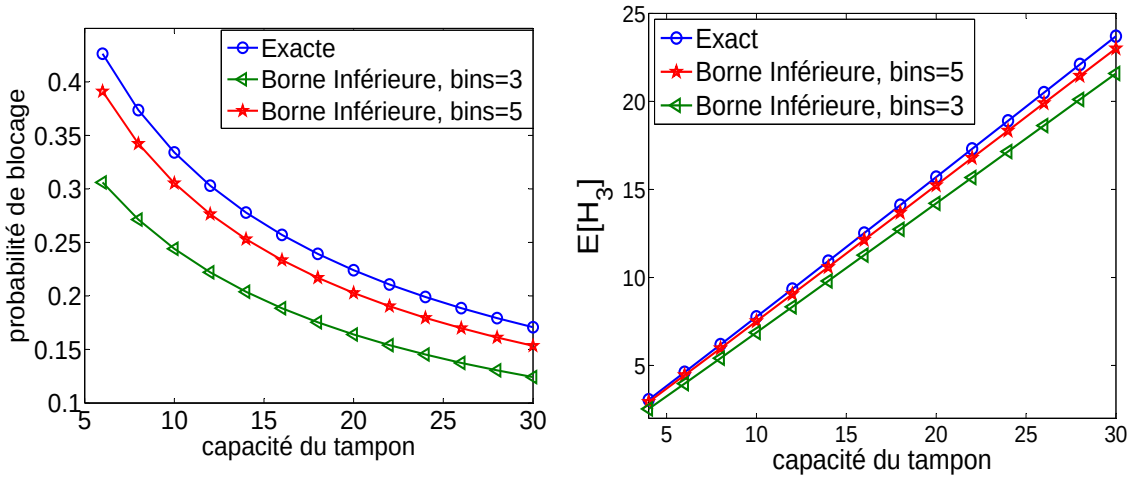


FIGURE 8.5 – Résultats sur les probabilités de blocage (à gauche) et la longueur moyenne du tampon (à droite).

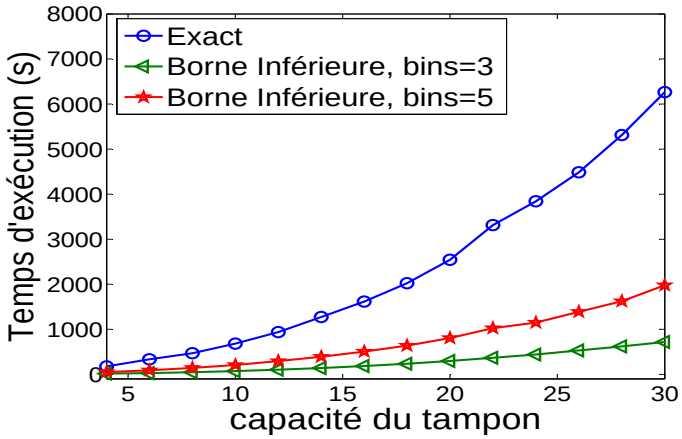


FIGURE 8.6 – Temps d'exécution (s).

Nous constatons que l'utilisation de la méthode de bornes, dans le cas du mécanisme AQM IRED, nous permet d'obtenir des résultats précis en un temps d'exécution réduit.

8.5 Éléments de routage

Nous nous intéressant dans cette partie à l'étude des éléments de routage dans le réseau tel que l'élément de division appelé *split* et l'élément de fusion appelé *merge* [Sch07]. Nous montrons ici que ces éléments du réseau sont des éléments monotones. Ceci nous permettra d'assurer que, si nous sommes en mesure de calculer des bornes inférieures et supérieures sur le trafic d'entrée d'un nœud, nous pouvons dériver des bornes inférieures et supérieures sur trafic sortant de ce nœud mais également sur les éléments de routage du réseau.

8.5.1 Élément de division : split

Dans cette section, nous étudions l'opération de division des flux dans le réseau également appelée *split*. Cet élément permet de diviser le flux en entrée de l'élément en de multiples flux en sortie (voir figure 8.7).

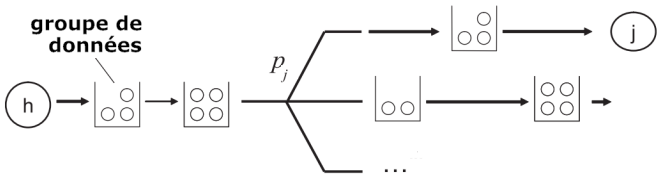


FIGURE 8.7 – Split du groupe entier.

Lorsque le flux d'entrée noté H_D traverse le nœud split, il est divisé en m flux : $H_{D,1}, \ldots, H_{D,m}$. Nous notons par E^{H_D} (resp. $E^{H_{D,i}}$) l'espace d'état de H_D (resp. $H_{D,i}$, $1 \leq i \leq m$). Nous définissons pour l'opérateur de split la H-monotonie comme suit :

Définition 8.4 *Un split est dit H-monotone, ssi*
$H_D^a \leq_{st} H_D^b \Rightarrow \forall i,\ H_{D,i}^a \leq_{st} H_{D,i}^b$.

Nous étudions deux cas de divisions :

- Le groupe de données qui arrive dans l'élément de split est entièrement envoyé vers la même sortie. Le flux en sortie est choisi de façon aléatoire selon une probabilité de routage.
- Le groupe de données est divisé sur toutes les sorties selon une distribution de répartition des unités de données.

▸ Modèle de division de tout le groupe de données avec une probabilité de routage

Nous étudions ici le cas où toutes les unités d'un groupe de données du flux en entrée sont dirigées vers une des sorties avec une probabilité de routage. Soit $p_i, 1 \leq i \leq m$ (tel que $\sum_{i=1}^{m} p_i = 1$), la probabilité de routage du groupe de données vers la sortie i du splitter. Nous supposons également que les groupes successifs sont acheminés indépendamment selon ces probabilités de routage. De plus, si l'ensemble des états de H_D n'inclut pas l'état 0, ce dernier sera ajouté avec une probabilité de 0, et l'ensemble des états des flux en sortie sera identique à celui de E^{H_D}.

$$E^{H_{D,i}} = \{0\} \cup E^{H_D}, \quad 1 \leq i \leq m$$

La distribution de probabilité de tout flux de sortie i peut être calculé comme suit :
pour tout $i \leq m$:

$$\begin{cases} H_{D,i}[k] & = p_i\ H_D[k],\ k \in E^{H_D},\ k > 0 \\ H_{D,i}[0] & = 1 - \sum_{k \neq 0} H_{D,i}[k]. \end{cases} \tag{8.6}$$

Exemple 8.5 *Nous considérons une distribution H_D définie sur l'ensemble des états E^{H_D} = $\{0, 3, 4, 7, 10\}$ et de vecteur de probabilités (histogramme) H_D = $[0.1, 0.2, 0.4, 0.1, 0.2]$. En supposant qu'un groupe de données peut être acheminé vers deux directions possibles avec la même probabilité (*i.e., $p_1 = p_2 = 0.5$*), chacun des groupes acheminés a la distribution suivante : les états sont $E^{H_{D,i}}$ = $\{0, 3, 4, 7, 10\}$ et les probabilités sont $H_{D,i}$ = $[0.55, 0.1, 0.2, 0.05, 0.1]$, où $1 \leq i \leq 2$.*

Nous notons que pour une implémentation efficace des histogrammes, uniquement les états ayant des probabilités non nulles sont considérés. Cependant, dans les preuves qui suivent, nous supposons que les histogrammes sont définis sur l'ensemble des états $E^H = \{0, \ldots, n\}$, par conséquent, les vecteurs de probabilités peuvent contenir des probabilités nulles.

Théorème 8.6 *Si dans un split, toutes les données sont envoyées vers une même sortie selon une probabilité de routage, alors le split est H-monotone.*

PREUVE. Pour chaque flux en sortie i , $1 \leq i \leq m$, nous avons l'équation suivante $1 \leq \forall l \leq n$:

$$\sum_{k=l}^{n} H^a_{D,i}[k] = \sum_{k=l}^{n} p_i \; H^a_D[k], \;\; \sum_{k=l}^{n} H^b_{D,i}[k] = \sum_{k=l}^{n} p_i \; H^b_D[k].$$

Vu que $H^a_D \leq_{st} H^b_D$, nous obtenons : $\sum_{k=l}^{n} H^a_D[k] \leq \sum_{k=l}^{n} H^b_D[k]$. Alors, pour tout flux i, $1 \leq i \leq m$:

$$\sum_{k=l}^{n} p_i H^a_{D,i}[k] \leq \sum_{k=l}^{n} p_i \; H^b_D[k].$$

qui peut également être écrit comme suit :

$\sum_{k=l}^{n} H^a_{D,i}[k] \leq \sum_{k=l}^{n} H^b_{D,i}[k]$, ce qui équivaut à dire que pour tout $i \leq m$, $\; H^a_{D,i} \leq_{st} H^b_{D,i}$. ■

▸ Modèle de division d'un groupe de données avec une répartition entre les liens

Nous supposons maintenant que les unités de données sont réparties entre les m flux de sortie. La proportion de données reçues par chaque flux est donnée par la probabilité p_i qui est maintenant définie comme étant un rapport. En raison de cette multiplication par p_i, cette quantité de données peut être une quantité non entière d'unités de données. Ensuite, nous supposons que des unités de données sont complétées avec des bits nuls afin d'obtenir un nombre entier d'unités de données.

$$E^{H_{D,i}} = \left\{k \mid \mathbb{1}_{\lceil p_i * q \rceil = k}\right\}_{q \in E^{H_D}}.$$

La distribution de probabilité d'un flux de sortie i peut être calculée comme suit $1 \leq \forall i \leq m$:

$$\begin{cases} H_{D,i}[k] & = \sum_{q \in E^{H_D}, q \neq 0} H_D[q] \mathbb{1}_{\lceil p_i * q \rceil = k}, \;\; k > 0 \\ \\ H_{D,i}[0] & = 1 - \sum_{k \neq 0} H_{D,i}[k]. \end{cases} \tag{8.7}$$

Exemple 8.6 *Nous considérons le même exemple que précédemment, mais nous supposons maintenant que les unités de données sont réparties entre les flux. Nous supposons également une répartition identique des flux (proportion égale $p_1 = p_2 = 0.5$). Les groupes de données résultant ont la même distribution : les états sont $E^{H_{D,i}} = \{0, 2, 4, 5\}$ et les probabilités sont $H_{D,i} = [0.1, 0.6, 0.1, 0.2]$. En effet, la probabilité pour que la taille d'un groupe de données soit de 2 correspond à la somme de la probabilité que la taille du groupe initial (avant la division en deux groupes égaux) soit de 3 ou 4. Le détail de calcul de ces distributions est donné comme suit.*

D'après l'équation 8.7 nous avons pour $i \in \{1, 2\}$:

$$\begin{cases} H_{D,i}[1] & = \sum_{q \in E^{H_D}, q \neq 0} H_D[q] \mathbb{1}_{\lceil 0.5 * q \rceil = 1} = 0; \\ H_{D,i}[2] & = \sum_{q \in E^{H_D}, q \neq 0} H_D[q] \mathbb{1}_{\lceil 0.5 * q \rceil = 2} = H_D[3] + H_D[4] = 0.6; \\ H_{D,i}[3] & = \sum_{q \in E^{H_D}, q \neq 0} H_D[q] \mathbb{1}_{\lceil 0.5 * q \rceil = 3} = 0; \\ H_{D,i}[4] & = \sum_{q \in E^{H_D}, q \neq 0} H_D[q] \mathbb{1}_{\lceil 0.5 * q \rceil = 4} = H_D[7] = 0.1; \\ H_{D,i}[5] & = \sum_{q \in E^{H_D}, q \neq 0} H_D[q] \mathbb{1}_{\lceil 0.5 * q \rceil = 5} = H_D[10] = 0.2; \\ \text{et enfin,} & \\ H_{D,i}[0] & = 1 - \sum_{k \neq 0} H_{D,i}[k] = 0.1. \end{cases}$$

Théorème 8.7 *Si dans un split, un groupe de données est réparti sur les sorties selon des probabilités de répartition, alors le split est H-monotone.*

PREUVE. Pour chaque flux i, $1 \leq i \leq m$, nous avons les équations suivantes $1 \leq \forall\, l \leq n$:

$$\sum_{k=l}^{n} H^a_{D,i}[k] = \sum_{k=l}^{n} \sum_{q=0}^{n} H^a_D[q] \mathbb{1}_{\lceil p_i * q \rceil = k}$$

Après avoir échangé les sommations, nous avons :

$$\sum_{k=l}^{n} H^a_{D,i}[k] = \sum_{q=0}^{n} H^a_D[q] \sum_{k=l}^{n} \mathbb{1}_{\lceil p_i * q \rceil = k}$$

mais

$$\sum_{k=l}^{n} \mathbb{1}_{\lceil p_i * q \rceil = k} = \mathbb{1}_{q \geq Q_i}$$

pour un Q_i. Alors,

$$\sum_{k=l}^{n} H^a_{D,i}[k] = \sum_{q=0}^{n} H^a_D[q]\ \mathbb{1}_{q \geq Q_i} = \sum_{q \geq Q_i} H^a_D[q]$$

Puisque $H^a_S \leq_{st} H^b_S$, en raison de l'ordre st nous obtenons

$$\sum_{q \geq Q_i} H^a_D[q] \leq \sum_{q \geq Q_i} H^b_D[q]$$

Par conséquent,

$$\sum_{k=l}^{n} H^a_{D,i}[k] \leq \sum_{k=l}^{n} H^b_{D,i}[k]$$

Ainsi, pour tout i, $H^a_{D,i} \leq_{st} H^b_{DS,i}$. ■

8.5.2 Élément de fusion : merge

Dans un élément de fusion (également appelé *merge*), un ensemble de flux indépendants avec des distributions $H_{F,i}, 1 \leq i \leq m$ sont agrégés en un flux de distribution H_F.

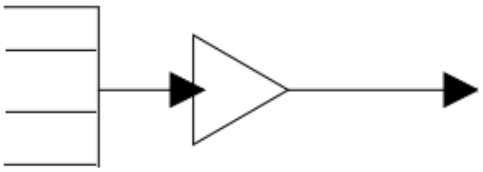


FIGURE 8.8 – Élément de fusion.

Nous supposons que tout lien i en entrée a une capacité finie C_i. Et que le lien de sortie a une capacité C. Nous présentons ici les propriétés de monotonie pour les éléments de fusion en utilisant des variables aléatoires correspondant à leurs histogrammes. Ainsi, la variable aléatoire X_i de *pmf* $H_{F,i}$ représente le nombre d'unités de données associées au flux d'entrée i de l'élément de fusion. La fusion peut être définie sous la forme d'une fonction comme suit :

Définition 8.5 *Le merge est une fonction*
$\mathscr{M} : \times_{i=1}^{m}\{0,\ldots,C_i\} \rightarrow \{0,\ldots,C\}$. $\mathscr{M}(X_1,\ldots,X_m)$ *représente l'état du flux de sortie de l'élément de fusion sous des flux d'entrée indépendants* X_i*. En fait, la fusion est une variable aléatoire de* pmf H_F *représentant le nombre d'unités de données sortant de l'élément de fusion et prenant des valeurs dans* $\{0,1,\cdots,C\}$ *où* $C \leq \sum_{i=1}^{m} C_i$.

Il est évident que, pour l'opération de fusion, le nombre d'unités de données en sortie doit être inférieur ou égale au nombre d'unités de données aux entrées.

Définition 8.6 *Le merge est causale si* $\mathscr{M}(X_1,\ldots,X_m) \leq \sum_{i=1}^{m} X_i$.

Nous pouvons également définir pour l'élément de fusion, la monotonie par rapport au trafic dit trafic monotone comme suit :

Définition 8.7 *Un élément de fusion est trafic monotone ssi*
pour tout couple $(X_1,\ldots,X_m)$ *et* $(Y_1,\ldots,Y_m)$*, si* $X_k \leq Y_k$*,* $\forall k$*, alors* $\mathscr{M}(X_1,\ldots,X_m) \leq \mathscr{M}(Y_1,\ldots,Y_m)$.

Dans la suite, nous considérons que les éléments de fusion sont causaux. L'opération de fusion peut avoir la propriété Tail Drop, qui est définie comme suit :

Définition 8.8 *Un élément de fusion est dit Tail drop ssi* $\mathscr{M}(X_1,\ldots,X_m) = \min(C, \sum_{i=1}^{m} X_i)$.

Nous étudions maintenant la propriété de monotonie de l'élément de fusion.

Définition 8.9 *Un élément de fusion est dit H-monotone, si et seulement si*

$$\forall i,\ H_{F,i}^{a} \leq_{st} H_{F,i}^{b} \Rightarrow H_F^a \leq_{st} H_F^b.$$

Théorème 8.8 *Si l'élément de fusion est trafic monotone alors il est H-monotone.*

PREUVE. Nous supposons que $\forall i,\ H^a_{F,i} \leq_{st} H^b_{F,i}$, ainsi les variables aléatoires correspondantes sont comparables : $\forall i,\ X^a_i \leq_{st} X^b_i$. La monotonie du trafic pour l'élément de fusion signifie en effet que la fonction $\mathscr{M}$ est une fonction croissante. Puisque les flux de sorties H^a_F et H^b_F sont définis comme des fonctions croissantes de variables aléatoires indépendantes comparables, ils sont également comparables (voir page 7 de [MS02]). ■

Corollaire 8.1 *Un élément de fusion avec une politique Tail Drop (i.e., $\mathscr{M}(X_1,\ldots,X_m) = \min(C, \sum_{i=1}^m X_i)$) est causal et trafic monotone. Par conséquent, il est H- monotone.*

Nous considérons maintenant les processus de perte des éléments de fusion. Un élément de fusion peut supprimer certaines unités de données en raison d'une limitation de la bande passante ou un contrôle d'accès. Nous définissons d'abord le nombre d'unités de données perdues par la fonction de perte l qui dépend de la fonction de fusion $\mathscr{M}$.

Définition 8.10 *Le nombre d'unités de données perdues dans un élément de fusion peut être défini par une fonction $l : \times_{i=1}^m \{0,\ldots,C_i\} \to \{0,\ldots,\sum_{i=1}^m C_i\}$. $l(X_1,\ldots,X_m) = \sum_{i=1}^m X_i - \mathscr{M}(X_1,\ldots,X_m)$.*

En effet, le nombre de pertes est la différence entre le nombre d'unités de données arrivées sur les m liens (*i.e.,* $\sum_{i=1}^m X_i$) et le nombre d'unités acceptées par l'élément de fusion (*i.e.,* $\mathscr{M}(X_1,\ldots,X_m)$). La distribution de perte peut être donnée comme suit, puisque les arrivées sont indépendantes. Notons que les lettres en minuscule désignent les réalisations des variables aléatoires correspondantes X_i.

Proposition 8.11 (Distribution de perte pour une fusion, H_L)

$$H_L[k] = \sum_{(x_1,\ldots,x_m)} \mathbb{1}_{(\sum_{j=1}^m x_j - \mathscr{M}(x_1,\ldots,x_m))=k} \prod_{i=1}^m H_{F,i}[x_i]$$

Propriété 8.3 *Si $C = \sum_i C_i$, et l'élément de fusion est Tail Drop alors il n'y a pas de pertes.*

PREUVE. L'élément est Tail Drop alors
$\mathscr{M}(X_1,\ldots,X_m) = \min(C, \sum_{i=1}^m X_i)$. Mais par construction $x_i \leq C_i$. Par conséquent $\sum_{i=1}^m X_i \leq \sum_{i=1}^m C_i = C$. Ainsi, il n'y a pas de pertes au niveau de l'élément de fusion. ■

Théorème 8.9 *Si la fonction de perte l est croissante, alors la distribution des pertes dans un élément de fusion est monotone : si $\forall i,\ H^a_{F,i} \leq_{st} H^b_{F,i}$, alors $H^a_L \leq_{st} H^b_L$.*

PREUVE. La preuve est similaire à celle du théorème 8.8. Puisque H^a_L et H^b_L sont définis comme des fonctions croissantes de flux d'entrées comparables, ils sont comparables. ■

Propriété 8.4 *Nous considérons un élément de fusion Tail Drop avec une capacité de sortie C avec $C < \sum_i C_i$. La distribution des pertes est monotone.*

PREUVE. Le nombre d'unités de données perdues est $l(X_1,\cdots X_m) = \max(0, \sum_{i=1}^m X_i - C)$ (car la fonction l est croissante). Comme il s'agit d'une fonction croissante, d'après le théorème 8.9 la distribution des pertes est monotone. ■

8.6 Conclusion

Dans le but de garantir des bornes sur les mesures de performance, nous avons étudié dans ce chapitre les propriétés de monotonie de différents éléments de réseau : file d'attente simple, élément de division et élément de fusion. Nous avons présenté et prouvé la pertinence de notre approche pour fournir une solution intéressante au problème de dimensionnement. La méthode de borne offre un compromis entre la précision des résultats et la complexité de calcul. Nous avons également abordé certains mécanismes de gestion active de files d'attente (AQM) et montré la notion de monotonie de ces derniers.

Le fait d'avoir montré ici que les éléments de réseaux FIFO sont stochastiquement monotones par rapport aux histogrammes, va nous permettre d'envisager l'obtention d'un encadrement stochastique de la taille des files d'attente dans un réseau en tandem ou dans un réseau feed-forward. Ce point représentera donc l'essentiel du travail que nous allons entreprendre dans le chapitre suivant.

Chapitre 9

Analyse des réseaux de files d'attente

Les réseaux de files d'attente ouverts sont souvent utilisés pour la modélisation et l'évaluation des performances des systèmes complexes tels que les systèmes informatiques, les réseaux de communication, les lignes de production ou les systèmes de fabrication. Cependant, les résultats analytiques exacts de ces systèmes ne sont disponibles que dans certaines situations sous des hypothèses restreintes. Dans le cas général, les solutions possibles ne peuvent être obtenues que par des méthodes d'approximations. Dans ce chapitre, nous proposons de nous intéresser aux réseaux de files d'attente de topologie DAG (Directed Acyclic Graph). Nous distinguons pour ce faire deux approches pour évaluer les performances de ces réseaux. Pour des processus de services déterministes, nous proposons d'étudier dans un premier temps l'approche par décomposition pour analyser et évaluer les performances des différents paramètres du réseau. Cette méthode ne fournissant pas de justification théorique quant à sa convergence au système exact ne représente qu'une approche approximative. La seconde approche abordée revient à étudier certains réseaux pour lesquels nous pouvons construire des chaînes de Markov bornantes. Contrairement à la première approche, la deuxième nous permet de définir des bornes plutôt que des approximations sur les processus de sortie du réseau.

Pour ces deux approches, nous proposons d'utiliser notre méthode de bornes stochastiques. Nous montrons que pour un histogramme d'entrée bornant, nous obtenons des encadrements très pertinents sur les distributions de sortie du réseau. Cette caractéristique représente l'une de nos contributions principales et elle est due à la monotonie stochastique prouvée pour les éléments du réseau. La garantie de la qualité de service représente l'autre apport principal. Ainsi, nous montrons grâce à certains résultats théoriques prouvés et sous certaines conditions la possibilité de déterminer des encadrements fiables sur les mesures de performance.

Ce chapitre est organisé comme suit : d'abord, nous décrivons de manière assez brève la composition d'un réseau de file d'attente ouvert, nous nous intéressons ici aux réseaux ayant une topologie de graphe orienté acyclique. Nous abordons dans la section 9.2, l'étude et l'analyse d'un réseau de files d'attente par la méthode d'approximation par décomposition. Nous démontrons quelques propriétés importantes de monotonie stochastique pour le calcul des bornes. Nous proposons également d'effectuer une évaluation numérique des réseaux de files d'attente à l'aide des traces de trafic réelles en considérant respectivement un réseau en tandem et un réseau feed-forward. Nous considérons ensuite dans la section 9.3 l'analyse de réseaux bornants. Nous nous limitons à l'analyse des réseaux en tandem. Nous présentons dans

cette section une approche permettant de définir des bornes sur les mesures de performances exactes du réseau.

9.1 Réseaux de files d'attente avec une topologie de DAG

Un réseau de files d'attente est un ensemble de files d'attente interconnectées où les sorties d'une (ou plusieurs) file entrent dans une (ou plusieurs) autre file, selon un routage défini, ou quitte le système. Nous considérons dans ce chapitre les réseaux de files d'attente à capacité finie. Dans le cas d'un réseau feed-forward : *i.e.,* un graphe orienté acyclique appelé également DAG (Directed Acyclic Graph), les nœuds du réseau se composent de :

- Sources de trafic (flux entrants) ;
- Files d'attente à capacités finies ;
- Splitters : séparation du flux en entrée (éléments de routage) ;
- Merge : fusion des flux en entrées (éléments de routage).

Les éléments du réseau sont numérotés en fonction du DAG. Les plus petits indices sont attribués aux sources du réseau. S'il existe un lien orienté de l'élément de réseau i à l'élément de réseau j, alors nous devons avoir $i < j$ (voir figure 9.1).

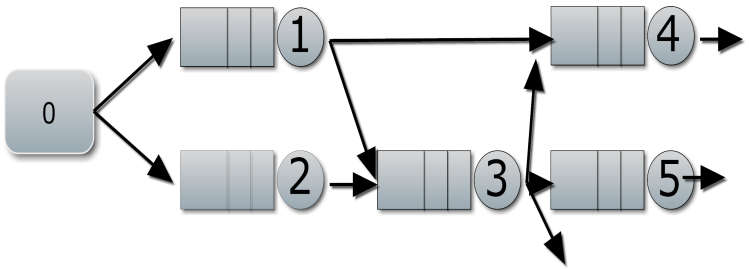


FIGURE 9.1 – Topologie Feed-Forward

Nous avons considéré ici l'hypothèse de réseaux orientés acycliques dans le but d'utiliser par la suite la méthode d'analyse par décomposition qui permet de décomposer le réseau et d'étudier ces éléments de manière séquentielle selon l'ordre topologique du DAG.

Avant de présenter l'analyse effectuée sur les réseaux, nous rappelons que les éléments de réseau tels que la file d'attente, la division ou la fusion sont des éléments monotones (voir le chapitre précédent). Ceci nous permet d'assurer que, si nous sommes en mesure de calculer des bornes inférieures et supérieures sur le trafic d'entrée d'un nœud, nous pouvons alors dériver des bornes inférieures et supérieures sur le trafic en sortie du même nœud et sur les éléments de routage dans le réseau.

9.2 Bornes sur l'analyse par décomposition des réseaux feed-forward

Nous proposons d'employer ici, l'une des méthodes les plus utilisées pour analyser les réseaux de files d'attente ouverts, à savoir, la méthode de décomposition. Cette méthode est une méthode approximative qui permet de réduire la taille du réseau en le décomposant en sous réseau et en analysant chaque sous-réseau de façon isolée. L'approche de décomposition suppose que les sous-réseaux peuvent être traités comme étant stochastiquement indépendants et que l'entrée de chaque sous-réseau est un processus stationnaire. L'analyse par décomposition comprend trois étapes :

1. Décomposition du réseau en sous-réseaux (files d'attente individuelles, dans la plupart des cas) ;
2. L'analyse de chaque sous-réseau ;
3. Recomposition des résultats pour calculer les performances du réseau.

Reposant sur cette approche, nous proposons d'appliquer notre méthode de bornes aux histogrammes d'entrée de chaque file et montrer le compromis intéressant qu'offre notre méthode à l'analyse de performance des réseaux. Cependant, il est important de noter que, comme pour la méthode de décomposition est une méthode d'approximation, les bornes calculées sur les différentes mesures de performance des files d'attente ne représentent que des bornes sur les approximations.

Nous énonçons à présent quelques résultats sur la monotonie des paramètres d'un réseau feed-forward. Nous présentons ensuite quelques exemples numériques pour observer et montrer l'applicabilité et l'intérêt de notre approche dans ces réseaux.

9.2.1 Bornes sur les paramètres d'une file d'attente

Afin d'établir un résultat de comparaison stochastique, nous considérons une file d'attente simple pour laquelle nous proposons d'injecter un trafic bornant (inférieur ou supérieur). Ainsi, pour un trafic d'entrée $\tilde{H_1}(k)$ décrivant le processus d'arrivée qui borne supérieurement ou inférieurement le trafic en entrée initial, nous notons par $\tilde{H_3}(k)$ l'état de la chaîne de Markov associée à ce processus d'arrivée. Le nombre de sorties sous les mêmes hypothèses est noté par $\tilde{H_5}(k)$. Le théorème 4.3.9 de Stoyan [MS02] permet de prouver la propriété suivante :

Proposition 9.1 *Si* $H_1(k) \leq_{st} \tilde{H_1}(k), \forall k \geq 0$ *alors* $H_3(k) \leq_{st} \tilde{H_3}(k), \forall k \geq 0$.

PREUVE. Par récurrence sur k, nous utilisons le fait que les fonctions `min` et `max` sont des fonctions croissantes et nous appliquons le théorème 4.3.9 de [MS02]. Ainsi, si nous avons $H_3(k) \leq_{st} \tilde{H_3}(k)$ et $H_1(k) \leq_{st} \tilde{H_1}(k)$, alors nous déduisons que :

$$H_3(k+1) \leq_{st} \tilde{H_3}(k+1).$$

■

Nous avons également le résultat pour les bornes inférieures.

Proposition 9.2 *Si* $\tilde{H_1}(k) \leq_{st} H_1(k)$ *pour tout* $k \geq 0$*, alors* $\tilde{H_3}(k) \leq_{st} H_3(k)$ *pour tout* $k \geq 0$.

D'après l'équation 7.2, nous déduisons que nous avons également des bornes sur le processus de sortie. Les preuves sont similaires à celle de la proposition 9.1 et sont donc omises.

Proposition 9.3 *Si* $H_1(k) \leq_{st} \tilde{H_1}(k)$ *pour tout* $k \geq 0$*, alors* $H_5(k) \leq_{st} \tilde{H_5}(k)$, $\forall k \geq 0$. *De même, si* $\tilde{H_1}(k) \leq_{st} H_1(k)$ $\forall k \geq 0$*, alors,* $\tilde{H_5}(k) \leq_{st} H_5(k)$, $\forall k \geq 0$.

Sachant que ces propositions sont vraies pour toutes les dates (slot) k, alors elles sont également vraies pour les versions stationnaires lorsque les chaînes sont ergodiques. En supposant que les arrivées $\tilde{H_1}(k)$ correspondent aux bornes stochastiques développées dans cette thèse, nous énonçons le théorème important suivant. Nous donnons le cas des bornes supérieures, les bornes inférieures peuvent être obtenues de manière similaire.

Théorème 9.1 *Soient* H_1 *(resp.* $\tilde{H_1}$*) l'histogramme d'entrée associé à la distribution stationnaire exacte (resp. borne supérieure) et* H_3, H_5 *(resp.* $\tilde{H_3}$, $\tilde{H_5}$*) la distribution stationnaire associée à la longueur du tampon et le processus de sortie sous la distribution d'entrée exacte* H_1 *(resp. borne supérieure* $\tilde{H_1}$*), alors nous avons :*

$$H_3 \leq_{st} \tilde{H_3} \text{ et } H_5 \leq_{st} \tilde{H_5}.$$

PREUVE. Par construction sur $H_1(k) \leq_{st} \tilde{H_1}(k)$, il découle des propositions 9.1 et 9.3 que nous avons des comparaisons pour tout k, donc également pour les processus stationnaires lorsque $k \to \infty$.
Remarquons que par construction H_3 et H_5 existent (en raison de l'hypothèse d'ergodicité). ■

9.2.2 Résultats sur les réseaux feed-forward

Pour un réseau feed-forward, il suffit de remarquer que le processus de sortie d'un élément de réseau i (file d'attente, élément de fusion ou élément de division) correspond au processus d'entrée de l'élément de réseau i+1. Les propositions précédentes ainsi que les résultats prouvés sur les éléments de routage dans le chapitre précédent, permettent de montrer que la relation d'ordre sur le trafic entrant dans la file 1 est préservé pour le trafic rentrant dans l'élément de réseau 2. Le calcul d'un trafic bornant à l'entrée de la file 1 peut donc se propager à toutes les files dans le réseau feed-forward. La formule de Little permet d'obtenir le temps moyen de réponse dans un élément du réseau. Le temps de bout en bout est obtenu par sommation grâce à la linéarité des espérances.

Proposition 9.4 *Dans un réseau feed-forward, si nous bornons le processus d'entrée* H_1 *dans la file source par deux bornes inférieures et supérieures au sens st, alors nous obtenons des bornes inférieures et supérieures au sens "st" sur les mesures de performance de tout élément du réseau. Ces bornes au sens st permettent de calculer des bornes inférieures et supérieures pour l'espérance de la longueur du tampon, car l'espérance est une fonction croissante, et grâce à la formule de Little [Kle76], pour les temps moyens. Par la linéarité des espérances, nous obtenons une borne inférieure et supérieure du temps moyen de traversée de bout en bout.*

Nous précisons que les bornes définies ici sont calculées sur une approximation du réseau réel, car l'analyse du réseau dans cette partie est effectuée par la méthode de décomposition qui est une méthode d'approximation. Cependant, il est utile de noter que l'analyse de la première file du réseau est exacte et que par conséquent les bornes calculées sur cette dernière représentent bien des bornes sur l'exact. Nous présentons ci-après quelques évaluations numériques de notre analyse.

Exemple 9.1 (Exemple numérique à partir de traces réelles pour un réseau série)
Nous étudions le réseau en tandem décrit dans la figure 9.2 sous la trace du trafic MAWI (figure 5.1). Le réseau est composé de trois files d'attente avec des capacités de tampon finies : $\mathrm{B}_1 = 2\,Mb$, $\mathrm{B}_2 = 1\,Mb$ *et* $\mathrm{B}_3 = 1\,Mb$, *respectivement. Le service est considéré déterministe dans chaque file d'attente et donné par :* $S_1 = 110\,Mbps$, $S_2 = 107.5\,Mbps$ *et* $S_3 = 106.5\,Mbps$. *Afin d'accélérer le temps de calcul, nous avons fixé l'unité de données à* $\mathrm{D} = 5\,Kb$. *Nous précisons que les paramètres considérés dans ces expériences sont pris de [HOVC10, HOVC07a] pour faciliter la comparaison des résultats.*

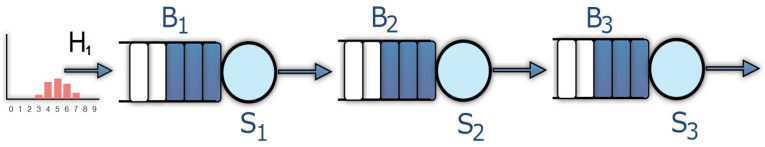


FIGURE 9.2 – Exemple de réseau en tandem.

Nous rappelons que l'analyse est réalisée file après file, c'est-à-dire que les entrées internes du système correspondent aux sorties des files d'attente qui les précèdent immédiatement. L'analyse du réseau est réalisée en considérant les trois méthodes suivantes : méthode de décomposition du réseau notée $\mathbf{Decomp}$., *méthode HBSP (Hernàndez et* al. *[HOVC10]) et notre méthode de borne (borne inférieure notée* $\mathbf{L.b}$ *et borne supérieure notée* $\mathbf{U.b}$*). Les algorithmes de bornes sont ainsi appliqués sur chaque histogramme d'entrée des files d'attente. Deux réductions de taille d'histogramme sont calculées :* $bins = 100$ *et* $bins = 200$ *pour les entrées de chaque file d'attente (voir le tableau 9.1 et le tableau 9.2).*

Nous spécifions pour chacune des files, certaines mesures de performance comme : la probabilité de blocage (la probabilité que le tampon soit plein, notée $\mathrm{Prob(B)}$*), l'espérance de la longueur du tampon (*$\mathbb{E}[H_3]$*) et l'espérance du nombre de données en sortie* $\mathbb{E}[H_5]$*). Nous calculons également les délais de transmission moyens (*$\mathbb{E}[T]$*) en utilisant la formule de Little [Kle76]. Dans la dernière ligne des tableaux, nous présentons les temps d'exécutions globaux nécessaires à l'analyse du réseau associé aux les différentes méthodes considérées.*

		Decomp.	**L. b**	**U. b**	**HBSP**
File 1	$\mathrm{Prob(B)}$	0.1818	0.1737	0.1967	0.2147
	$\mathbb{E}[H_3]$ $(bits)$	938529	916776	977657	1019260
	$\mathbb{E}[H_5]$ $(bits)$	4261850	4253810	4275090	4244160
	$\mathbb{E}[T]$	0.2202	0.2155	0.2287	0.2402
File 2	$\mathrm{Prob(B)}$	0.1735	0.1551	0.2127	0.1551
	$\mathbb{E}[H_3]$ $(bits)$	488719	465232	530817	468094
	$\mathbb{E}[H_5]$ $(bits)$	4246920	4239950	4257230	4234410
	$\mathbb{E}[T]$	0.1151	0.1097	0.1247	0.1105
File 3	$\mathrm{Prob(B)}$	0.1635	0.1243	0.2485	$1.18\ .10^{-6}$
	$\mathbb{E}[H_3]$ $(bits)$	505240	465232	594579	39418.4
	$\mathbb{E}[H_5]$ $(bits)$	4240800	4234810	4248200	4234410
	$\mathbb{E}[T]$	0.1191	0.1098	0.1399	0.0093
Temps d'exécution (s)		21868	2.94	2.89	0.13

TABLE 9.1 – Résultats numériques pour un réseau en tandem avec un nombre de bins égal à 100

Nous constatons que notre méthode de bornes fournit de meilleurs résultats que la méthode HBSP qui on peut le dire se dégrade à mesure qu'on avance dans le réseau. Chose qui ne se produit pas avec notre méthode, car cette dernière offre au niveau de chaque file un encadrement très pertinent sur les mesures exactes par décomposition. Nous remarquons également que notre méthode est très rapide comparée à l'approche sans bornes, même si elle reste légèrement supérieure en temps de calcul à la méthode HBSP. Mais au final, nous pouvons dire que le compromis complexité / précision est nettement avantageux pour notre méthode.

		Decomp.	**L. b**	**U. b**	**HBSP**
File 1	$\mathrm{Prob(B)}$	0.1818	0.1795	0.1915	0.1974
	$\mathbb{E}[H_3]$ $(bits)$	938529	931775	962347	966804
	$\mathbb{E}[H_5]$ $(bits)$	4261850	4259190	4269830	4253710
	$\mathbb{E}[T]$	0.2202	0.2187	0.2254	0.2273
File 2	$\mathrm{Prob(B)}$	0.1735	0.1683	0.1963	0.1109
	$\mathbb{E}[H_3]$ $(bits)$	488719	481468	513482	424992
	$\mathbb{E}[H_5]$ $(bits)$	4246920	4244590	4253090	4246330
	$\mathbb{E}[T]$	0.1151	0.1134	0.1207	0.1001
File 3	$\mathrm{Prob(B)}$	0.1635	0.1512	0.2106	0.1229
	$\mathbb{E}[H_3]$ $(bits)$	505240	488868	556369	456314
	$\mathbb{E}[H_5]$ $(bits)$	4240800	4238830	4245290	4242080
	$\mathbb{E}[T]$	0.1191	0.1153	0.1310	0.1076
Temps d'exécution (s)		21868	5.34	6.57	0.15

TABLE 9.2 – Résultats numériques pour un réseau en tandem avec un nombre de bins égal à 200

Nous nous intéressons à présent aux réseaux feed-forward composés respectivement de files d'attente et d'éléments de routage. En considérant toujours une trace de trafic réelle comme processus d'entrée, nous proposons à travers l'exemple suivant de déterminer les paramètres de performance du réseau. Nous notons que contrairement à notre méthode de bornes, la méthode HBSP ne peut être étendue aux réseaux de files d'attente composés d'éléments de division. Ceci s'explique par le fait que le procédé de réduction utilisé par cette méthode construit des histogrammes définit sur des états réels ce qui empêche l'application directe des modèles de routage sans apporter un changement à l'espace d'état des histogrammes. Malgré que les auteurs ont mentionné l'applicabilité de leur méthode sur les réseaux ouverts, aucune étude ni expérimentation n'a été présentée dans leurs articles.

Exemple 9.2 (Exemple numérique à partir de traces réelles pour un réseau feed-forward) *Nous considérons un modèle de réseau feed-forward représenté dans la figure 9.3 avec* 6 *nœuds, avec comme histogramme d'entrée* (H_1) *obtenu à partir de la trace de trafic MAWI. Chaque nœud est un élément de division (resp. fusion) ou une file d'attente à capacité finie (*$\mathrm{B_i} = 10\,\mathrm{Mb}$, $i = 1, 3, 4, 6$*). Le service de chaque file d'attente est pris respectivement égal à* $110\,Mb/s$, $67.5\,Mb/s$, $90\,Mb/s$ *et* $117.5\,Mb/s$.

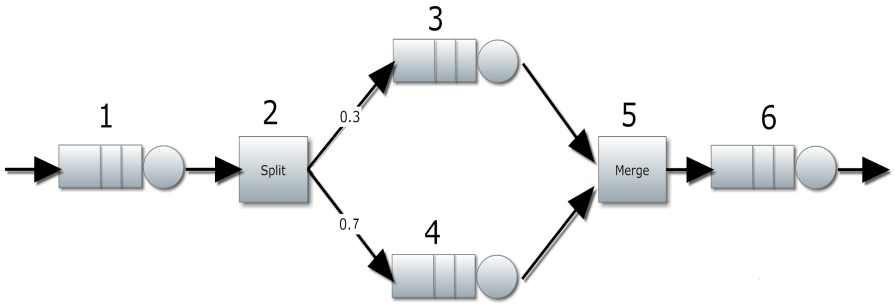


FIGURE 9.3 – Exemple de réseau Feed Forward.

		$\mathbb{E}[H_1]$	$\mathbb{E}[H_3]$	$\mathbb{E}[H_5]$	P_L
File 1	Decomp.	4375620	4332130	4352390	0.00530
	$L.b$	4356310	3850300	4339640	0.00382
	$U.b$	4397080	4890670	4364590	0.00739
File 2	Decomp.	1305720	875834	1305030	0.00052
	$L.b$	1300650	863705	1300010	0.00049
	$U.b$	1309460	884739	1308730	0.00055
File 3	Decomp.	3046670	2256310	3037710	0.00293
	$L.b$	3034840	2190650	3026910	0.00260
	$U.b$	3055400	2304630	3045640	0.00318
File 4	Decomp.	4342730	2519100	4327660	0.00346
	$L.b$	4313200	2340670	4301450	0.00272
	$U.b$	4357650	2593470	4341260	0.00375

TABLE 9.3 – Résultats pour bins=100.

Dans les tableaux 9.3 et 9.4, nous donnons les résultats d'approximations obtenus par la méthode de décomposition pour les quatre files d'attente du réseau et ceux calculés en utilisant nos bornes stochastiques. Nous considérons respectivement un nombre de bins égal à 100 *et un nombre de bins égal à* 200*.*

		$\mathbb{E}[H_1]$	$\mathbb{E}[H_3]$	$\mathbb{E}[H_5]$	P_L
File 1	Decomp.	4375620	4332130	4352390	0.00530
	$L.b$	4366450	4099970	4346550	0.00455
	$U.b$	4386860	4622920	4359020	0.00633
File 2	Decomp.	1305720	875834	1305030	0.00052
	$L.b$	1302780	868692	1302120	0.00050
	$U.b$	1307780	880729	1307060	0.00054
File 3	Decomp.	3046670	2256310	3037710	0.00293
	$L.b$	3039820	2217560	3031480	0.00273
	$U.b$	3051480	2282840	3042080	0.00306
File 4	Decomp.	4342730	2519100	4327660	0.00346
	$L.b$	4322810	2398430	4310020	0.00295
	$U.b$	4350010	2555410	4334300	0.00360

TABLE 9.4 – Résultats pour bins=200.

D'après ces tableaux, nous remarquons que nos bornes sont garanties à chaque étape intermédiaire (en raison de la H-monotonie des éléments du réseau). Nous pouvons également voir que les résultats fournis par nos bornes sont très précis. Les temps d'exécution des bornes pour un nombre de bins égal à 100 *(resp.* 200*) sont obtenus respectivement en* $14.4\,s$ *(resp.* 22.1*) pour la borne inférieure et* 15.9 *s (resp.* $25.9\,s$*) pour la borne supérieure. Le calcul par décomposition des différentes mesures de performance du réseau a été obtenu en un peu plus de trois jours :* $314248\,s$*. Nous pouvons donc conclure que les mesures du réseau obtenues par la méthode de décomposition peuvent être bornées en utilisant la méthode proposée avec une complexité de calcul relativement faible.*

Nous allons maintenant passer à la présentation et l'étude de l'autre approche d'analyse des réseaux DAG. Cette approche valable pour les réseaux ayant une topologie d'arbre, permet de construire des chaînes de Markov bornantes du réseau de files d'attente.

9.3 Bornes sur les mesures exactes des réseaux de files d'attente en tandem

Considérant un réseau de files d'attente en tandem, nous nous intéressons dans ce qui suit à la détermination de bornes sur les paramètres de sortie du réseau. Ayant pour but de dériver des bornes plutôt que des approximations sur les mesures de performance, nous proposons ici différentes approches permettant de construire des encadrements sur certaines mesures telles

que : le nombre de donnés en sortie, les délais de bout en bout et les pertes totales dans le réseau. Cependant, pour chaque approche étudiée, nous ne perdons pas de vue notre objectif principal qui est, d'utiliser notre approche de bornes stochastiques sur les distributions. Nous voulons en effet observer le compromis (précision et temps de calcul) offert par notre méthode et confirmer l'attrait que peut représenter cette dernière dans le domaine de l'analyse de files d'attente.

Nous allons pour commencer présenter le modèle étudié. Nous considérons un réseau de file d'attente en série noté comme suit : $H_1/D/S_1/B_1 \rightarrow /D/S_2/B_2 \rightarrow \ldots \rightarrow /D/S_i/B_i \rightarrow \ldots \rightarrow /D/S_N/B_N$, composé de N file d'attente en série. Chaque file d'attente i possède un tampon de longueur B_i et un service déterministe de capacité S_i. Le système est supposé vide au départ. Les données entrent dans la première file d'attente selon une distribution d'arrivée H_1, puis passent à travers les files d'attente dans l'ordre : les données sortantes de la première file d'attente entrent au bout d'au moins un intervalle de temps dans la deuxième file d'attente, et ainsi de suite (voir la figure 9.4).

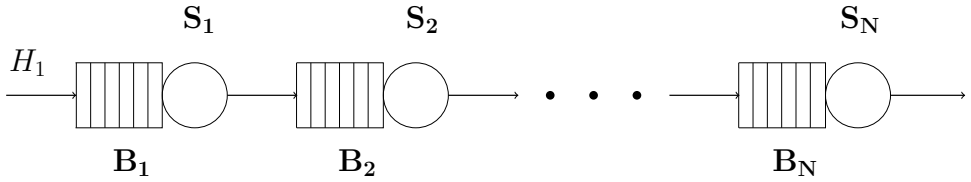


FIGURE 9.4 – Réseau en tandem composé de N files d'attente.

L'analyse numérique exacte de ce système est très difficile voire impossible à effectuer vu la taille du modèle associé. Par contre, la simulation est simple. Dans cette partie et contrairement aux méthodes souvent utilisées dans la littérature qui reposent sur une analyse approximative, nous proposons d'avoir une démarche tout à fait différente qui consiste à déterminer des bornes sur les différents paramètres du réseau : délais de bout en bout, distribution de sortie, nombre total des pertes. Nous avons pour ce faire, développé quatre approches d'analyse distinctes répondant à cet objectif.

Cependant, avant d'entamer la description de ces différentes approches, nous allons tout d'abord étudier l'influence que peut avoir la séquence des capacités de service des files d'attente dans l'analyse d'un réseau en tandem.

9.3.1 Influence de la séquence des capacités de service du réseau sur l'analyse de performance

Lors de l'analyse de réseaux en tandem, nous remarquons que selon la composition de la séquence des capacités de service des files d'attente, des modifications peuvent être apportées au réseau initial simplifiant ainsi sa résolution. Nous distinguons à cet effet deux cas : séquence des capacités de service croissante, c.-à-d. $S_1 \leq S_2 \leq S_3 \leq \ldots \leq S_N$ et séquence des capacités de service générale. Ces deux points sont présentés ci-après.

a) Séquence des capacités de service croissante

Il est clair que pour cette configuration l'analyse exacte du réseau se résume à l'analyse de la première file d'attente uniquement. Pour les autres files, aucune perte ne sera observée et seul le temps de traversée sera pris en considération. Ce temps est donné par un intervalle de temps (slot) pour chaque file d'attente $i > 1$.

b) Séquence des capacités de service générale

Dans ce cas, nous présentons d'abord certaines définitions utiles à notre étude. Nous introduisons pour ce faire la notion de goulot d'étranglement plus communément appelée *Bottleneck*. Les nœuds (files d'attente) qui ont la particularité de limiter les performances globales du réseau et correspondant aux nœuds surchargés sont dits *Bottleneck*. Nous précisons que la notion de bottleneck qui nous intéresse n'est pas tout à fait équivalente à celle étudiée en analyse opérationnelle [Jai91], nous donnons ci-après la définition exacte utilisée dans notre étude.

Dans un réseau de files d'attente en tandem, nous pouvons distinguer deux types de bottleneck : *Bottleneck* local et *Bottleneck* global. Nous donnons les définitions respectives ci-après.

Soit S_i la capacité de service de la file i et B_i la longueur du tampon. Pour tout $i > 1$, un bottleneck local est défini comme suit :

Définition 9.1 (Bottleneck local) *Une file i est dite bottleneck local (noté* BL*) si*

$$\forall\, j < i, \quad S_j > S_i.$$

De cette définition découle une propriété simple, énoncée ci-après.

Propriété 9.1 *Si la file i n'est pas un bottleneck local, alors elle est vide à chaque instant $n \geq 1$. Nous notons que par hypothèse la file i est vide à l'instant $n = 0$.*

PREUVE. Soit $H_3^i(n+1)$ la longueur de la file i à l'instant $(n+1)$ définie par l'équation d'évolution suivante :

$$H_3^i(n+1) = \min\left(B_i,\, (H_3^i(n) + H_1^i(n+1) - S_i)^+\right), \tag{9.1}$$

où $H_1^i(n+1)$ est le processus d'entrée de la file i à l'instant $n+1$. Nous avons $H_3^i(0) = 0$ (par hypothèse) et $H_3^i(1) = \min\left(B_i,\, (H_1^i(1) - S_i)^+\right)$ d'après l'équation 9.1. Comme la file i n'est pas un *Bottleneck* local, nous avons :

$$H_1^i(1) \leq S_i \implies H_3^i(1) = 0.$$

Par récurrence, nous obtenons le résultat suivant :

$$H_1^i(n+1) \leq S_i \implies H_3^i(n+1) = 0, \quad \forall\, n.$$

Ainsi, les données en sortie sont égales aux données en entrée. De plus, si les données en entrée sont indépendantes alors les données en sortie le sont également. ■

Nous passons à présent à la définition d'un *bottleneck* global.

Définition 9.2 (*Bottleneck* global) *Une file i ($i \geq 1$) est dite bottleneck global, noté* BG *si*

1. $\forall j > 1,\ j \neq i :\quad S_j > S_i$;
2. *ou si* $S_j = S_i$, *alors* $i < j$.

Après avoir donné la définition des différents *bottleneck*, nous allons nous intéresser à leurs emplacements dans le réseau. En reprenant la configuration d'un réseau en tandem composé de N files d'attente, nous distinguons deux cas selon la position du *bottleneck* dans le réseau :

1. *Bottleneck* global en tête du réseau. Le fait que la file d'entrée du réseau soit un *Bottleneck* global revient à dire que les files de 2 à N ne perdent aucune données. Dans ce cas, l'analyse du système peut être effectuée de façon exacte. La première file est analysée. Pour les autres files, seuls les délais de traversées seront considérés (les données en entrée des files $i > 1$ sont plus petites que leurs capacité de service ($H_1^i < S_i$), ce qui implique qu'elles sortent au bout d'un intervalle de temps).
2. Sinon, nous indiquons au préalable les positions des bottlenecks locaux (*BL*) et du bottleneck global (*BG*) dans le réseau. Nous illustrons dans la figure 9.7 un exemple de position des bottlenecks.

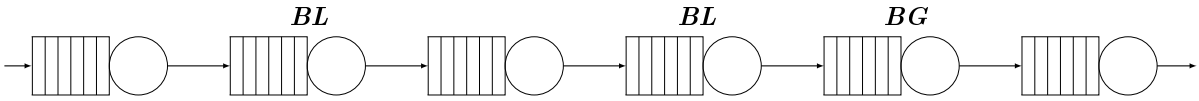


FIGURE 9.5 – Exemple de positions des bottlenecks dans un réseau en tandem.

Nous proposons de dériver dans un premier temps une forme réduite du réseau initial (nous effectuons un postprocessing sur le délai et la taille du réseau). Nous construisons à cet effet un nouveau réseau selon les étapes suivantes :

(a) Si nous avons une file i telle que i >BL (ou i >BG) et i n'est ni bottleneck local ni un bottleneck global, alors nous proposons d'enlever la file i, comme illustré dans la figure 9.6.

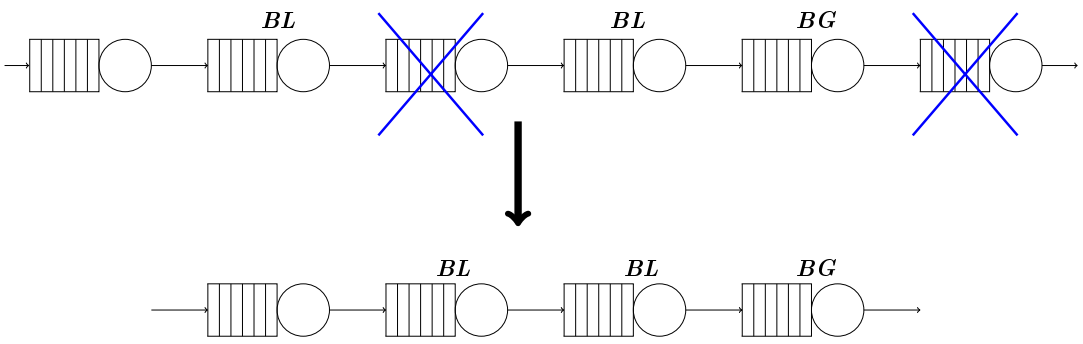


FIGURE 9.6 – Réseau réduit -étape 1-.

En effet, si une file d'attente i vérifie ces conditions, ceci signifie que $S_i \geq S_{BL}$ (ou $S_i \geq S_{BG}$). Nous retombons donc dans une configuration de séquence de service croissante, qui se résume à l'analyse de la file BL (ou BG).

(b) Si nous avons une file i telle que $i > 1$ (différente de la première file du réseau) et qui est située avant le premier bottleneck local, alors nous pouvons également l'éliminer (conséquence du résultat énoncé dans la sous-section 9.3.1.a).

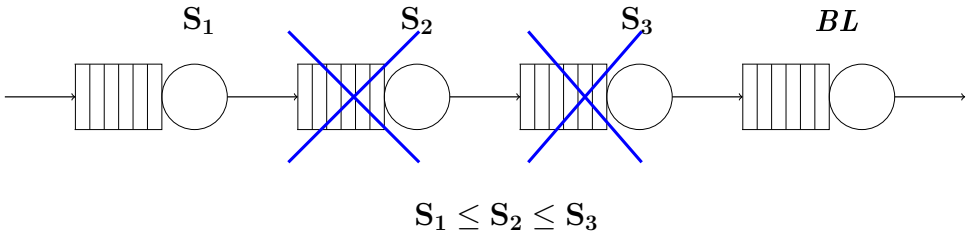


FIGURE 9.7 – Réseau réduit -étape 2-.

Nous définissons au final un nouveau réseau réduit, composé uniquement de la première file d'attente du réseau initial, des bottleneck locaux et du bottleneck global. Notre but consiste maintenant à analyser ce nouveau réseau et déterminer des bornes sur ses mesures de performances. Une généralisation au réseau initial sera ensuite effectuée en ajoutant les temps de traversée des files non considérées (supprimées). Dans cette optique, nous proposons quatre approches d'analyse permettant de définir des bornes supérieures ou inférieures sur les paramètres du réseau en tandem : nombre de données traitées, délais de bout en bout et nombre de pertes dans le réseau. Ces approches sont présentées de manière détaillée ci-après.

9.3.2 Approche 1 : Détermination de bornes supérieures

Cette approche à pour but de construire des bornes supérieures sur les paramètres de sortie d'une file d'attente i du réseau, $1 < i \leq N$. Par souci de clarté, nous proposons de présenter cette approche selon deux étapes. Ces étapes ont pour but de synthétiser l'idée ainsi que le procédé régi par cette dernière.

Étape 1 : modèle préliminaire

Soit un réseau en tandem composée de N files d'attente à capacités finies, la détermination de bornes supérieures sur les mesures de performance d'une file d'attente i, $1 < i \leq N$, consiste à poser tous les tampons des files précédent la file i à l'infini (c.-à-d. $B_j = \infty, \quad j < i$). Cette démarche permet de définir des bornes supérieures sur : le nombre total de données en sortie, les pertes et les délais de bout en bout, de la file i. Nous prouvons ces résultats ci-après.

a) Borne supérieure sur le nombre total de données en sortie : Nous commençons par analyser un réseau en tandem simple composé de deux files d'attente.

Nous considérons le réseau de files d'attente en tandem suivant : $H_1/D/S_1/B_1 \rightarrow /D/S_2/B_2$ (noté **Réseau A**). Nous définissons un second réseau ayant les mêmes paramètres sauf pour la longueur du tampon de la file 1. Nous proposons en effet d'augmenter d'une unité la longueur

du tampon B_1, ce qui nous donne le réseau $H_1/D/S_1/B_1+1 \rightarrow /D/S_2/B_2$ (noté **Réseau B**). En effectuant ce changement, nous voulons montrer la comparaison stochastique existante entre les processus de sortie finaux du premier et du deuxième réseau.

Les notations suivantes sont utilisées durant nos preuves. Pour le premier réseau, $H_3^i(n)$ représente le nombre de données à l'instant n dans le tampon B_i et $H_5^i(n)$ décrit le nombre de données en sortie dans la file i à la date n. Pour le deuxième réseau, nous notons par $\widehat{H}_3^i(n)$ le nombre d'unité de donnée dans le tampon B_i et par $\widehat{H}_5^i(n)$ le nombre de données en sortie dans la file i.

Lemme 9.1 *Soient deux réseaux de files d'attente en tandem $H_1/D/S_1/B_1 \rightarrow /D/S_2/B_2$ et $H_1/D/S_1/B_1+1 \rightarrow /D/S_2/B_2$, nous avons :*

$$H_5^2(n) \leq_{st} \widehat{H}_5^2(n), \quad \forall\, n.$$

PREUVE. Soient les équations d'évolution du nombre de données dans le tampon et du nombre de données en sortie des deux files d'attente des réseaux A et B :

Réseau A

$$\begin{cases} H_3^1(n) = \min\left(B_1,\, (H_3^1(n-1) + H_1(n) - S_1)^+\right); & \text{(9.2a)} \\ H_5^1(n) = \min\left(S_1,\, H_3^1(n-1) + H_1(n)\right); & \text{(9.2b)} \\ H_3^2(n+1) = \min\left(B_2,\, (H_3^2(n) + H_1^2(n+1) - S_2)^+\right); & \text{(9.2c)} \\ H_5^2(n+1) = \min\left(S_2,\, H_3^2(n) + H_1^2(n+1)\right). & \text{(9.2d)} \end{cases}$$

Réseau B

$$\begin{cases} \widehat{H}_3^1(n) = \min\left(B_1+1,\, (\widehat{H}_3^1(n-1) + H_1(n) - S_1)^+\right); & \text{(9.3a)} \\ \widehat{H}_5^1(n) = \min\left(S_1,\, \widehat{H}_3^1(n-1) + H_1(n)\right); & \text{(9.3b)} \\ \widehat{H}_3^2(n+1) = \min\left(B_2,\, (\widehat{H}_3^2(n) + \widehat{H}_1^2(n+1) - S_2)^+\right); & \text{(9.3c)} \\ \widehat{H}_5^2(n+1) = \min\left(S_2,\, \widehat{H}_3^2(n) + \widehat{H}_1^2(n+1)\right). & \text{(9.3d)} \end{cases}$$

Nous notons par $H_1^2(n+1)$ la distribution d'entrée de la file 2 à la date $n+1$. Nous précisons que l'analyse séquentielle du réseau, entraîne un décalage temporel d'un intervalle de temps entre la file 1 et la file 2 du réseau.
Nous avons également les hypothèses de départ suivantes : $H_3^1(0) = 0$, $\widehat{H}_3^1(0) = 0$ et $H_3^2(i) = 0$, $\widehat{H}_3^2(i) = 0$ pour $i = 0, 1$.

- Pour $n = 1$, nous avons d'après les équations 9.2b et 9.3b, le nombre de données en sortie de la file 1 :

  $$H_5^1(1) = \min(S_1,\, 0 + H_1) = \widehat{H}_5^1(1),$$

  car le flux à l'entrée des deux réseaux est identique. Le passage à la file 2 nous permet d'avoir le résultat suivant :

  $$H_5^2(2) = \min(S_2,\, 0 + H_5^1(1)) = \min(S_2,\, 0 + \widehat{H}_5^1(1)) = \widehat{H}_5^2(2)$$

  Par conséquent, $H_5^2(2) \leq_{st} \widehat{H}_5^2(2)$.

- À présent, nous allons raisonner par induction sur n. Nous supposons que $H_3^1(n-1) \leq_{st} \widehat{H}_3^1(n-1)$ et $H_3^2(n) \leq_{st} \widehat{H}_3^2(n)$ et nous voulons montrer que $H_3^2(n+1) \leq_{st} \widehat{H}_3^2(n+1)$ est vrai. Ce résultat nous permet de déduire après que si $H_5^2(n) \leq_{st} \widehat{H}_5^2(n)$ alors, $H_5^2(n+1) \leq_{st} \widehat{H}_5^2(n+1)$ est également vrai.
  Pour commencer, nous avons :
  . **Réseau A** : $H_5^1(n) = \min\left(S_1,\, H_3^1(n-1) + H_1(n)\right).$

  . **Réseau B** : $\widehat{H}_5^1(n) = \min\left(S_1,\, \widehat{H}_3^1(n-1) + H_1(n)\right).$

  Sachant que la capacité de service et le processus d'entrée dans les deux modèles sont identiques, que $H_3^1(n-1) \leq_{st} \widehat{H}_3^1(n-1)$ (par hypothèse), et que la fonction $\min$ est une fonction croissante, nous déduisons le résultat suivant :

$$H_5^1(n) \leq_{st} \widehat{H}_5^1(n). \tag{9.4}$$

  Nous étudions ensuite à la file 2 à l'instant $n+1$. La distribution du nombre de données dans cette file est donnée respectivement par l'équation 9.2c pour le réseau A et par l'équation 9.3c pour le réseau B. Ces deux distributions sont comparables au sens de l'ordre stochastique fort "st" comme suit :

$$H_3^2(n+1) \leq_{st} \widehat{H}_3^2(n+1),$$

  car $H_3^2(n) \leq_{st} \widehat{H}_3^2(n)$ (par hypothèse), les distributions d'arrivée des deux réseaux $H_5^1(n)$ et $\widehat{H}_5^1(n)$ sont comparable stochastiquement $H_5^1(n) \leq_{st} \widehat{H}_5^1(n)$ (équation 9.4) et que la fonction $\min$ est monotone.
  Pour ce qui est du processus de sortie de la deuxième file d'attente, ce dernier est donné par :

  . **Réseau A** : $H_5^2(n+1) = \min\left(S_2,\, H_3^2(n) + H_5^1(n)\right).$

  . **Réseau B** : $\widehat{H}_5^2(n+1) = \min\left(S_2,\, \widehat{H}_3^2(n) + \widehat{H}_5^1(n)\right).$

  Le processus d'entrée de la file 2 est remplacé par le processus de sortie de la file qui la précède ($H_1^2(n+1) = H_5^1(n)$ pour le réseau A et $H_1^2(n+1) = \widehat{H}_5^1(n)$ pour le réseau B). Nous avons : la capacité de service est identique dans les deux réseaux (S_2), $H_3^2(n) \leq_{st} \widehat{H}_3^2(n)$ (par hypothèse) et $H_5^1(n) \leq_{st} \widehat{H}_5^1(n)$ (équation 9.4). Ainsi, comme la fonction $\min$ est une fonction croissante, nous déduisons le résultat suivant :

$$H_5^2(n+1) \leq_{st} \widehat{H}_5^2(n+1).$$

Ainsi, pour tout $n \geq 0$, nous avons $H_5^2(n) \leq_{st} \widehat{H}_5^2(n)$. ■

D'après le lemme 9.1, nous énonçons les deux résultats suivants :

Corollaire 9.1 *Soient deux réseaux de files d'attente en tandem* $H_1/D/S_1/B_1 \to /D/S_2/B_2$ *et* $H_1/D/S_1/\infty \to /D/S_2/B_2$*, nous avons :*

$$H_5^2(n) \leq_{st} \widehat{H}_5^2(n), \quad \forall\, n.$$

Lemme 9.2 *Soient deux réseaux de files d'attente en tandem* $H_1/D/S_1/B_1 \rightarrow /D/S_2/B_2$ *et* $H_1/D/S_1/\infty \rightarrow /D/S_2/B_2$*, nous avons :*

$$\sum_n \mathbb{E}[H_5^2(n)] \leq \sum_n \mathbb{E}[\widehat{H}_5^2(n)].$$

PREUVE. Conséquence du théorème 2.5 et du fait que l'espérance soit une fonction linéaire. ■

Nous pouvons également généraliser ces résultats sur un réseau en tandem à N files d'attente ($N > 1$).

Lemme 9.3 *Soient deux réseaux de files d'attente en tandem* $H_1/D/S_1/B_1 \rightarrow /D/S_2/B_2 \ldots \rightarrow /D/S_N/B_N$ *et* $H_1/D/S_1/\infty \rightarrow /D/S_2/\infty \ldots \rightarrow /D/S_{N-1}/\infty \rightarrow /D/S_N/B_N$*, nous avons :*

$$H_5^N(n) \leq_{st} \widehat{H}_5^N(n), \ \ \forall\, n \ \textit{et} \ \sum_n \mathbb{E}[H_5^N(n)] \leq \sum_n \mathbb{E}[\widehat{H}_5^N(n)].$$

b) Borne supérieure sur le nombre total de données perdues : Nous considérons deux modèles de réseaux en tandem. Le premier noté réseau A, est composé de deux files d'attente à capacité finies. Le second est composé quand à lui d'une première file à capacité infinie et d'une seconde de capacité finie (noté réseau B). Les deux modèles sont représentés dans la figure ci-après.

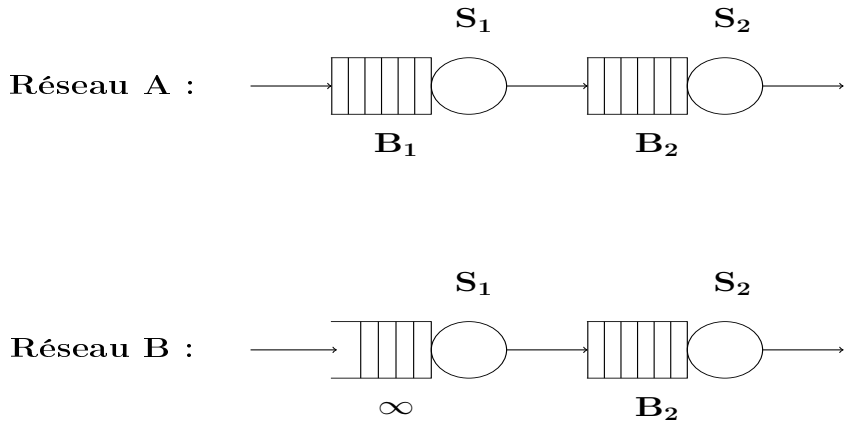


FIGURE 9.8 – Deux modèles de réseaux en tandem.

Nous avons le premier résultat présenté dans le lemme suivant.

Lemme 9.4 *Soient deux réseaux de files d'attente en tandem* $H_1/D/S_1/B_1 \rightarrow /D/S_2/B_2$ *et* $H_1/D/S_1/\infty \rightarrow /D/S_2/B_2$*, nous avons :*

$$H_L^2(n) \leq_{st} \widehat{H}_L^2(n), \ \ \forall\, n, \ \textit{et} \ \sum_n \mathbb{E}[H_L^2(n)] \leq \sum_n \mathbb{E}[\widehat{H}_L^2(n)].$$

PREUVE. Les équations d'évolution du nombre de données perdues à l'instant n dans les deux modèles sont données par :

$$\textbf{Réseau A :} \quad \begin{cases} H_L^1(n) = \max\left(0,\, H_3^1(n-1) + H_1(n) - S_1 - B_1\right); & \text{(9.5a)} \\ H_L^2(n+1) = \max\left(0,\, H_3^2(n) + H_1^2(n+1) - S_2 - B_2\right). & \text{(9.5b)} \end{cases}$$

$$\textbf{Réseau B :} \quad \widehat{H}_L^2(n+1) = \max\left(0,\, \widehat{H}_3^2(n) + \widehat{H}_1^2(n+1) - S_2 - B_2\right). \quad \text{(9.6)}$$

Nous avons les hypothèses de départ suivantes : $H_3^1(0) = 0$, $\widehat{H}_3^1(0) = 0$ et $H_3^2(i) = 0$, $\widehat{H}_3^2(i) = 0$ pour $i = 0, 1$. Nous rappelons qu'il y a un décalage d'une unité de temps entre chaque file du réseau.

- Pour $n = 2$, les pertes au niveau de la deuxième file sont données par :

  $$H_L^2(2) = \max\left(0,\, 0 + H_1^2(2) - B_2 - S_2\right); \quad \text{et} \quad \widehat{H}_L^2(2) = \max\left(0,\, 0 + \widehat{H}_1^2(2) - B_2 - S_2\right).$$

  Nous notons que $H_1^2(2)$ (resp. $\widehat{H}_1^2(2)$) correspond à $H_5^1(1)$ (resp. $\widehat{H}_5^1(1)$), que $H_5^2(2) \leq_{st} \widehat{H}_5^2(2)$ (d'après le corollaire 9.1) et que la fonction $\max$ est une fonction croissante. De ces différents points, nous déduisons le résultat suivant : $H_L^2(2) \leq_{st} \widehat{H}_L^2(2)$.
- Nous raisonnons par la suite par induction sur n. Nous supposons que $H_3^1(n-1) \leq_{st} \widehat{H}_3^1(n-1)$ et $H_3^2(n) \leq_{st} \widehat{H}_3^2(n)$ et nous voulons montrer que $H_3^2(n+1) \leq_{st} \widehat{H}_3^2(n+1)$ est vrai. Ce résultat nous permet de déduire après que $H_L^2(n+1) \leq_{st} \widehat{H}_L^2(n+1)$ est également vrai.
  Reposant sur les résultats démontrés dans la preuve du lemme 9.1, nous avons $H_3^2(n) \leq_{st} \widehat{H}_3^2(n)$ et $H_5^1(n) \leq_{st} \widehat{H}_5^1(n)$. Ainsi, d'après la propriété 2.4 nous avons : $H_3^2(n) + H_5^1(n) \leq_{st} \widehat{H}_3^2(n) + \widehat{H}_5^1(n)$. Sachant que $\max$ est une fonction croissante, ce résultat nous permet d'avoir :

  $$H_L^2(n+1) \leq_{st} \widehat{H}_L^2(n+1).$$

  Ainsi, $\forall\, n \geq 0$, nous avons $\quad H_L^2(n) \leq_{st} \widehat{H}_L^2(n)$.

Sachant que l'espérance est une fonction linéaire nous dérivons le résultat suivant sur le nombre de pertes total depuis l'instant 0 dans les réseaux :

$$\sum_{n\geq 0} \mathbb{E}[H_L^2(n)] \leq \sum_{n\geq 0} \mathbb{E}[\widehat{H}_L^2(n)].$$

■

Une conséquence directe de ce lemme nous permet d'énoncer le résultat important sur le nombre total de pertes dans les deux réseaux présenté ci-après. Nous notons par $\mathbf{PS}_A$ le nombre de pertes totales dans le réseau A depuis l'instant 0. Pour le réseau A (resp. réseau B), nous définissons par $\mathbf{ps}_A^i(n)$ (resp. $\mathbf{ps}_B^i(n)$) le nombre total de données perdues dans la files i depuis l'instant 0.

Lemme 9.5 *Soient deux réseaux de files d'attente en tandem $H_1/D/S_1/B_1 \to /D/S_2/B_2$ et $H_1/D/S_1/\infty \to /D/S_2/B_2$, nous avons :*

$$\mathbf{PS}_A(n) \leq \mathbf{ps}_A^1(n) + \mathbf{ps}_B^2(n), \quad \forall\, n \geq 0.$$

PREUVE. Nous précisons que $\mathbf{ps}_A^i(n) = \sum_{t\leq n} \mathbb{E}[H_L^i(t)]$ et $\mathbf{ps}_B^i(n) = \sum_{t\leq n} \mathbb{E}[\widehat{H}_L^i(t)]$ avec $i = \{1, 2\}$ et $\forall\, n \geq 0$.

Nous avons : $\mathbf{PS}_A(n) = \mathbf{ps}_A^1(n) + \mathbf{ps}_A^2(n)$.

D'après le lemme 9.4, nous déduisons que pour tout $n \geq 0$,

$$\mathbf{PS}_A(n) = \mathbf{ps}_A^1(n) + \mathbf{ps}_A^2(n) \leq \mathbf{ps}_A^1(n) + \mathbf{ps}_B^2(n).$$

Et la preuve est ainsi terminée. ■

Une généralisation de ce résultat à un réseau à N files d'attente est donnée par le lemme suivant.

Lemme 9.6 *Soient deux réseaux de files d'attente en tandem A : $H_1/D/S_1/B_1 \rightarrow \ldots \rightarrow /D/S_N/B_N$ et B : $H_1/D/S_1/\infty \rightarrow \ldots \rightarrow /D/S_{N-1}/\infty \rightarrow /D/S_N/B_N$, nous avons :*

$$\mathbf{PS}_A(n) \leq \sum_{j<N} \mathbf{ps}_A^j(n) + \mathbf{ps}_B^N(n), \quad \forall\, n \geq 0.$$

c) Borne stochastique supérieure sur les délais de bout en bout : Nous allons à présent déterminer une borne supérieure sur les délais. Pour ce faire, nous considérons deux réseaux en tandem composés de i files d'attente. Le premier modèle considéré est composé de i files d'attente de capacité finies. Pour le second modèle, le réseau est composé de i files d'attente avec un tampon de capacité finie pour la file i et des tampons de capacité infinie pour les files $j < i$ (voir figure 9.9).

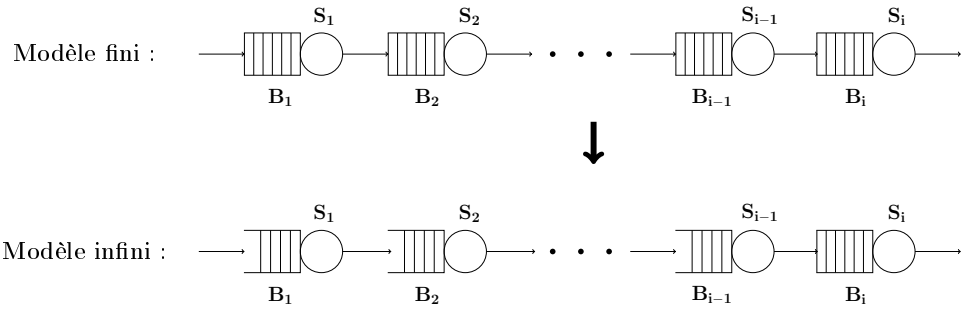


FIGURE 9.9 – Modèles étudiés.

Nous notons par : $T_i^F(k)$ la date de sortie de la file i pour l'unité de donnée k, dans le cas du modèle avec files finies et par $T_i^I(k)$ la date de sortie de la file i pour l'unité de donnée k, dans le cas du modèle avec de files infinies.

Un premier résultat simple est énoncé dans la propriété qui suit.

Propriété 9.2 *Pour toute file d'attente i du réseau, nous avons :*

$$T_i^F(k) \leq T_i^I(k),$$

tel que k est une donnée non perdue dans le réseau composé de la file 1 à la file i.

Dans le cas où la donnée k est perdue, nous pouvons distinguer les deux constructions suivantes :

Construction (A) : Nous considérons que $T_i^F(k) = 0, \forall\, i$.

Construction (B) : Soit j la file où la donnée k est perdue. Deux cas peuvent se présenter :

- Si $j = 1$, nous posons : $T_i^F(k) = 0$;
- Si $j > 1$, $T_i^F(k) = T_{j-1}^F(k)$. Ce qui correspond à la durée de vie de la donnée k dans le réseau.

Avant de présenter le résultat important sur les délais dans le réseau. Nous donnons le résultat trivial suivant :

Lemme 9.7 *Si $j < i$ alors nous avons $T_j^I(k) < T_i^I(k)$.*

Lemme 9.8 *Pour les deux constructions **(A)** et **(B)**, nous avons :*

$$T_i^F(k) \leq T_i^I(k), \quad \forall\, i,\ \forall\, k.$$

PREUVE. Pour cette preuve, nous distinguons les cas suivants :

- Si k n'est pas perdu, l'inégalité est vérifiée (propriété 9.2).
- Si k est perdu et que la construction utilisée est **(A)**, nous avons $T_i^F(k) = 0 < i \leq T_i^I(k)$. Dans le modèle infini la donnée k passe au minimum 1 unité de temps dans chaque file (temps de traversé du serveur).
- Si la construction considérée est **(B)** et que la donnée k est perdu au niveau de la file j, deux cas peuvent se présenter selon la position de j :
  - Si $j = 1$, nous nous retrouvons dans le cas où le modèle admet la construction **(A)**. C'est-à-dire que nous avons $T_i^F(k) = 0 < i \leq T_i^I(k)$.
  - Si $j > 1$, alors la donnée k n'est par perdu au niveau de la file $j-1$. D'après la propriété 9.2, nous avons $T_{j-1}^F(k) \leq T_{j-1}^I(k)$. De plus, nous avons

$$\left.\begin{array}{rl} \text{Par construction :} & T_{j-1}^F(k) = T_i^F(k) \\ \text{D'après le lemme 9.7 :} & T_{j-1}^I(k) \leq T_i^I(k) \end{array}\right\} \quad \Longrightarrow \quad T_i^F(k) \leq T_i^I(k).$$

■

Étape 2 : Modèle résiduel

Après avoir prouvé que le nouveau réseau noté modèle préliminaire pouvait déterminer et garantir des bornes supérieures sur les mesures de performance du réseau initial, la seconde étape que nous entreprenons revient à utiliser un résultat très important dans les réseaux en tandem qui est la propriété d'interchangeabilité.

La notion d'interchangeabilité des files d'attente dans un réseau en tandem signifie que l'ordre des files d'attente n'affecte pas les distributions de sortie du réseau. L'utilisation de la propriété d'interchangeabilité dans les réseaux en tandem est essentiellement due à Friedman en 1965 dans [Fri65], Whitt en 1985 dans [Whi85] et Weber en 1979 [Web]. Friedman a établi dans [Fri65] un résultat très pertinent qui stipule que pour un processus d'arrivée arbitraire, un service déterministe et des tampons infinis dans chaque file, le processus de départ final

du réseau en tandem est indépendant de l'ordre des files d'attente. De plus, une conséquence de ce résultat induit que les distributions des temps de séjour dans l'ensemble du réseau sont également indépendantes de l'ordre des files d'attente. Un résultat semblable a été également établi pour des files avec des temps de service exponentiel par Weber dans [Web]. Cependant, lorsqu'on est en présence de serveurs qui sont à la fois exponentielles et déterministes, la propriété d'interchangeabilité n'est pas vérifiée. Ce résultat est exprimé dans le théorème 9.2. Nous proposons cependant dans cette thèse de réécrire la preuve de Friedman de manière plus détaillée et plus en adéquation avec notre modèle. Dans la preuve de Friedman, aucune information explicite sur les dates d'arrivée des clients ou sur le fait que les clients qui arrivent soient distincts ou égaux n'est donnée. Ainsi, nous proposons de reprendre la preuve en prenant bien en compte le fait que nous avons des arrivées par groupe et que chaque groupe est composé d'un certain nombre d'unités de données ayant toutes les mêmes dates d'arrivée. Nous exposons cette preuve en annexe.

Théorème 9.2 *Deux files d'attente en tandem $./D/S/\infty$ sont interchangeables, nous avons pour toute distribution d'entrées, H_1, la distribution de sortie et la distribution des temps de séjours dans le réseau $H_1/D/S_1/\infty \rightarrow /D/S_2/\infty$ et le réseau $H_1/D/S_2/\infty \rightarrow /D/S_1/\infty$ sont identiques.*

PREUVE. Voir l'annexe. ■

Nous proposons ici d'exploiter le résultat de Friedman sur le modèle préliminaire défini lors de l'étape 1. Nous allons pour ce faire utiliser la notion d'interchangeabilité afin de réorganiser les files d'attente ayant les capacités de tampon infinies. Nous déplaçons la file ayant la capacité de service la plus faible en début de réseau ce qui nous amène à définir une nouvelle séquence de capacité de service des files d'attente qui est croissante.

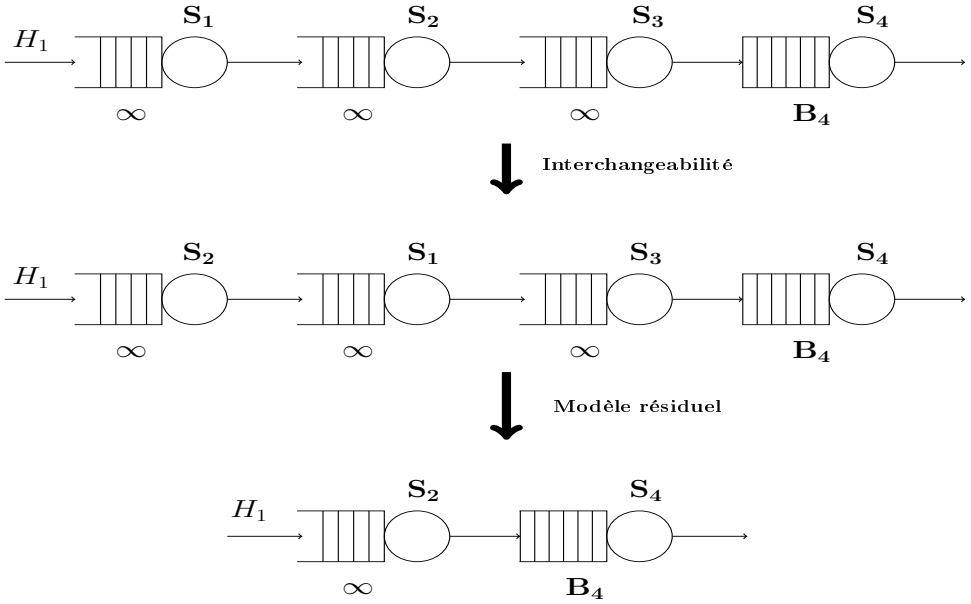


FIGURE 9.10 – Exemple illustrant l'étape 2 de l'approche 1.

D'après la section 9.3.1.a, cette démarche nous permet d'avoir un modèle résiduel restreint constitué uniquement de deux files d'attente. La première correspond à la file ayant la capacité de service la plus faible (Bottleneck global dans le réseau composé des files infinies) et la seconde file correspond à la file étudiée i. Cette démarche est représentée via la figure 9.10. Pour une file d'attente $i = 4$, cette figure illustre notre approche sur un exemple de réseau en tandem ayant la séquence de capacité de service suivante : $S_1 = 4$, $S_2 = 2$, $S_3 = 3$ et $S_4 = 1$.

9.3.3 Approche 2 : détermination de bornes supérieures sur les paramètres du réseau

Dans cette approche, nous proposons de déterminer de nouvelles bornes supérieures sur les nombres de données en sortie du réseau et les délais de bout en bout. Reposant sur le théorème de Friedman, l'idée développée dans cette approche consiste à fixer toutes les longueurs des tampons des files d'attente du réseau à l'infinie. Employant la propriété d'interchangeabilité, nous déplaçons la file *bottleneck* global en tête du réseau. Cette approche nous permet ainsi de dériver des bornes supérieures sur le processus de sortie final du réseau ainsi que sur les délais.

Le fait de déplacer la file bottleneck global en début de réseau, nous ramène à l'analyse de la première file (bottleneck global) uniquement. En effet, la capacité de service de cette file étant la plus faible, son processus de sortie est toujours inférieur aux capacités de service des files qui suivent. Ce processus de sortie ne fait donc que traverser les files jusqu'à sa sortie du réseau. Le procédé de cette approche est illustré dans la figure 9.11.

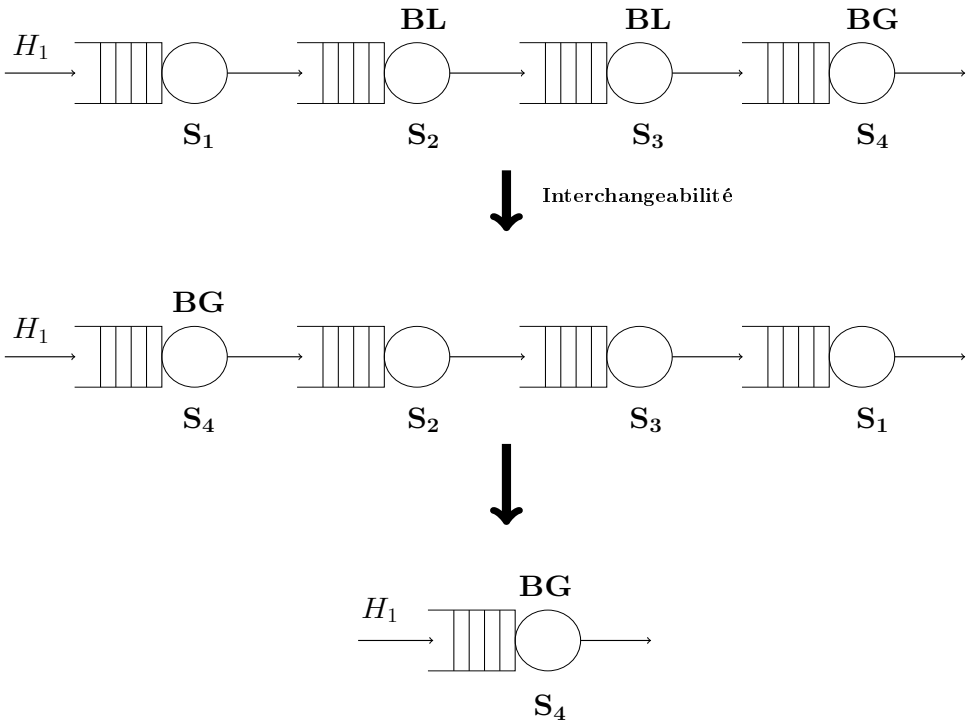


FIGURE 9.11 – Exemple illustrant l'approche 2. *Suite à la permutation des files, les files 2 et 3 qui étaient précédemment bottleneck locaux ne le sont plus.*

Les principaux résultats relatifs à cette approche sont donnés ci-après.

a) Borne supérieure sur le nombre total de données en sortie : Nous notons par $H_5^i(n)$ (resp. $\widehat{H}_5^i(n)$) le nombre de données en sortie de la file i à l'instant n pour le réseau $H_1^1/D/S_1/B_1 \rightarrow /D/S_2/B_2 \rightarrow \ldots \rightarrow /D/S_N/B_N$ (resp. $H_1^1/D/S_1/\infty \rightarrow /D/S_2/\infty \ldots \rightarrow /D/S_N/\infty$).

Lemme 9.9 *Soient deux réseaux de files d'attente en tandem $H_1^1/D/S_1/B_1 \rightarrow /D/S_2/B_2 \rightarrow \ldots \rightarrow /D/S_N/B_N$ et $H_1^1/D/S_1/\infty \rightarrow /D/S_2/\infty \ldots \rightarrow /D/S_N/\infty$, nous avons :*

$$H_5^i(n) \leq_{st} \widehat{H}_5^i(n), \ \ \forall\, n \ \textit{et} \ \sum_n \mathbb{E}[H_5^i(n)] \leq \sum_n \mathbb{E}[\widehat{H}_5^i(n)].$$

PREUVE. Conséquence directe du lemme 9.3. ■

b) Borne stochastique supérieure sur les délais de bout en bout :

Lemme 9.10 *Soit $T^F(k)$ (resp. $T^F(k)$) la date de sortie de l'unité de données k dans le réseau $H_1^1/D/S_1/B_1 \rightarrow /D/S_2/B_2 \rightarrow \ldots \rightarrow /D/S_N/B_N$ (resp. $H_1^1/D/S_1/\infty \rightarrow /D/S_2/\infty \ldots \rightarrow /D/S_N/\infty$), nous avons :*

$$T^F(k) \leq T^I(k), \ \ \forall\, k > 0.$$

Ce résultat est une conséquence directe du lemme 9.8.

9.3.4 Approche 3 : bornes inférieures sur les paramètres de sortie du réseau

Dans le but de définir des bornes inférieures sur les mesures de performances d'une file d'attente i, $i > 1$ dans un réseau en tandem, nous proposons de fixer cette fois-ci les longueurs des tampons de chaque file d'attente à zéro excepté la file i étudiée (voir figure 9.12).

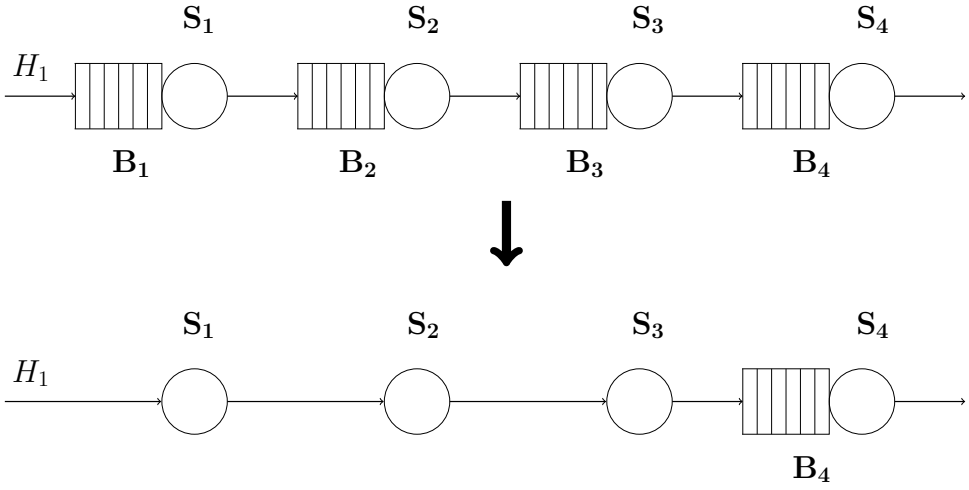


FIGURE 9.12 – Principe de l'approche 3. Exemple de réseau en tandem pour $i = 4$.

Cette démarche nous permet de définir non seulement des bornes inférieures sur les paramètres de sortie finaux du réseau, mais également sur chaque file d'attente i ($i > 1$) qui le compose.

Partant d'un réseau de files d'attente en tandem composé de N files, l'analyse de performance d'une file d'attente i, $1 < i \leq N$ via l'approche 3 revient à fixer les capacités des tampons des files d'attente précédent la file i ($\forall \ell < i$) à zéro. Cette démarche permet de dériver des bornes inférieures sur les mesures de performance de la file i. Les principaux résultats établis pour cette approche sont démontrés ci-après.

a) Borne supérieure sur le nombre total de données en sortie : Le premier résultat établi concerne les unités de données en sortie de la file i, donné par le lemme 9.11. Nous notons que $H_5^N(n)$ (resp. $\widehat{H}_5^N(n)$) représente le nombre de données en sortie de la file N à l'instant n pour le réseau $H_1/D/S_1/B_1 \rightarrow /D/S_2/B_2 \ldots \rightarrow /D/S_N/B_N$ (resp. $H_1/D/S_1/0 \ldots \rightarrow /D/S_{N-1}/0 \rightarrow /D/S_N/B_N$).

Lemme 9.11 *Soient deux réseaux en tandem* $H_1/D/S_1/B_1 \rightarrow \ldots \rightarrow /D/S_N/B_N$ *et* $H_1/D/S_1/0 \ldots \rightarrow /D/S_{N-1}/0 \rightarrow /D/S_N/B_N$*, nous avons :*

$$\widehat{H}_5^N(n) \leq_{st} H_5^N(n), \quad \forall n, \quad \textit{et} \quad \sum_n \mathbb{E}[\widehat{H}_5^N(n)] \leq_{st} \sum_n \mathbb{E}[H_5^N(n)].$$

PREUVE. Conséquence directe du lemme 9.3. ■

b) Borne supérieure sur le nombre total de données perdues : Pour le nombre total de pertes dans le réseau, le lemme ci-après permet de définir la borne inférieure.

Lemme 9.12 *Soient deux réseaux en tandem A :* $H_1/D/S_1/B_1 \rightarrow \ldots \rightarrow /D/S_N/B_N$ *et B :* $H_1/D/S_1/0 \ldots \rightarrow /D/S_{N-1}/0 \rightarrow /D/S_N/B_N$*. Nous notons par* $\mathbf{SP}_A(t)$ *le nombre de données total perdu depuis l'instant* 0 *dans le premier réseau. Pour le réseau A (resp. réseau B),* $\mathbf{ps}_A^i(t)$ *(resp.* $\mathbf{ps}_B^i(t)$*) représente le nombre total de données perdues dans la file* i *depuis l'instant 0. Nous avons :*

$$\mathbf{SP}_A(t) = \sum_{j<N} \mathbf{sp}_A^j(t) + \mathbf{sp}_A^N(t) \geq \sum_{j<N} \mathbf{sp}_A^j(t) + \mathbf{sp}_B^N(t), \quad \textit{pour tout } t.$$

PREUVE. Même schéma de preuve que le lemme 9.6. ■

c) Borne stochastique supérieure sur les délais de bout en bout : La borne inférieure sur le temps de traversée d'une unité de donnée k, $k > 0$, de la file d'entrée à la file de sortie i est donnée par le corollaire 9.2. Nous notons par $T_i^B(k)$ (resp. $T_i^A(k)$) la date de sortie de la file i de l'unité de donnée k dans le modèle avec files à capacité nulles (resp. modèle fini).

Corollaire 9.2 *Soit* $i = \{1, \ldots, N\}$ *et soient deux réseaux de files d'attente en tandem* $H_1/D/S_1/B_1 \rightarrow \ldots \rightarrow /D/S_i/B_i$ *et* $H_1/D/S_1/0 \rightarrow \ldots \rightarrow H_1/D/S_{i-1}/0 \rightarrow /D/S_i/B_i$*, nous avons :*

$$T_i^B(k) \leq T_i^A(k), \quad \forall\, k \geq 1.$$

PREUVE. Même schéma de preuve que celle de la propriété 9.2. ■

La détermination des mesures de performance de la file d'attente i via cette approche se résume à l'analyse de la file de manière isolée. En effet, le processus d'entrée de la file i correspond au filtrage de la séquence d'entrée du réseau H_1 sur tous les services des files d'attente qui précèdent la file i. La nouvelle séquence d'entrée notée H_1^i est donnée par :

$$H_1^i = \ominus_{\ell<i}(H_1, S_\ell), \quad i = \{1, \ldots, N\};$$

où l'opérateur $\ominus$ permet de faire le filtrage de l'histogramme d'entrée H_1 sur le service S_ℓ ($\ell < i$). Cet opérateur est défini comme suit : $H = \ominus(H_1, S_i)$ avec

$$\begin{cases} H(x) &= 0, \ \forall x > S_i, \\ H(x) &= H_1(x), \ \forall x < S_i, \\ H(S_i) &= \sum H_1(x), \ \forall x \geq S_i. \end{cases} \tag{9.7}$$

9.3.5 Approche 4 : bornes supérieures sur les paramètres de sortie du réseau

Dans cette approche, nous construisons un modèle pour lequel des bornes supérieures sont dérivées sur les nombres de données en sortie et le nombre total de pertes dans le réseau en tandem. Ainsi, afin de déterminer des bornes supérieures sur les mesures de performance d'une la file i ($i > 1$) du réseau, nous proposons de fixer les capacités de service des files $j < i$ à l'infini (voir figure 9.13).

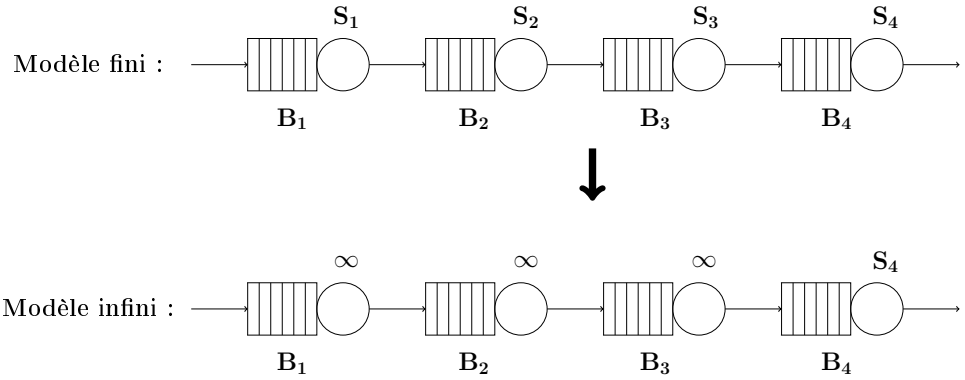


FIGURE 9.13 – Principe de l'approche 4. Exemple de réseau en tandem pour $i = 4$.

Les différents résultats établis sur cette approche sont démontrés ci-après. Le premier concerne le processus de départ de la file i.

a) Borne supérieure sur le nombre total de données en sortie : Soit $H_5^i(n)$ (resp. $\widehat{H}_5^i(n)$) le nombre de données en sortie de la file i à l'instant n pour le réseau $H_1/D/S_1/B_1 \rightarrow /D/S_2/B_2 \ldots \rightarrow /D/S_N/B_N$ (resp. $H_1/D/\infty/B_1 \rightarrow /D/\infty/B_2 \ldots \rightarrow /D/\infty/B_{i-1} \rightarrow /D/S_i/B_i$). Nous avons le résultat suivant :

Lemme 9.13 *Soient deux réseaux de files d'attente en tandem* $H_1/D/S_1/B_1 \rightarrow /D/S_2/B_2 \ldots \rightarrow /D/S_N/B_i$ *et* $H_1/D/\infty/B_1 \rightarrow /D/\infty/B_2 \ldots \rightarrow /D/\infty/B_{i-1} \rightarrow /D/S_i/B_i$,

nous avons :

$$H_5^i(n) \leq_{st} \widehat{H}_5^i(n), \ \ \forall\, n \ \textit{et} \ \ \sum_n \mathbb{E}[H_5^i(n)] \leq \sum_n \mathbb{E}[\widehat{H}_5^i(n)].$$

PREUVE. Par soucis de clarté, nous considérons le cas d'un réseau en tandem composé de deux files d'attente. La généralisation à i files d'attente est simple par la suite.
Les équations d'évolution du nombre de données en sortie des deux modèles sont données par :

$$\text{Modèle fini} \quad \begin{cases} H_5^1(n) = \min\big(S_1,\, H_3^1(n-1) + H_1(n)\big); & \text{(9.8a)} \\ H_3^2(n+1) = \min\big(B_2,\, (H_3^2(n) + H_1^2(n+1) - S_2)^+\big); & \text{(9.8b)} \\ H_5^2(n+1) = \min\big(S_2,\, H_3^2(n) + H_5^1(n)\big). & \text{(9.8c)} \end{cases}$$

$$\text{Modèle infini} \quad \begin{cases} \widehat{H}_5^1(n) = H_1(n); & \text{(9.9a)} \\ \widehat{H}_3^2(n+1) = \min\big(B_2,\, (\widehat{H}_3^2(n) + H_1(n) - S_2)^+\big); & \text{(9.9b)} \\ \widehat{H}_5^2(n+1) = \min\big(S_2,\, \widehat{H}_3^2(n) + H_1(n)\big). & \text{(9.9c)} \end{cases}$$

Nous avons les hypothèses de départ suivantes : $H_3^2(0) = 0$ et $\widehat{H}_3^2(1) = 0$.

- Pour $n = 2$, nous avons $H_5^2(2) = \min\,(S_2,\, H_5^1(1))$ et $\widehat{H}_5^2(2) = \min\,(S_2,\, H_1(1))$. Sachant que $H_5^1(1) \leq_{st} H_1(1)$, nous avons alors $H_5^2(2) \leq_{st} \widehat{H}_5^2(2)$.
- Par récurrence sur n, nous supposons que $H_3^2(n) \leq_{st} \widehat{H}_3^2(n)$ et nous voulons montrer que $H_3^2(n+1) \leq_{st} \widehat{H}_3^2(n+1)$ est vrai. Par déduction nous démontrons également que $H_5^2(n+1) \leq_{st} \widehat{H}_5^2(n+1)$.
  Sachant que $H_3^2(n) \leq_{st} \widehat{H}_3^2(n)$ (par hypothèse), $H_5^1(n) \leq_{st} H_1(n)$ et que la fonction $\min$ est une fonction croissante, nous avons :

  $$H_3^2(n+1) \leq_{st} \widehat{H}_3^2(n+1).$$

  De ce résultat, nous pouvons déduire que :

  $$H_5^2(n+1) \leq_{st} \widehat{H}_5^2(n+1).$$

Ainsi, $\forall\, n \geq 0$, nous avons $H_5^2(n) \leq_{st} \widehat{H}_5^2(n)$. De plus, $\sum_n \mathbb{E}[H_5^2(n)] \leq \sum_n \mathbb{E}[\widehat{H}_5^2(n)]$, car l'espérance est une fonction linéaire. ■

b) Borne supérieure sur le nombre total de données perdues : Le second résultat concerne le nombre total de pertes à la sortie de la file i depuis l'instant 0. Pour une raison de lisibilité et de simplification, la preuve développée ici considère le cas d'un réseau en tandem composé de deux files d'attente, cependant ce résultat est facilement généralisable à N files d'attente.

Nous considérons une date T quelconque et une séquence quelconque $H(0) \ldots H(T-1)$. Il n'est pas nécessaire que la séquence $H()$ soit stochastique. Pour la séquence H, nous notons pour chaque date $t < T$:
$\mathbf{X}^H(t)$: le nombre de données dans la file d'entrée,
$\mathbf{K_I}^H(t)$: la taille du groupe de données entre la file 1 et la file 2 dans modèle infini,

$\mathbf{K_F}^H(t)$: la taille du groupe de données entre la file 1 et la file 2 dans le modèlle fini,
$\mathbf{N_I}^H(t)$: la longueur de la file d'attente de sortie, dans le modèle fini,
$\mathbf{N_I}^H(t)$: la longueur de la file d'attente de sortie, dans le modèle infini,
$\mathbf{SP}_F^H(t)$: le nombre de pertes depuis l'instant 0 dans le modèle fini,
$\mathbf{SP}_I^H(t)$: le nombre de pertes depuis l'instant 0 dans le modèle infini,
$\mathbf{SS}_F^H(t)$: le nombre de données traités depuis l'instant 0 dans le modèle fini,
$\mathbf{SS}_I^H(t)$: le nombre de données traités depuis l'instant 0 dans le modèle infini.

Les modèles sont décrits par leurs équations d'évolution.
Modèle fini :

$$\begin{aligned}
\mathbf{X}^{H_1}(t+1) &= (\mathbf{X}^H(t) + H_1(t) - S_1)^+ \\
\mathbf{K_F}^{H_1}(t+1) &= \min(S_1, \mathbf{X}^{H_1}(t) + H_1(t)) \\
\mathbf{N_F}^{H_1}(t+1) &= \min\left(B_2, (\mathbf{N_F}^{H_1}(t) + \mathbf{K_F}^{H_1}(t) - S_2)^+\right) \\
\mathbf{SP}_F^{H_1}(t+1) &= \mathbf{SP}_F^{H_1}(t) + (\mathbf{N_F}^{H_1}(t) + \mathbf{K_F}^{H_1}(t) - S_2 - B_2)^+ \\
\mathbf{SS}_F^{H_1}(t+1) &= \mathbf{SS}_F^{H_1}(t) + \min\left(S_2, \mathbf{N_F}^{H_1}(t) + \mathbf{K_F}^{H_1}(t)\right)
\end{aligned}$$

Modèle infini :

$$\begin{aligned}
\mathbf{K_I}^{H_1}(t+1) &= H_1(t) \\
\mathbf{N_I}^{H_1}(t+1) &= \min\left(B_2, (\mathbf{N_I}^{H_1}(t) + \mathbf{K_I}^{H_1}(t) - S_2)^+\right) \\
\mathbf{SP}_I^{H_1}(t+1) &= \mathbf{SP}_I^{H_1}(t) + (\mathbf{N_I}^{H_1}(t) + \mathbf{K_I}^{H_1}(t) - S_2 - B_2)^+ \\
\mathbf{SS}_I^{H_1}(t+1) &= \mathbf{SS}_I^{H_1}(t) + \min\left(S_2, \mathbf{N_I}^{H_1}(t) + \mathbf{K_I}^{H_1}(t)\right)
\end{aligned}$$

Nous avons un premier résultat très simple dû au principe de conservation dans les files d'attente.

Lemme 9.14 (Loi de conservation) *Pour tout t, nous avons :*

$$\mathbf{X}^H(t)+\mathbf{K_F}^H(t)+\mathbf{N_F}^H(t)+\mathbf{SP}_F^H(t)+\mathbf{SS}_F^H(t) = \mathbf{K_I}^H(t)+\mathbf{N_I}^H(t)+\mathbf{SP}_I^H(t)+\mathbf{SS}_I^H(t) = \sum_{i=0}^{t-1} H(t).$$

PREUVE. Nous remarquons simplement que les entrées doivent être égales aux sorties dans les deux modèles et les entrées des deux modèles sont identiques par construction. ■

Nous énonçons dans ce qui suit un ensemble de résultats nécessaires pour notre preuve.

Lemme 9.15 *Comparaison sur la file 1. Pour tout t nous avons :*

$$\mathbf{X}^H(t) + \mathbf{K_F}^H(t) \geq \mathbf{K_I}^H(t).$$

PREUVE. Pour une unité de données il faut au moins une unité de temps pour sortir de la file 1 dans le modèle fini et il faut exactement une unité de temps dans le modèle infini. ■

Lemme 9.16 *Soit H une séquence d'entrée quelconque, et soit G définie par :*

$$\forall t \neq t_0, G(t) = H(t) \;\; et \;\; G(t_0) = H(t_0) + 1$$

alors $\mathbf{SP}_F^H(t) \leq \mathbf{SP}_F^G(t) \leq \mathbf{SP}_F^H(t) + 1$ *pour tout t.*

PREUVE. Nous allons examiner les différents intervalles de temps.

- $t < t_0$. Avant t_0 les séquences d'arrivée sont identiques et les états initiaux sont les mêmes. Donc, dans les deux systèmes sous le modèle fini, nous perdons les mêmes données. Et nous avons donc $\mathbf{SP}_F^H(t) = \mathbf{SP}_F^G(t)$ pour $t < t_0$.
- $t = t_0$. Nous faisons rentrer une unité de donnée supplémentaire. Sans perte de généralité, nous considérons que c'est la dernière du groupe d'arrivée et donc la dernière dans la file d'entrée à l'instant t_0. Nous supposons qu'elle sort de la file d'entrée (file 1) à l'instant $t_1 > t_0$. De t_0 à $t_1 - 1$, il n' y a pas de différence entre les groupes de données à l'entrée de la file de sortie (file 2) pour les deux séquences G et H :

  $$\mathbf{K_F}^G(t) = \mathbf{K_F}^H(t), \quad \forall\, t \in t_0, \ldots, t_1 - 1.$$

  Pour la même raison, les longueurs de la file de sortie sont égales.

  $$\mathbf{N_F}^G(t) = \mathbf{N_F}^H(t), \quad \forall\, t \in t_0, \ldots, t_1.$$

  Et nous avons le même résultat sur les pertes,

  $$\mathbf{SP}_F^H(t) = \mathbf{SP}_F^G(t), \quad \forall\, t \in t_0, \ldots, t_1.$$

- À partir de t_1, la séquence de sortie de la file d'entrée associée à H est une séquence de longueur l de valeurs égales à $S1$ suivi d'une valeur $f < S1$. la longueur l peut être égale à 0. Pour la séquence d'entrée G, on observe la même séquence de l valeurs égales à $S1$ mais, elle est suivie d'une valeur $f+1$. Comme $f < S1$, $f+1$ est une valeur possible pour le groupe de données en entrée de la file 2 et peut valoir $S1$.
  Puisque les files de sorties sont égales à l'instant t_1 pour les deux séquences et qu'il y a la même séquence d'arrivée dans la file 2 jusqu'à $t_1 + l$, on pert le même nombre d'unité de donnée jusqu'à cet instant.

  $$\mathbf{SP}_F^H(t) = \mathbf{SP}_F^G(t), \quad \forall\, t \in t_1 + 1, \ldots, t_1 + l.$$

- À l'instant $t_1 + l + 1$,nous avons donc une arrivée de plus qui se présente devant la file de sortie. Nous avons deux cas selon que l'arrivée est refusée ou acceptée.
  - Accès refusé : Nous perdons une unité de donnée de plus avec la séquence G qu'avec la séquence H mais, après cette perte, les deux tampons $\mathbf{N_F}^H$ et $\mathbf{N_F}^G$ sont égaux et les entrées suivantes seront égales. Donc, nous avons :

    $$\mathbf{SP}_F^G(t) = \mathbf{SP}_F^H(t) + 1, \quad \forall\, t \in t_1 + 1, \ldots, t_1 + l.$$

  - Accès accepté : Nous avons

    $$\mathbf{N_F}^G(t + l + 1) = \mathbf{N_F}^H(t + l + 1) + 1.$$

    et les séquences d'arrivées en file de sortie sont alors égales dans le futur. Il y a encore deux cas selon que la file se vide ou pas avant la première perte.

– S'il y a, avec la séquence G, une perte avant que la file de sortie se vide, les tampons ${\mathbf{N_F}}^G$ et ${\mathbf{N_F}}^H$ seront égaux dans la suite de la trajectoire. Si t_2 est la date de cette perte, nous avons alors :

$$\mathbf{SP}_F^G(t) = \mathbf{SP}_F^H(t), \;\; t_1 < t \leq t_2 - 1 \;\; et \;\; \mathbf{SP}_F^G(t) = \mathbf{SP}_F^H(t) + 1, \;\; \forall\, t \geq t_2.$$

– Sinon, la file de sortie se vide avec la séquence G avant la première perte. Donc, elle se vide également pour la séquence H. Soit t_3 cette date. Puisqu'il n' y a pas de perte entre $t_1 + l$ et t_3 par hypothèse, nous avons :

$$\mathbf{SP}_F^G(t) = \mathbf{SP}_F^H(t), \;\; t_1 + 1 + l < t \leq t_3.$$

Puisque les arrivées sont les mêmes depuis $t_0 + 1$, nous aurons pour tout $t \geq t_3$:

$${\mathbf{N_F}}^G(t) = {\mathbf{N_F}}^H(t), \;\; \text{et} \;\; \mathbf{SP}_F^G(t) = \mathbf{SP}_F^H(t).$$

Ce qui conclut la preuve.

■

Lemme 9.17 *Pour tout T, et pour toute séquence H entre 0 et $T-1$, s'il n'y a pas de perte entre 0 et T dans le modèle infini, alors il n'y a pas de perte dans le modèle fini. Formellement, si $\mathbf{SP}_I^H(T) = 0$, alors $\mathbf{SP}_F^H(T) = 0$.*

PREUVE. La preuve se fera par récurrence sur t et consistera à démontrer que pour toute date t strictement inférieure à la date de la première perte dans le modèle infini nous avons :

$$\mathbf{SP}_I^H(t) = \mathbf{SP}_F^H(t) = 0 \tag{9.10}$$

$$\mathbf{SS}_I^H(t) = \mathbf{SS}_F^H(t) \tag{9.11}$$

$$\mathbf{X}^H(t) + {\mathbf{K_F}}^H(t) + {\mathbf{N_F}}^H(t) = {\mathbf{K_I}}^H(t) + {\mathbf{N_I}}^H(t) \tag{9.12}$$

Notons que la relation 9.12 découle immédiatement des relations 9.10 et 9.11 et de la relation de conservation énoncée au lemme 9.14. De plus, si la relation 9.12 est vraie, alors compte tenu de l'équation sur la comparaison dans la file 1 établie au lemme 9.15, nous avons ${\mathbf{N_F}}^H(t) \leq {\mathbf{N_I}}^H(t)$ pour toute date t inférieure à la date de la première perte dans le modèle infini.

Les trois relations sont clairement vraies à la date 0. Supposons maintenant que les relations sont vraies à la date t. D'après la relation 9.12, et puisque $\mathbf{X}^H(t) \geq 0$, nous avons :

$${\mathbf{K_F}}^H(t) + {\mathbf{N_F}}^H(t) \leq {\mathbf{K_I}}^H(t) + {\mathbf{N_I}}^H(t). \tag{9.13}$$

La soustraction de S_2 aux deux membres de l'inégalité nous donne :

$${\mathbf{K_F}}^H(t) + {\mathbf{N_F}}^H(t) - S_2 \leq {\mathbf{K_I}}^H(t) + {\mathbf{N_I}}^H(t) - S_2. \tag{9.14}$$

Mais puisqu'il n'y a pas encore eu de perte dans le modèle infini, nous avons ${\mathbf{K_I}}^H(t) + {\mathbf{N_I}}^H(t) - S_2 \leq B_2$. Par transitivité, nous obtenons : ${\mathbf{K_F}}^H(t) + {\mathbf{N_F}}^H(t) \leq B_2$ et donc nous ne perdons pas d'unité de donnée à la date t dans le modèle fini. Comme $\mathbf{SP}_I^H(t+1)$ est nul par définition, la relation 9.10 est établie à la date $t + 1$.

Montrons maintenant que l'équation 9.11 est vraie à cette même date. Nous regardons la taille du groupe de données ${\mathbf{K_I}}^H(t)$ dont on rappelé qu'il est égal à $H(t-1)$.

- Si $H(t-1) \geq S_1$, ${\mathbf{K_F}}^H(t) = S_1$ car la file est work conserving. Et par construction on a ${\mathbf{K_I}}^H(t) = H(t-1)$. Donc, on sort S_2 unités de données de la file de sortie dans les deux modèles, puisque $S_1 > S_2$ et qu'il arrive $S1$ unités de données dans cette file.
  Nous avons par induction :
  $$\mathbf{SS}_I^H(t) = \mathbf{SS}_F^H(t),$$
  ce qui donne
  $$\mathbf{SS}_I^H(t+1) = S_2 + \mathbf{SS}_I^H(t) = S_2 + \mathbf{SS}_F^H(t) = \mathbf{SS}_F^H(t+1)$$
- Sinon $H(t-1) < S_1$ et nous avons encore deux cas :
  - Si ${\mathbf{K_F}}^H(t) < S_1$, alors $\mathbf{X}^H(t) = 0$ puisque la file d'entrée est work conserving. Donc,
    $${\mathbf{K_F}}^H(t) + {\mathbf{N_F}}^H(t) = {\mathbf{K_I}}^H(t) + {\mathbf{N_I}}^H(t) \tag{9.15}$$
    Et
    $$\min(S_2, {\mathbf{K_F}}^H(t) + {\mathbf{N_F}}^H(t)) = \min(S_2, {\mathbf{K_I}}^H(t) + {\mathbf{N_I}}^H(t)). \tag{9.16}$$
    et on a donc les même sorties dans les deux modèles à la date $t+1$. Et puisque les cumuls à la date t étaient égaux dans les deux modèles, ils sont également égaux à la date $t+1$.
  - Sinon ${\mathbf{K_F}}^H(t) = S_1$. Dans le modèle fini, la file de sortie pourra sortir S_2 unités de données, car il y a S_1 unités de données qui rentre et $S_1 > S_2$.
    Dans le modèle infini, la file de sortie voit entrer ${\mathbf{K_I}}^H(t) < S_1$ unité de données. Nous avons toujours la relation 9.12, et comme ${\mathbf{K_F}}^H(t) = S_1$, nous avons ${\mathbf{K_I}}^H(t) + {\mathbf{N_I}}^H(t) \geq S_1 > S_2$. Donc, dans la file de sortie du modèle infini, il y a assez de données pour sortir S_2 unités de données. Grâce au même argument que précédement, nous obtenons :
    $$\mathbf{SS}_I^H(t+1) = \mathbf{SS}_F^H(t+1)$$

Donc, les relations 9.10 et 9.11 sont vérifiées à la date $t+1$, ce qui implique que l'équation 9.12 est également prouvée. ■

Théorème 9.3 $\mathbf{SP}_F^H(t) \leq \mathbf{SP}_I^H(t)$*, pour tout* t*.*

PREUVE. Nous considérons une séquence $H(0), \ldots, H(t-1)$ quelconque. La preuve repose sur les 4 étapes suivantes :

- Étape 1 : On simule le modèle infini avec la séquence H. On obtient $\mathbf{SP}_I^H(t) = g$. Si g est nul alors le Lemme 9.17 montre que $\mathbf{SP}_F^H(t) = 0$ et on a donc terminé la preuve. On suppose donc maintenant que $g > 0$. On marque toutes les "unités de données" perdues dans cette simulation.
- Étape 2 : On enlève de la séquence d'arrivée H toutes les "unité de données" perdues durant l'étape 1. On obtient ainsi une nouvelle séquence d'arrivée, nommée G. Cette séquence est une vraie séquence d'arrivée. En effet, dans le modèle infini, quand on perd une "unité de données", on perd également toutes celles qui font partie du même groupe mais qui sont en fin de groupe. Le rejet des données par le modèle infini fait donc un écrémage des groupes d'arrivées. Cette séquence d'arrivée est valide pour les deux modèles, fini ou infini.

– Étape 3 : On simule les deux modèles avec la séquence G. Par construction, on a :

$$\mathbf{N_I}^G(t) = \mathbf{N_I}^H(t), \;\; \mathbf{SS}_I^G(t) = \mathbf{SS}_I^H(t), \;\; \mathbf{SP}_I^G(t) = 0.$$

Et le Lemme 9.17 implique alors que $\mathbf{SP}_F^G(t) = 0$.

– Étape 4 : On ajoute de nouveau les g unités de données perdues dans le système en les ajoutant à partir de la séquence G pour reconstruire la séquence H. A chaque modification de la séquence, on utilise le Lemme 9.16. On obtient finalement :

$$\mathbf{SP}_F^H(t) \leq \mathbf{SP}_F^G(t) + g \leq 0 + \mathbf{SP}_I^H(t) = \mathbf{SP}_I^H(t).$$

Et la preuve est terminée. ■

Nous avons ainsi prouvé que ce nouveau réseau construit via l'approche 4 définit clairement des bornes supérieures sur les mesures de performance du réseau en tandem initial.

Ce nouveau réseau peut être représenté par une seule et unique file d'attente ayant H_1 comme séquence d'entrée de la file. Ainsi, pour une file d'attente i du réseau, la détermination de bornes supérieures sur cette file se résume à son étude de façon isolée selon le procédé illustré dans la figure 9.14.

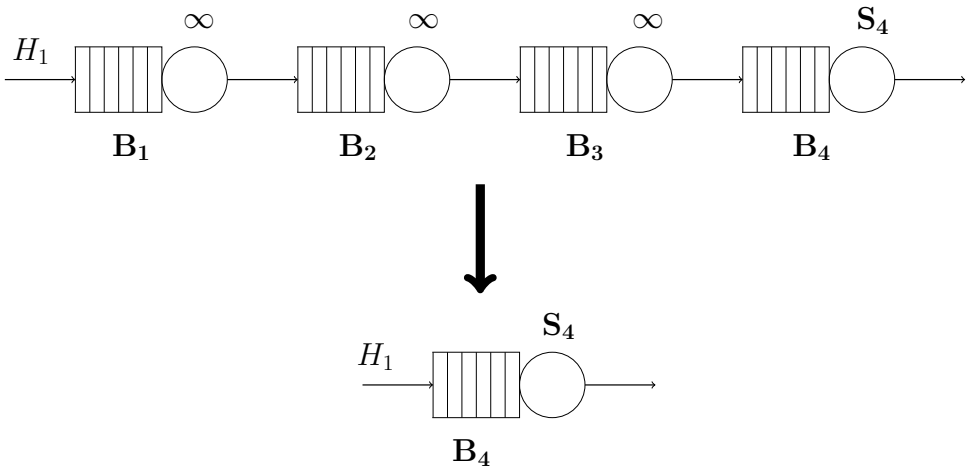


FIGURE 9.14 – Approche 4. Exemple de réseau en tandem pour $i = 4$.

9.3.6 Exemple numérique

Nous considérons le réseau en tandem présenté dans l'exemple 9.1, composé de trois files d'attente munie de tampons de longueur finie : $\mathrm{B}_1 = 2\,Mb$, $\mathrm{B}_2 = 1\,Mb$ et $\mathrm{B}_3 = 1\,Mb$ et des services déterministe donné par $110\,Mbps$ pour la file 1, $107.5\,Mbps$ pour la file 2 et $106.5\,Mbps$ pour la file 3 (voir 9.15).

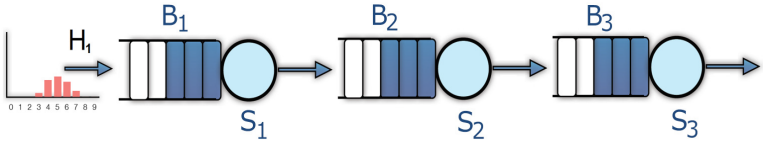


FIGURE 9.15 – Exemple de réseau en tandem.

Nous proposons de dériver via les approches 1, 2, 3 et 4 développées, des bornes supérieures et inférieures sur l'espérance du nombre de données en sortie (noté $\mathbb{E}[H_5]$), le nombre moyen des pertes au niveau de la file de sortie (noté **SP**) et le temps moyen de séjours, noté **Tm**. Nous spécifions que les temps de séjours sont calculés en utilisant la formule de Little. Nous donnons également le temps de calcul global (noté Temps Exc.) nécessaire à la détermination des différentes mesures de performance du réseau. Pour nos évaluations numériques, l'utilisation des différentes approches est effectuée pour les séquences d'entrée suivantes : histogramme d'entrée exact H_1 associé à la trace de trafic réelle MAWI et histogrammes bornants obtenus par notre approche de bornes stochastique. Les histogrammes bornants sont définis au travers nos algorithmes de bornes stochastiques supérieure ou inférieure pour deux tailles de réduction : bins égal à 100 et bins égal à 200. Les histogrammes bornes inférieures et supérieures sont notés respectivement par H_1^l et H_1^u. Nous présentons également ici les résultats obtenus par la méthode de simulation et l'approche d'analyse par décomposition, notée **Appr. Décomposition**. Le but étant de comparer les mesures fournies par les différentes approches.

Les mesures de performance des files 2 et 3 sont données respectivement dans les tableaux 9.5, 9.8 et 9.7, 9.8. Nous précisons que la résolution du modèle décrit par l'approche 1 est effectuée par la méthode de simulation, contrairement aux autres approches qui ont toutes étaient résolues de manière exacte. Le simulateur développé repose sur l'utilisation de la simulation équationnelle qui simule les équations de récurrence 7.1 et 7.2 sur le nombre de données dans le tampon et le nombre de données en sortie de la file d'attente. Les expériences ont été dupliquées 50 fois pour un intervalle de confiance à 95%. La largeur des différents intervalles de confiance est inférieure à 0.01. Nous notons que l'utilisation d'une simulation équationnelle est légèrement plus rapide que la simulation événementielle.

		$\mathbb{E}[H_5]$	**SP**	**Tm**	Temps Exc. (s)
Approche 1	H_1	4295530	794715	0.2058	2.8784
	H_1^u avec bins=100	4297610	883093	0.2144	1.435
	H_1^u avec bins=200	4296560	834361	0.2097	1.788
Approche 4	H_1	4147710	568647	/	184.356
	H_1^u avec bins=100	4159080	593788	/	1.345
	H_1^u avec bins=200	4154510	583321	/	2.658
Simulation		4065400	310190	0.1242	3.0446
Appr. Décomposition		4246920	663140	0.1151	6214

TABLE 9.5 – Bornes supérieures sur les paramètres de sorties de la file 2.

		$\mathbb{E}[H_5]$	**SP**	**Tm**	Temps Exc. (s)
Approche 3	H_1	4033420	1046	0.0695	76.396
	H_1^l avec bins=100	4030710	917	0.0683	0.331
	H_1^l avec bins=200	4033370	1042	0.0685	0.571
Simulation		4065400	310190	0.1242	3.0446
Appr. Décomposition		424692	663140	0.1151	6214

TABLE 9.6 – Borne inférieure sur les paramètres de sorties de la file 2.

		$\mathbb{E}[H_5]$	**PS**	**Tm**	Temps Exc. (s)
Approche 1	H_1	4259580	934694	0.2259	2.8595
	H_1^u avec bins=100	4259740	959962	0.2281	1.458
	H_1^u avec bins=200	4259660	946623	0.2269	1.942
Approche 2	H_1	4257480	/	3.669	1561
	H_1^u avec bins=100	4258030	/	3.6314	1.984
	H_1^u avec bins=200	4257730	/	3.6296	3.363
Approche 4	H_1	4125400	613649	/	151.023
	H_1^u avec bins=100	4136060	638011	/	1.499
	H_1^u avec bins=200	4131760	629640	/	2.633
Simulation		4048900	326570	0.1221	5.9763
Appr. Décomposition		4240800	835020	0.1191	21868

TABLE 9.7 – Bornes supérieures sur les paramètres de sorties de la file 3.

		$\mathbb{E}[H_5]$	**SP**	**Tm**	Temps Exc. (s)
Approche 3	H_1	3984820	0	0.09162	30.964
	H_1^l avec bins=100	3980710	0	0.09151	0.302
	H_1^l avec bins=200	3983080	0	0.09157	0.526
Simulation		4048900	326570	0.1221	5.9763
Appr. Décomposition		4240800	835020	0.1191	21868

TABLE 9.8 – Borne inférieure sur les paramètres de sorties de la file 3.

Nous constatons à travers les différents résultats obtenus que l'approche 4 représente l'approche qui détermine les meilleures bornes supérieures sur les mesures de performance des files d'attente du réseau. Que ce soit en terme de nombre de données en sortie ou en terme de nombre de pertes, les valeurs calculées s'avèrent être très pertinentes. L'approche 3 permet quant à elle de dériver des bornes inférieures sur les mesures de performances. Nous remarquons que cette dernière offre des bornes assez intéressantes sur l'espérance du nombre de données en sortie et sur les temps de séjours, cependant les bornes obtenues sur le nombre moyen de pertes dans la file de sortie sont loin d'être significatives. Nous notons qu'il serait tout de même intéressant de regarder si ces comportements sont également observés sur d'autres exemples de réseau en tandem. Cette étude sera investiguée dans des travaux futurs.

Concernant l'utilisation de nos bornes stochastiques sur l'histogramme d'entrée du réseau pour chaque approche étudiée, nous remarquons que les résultats calculés via notre méthode sont très proches des valeurs obtenues par l'histogramme d'entrée exact H_1 et cela en des temps de calcul très courts. De plus, ces temps sont de même ordre voir plus rapide que ceux de la méthode de simulation. Ainsi, nous pouvons conclure en mettant l'accent sur le compromis très intéressant entre la précision et le temps de calcul qu'offre notre méthode à l'analyse de performance des files d'attente. Nous précisons également que contrairement à la méthode de simulation, en combinant les approches proposées ici avec notre méthode de bornes stochastiques, nous garantissons l'obtention de bornes et non d'approximations sur les mesures de performances des files du réseau.

9.4 Conclusion

Nous avons montré dans ce chapitre l'impact que pouvait avoir l'utilisation de notre méthode de bornes sur l'analyse de réseaux de files d'attente. Ainsi, à travers l'étude de deux approches d'analyse nous avons montré les différents avantages que pouvait offrir notre méthode. Dans un premier temps, nous avons abordé l'approche d'analyse par décomposition. Reposant sur une analyse séquentielle du réseau (file par file), nous avons mis en évidence le fait que quitte à faire une approximation pour évaluer les performances d'un réseau feed-forward, notre méthode pouvait accélérer la vitesse de l'approche de décomposition sans modifier fondamentalement l'erreur faite par cette approche. Dans un second temps, nous nous sommes

intéressés à la détermination de bornes et non d'approximations sur les mesures de performance d'un réseau en tandem. Nous avons proposé à cet effet, quatre approches d'analyse permettant de dériver des bornes inférieures ou supérieures prouvées sur des indices de performance d'un réseau en tandem tels que : le nombre total de données en sortie du réseau, le nombre total de pertes depuis l'instant initial et les délais dans le réseau. Nous avons montré clairement à travers cette étude la pertinence des résultats obtenus via notre approche de bornes stochastiques.

Nous avons ainsi démontré dans ce chapitre l'attrait que pouvait représenter notre nouvelle approche de bornes stochastiques sur l'évaluation de performance des réseaux de files d'attente ayant des mesures générales de traces de trafic. La pertinence de notre travail consiste premièrement à obtenir des histogrammes bornant dont la taille est ajustable en fonction de la complexité et de la précision des résultats. Et deuxièmement, à garantir l'obtention de bornes sur les mesures de performance du réseau de files d'attente, grâce à la propriété de monotonie stochastique prouvée des histogrammes de bornes. Il reste cependant à étudier les réseaux qui ne sont pas feed-forward, car ils requièrent une étude mathématique plus poussée pour prouver la convergence de l'analyse numérique pour des files d'attente ayant des boucles. Une autre étude pourrait également être envisagée et consisterait à analyser des réseaux spécifiques comme le *Cloud* afin d'étudier l'impact de la variation et d'évolution des ressources d'approvisionnement sur la performance. Une extension de notre méthode pour la gestion des flux non stationnaires nous semble également très intéressante. Nous proposons donc d'investiguer ce point dans le chapitre suivant et d'étudier le cas d'une file d'attente simple.

Chapitre 10

Analyse d'une file d'attente simple avec processus d'arrivée non-stationnaire

Les études que nous avons menées jusqu'à présent reposaient essentiellement sur l'hypothèse de stationnarité du processus en entrée. Dans ce chapitre, nous remplaçons cette hypothèse avec la notion de non-stationnarité. Nous proposons de nous intéresser ici au processus d'arrivée *Switch Batch Bernoulli Process* noté SBBP [HTS91] pour lequel nous proposons d'étudier certaines mesures de performance pour une file d'attente simple alimentée par une trace de trafic réelle. Le processus SBBP nous permet de prendre en considération l'aspect sporadique du trafic en entrée ainsi que les caractéristiques du système étudié. Ainsi, pour une file d'attente SBBP/D/S/B, dont le service est déterministe (S unités de données sont servies durant un intervalle de temps) et la longueur de tampon est finie B, nous envisageons d'appliquer notre méthode de bornes stochastiques et de montrer la notion de monotonie des paramètres du système.

Pour ce faire, nous organisons ce chapitre comme suit : Dans la section 10.1, nous rappelons ce qu'est un processus SBBP et présentons la méthodologie suivie pour trouver un SBBP approprié pour modéliser une trace de trafic réelle. La section 10.2, résume l'analyse d'une file d'attente simple SBBP/D/S/B ainsi que la chaîne de Markov associée au modèle. Dans la section 10.3, nous abordons la notion de monotonie des paramètres de l'élément de réseau, nous montrons que si deux DTMCs sont comparables au sens de l'ordre stochastique fort, alors leurs mesures de performance sont également comparables au sens "st". Et pour finir, nous illustrons à travers un exemple numérique certains résultats pour lesquels il est aisé de constater l'apport qu'offre notre méthode à l'analyse d'une file d'attente isolée.

10.1 Processus SBBP

Un processus SBBP est caractérisé par un processus d'arrivée modulé par une chaîne de Markov à p états représentant les phases. Le processus d'arrivée est représenté par des arrivées qui dépendent de la phase. Une phase représente l'état courant de la chaîne de Markov. Pour représenter la sporadicité des arrivées dans le cas d'une trace de trafic réelle, nous proposons de distinguer trois phases, la phase 1 qui représente l'état où le trafic est faible, la phase 2, pour le

trafic moyen, et la phase 3, pour représenter un trafic intense. Dans chaque phase, le processus d'arrivée est stationnaire et est modélisé par un histogramme.

Nous proposons d'illustrer le processus d'arrivée SBBP, en présentant un exemple de trace de trafic réel pour laquelle nous définissons les phases d'arrivées des groupes de données. Nous considérons la trace de trafic MAWI [SC00] qui correspond à une mesure de trafic IP sur une ligne transpacifique avec des capacités de liens de 128 bytes/s, réalisée entre le jeudi 6 mars 2007 à 18 : 00 et le vendredi 7 mars 2007 à 04 : 24 : 27. Pour un échantillonnage avec une période de 40 ms, nous obtenons la trace illustrée dans la figure 10.1 avec un nombre d'états de 922873.

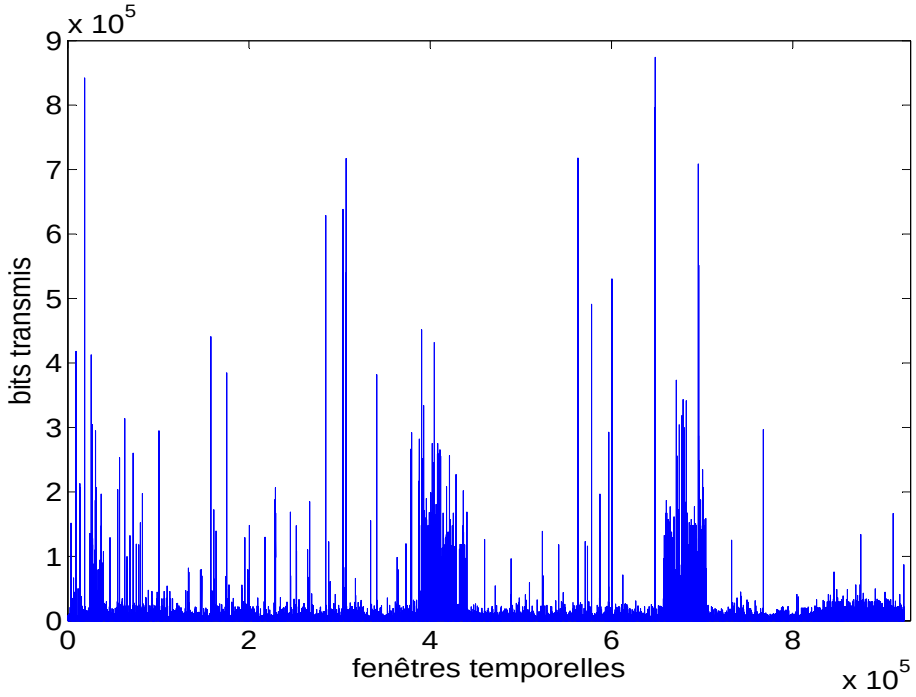
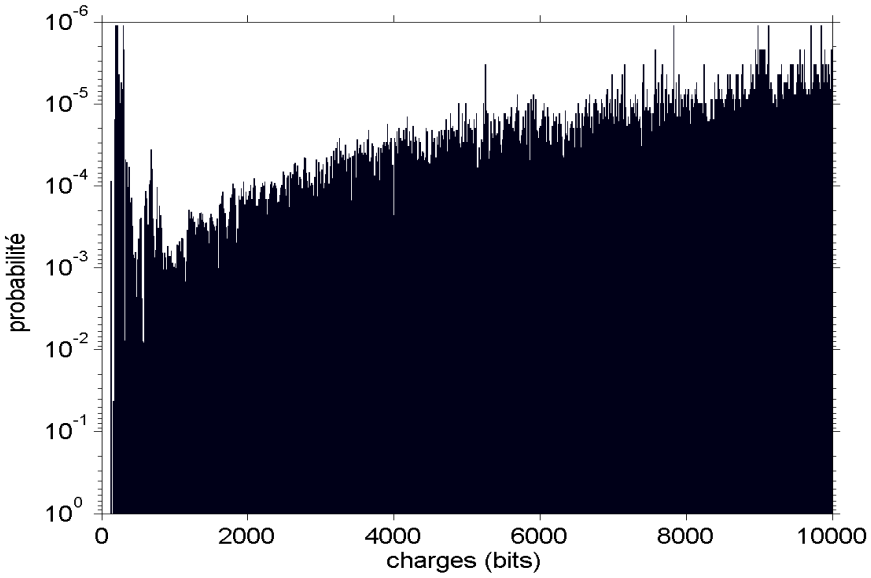


FIGURE 10.1 – Trace de trafic MAWI.

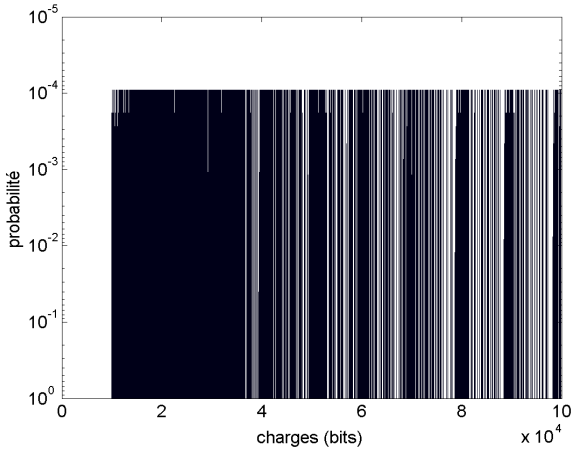
Nous avons supposé ici que pour chaque slot, la trace est caractérisée par un volume (bits/s) et une phase i, $i = \{1, 2, 3\}$. La détermination de la matrice de transition associée à la chaîne de Markov représentant les phases est définie par :

$$\mathbf{M}[i, j] = \frac{\text{nombre de transition entre la phase } i \text{ et la phase } j}{\text{nombre d'instant (slot) en phase } i}.$$

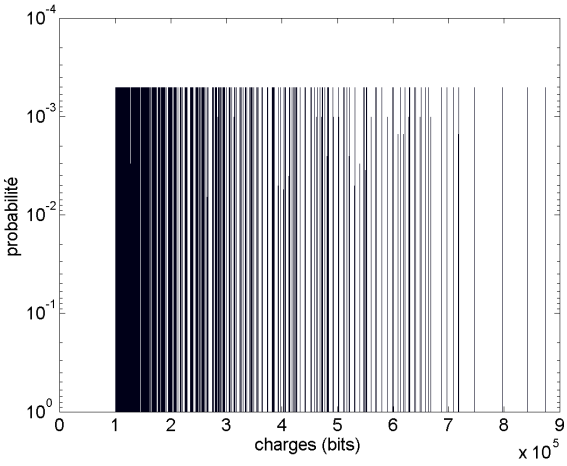
En considérant un seuil minimal (*Smin*=10 Kbps) et un seuil maximal (*Smax*=100 Kbps), nous avons caractérisé la trace de trafic MAWI selon trois phases : trafic faible, trafic moyen et trafic intense. Nous effectuons une caractérisation sur la valeur instantanée du trafic. Les histogrammes correspondants aux volumes des différentes phases sont représentés dans les figures suivantes.



(a) Phase 1



(b) Phase 2



(c) Phase 3

FIGURE 10.2 – Phases de la trace de trafic MAWI.

La matrice de transition M résultante est :

$$\mathbf{Q} = \begin{pmatrix} 0.9982 & 0.0018 & 0.0000 \\ 0.5563 & 0.4163 & 0.0274 \\ 0.2706 & 0.2615 & 0.4679 \end{pmatrix}$$

La taille des différents histogrammes est de : 1228 états pour la phase 1, 2568 états pour la phase 2 et 783 états pour la phase 3. Ces histogrammes sont caractérisés par les descriptions statistiques suivantes : une espérance de 433.56 bits pour la phase 1, 28953 bits pour la phase 2 et $2.1515\,10^5$ bits pour la phase 3, un écart-type de $1.0503\,10^3$ bits pour la phase 1, $2.13\,10^4$ bits pour la phase 2 et $1.2844\,10^5$ bits pour la phase 3. Le coefficient de variation

(scv) correspondant au rapport entre l'écart-type et l'espérance, donne les mesures relatives de dispersion suivantes :5.8684 pour la phase 1, 0.5413 pour la phase 2 et 0.3564 pour la phase 3.

Nous précisons que cette décomposition et ces valeurs sont arbitraires. Elles ont pour but de montrer comment fonctionne notre approche. Il n'est pas dans notre intention d'étudier la caractérisation statistique d'une SBBP à partir d'une trace.

10.2 Analyse d'une file d'attente SBBP/D/S/B

On considère un système de file d'attente simple muni d'un processus d'arrivée de type SBBP, un service déterministe de capacité S et un tampon de longueur finie, égal à B. Les arrivées sont supposées se produire au début de chaque slot (intervalle de temps) et les sorties se font en fin de slot (mêmes hypothèses que les chapitres précédents).

Le système SBBP/D/S/B est une chaîne de Markov à temps discret, notée $\{QP(k), k \geq 0\}$. La DTMC $QP(k)$ est décrite par le couple $(Q(k), \phi(k))$ où la première composante $Q(k)$ représente le nombre d'entités de données dans le tampon et la deuxième composante $\phi(k)$ représente la phase des arrivées à l'instant k (voir figure 10.3).

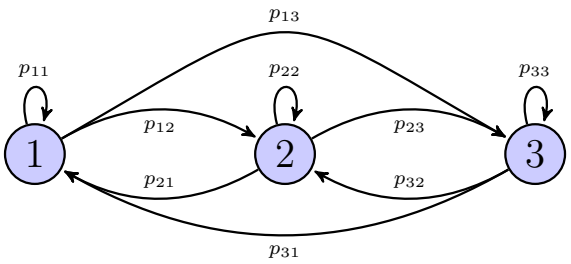


FIGURE 10.3 – Exemple de graphe de transition d'une chaîne de Markov avec trois phases.

Le processus $Q(k)$ prend ses valeurs dans l'ensemble $\mathcal{N} = \{0, \ldots, B\}$. L'évolution de $Q(k)$ est identique au cas des arrivées stationnaires où la chaîne de Markov est unidimensionnelle. Cependant, au lieu d'avoir un processus d'arrivée stationnaire A, nous considérons le processus d'arrivée pendant la phase $\phi(k)$, noté A^i, si $\phi(k) = i$.

L'évolution de la seconde composante $\phi(k)$ est quant à elle régie par une DTMC qui est indépendante de $Q(k)$. Soit p le nombre de phases du processus d'arrivée, l'espace d'état de la DTMC $\phi(k)$ est donné par $\mathcal{P} = \{1, \ldots, p\}$. Et le processus d'arrivée stationnaire à la phase $i \in \mathcal{P}$ est noté par A^i.

L'espace d'états de la DTMC $\{QP(k),\ k \geq 0\}$ est donné par l'espace produit $\mathcal{S} = \mathcal{N} \times \mathcal{P}$. Nous définissons l'ordre partiel sur $\mathcal{S}$ noté par $\preceq$ comme suit : soient $x = (x_Q, x_\phi)$ et $y = (y_Q, y_\phi) \in \mathcal{S}$,

$$x \preceq y \ \text{ ssi } \ x_Q \leq y_Q \ \text{ et } \ x_\phi = y_\phi.$$

De plus, nous définissons par $g : \mathcal{S} \to \mathcal{N}$ la projection du nombre tel que : $g(x_Q, x_\phi) = x_Q$. Ainsi,

$$\text{si } x \preceq y, \ \text{ alors } \ g(x) \leq g(y).$$

10.3 Bornes sur les paramètres de la file d'attente

Nous considérons une autre DTMC notée par $\tilde{QP}(k)$ où les processus d'arrivée de chaque phase i représentent des bornes supérieures notées $\tilde{A}^i$ du trafic réel. Formellement, nous avons

$$\forall i \in \mathcal{P}, \quad A^i \leq_{st} \tilde{A}^i \tag{10.1}$$

Nous remarquons que la deuxième composante des deux modèles sont régies par la même DTMC indépendamment de la première composante. Ainsi, si dans les deux modèles nous commençons par le même vecteur d'états initial, l'évolution de la deuxième composante sera la même à chaque instant k. Par conséquent, nous noterons par $\phi(k)$ la phase des arrivées à l'instant k pour les deux systèmes. Les équations d'évolutions de la deuxième composante pour $g(QP(k+1))$ et $g(\tilde{QP}(k+1))$ sont décrites comme suit :

$$Q(k+1) = \min\left(B, (Q(k) + A^{\phi(k)} - S)^+\right). \tag{10.2}$$

$$\tilde{Q}(k+1) = \min\left(B, (\tilde{Q}(k) + \tilde{A}^{\phi(k)} - S)^+\right). \tag{10.3}$$

Nous montrons ci-après la comparaison stochastique existante entre la première composante associée à la DTMC original $\{QP(k), k \geq 0\}$ et celle de la DTMC ayant un processus d'arrivée bornant et qui satisfait l'équation 10.1.

Théorème 10.1 *Si* $\forall i \in \mathcal{P}, \quad A^i \leq_{st} \tilde{A}^i$, $Q(0) \leq \tilde{Q}(0)$ *et* $\phi(0) = \tilde{\phi}(0)$, *alors*

$$\forall k, \quad g(QP(k)) \leq_{st} g(\tilde{QP}(k)).$$

PREUVE. Nous utilisons pour cette preuve le théorème 2.6 qui fournit des conditions suffisantes pour établir une relation d'ordre $\leq_{st}$ entre deux DTMCs.

- Nous avons considéré les mêmes conditions initiales : $QP(0) \preceq \tilde{QP}(0)$, ainsi la première condition du théorème 2.6 est vérifiée.
- La deuxième condition consiste à prouver la monotonie de l'une des DTMCs. Nous montrons ici la monotonie de la DTMC originale $g(QP(k))$.
  Nous avons $\forall i, j \in \mathcal{S}$ tels que $i \preceq j$ et $\forall\, l \in \mathcal{N}$:

$$\text{Prob}(g(QP(k+1)) > l \mid QP(k) = i) \leq \text{Prob}(g(QP(k+1)) > l \mid (QP(k) = j)) \tag{10.4}$$

  Comme $QP(k)$ est homogène dans le temps, ces probabilités conditionnelles ne dépendent pas des instants k et sont donc données par les probabilités de transition.
  Il résulte de la relation d'ordre sur l'espace d'états $\mathcal{S}$ que, si $i \preceq j$ alors $i_Q \leq j_Q$, et $i_\phi = j_\phi$. Ainsi, les probabilités d'arrivée dans l'état i et j sont les mêmes. De plus, il découle de l'équation d'évolution 10.2 que l'évolution du nombre d'entités dans le tampon quand on démarre de l'état i est inférieure que pour le cas où l'état de départ est j. Par conséquent, les inégalités 10.2 sont satisfaites et $g(QP(k))$ est monotone au sens de l'ordre stochastique fort $\leq_{st}$. Nous pouvons facilement montrer de manière similaire que $g(\tilde{QP}(k))$ est également monotone.

- La troisième condition revient à prouver que les deux DTMCs sont comparables. Nous avons pour cela, $\forall i \in \mathcal{S}$ et $\forall\, l \in \mathcal{N}$:

$$\text{Prob}(g(QP(k+1)) > l \mid QP(k) = i) \leq \text{Prob}(g(\tilde{QP}(k+1)) > l \mid \tilde{QP}(k) = i) \quad (10.5)$$

  Dans ce cas, nous considérons l'évolution des DTMCs $QP(k+1)$ et $\tilde{QP}(k+1)$ lorsque l'état à l'instant k est le même ($QP(k) = \tilde{QP}(k)$). Les phases d'arrivée sont les mêmes dans les deux systèmes, cependant les processus d'arrivée sont différents. La différence entre ces deux systèmes est le processus en entrée de chaque phase d'arrivée, ils sont comparable au sens de l'ordre $\leq_{st}$ (voir l'équation 10.1). Une démarche similaire à la preuve sur la monotonie, nous considérons les équations d'évolution de ces systèmes (équations 10.2 et 10.3). Il résulte de l'équation 10.1 que si nous avons une probabilité d'arrivée non-nulle du processus d'arrivée A dans l'état m alors, cette probabilité apparaît dans la pmf de $\tilde{A}$ dans un état n tel que $n \geq m$. Ainsi, les inégalités de probabilité conditionnelle de l'équation 10.5 sont vérifiées.

Par conséquent, nous avons la comparaison stochastique des DTMCs $\{QP(k),\, k \geq 0\}$ et $\{\tilde{QP}(k),\, k \geq 0\}$. ■

Suite au résultat démontré sur la comparaison des chaînes et au fait que g soit une fonction croissante, nous pouvons alors énoncer le théorème suivant :

Théorème 10.2 (Comparison sur les mesures) *soit* $\mathbf{\Pi}$ *(resp.* $\tilde{\mathbf{\Pi}}$*) le vecteur de distribution stationnaire de* $QP(k)$ *(resp.* $\tilde{QP}(k)$*), et* $\mathbb{E}[Q]$ *(resp.* $\mathbb{E}[\tilde{Q}]$*) l'espérance du nombre d'entités de données dans le tampon calculées à partir de* $\mathbf{\Pi}$ *(resp.* $\tilde{\mathbf{\Pi}}$*). Nous avons alors*

$$\mathbb{E}[Q] = \sum_{a \in \mathcal{S}} g(a)\mathbf{\Pi}[a] \leq \sum_{a \in \mathcal{S}} g(a)\tilde{\mathbf{\Pi}}[a] = \mathbb{E}[\tilde{Q}].$$

De même pour les probabilités de blocage :

$$\sum_{a \in \mathcal{S} | g(a) = B} \mathbf{\Pi}[a] \leq \sum_{a \in \mathcal{S} | g(a) = B} \tilde{\mathbf{\Pi}}[a].$$

10.4 Exemple numérique

Pour un modèle de file d'attente simple SBBP/D/S/B, nous proposons dans cette évaluation numérique de comparer les deux approches suivantes : résolution exacte du modèle alimenté par les processus d'arrivée originaux, résolution exacte du modèle alimenté par les processus d'arrivée bornants pour chaque phase. En considérant certaines mesures de performance, nous allons montrer à travers cet exemple l'impact et l'utilité qu'offre notre méthode de bornes à l'analyse de performance d'une file d'attente simple dont le processus d'arrivée est non stationnaire. Nous présentons également ici les résultats pour une file A/D/S/B alimentée par un trafic d'entrée stationnaire. Le but de cette démarche consiste à observer les modifications qu'entraîne l'utilisation du processus SBBP sur l'analyse de performance d'une file d'attente.

Nous considérons une file d'attente alimentée par la trace de trafic réelle MAWI présentée dans la section 10.1 (figure 10.1) munie d'un service déterministe de capacité $S = 600$ Kbps.

Afin de distinguer les deux modèles A/D/S/B et SBBP/D/S/B, nous définissons respectivement le trafic stationnaire et le trafic SBBP associé à la trace. Le processus d'arrivée SBBP considéré est présenté dans la section 10.1 (voir figure 10.2). Nous proposons de faire varier la longueur du tampon entre 100 Kb et 3 Mb et déterminer les mesures de performances associées aux deux modèles (espérance de la longueur du tampon et probabilité de blocage). Les tailles de réduction d'histogrammes considérées pour l'approche de bornes sont les suivantes : une réduction sur 20 états et une réduction sur 100 états pour les histogrammes de chaque phase. Les résultats obtenus sur l'espérance de la longueur du tampon, la probabilité de blocage ainsi que les temps de calcul (en secondes) sont présentés dans les figures 10.1, 10.2 et 10.3. Nous précisons que pour nos calculs nous utilisons comme unité de données D = 1 Kb.

	Arr. Stationnaire	Arrivées SBBP				
	Exact	**Exact**	20		100	
			$L.b$	$U.b$	$L.b$	$U.b$
10^5	0.001701	0.002233	0.002212	0.002284	0.002231	0.002254
$2\,10^5$	6.8862e-4	0.001586	0.001552	0.001645	0.001585	0.001607
$5\,10^5$	8.8405e-5	7.1293e-4	6.8067e-4	7.6195e-4	7.1140e-4	7.2380e-4
10^6	1.3199e-7	1.7397e-4	1.5378e-4	1.9987e-4	1.7311e-4	1.7868e-4
$2\,10^6$	8.1135e-13	1.0925e-5	8.1727e-6	1.3865e-5	1.0470e-5	1.1086e-5
$3\,10^6$	4.4635e-18	6.4397e-7	4.35007e-7	9.6111e-7	6.3414-7	6.8729e-7

TABLE 10.1 – Probabilités de blocage.

	Arr. Stationnaire	Arrivées SBBP				
	Exact	**Exact**	20		100	
			$L.b$	$U.b$	$L.b$	$U.b$
10^5	726.993	648.457	638.001	682.785	648.385	666.66
$2\,10^5$	1392.65	1329.01	1310.74	1388.52	1328.4	1360.83
$5\,10^5$	2703.81	3689.92	3604.04	3890.79	3685.09	3777.34
10^6	2892.89	6371.5	6074.87	6879.49	6356.7	6543.62
$2\,10^6$	2893.47	8063.9	7339.35	8678.78	7836.01	8099.39
$3\,10^6$	2894.41	8019.54	7457.37	8896.6	7992.75	8203.36

TABLE 10.2 – Espérance de la longueur du tampon.

	Arr. Stationnaire	Arrivées SBBP				
	Exact	**Exact**	20		100	
			$L.b$	$U.b$	$L.b$	$U.b$
10^5	5.7369	8.9343	1.0528	0.1605	2.2393	0.2176
$2\,10^5$	15.7843	27.3395	1.3981	0.2096	2.1273	0.2590
$5\,10^5$	72.8370	102.8670	1.7235	0.3854	2.2599	0.6286
10^6	232.696	487.3540	1.9172	0.8645	2.5104	0.8089
$2\,10^6$	1163.48	1759.25	2.5062	1.6919	2.8416	1.7036
$3\,10^6$	1859.63	3380.59	2.8293	2.1639	3.6933	2.5346

TABLE 10.3 – Temps de calcul (s).

Pour le modèle SBBP/D/S/B, en observant les résultats obtenus sur les deux mesures de performance, nous constatons que notre méthode de bornes fournit un encadrement très pertinent des résultats et cela en des temps très courts. De plus, quand nous augmentons la taille de la réduction, nous remarquons que l'encadrement des mesures se rapproche fortement des résultats exacts. Pour ce qui est des temps de calcul, nous pouvons dire que ces derniers sont fortement intéressants en vue de la précision des bornes obtenues. Ces observations nous permettent donc de conclure que notre méthode de bornes fournit toujours des résultats très pertinents et garde tout son attrait quant à son application dans le domaine des files d'attente.

Concernant la différence entre le trafic d'entrée stationnaire et le trafic SBBP, nous remarquons que les probabilités de blocage sont beaucoup plus fortes pour le trafic SBBP. Les espérances de la longueur de tampon le sont également sauf pour les petites valeurs de tampon. Ce phénomène peut se traduire par le fait qu'il existe une dépendance de la variance du processus d'arrivée. Nous notons que la moyenne du flux en entrée pour les deux trafics (stationnaire et SBBP) est identique, cependant nous avons plus de variance dans le modèle SBBP ce qui se retranscrit dans ses mesures de performance.

10.5 Conclusion

L'utilisation de notre méthode de bornes stochastique permet également de garantir des bornes sur les paramètres d'une file d'attente simple avec processus d'arrivée non stationnaire. Les résultats prouvés et observés dans ce chapitre montrent que nous pouvons remplacer les distributions d'arrivées de chaque phase par une distribution bornante (de taille plus petite) inférieure et/ou supérieure et assurer l'obtention de bornes pour les mesures de performance du système et cela en des temps relativement courts.

Une extension possible pour notre approche, mais qui cette fois-ci peut s'avérer beaucoup plus difficile serait de considérer des réseaux avec de multiples classes de client. Cela permettrait de modéliser des mécanismes proposés pour faire de la différentiation de service :

les disciplines équitables pour le service dans les files d'attente, les accès privilégiés au tampon pour minimiser les pertes, les disciplines de destructions en cas de débordement du tampon.

Chapitre 11

Conclusion

Cette thèse a permis d'apporter des solutions de modélisation de systèmes incertains et de résolution numérique reposant sur les techniques de bornes. Nous avons en effet abordé le problème d'imprécision souvent rencontré dans l'analyse des systèmes laissant ainsi entrevoir le développement de nouvelles approches et de nouvelles méthodes et algorithmes de résolution. Nous synthétisons dans ce chapitre les différents résultats obtenus au cours de nos travaux ainsi que leurs apports. Des perspectives et exemples d'études futures sont également proposés.

Les deux aspects d'incertitude étudiés dans cette thèse ont été distingués en deux parties. La première est consacrée à l'étude de chaînes de Markov imprécises et vise à apporter des réponses quant aux performances des systèmes. Dans cette partie, nous avons commencé dans le chapitre 3 par formaliser et présenter les résultats existants pour l'analyse de chaînes de Markov partiellement spécifiées. Parmi les résultats présentés, nous avons mis l'accent sur le fait que ces méthodes n'ont pas le même critère d'optimalité et ne renvoient pas les mêmes résultats. Nous avons pour ce faire, effectué une analyse comparative de ces algorithmes en terme de vitesse et de précision, étude qui, à notre connaissance, n'existe pas. Considérant une récompense positive croissante, nous avons énoncé quelques lignes directrices qui peuvent, en fonction de l'usage que nous voulons avoir des bornes, aider au choix des méthodes et algorithmes appropriés. Nous avons ensuite présenté dans le chapitre 4 de nouveaux algorithmes itératifs pour calculer des bornes supérieures et inférieures par élément de la distribution stationnaire d'une chaîne de Markov partiellement spécifiée. Les algorithmes construits ont la caractéristique d'être plus rapides que les approches existantes (approche de Muntz, approche de Courtois et Semal ou approche de Buchholz). Fondés sur la théorie polyédrale développée par Courtois et Semal et sur le nouvel algorithme itératif de Bušić et Fourneau, nos algorithmes donnent des bornes de la distribution stationnaire à chaque étape de calcul. Une extension de l'applicabilité de nos algorithmes pour l'analyse de chaînes de Markov absorbantes et partiellement spécifiées a été effectuée. La construction de bornes supérieures et inférieures sur les temps moyens d'absorption (MTTF), notion très utile dans le contexte d'un problème de fiabilité, a ainsi été développée.

Nous avons de ce fait défini un ensemble d'algorithmes qui semble suffisant pour aborder la résolution numérique de modèles reposant sur des chaînes de Markov à temps discret

imprécises. Comme perspectives, il serait intéressant de construire un "model checker" pour des problèmes modélisés par des chaînes de Markov imprécises et qui serait fondé sur les résultats et les algorithmes présentés ici.

La seconde partie s'intéresse quant à elle au développement d'une nouvelle méthodologie fondée respectivement sur la notion d'histogramme et de comparaison stochastique. Considérant des traces de trafic réelles, le problème fréquemment rencontré lors de l'étude de réseaux transportant ce genre de trafic, est leurs volumes. Souvent de l'ordre de plusieurs milliers d'états, l'analyse exacte de ces systèmes est généralement très difficile, voire impossible à envisager. Nous proposons donc dans cette thèse de réduire la taille de ces traces et de définir des encadrements sur les mesures de performance. Ainsi, contrairement aux approches déjà présentées dans la littérature, qui face à ce problème présentent leurs limites ou ne reflètent pas correctement le comportement du réseau, nous avons proposé ici une nouvelle méthode d'analyse permettant de définir des encadrements plutôt que des approximations sur les mesures de performances des systèmes.

Nous nous sommes intéressés pour notre étude à la représentation par histogrammes des traces de trafic réelles. Utilisant cette caractérisation du trafic nous avons proposé de réduire la taille de l'histogramme associé à la trace initiale en utilisant la comparaison stochastique. Nous avons ainsi développé dans le chapitre 6, différents algorithmes répondant à cet objectif. Pour une fonction de récompense positive croissante, nous avons construit deux types d'algorithmes : algorithmes gloutons et algorithmes optimaux qui reposent essentiellement sur les propriétés fortes de la comparaison stochastique. Nous avons illustré l'intérêt et l'impact de cette démarche en considérant deux exemples portant respectivement sur un problème de recherche opérationnelle et un problème de files d'attente. Nous avons montré via ces deux exemples que les algorithmes bornants développés diminuent de manière radicale le temps de calcul des mesures de performance, et offre un encadrement très pertinent sur les résultats exacts.

Dans le chapitre 7, nous avons abordé l'étude de files d'attente. Nous avons considéré le cas d'une file d'attente simple avec service déterministe et arrivées iid avec comme hypothèse, la stationnarité du trafic en entrée. Nous avons montré à travers différentes expériences numériques que notre méthode de bornes stochastiques offre un compromis très intéressant entre la précision et la vitesse de calcul. En effet, selon le choix de la réduction apporté à la taille des distributions, nous pouvons déterminer des bornes pertinentes et un encadrement très fin du résultat exact en des temps relativement courts. Nous avons montré également que l'élément du réseau (file d'attente) est stochastiquement monotone ce qui nous permet d'assurer que les distributions bornantes calculées représentent bien des bornes de la distribution exacte. Ainsi, en appliquant nos algorithmes de bornes sur l'histogramme représentant la trace de trafic en entrée, nous avons défini des histogrammes bornants sur les différentes mesures de performance : les probabilités de blocage, l'espérance de la longueur du tampon, etc.. Nous avons ensuite effectué dans le chapitre 8 une généralisation sur le service et étudié la propriété de monotonie des éléments de routages, ainsi que certains mécanismes de gestion active de files d'attente. Nous avons montré les propriétés de monotonie des différents éléments de réseau : file d'attente simple, élément de division et élément de fusion. La propriété de monotonie a également été prouvée pour le mécanisme de gestion active de files d'attente (Tail Drop et IRED). Nous avons ainsi présenté et prouvé la pertinence de notre approche pour fournir

une solution intéressante au problème de dimensionnement. Notre méthode de borne offre un compromis entre la précision des résultats et la complexité de calcul.

Le fait d'avoir montré que les éléments de réseaux FIFO sont stochastiquement monotones par rapport aux histogrammes nous a permis par la suite d'aborder l'analyse de réseaux. Dans le chapitre 9, nous avons en effet proposé d'utiliser notre méthode de bornes stochastiques pour calculer les performances des réseaux. Nous avons proposé d'appliquer notre méthode sur deux approches de résolution de réseaux, la première repose sur l'approche d'approximation par décomposition et permet de définir des encadrements sur les mesures de performances des réseaux feed-forward telles que les pertes, l'espérance du flux en sortie, etc.. Les résultats obtenus pour ce cas confirment bien le fait que notre méthode fournit des encadrements pertinents sur les résultats d'approximation par décomposition et ceci en des temps relativement courts. En effet, en optant pour une résolution par l'approche par décomposition, nous nous confrontons à une analyse lente due au fait que les histogrammes considérés soient très grands. Ainsi, l'utilisation de notre approche dans ce cas va nous permettre de gagner non pas en précision, mais en vitesse de résolution par rapport à l'approximation par décomposition, car nous construisons des bornes rapides sur les résultats approximatifs. Il reste cependant à étudier les réseaux qui ne sont pas feed-forward. Ces derniers requièrent une étude mathématique approfondie permettant de prouver la convergence de l'analyse numérique pour des files d'attente admettant des boucles. La deuxième approche abordée consiste quant à elle à déterminer des bornes sur les paramètres de sortie du réseau et non pas des encadrements sur l'approximation. Cette approche prouvée pour les réseaux en tandem permet de calculer des bornes sur les mesures de performances comme les délais, les pertes ou les distributions de départ. Cette approche pourra également être étendue aux réseaux en arbres, car dans le cas des réseaux ayant une topologie en arbre, nous remarquons que leur analyse peut s'effectuer en décomposant le réseau initial en sous-réseau en tandem. Ainsi, l'analyse isolée de chaque sous-réseau par cette approche nous permet d'évaluer les mesures de performance du réseau en arbre tout en gardant les caractéristiques et propriétés du réseau initial (voir figure 11.1).

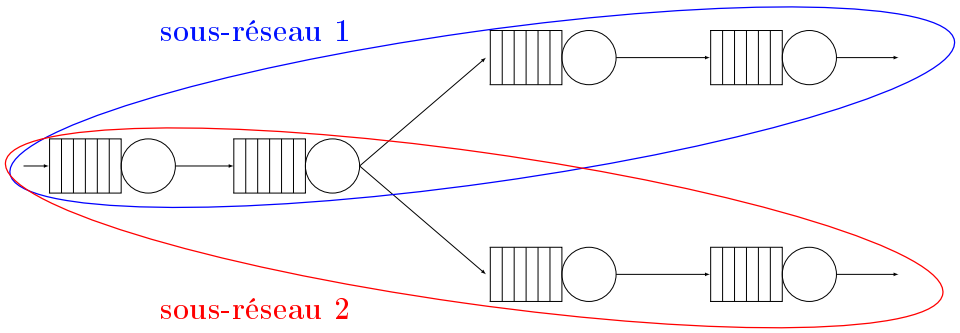


FIGURE 11.1 – Exemple de réseau en arbre.

L'étude de réseaux feed-forward ou de topologie générale via cette approche reste tout de même très compliquée et nécessite des résultats et propriétés très fortes et une démarche d'analyse poussée. Néanmoins, nous avons montré pour les deux approches d'analyse de réseaux, que pour un histogramme d'entrée bornant, nous pouvons garantir l'obtention d'encadrements très pertinents sur les mesures de performance du réseau. Ce résultat, dû

à la monotonie stochastique prouvée pour les éléments du réseau, représente l'une de nos contributions principales avec la garantie de la qualité de service.

L'analyse des réseaux spécifiques tels que le *Cloud* pourrait également s'avérer très intéressante. En effet, nous notons que les environnements de *Cloud* sont difficiles à analyser à cause de leurs dynamicités dues essentiellement à la variabilité des demandes et aussi des ressources utilisées (leur type et leur nombre). Ainsi, l'utilisation de notre méthode de bornes stochastiques pour développer des encadrements sur les performances du réseau pourrait s'avérer très avantageuse. Nous précisons que notre approche est compatible avec l'analyse des transitoires. Une extension aux réseaux mobiles par exemple, pourrait également présenter un intérêt. De nos jours, les réseaux mobiles offrent plusieurs types de service comme la voix, les données ou le multimédia. Chaque type de service considéré dans ce type de réseau a sa propre spécificité en terme de qualité de service. Afin d'avoir une maîtrise de la qualité de service dans ces réseaux, différents protocoles ont été proposés dans la littérature. L'évaluation de performance de ce type de protocole par des méthodes analytiques est très complexe, voir irréalisable du fait de l'explosion combinatoire du nombre d'états qui est dû à la diversité des services considérés et la mobilité de l'utilisateur (handover). L'utilisation de la méthode de bornes développée dans cette thèse pour calculer une borne du dropping handover ou des bornes sur les indices de performance pourrait être très intéressante.

Nous finissons dans le chapitre 10, par considérer des flux non stationnaires. Pour une file d'attente simple alimentée par une trace de trafic réelle, nous avons montré que pour un modèle SBBP, l'utilisation de notre méthode de bornes stochastique permet également de garantir des bornes sur les paramètres. Ainsi, pour un processus d'arrivée non stationnaire les résultats prouvés et observés montrent que nous pouvons assurer l'obtention d'encadrements très intéressants pour les mesures de performance du système et cela en des temps relativement courts. Une extension possible pour notre approche, mais qui cette fois-ci peut s'avérer beaucoup plus difficile serait de considérer des réseaux avec de multiples classes de client. Cela permettrait de modéliser des mécanismes pour faire de la différentiation de service : les disciplines équitables pour le service dans les files d'attente, les accès privilégiés au tampon pour minimiser les pertes, les disciplines de destructions en cas de débordement du tampon, etc.. Nous notons que dans cette thèse, nous nous sommes particulièrement intéressés aux traces de trafic MAWI (Measurement and Analysis on the WIDE Internet) et CAIDA (Cooperative Association for Internet Data Analysis) pour nos expériences numériques. Cependant, d'autres traces telles que les traces vidéo ont été testées et confirment parfaitement les propriétés et les principaux résultats démontrés dans la thèse.

Les contributions apportées par notre méthode de bornes stochastiques et présentées dans la deuxième partie du manuscrit ouvrent plusieurs axes d'amélioration. En effet, nous avons considéré dans cette thèse le cas des chaînes de Markov finies à temps discret, cependant une extension aux chaînes de Markov infinies ou en temps continues serait tout aussi intéressante. Une adaptation et une amélioration de notre méthode de bornes stochastiques pourraient avoir une contribution non-négligeable pour assurer le fonctionnement et la qualité de service de ces systèmes. De plus, nous nous sommes intéressés ici à l'utilisation de l'ordre stochastique fort, la considération d'un ordre autre que l'ordre $\leq_{st}$, qui impose moins de contraintes peut également constituer un axe de recherche. Étudier des ordres plus faibles que l'ordre $\leq_{st}$

de manière à fournir des bornes plus précises sur des fonctions particulières (les moyennes, certains quantiles de la fonction de répartition, ou certains moments) et non pas sur toute la distribution pourrait présenter un intérêt. Également, nous pouvons constater que notre étude s'est fondamentalement focalisée sur l'utilisation de fonctions de récompense positives croissantes. Une voie qui pourrait représenter un intérêt serait de ne pas se limiter à ce critère et de considérer d'autres types de fonctions. Chercher et démontrer d'autres résultats qui pourraient s'avérer tout aussi importants que ceux développés ici.

Pour finir, nous pouvons dire que cette thèse apporte une contribution prometteuse et novatrice aux problèmes de modélisation de systèmes incertains et de leur dimensionnement. Nous avons pu par exemple développer des solutions et méthodes de résolutions nouvelles pour construire des modèles bornants. Nous avons également montré la propriété de monotonie stochastique sur les histogrammes de mesures de trafic réel, qui permet de garantir et préserver la relation d'ordre des d'encadrements calculés sur les mesures de performance des systèmes. Et bien évidemment, d'autres études et d'autres extensions restent et aspirent toujours à être élargies pas forcement dans le domaine des réseaux, mais pour des approches d'architecture ou de systèmes de production par exemple où nous pouvons éventuellement avoir la même démarche fondée sur la propriété de monotonie.

Annexe A

Preuve

A.1 Preuve du théorème 9.2 (page 195)

Pour une file d'attente i du réseau munie d'un tampon de longueur B_i et d'un service de capacité S_i, nous considérons la description de file d'attente présentée dans la figure A.1. Le service est représenté par S_i serveurs de capacité 1 (une unité de donnée traité par serveur). Nous introduisons cette caractérisation du service pour faciliter notre schéma de preuve. Cependant, nous précisons que cette dernière est équivalente à celle précédemment utilisée.

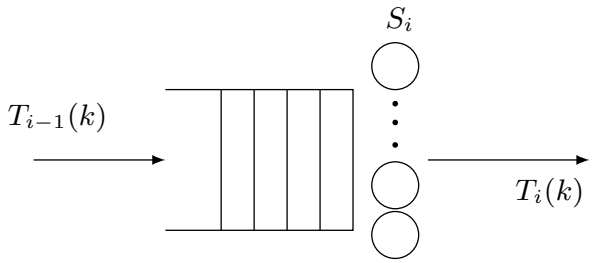


FIGURE A.1 – Représentation de la file i.

Nous notons par $T_i(k)$ la date de sortie de l'unité de données k de la file i, pour tout $i = \{0, \ldots, N\}$ et $k > 0$. La date $T_{0,k}$ représente donc la date d'arrivée de l'unité de données k dans la file 1 (sortie de la file source 0). Nous supposons que les files d'attente sont de discipline FIFO et ont un mécanisme de gestion round robin. Ainsi, pour une file d'attente i, la date de sortie de l'unité de données k est donnée comme suit :

$$T_i(k) = 1 + \max\left(T_i(k-1); T_{i-1}(k)\right). \tag{A.1}$$

En effet, si la longueur de la file i vu par le client k est inférieure ou égale à S_i (en se comptant lui même) alors

$$T_i(k) = 1 + T_{i-1}(k).$$

Il faut une unité de temps pour traverser la file i.

Sinon (longueur de la file i est $> S_i$), tous les serveurs sont occupés et nous pouvons supposer en faisant une affectation round robin des serveurs que le client k utilisera le même

serveur que le client $k - S_i$. et donc

$$T_i(k) = 1 + T_i(k - S_i).$$

Après avoir énoncé ce résultat, nous considérons les deux systèmes de files d'attente suivants :

Système A : réseau de deux files d'attente en tandem $H_1/D/S_1/\infty \rightarrow /D/S_2/\infty$ avec $t_i(k)$ la date de sortie du client k de la file i, $i = 1, 2$ et $k > 0$;

Système B : réseau de deux files d'attente en tandem $H_1/D/S_2/\infty \rightarrow /D/S_1/\infty$ avec $u_i(k)$ la date de sortie du client k de la file i, $i = 1, 2$ et $k > 0$.

Nous utilisons comme hypothèses $u_0(k) = T_0(k)$, $\forall k$ et les séquences d'arrivées dans les sources des deux réseaux son identiques. Nous voulons démontrer que $\forall\, k\ T_2(k) = u_2(k)$.

Les équations des dates de sorties de chaque file dans les deux systèmes sont données comme suit :

$$\textbf{Système A} \quad \begin{cases} T_2(k) = & 1 + \max\left(T_2(k - S_2);\, T_1(k)\right); & \text{(A.3a)} \\ T_1(k) = & 1 + \max\left(T_1(k - S_1);\, T_0(k)\right). & \text{(A.3b)} \end{cases} \tag{A.3}$$

$$\textbf{Système B} \quad \begin{cases} u_2(k) = & 1 + \max\left(u_2(k - S_1);\, u_1(k)\right); & \text{(A.5a)} \\ u_1(k) = & 1 + \max\left(u_1(k - S_2);\, u_0(k)\right). & \text{(A.5b)} \end{cases} \tag{A.5}$$

Nous distinguons deux cas selon la valeur de k. Nous posons comme hypothèse $t_i(k) = 0$ (resp. $u_i(k) = 0$) $\forall\, k \leq 0$.

- Pour $k < S_1 + S_2$, nous distinguons également ici trois autres cas. Pour le déroulement de notre preuve, nous supposons que $S_1 > S_2$. Les mêmes conclusions faites ici peuvent également être tirés si l'on considère $S_1 < S_2$ (en suivant la même démarche).
  - Pour $k \leq \min(S_1, S2)$, les équations des dates de sorties dans chaque système sont définies comme suit :

$$\textbf{Système A} \quad \begin{cases} T_2(k) = & 1 + T_1(k); \\ T_1(k) = & 1 + T_0(k). \end{cases} \qquad \Longrightarrow T_2(k) = 2 + T_0(k). \tag{A.6}$$

$$\textbf{Système B} \quad \begin{cases} u_2(k) = & 1 + u_1(k); \\ u_1(k) = & 1 + u_0(k). \end{cases} \qquad \Longrightarrow u_2(k) = 2 + u_0(k). \tag{A.7}$$

    Connaissant l'hypothèse de départ qui $u_0(k) = T_0(k)$, $\forall k$, nous pouvons conclure que le résultat des équations A.8 et A.9 est identique.
  - Pour $\min(S_1, S_2) < k \leq \max(S_1, S_2)$, les équations des dates de sorties des deux systèmes sont exprimées comme suit :

$$\textbf{Système A} \quad \begin{cases} T_2(k) = 1 + \max\left(T_2(k - S_2);\, T_1(k)\right); \\ T_1(k) = 1 + T_0(k). \end{cases}$$

$$\Longrightarrow T_2(k) = 1 + \max\left(T_2(k - S_2);\ 1 + T_0(k)\right). \tag{A.8}$$

Système B
$$\begin{cases} u_2(k) = 1 + u_1(k); \\ u_1(k) = 1 + \max\left(u_1(k - S_2);\ u_0(k)\right). \end{cases}$$

$$\Longrightarrow u_2(k) = 2 + \max\left(u_1(k - S_2);\ u_0(k)\right). \tag{A.9}$$

Procédons par induction, nous supposons que $T_2(k - S_2) = u_2(k - S_2)$ est vraie, et montrons que $\forall\, k \leq \max(S_1,\, S_2)$, $T_2(k) = u_2(k)$ est également vraie. Pour ce faire, nous avons $T_2(k - S_2) = u_2(k - S_2) = 2 + \max\left(u_1(k - 2\,S_2);\ u_0(k - S_2)\right)$, par hypothèse. En remplaçant cette quantité dans l'équation A.8, nous obtenons :

$$T_2(k) = 1 + \max\left(2 + u_1(k - 2\,S_2);\ 2 + u_0(k - S_2);\ 1 + T_0(k)\right).$$

Cette équation peut être simplifié de la manière suivante :

$$T_2(k) = 2 + \max\left(u_1(k - S_2);\ u_0(k)\right).$$

Ce résultat est identique à celui de $u_2(k)$, par conséquent nous avons $u_2(k) = T_2(k)$, $\forall\, k \leq \max(S_1,\, S_2)$.

- Pour $\max(S_1,\, S_2) < k \leq S1 + S2$, les équations des deux systèmes peuvent être résumés comme suit :
  Système A : $T_2(k) = 1 + \max\left(T_2(k - S_2);\ 1 + T_1(k - S_1);\ 1 + T_0(k)\right)$.
  Système B : $u_2(k) = 1 + \max\left(u_2(k - S_1);\ 1 + u_1(k - S_2);\ 1 + u_0(k)\right)$.
  Nous procédons également par induction, nous supposons que $\forall\, k < \max(S_1,\, S_2)$, $T_{2,\,k} = u_2(k)$ et nous voulons démontrer que c'est vrai pour tout $k < S_1 + S2$. Prenons par exemple le cas $T_2(k - S_2) = u_2(k - S_2)$. Nous avons :

$$\begin{aligned} T_2(k) = 1 + \max(&1 + u_2(k - S_1 - S_2);\ 2 + u_1(k - 2 - S_2); \\ &2 + u_0(k - S_2);\ 1 + T_1(k - S_1);\ 1 + T_0(k)). \end{aligned} \tag{A.10}$$

$$T_2(k) = 2 + \max\left(1 + u_1(k - 2\,S_2);\ 1 + u_0(k - S_2);\ T_1(k - S_1);\ T_0(k)\right).$$

$$T_2(k) = 2 + \max\left(u_1(k - S_2);\ T_1(k - S_1);\ u_0(k)\right).$$

Pour la deuxième equation, nous considérons $T_{2,\,k-S_1} = u_{2,\,k-S_1}$.

$$\begin{aligned} u_2(k) = 1 + \max(&1 + T_2(k - S_1 - S_2);\ 2 + T_1(k - 2\,S_1); \\ &2 + T_0(k - S_1);\ 1 + u_1(k - S_2);\ 1 + u_0(k)). \end{aligned} \tag{A.11}$$

$$T_2(k) = 2 + \max\left(1 + u_1(k - 2\,S_1);\ 1 + T_0(k - S_1);\ u_1(k - S_2);\ u_0(k)\right).$$

$$T_2(k) = 2 + \max\left(T_1(k - S_1);\ u_1(k - S_2);\ u_0(k)\right).$$

- Nous procédons par récurrence. Nous supposons que l'hypothèse est vraie pour $l \geq S_1 + S_2$ et $l < k$.
  En remplaçant l'équation A.3b dans A.3a, nous obtenons :

$$T_2(k) = 1 + \max\left(T_2(k - S_2);\ 1 + T_1(k - S_1);\ 1 + T_0(k)\right). \tag{A.12}$$

  Par induction, nous supposons que $T_2(k - S_2) = u_2(k - S_2)$ est vrai. En utilisant cette égalité et l'équation A.5a, nous obtenons :

$$T_2(k) = 1 + \max\left(1 + u_2(k - S_1 - S_2);\ 1 + u_1(k - S_2);\ 1 + T_1(k - S_1);\ 1 + T_0(k)\right).$$

  La factorisation de cette équation, nous permet d'écrire la simplification suivante :

$$T_2(k) = 2 + \max\left(u_2(k - S_1 - S_2);\ u_1(k - S_2);\ T_1(k - S_1);\ T_0(k)\right). \tag{A.13}$$

  Maintenant, nous effectuons le même raisonnement sur l'équation A.5a en injectant l'équation A.5b puis l'équation A.3a. Cette démarche nous permet d'avoir l'équation suivante :

$$u_2(k) = 2 + \max\left(T_2(k - S_1 - S_2);\ u_1(k - S_2);\ T_1(k - S_1);\ u_0(k)\right). \tag{A.14}$$

  Ainsi, d'après l'hypothèse de départ $u_0(k) = T_0(k)$, $\forall\, k$ et l'induction sur $u_2(l) = t_2(l)$, $\forall\, l < k$ (ici $l = k - S_1 - S_2$)., nous avons $u_2(k) = t_2(k)$, $\forall\, k$.

Bibliographie

[AA92] Bobbio A and Cumani A. Ml estimation of the parameters of a ph distribution in triangular canonical form. pages 33–46, 1992.

[AA95] O. Abu-Amsha. Méthodes d'encadrement stochastiques dans les réseaux d'automates stochastiques. Mémoire de dea de statistique et processus stochastiques, INPG-Université Joseph Fourier à Grenoble, Lmc-Imag, June 1995.

[AAM05] Bobbio A, Horvath A, and Telek M. Matching three moments with minimal acyclic phase type distributions. *STOCHASTIC MODELS*, 21(2-3) :303–326, 2005.

[AAV98] O. Abu-Amsha and J. M. Vincent. An algorithm to bound functionals on Markov chains with large state space. In *4th INFORMS Conference on Telecommunications*, Boca Raton, Floride, E.U, 1998. INFORMS.

[ACC+12] F. Aït-Salaht, J. Cohen, H. Castel Taleb, J. M. Fourneau, and N. Pekergin. Accuracy vs. complexity : the stochastic bound approach. In *11th International Workshop on Disrete Event Systems (WODES 2012)*, number 8, pages 343–348, 2012.

[ACFP13a] F. Aït-Salaht, H. Castel Taleb, J. M. Fourneau, and N. Pekergin. A bounding histogram approach for network performance analysis. In *15th IEEE International Conference on High Performance Computing and Communications (HPCC'13), China*, pages 458–465, 2013.

[ACFP13b] F. Aït-Salaht, H. Castel Taleb, J. M. Fourneau, and N. Pekergin. Modeling networks and active queues management with stochastic bounds and histograms. In *7th International Workshop on Verification and Evaluation of Computer and Communication Systems (VeCoS 2013), Florence, Italy*, 2013.

[ACFP13c] F. Aït-Salaht, H. Castel Taleb, J. M. Fourneau, and N. Pekergin. Stochastic bounds and histograms for network performance analysis. In *10th European Workshop on Performance Engineering (EPEW'13)*, volume 8168, pages 13 – 27, 2013. Collection : Lecture Notes in Computer science.

[ACFP13d] F. Aït-Salaht, H. Castel Taleb, J. M. Fourneau, and N. Pekergin. Une approche combinant bornes stochastiques, traces et histogrammes pour l'analyse de

performance des réseaux. In *Journal européen des systèmes automatisés Volume 47 N° 1-2-3/Janvier-Mai 2013 : Modélisation des systèmes réactifs (MSR 2013)*, pages 165–179. Germes Lavoisier, 2013.

[AFCP14] F. Aït-Salaht, J. M. Fourneau, H. Caltel-Taleb, and N. Pekergin. Approche par bornes stochastiques et histogrammes pour l'analyse de performance des réseaux. In *10ième Atelier en Évaluation de Performances*, Sophia-Antipolis, France, 2014.

[AFP12] Farah Aït-Salaht, Jean-Michel Fourneau, and Nihal Pekergin. Computing entry-wise bounds of the steady-state distribution of a set of markov chains. In *27th International Symposium on Computer and Information Sciences, Paris, France*, pages 115–122. Springer, 2012.

[AFP13] Farah Aït-Salaht, Jean-Michel Fourneau, and Nihal Pekergin. Computing bounds of the mttf for a set of markov chains. In *28th International Symposium on Computer and Information Sciences, Paris, France*, Lecture Notes in Electrical Engineering, pages 67–76. Springer, 2013.

[BBFP06] Mouad Ben Mamoun, Ana Busic, Jean-Michel Fourneau, and Nihal Pekergin. Increasing convex monotone Markov chains : theory, algorithms and applications. In *Markov Anniversary Meeting*, pages 189–210. Boson Books, Raleigh, North Carolina, 2006.

[BCOQ92] F. Baccelli, G. Cohen, G. J. Olsder, and J.-P. Quadrat. *Synchronization and Linearity : An Algebra for Discrete Event Systems*. Willey, New York, 1992.

[BF10] A. Busic and J.-M. Fourneau. Iterative component-wise bounds for the steady-state distribution of a markov chain, 2010.

[BF11] A. Busic and J.-M. Fourneau. Iterative component-wise bounds for the steady-state distribution of a markov chain. *Numerical Linear Algebra with Applications*, 18(6) :1031–1049, 2011.

[BGDMT98] G. Bolch, S. Greiner, H. De Meer, and K. Trivedi. *Queueing Networks and Markov Chains*. John Wiley & Sons, 1998.

[Bil95] Patrick Billingsley. *Probability and Measure*. Wiley, New York, NY, 3rd edition, 1995.

[BJmL93] Francois Baccelli, Alain Jean-marie, and Zhen Liu. A survey on solution methods for task graph models. INRIA Report, Sophia Antipolis, France, 1993.

[Buc02] Peter Buchholz. An iterative bounding method for stochastic automata networks. *Perform. Eval.*, 49(1/4) :211–226, 2002.

[Buc05] P. Buchholz. An improved method for bounding stationary measures of finite Markov processes. *Performance Evaluation*, 62 :349–365, 2005.

[Buc06] P. Buchholz. Bounding stationary results of tandem networks with map input and ph service time distributions. In Raymond A. Marie, Peter B. Key, and Evgenia Smirni, editors, *Proceedings of the Joint International Conference on Measurement and Modeling of Computer Systems, SIGMETRICS/Performance 2006, Saint Malo, France*, pages 191–202. Acm, 2006.

[Buc10] P. Buchholz. Bounding reward measures of Markov models using Markov decision processes. In *Numerical Solution of Markov Chains Conference (NSMC 2010)*, pages 13–16, W. sburg, Virginia, 2010.

[Bus07] A. Busic. *Comparaison Stochastique de modèles Markoviens : une approche algorithmique et ses applications en fiabilité et en évaluation de performances*. PhD thesis, Université de Versailles Saint-Quentin, 2007.

[CAI] CAIDA. Traces of oc48 link at ames internet exchange (aix) (april 24, 2003), accessed via datcat - internet data measurement catalog,. http ://imdc.datacat.org.

[Cha00] Ch-Sh. Chang. *Performance guarantees in Communication Networks*. Springer, 2000.

[CS84] P.-J. Courtois and P. Semal. Bounds for the positive eigenvectors of nonnegative matrices and for their approximations by decomposition. *Journal of the ACM*, 31(4), 1984.

[CS85] P.-J. Courtois and P. Semal. On polyhedra of Perron-Frobenius eigenvectors. *Linear Algebra and Applications*, 65 :157–170, 1985.

[CS86] P.-J. Courtois and P. Semal. Computable bounds for conditional steady-state probabilities in large Markov chains and queueing models. *IEEE Journal on Selected Areas in Communications*, 4(6) :926–937, 1986.

[DFPV06] T. Dayar, J.-M. Fourneau, N. Pekergin, and J.-M. Vincent. Polynomials of a stochastic matrix and strong stochastic bounds. In *Markov Anniversary Meeting*, pages 211–228, Charleston, 2006. Boson Books, Raleigh, North Carolina.

[DGK+02] José Luis Díaz, Daniel F. García, Kanghee Kim, Chang-Gun Lee, Lucia Lo Bello, José María López, Sang Lyul Min, and Orazio Mirabella. Stochastic analysis of periodic real-time systems. In *RTSS*, pages 289–300. IEEE Computer Society, 2002.

[Don93] Susanna Donatelli. Superposed stochastic automata : A class of stochastic Petri nets with parallel solution and distributed state space. *Perform. Eval.*, 18(1) :21–36, 1993.

[Doo53] J.L. Doob. *Stochastic Processes*. Wiley Publications in Statistics. John Wiley & Sons, 1953.

[Dud02] R.M. Dudley. *Real Analysis and Probability*. Cambridge Studies in Advanced Mathematics. Cambridge University Press, 2002.

[FACP14] J. M. Fourneau, F. Aït-Salaht, H. Caltel-Taleb, and N. Pekergin. Bornes sur les histogrammes et équation de loynes pour l'analyse rapide d'une file fifo. In *ROADEF - 15ème congrès annuel de la Société française de recherche opérationnelle et d'aide à la décision*, Bordeaux, France, February 2014.

[FJ93] Sally Floyd and Van Jacobson. Random early detection gateways for congestion avoidance. *IEEE/ACM Trans. Netw.*, 1(4) :397–413, August 1993.

[FM94] G. Franceschinis and R. R. Muntz. Bounds for quasi-lumpable markov chains. In *Proceedings of the 16th IFIP Working Group 7.3 international symposium*

on Computer performance modeling measurement and evaluation, Performance '93, pages 223–243. Elsevier Science Publishers B. V., 1994.

[FQ12] J.-M. Fourneau and F. Quessette. Some improvements for the computation of the steady-state distribution of a Markov chain by monotone sequences of vectors. In *ASMTA*, Lecture Notes in Computer Science. Springer, 2012.

[Fri65] H.D. Friedman. Reduction methods for tandem queueing systems. **13** :121–131, 1965.

[GHBDZ10] Varun Gupta, Mor Harchol-Balter, J. G. Dai, and Bert Zwart. On the inapproximability of m/g/k : why two moments of job size distribution are not enough. *Queueing Syst.*, 64(1) :5–48, 2010.

[GO02] Roch Guérin and Ariel Orda. Computing shortest paths for any number of hops. *IEEE/ACM Trans. Netw.*, 10(5) :613–620, 2002.

[GTH85] W.K. Grassman, M.I. Taksar, and D.P. Heyman. Regenerative analysis and steady state distributions for markov chains. *Operations Research*, 33(5) :1107–1116, 1985.

[Hag89] W. W. Hager. Updating the inverse of a matrix. *SIAM Rev.*, 31(2) :221–239, 1989.

[HK01] J. Hillston and L. Kloul. An efficient Kronecker representation for PEPA models. In L. De Alfaro and S. Gilmore., editors, *Process Algebra and Probabilistic Methods, Performance Modeling and Verification : Joint International Workshop, PAPM-PROBMIV, Aachen, Germany*, volume 2165 of *Lecture Notes in Computer Science*, pages 120–135. Springer, 2001.

[HM07] S. Haddad and P. Moreaux. Sub-stochastic matrix analysis for bounds computation - theoretical results. *European Journal of Operational Research*, 176(2) :999–1015, 2007.

[HOVC07a] Enrique Hernández-Orallo and Joan Vila-Carbó. A histogram-based stochastic process for finite buffer occupancy analysis. In *VALUETOOLS*, page 44, 2007.

[HOVC07b] Enrique Hernández-Orallo and Joan Vila-Carbó. Network performance analysis based on histogram workload models. In *MASCOTS*, pages 209–216, 2007.

[HOVC09] Enrique Hernández-Orallo and Joan Vila-Carbó. Web server performance analysis using histogram workload models. *Computer Networks*, 53(15) :2727–2739, 2009.

[HOVC10] Enrique Hernández-Orallo and Joan Vila-Carbó. Network queue and loss analysis using histogram-based traffic models. *Computer Communications*, 33(2) :190–201, 2010.

[HTB05] G. Horváth, M. Telek, and P. Buchholz. A map fitting approach with independent approximation of the inter-arrival time distribution and the lag correlation. In *QEST*, pages 124–133. IEEE Computer Society, 2005.

[HTS91] On Hashida, Yoshitaka Takahashi, and Shinsuke Shimogawa. 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*, 9(3) :394–401, 1991.

[Jai91] Raj Jain. *The art of computer systems performance analysis - techniques for experimental design, measurement, simulation, and modeling.* Wiley professional computing. Wiley, 1991.

[JT88] M.A. Johnson and M.R. Taaffe. *Matching Moments to Phase Distributions : Mixtures of Erlang Distributions of Common Order*. Research memorandum. School of Industrial Engineering, Purdue University, 1988.

[JT89] M.A. Johnson and M.R. Taaffe. *Matching Moments to Phase Distributions : Nonlinear Programming Approaches*. Research memorandum. School of Industrial Engineering, Purdue University, 1989.

[KAC04] Jahwan Koo, Seong-Jin Ahn, and Jin-Wook Chung. Performance analysis of active queue management schemes for ip network. In *International Conference on Computational Science*, volume 3036 of *Lecture Notes in Computer Science*, pages 349–356. Springer, 2004.

[Kij97] M. Kijima. *Markov Processes for Stochastic Modeling*. Chapman & Hall, London, UK, 1997.

[KK77] J. Keilson and A. Kester. Monotone matrices and monotone markov processes. *Stochastic Processes and Their Applications*, 5 :231–241, 1977.

[Kle76] L. Kleinrock. *Queueing systems, Computer applications, vol 2*. Wiley Interscience, New York, 1976.

[KLL+02] Alexander KLEMM, Christoph LINDEMANN, Marco LOHMANN, Tony Field, Peter G. Harrison, Jeremy Bradley, and Uli Harder. Traffic modeling of ip networks using the batch markovian arrival process. *Lecture notes in computer science*, 2324(12) :92–110, 2002.

[KS01] Seok-Kyu Kweon and Kang G. Shin. Real-time transport of mpeg video with a statistically guaranteed loss ratio in atm networks. *IEEE Transactions on Parallel and Distributed Systems*, pages 12–4, 2001.

[Lie] Add-on multimedia : Spécialiste du multimédia professionnel. http://www.add-on-multimedia.fr/Connectivité/LiensIP/tabid/185/Default.aspx.

[LM94] J. C. S. Lui and R. R. Muntz. Computing bounds on steady state availability of repairable computer systems. *Journal of the ACM*, 41(4) :676–707, 1994.

[Loy62] R. M. Loynes. Stationary waiting-time distributions for single-server queues. *Ann. Math. Statist.*, 33(4) :1227–1527, 1962.

[Mah97] S. Mahevas. *Modèles markoviens de grande taille : calculs de bornes*. PhD thesis, Univeristé de Rennes I, 1997.

[MDG89] R. R. Muntz, E. De Souza E Silva, and A. Goyal. Bounding availability of repairable computer systems. *IEEE Trans. on Computers*, 38(12) :1714–1723, 1989.

[MEP01] Sorin Manolache, Petru Eles, and Zebo Peng. Memory and time-efficient schedulability analysis of task sets with stochastic execution time. In *ECRTS*, pages 19–, 2001.

[MR01] S. Mahevas and G. Rubino. Bound computation of dependability and performance measures. *IEEE Trans. Comput.*, 50(5) :399–413, 2001.

[MS02] A. Müller and D. Stoyan. *Comparison Methods for Stochastic Models and Risks*. Wiley, New York, NY, 2002.

[O'C06] D. O'Connor. Exact and approximate distributions of stochastic PERT networks. Dublin University College, 2006.

[Pla85] B. Plateau. On the stochastic structure of parallelism and synchronization models for distributed algorithms. In *Proc. of the SIGMETRICS Conference*, pages 147–154, Texas, 1985.

[PS80] W. C. Parr and W. R. Schucany. Minimum distance and robust estimation. *Journal of the American Statistical Association*, 75(371) :616–624, 1980.

[PV91] N. Pekergin and J.-M. Vincent. Stochastic bounds on execution times of parallel programs. *IEEE Trans. Software Eng.*, 17(10) :1005–1012, 1991.

[Rob92] John P Robertson. The computation of aggregate loss distributions. In *In Proceedings of the Casualty Actuarial Society*, pages 57–133, 1992.

[SC00] Kenjiro Cho Sony and Kenjiro Cho. Traffic data repository at the wide project. In *In Proceedings of USENIX 2000 Annual Technical Conference : FREENIX Track*, pages 263–270, 2000.

[Sch02] Hans Schneider. Wielandt's proof of the exponent inequality for primitive nonnegative matrices. *j-LINEAR-ALGEBRA-APPL*, 353(1–3) :5–10, sep 2002.

[Sch07] M. Schleyer. *Discrete Time Analysis of Batch Processes in Material Flow Systems*. Wissenschaftliche Berichte des Institutes für Fördertechnik und Logistiksysteme des Karlsruher Instituts für Technologie. Univ.-Verlag Karlsruhe, 2007.

[Sem95] P. Semal. Refinable bounds for large Markov chains. *IEEE Trans. on Computers*, 44(10) :1216–1222, 1995.

[She95] Jian Shen. Proof of a conjecture about the exponent of primitive matrices. *j-LINEAR-ALGEBRA-APPL*, 216(1–3) :185–203, feb 1995.

[She96] Jian Shen. A bound on the exponent of primitivity in terms of diameter. *j-LINEAR-ALGEBRA-APPL*, 244(1–3) :21–33, sep 1996.

[SS94] M. Shaked and J. G. Shantikumar. *Stochastic Orders and their Applications*. Academic Press, San Diego, CA, 1994.

[SSD93] Paul Skelly, Mischa Schwartz, and Sudhir S. Dixit. A histogram-based model for video traffic behavior in an atm multiplexer. *IEEE/ACM Trans. Netw.*, 1(4) :446–459, 1993.

[Ste95] W.J. Stewart. *Introduction to the numerical Solution of Markov Chains*. Princeton University Press, New Jersey, 1995.

[Sto83] D. Stoyan. *Comparaison Methods for Queues and Other Stochastic Models*. John Wiley and Sons, Berlin, 1983.

[Tal96] H. Taleb. *Bornes stochastiques pour l'évaluation de performance des réseaux informatiques*. PhD thesis, Univérsité de ParisVI, January 1996.

[TDS+95] Too-Seng Tia, Zhong Deng, Mallikarjun Shankar, Matthew F. Storch, Jun Sun 0002, L.-C. Wu, and Jane W.-S. Liu. Probabilistic performance guarantee for real-time tasks with varying computation times. In *IEEE Real Time Technology and Applications Symposium*, pages 164–173. IEEE Computer Society, 1995.

[Tij03] H.C. Tijms. *A First Course in Stochastic Models*. Wiley, 2003.

[Tri02] K. S. Trivedi. *Probability and Statistic with Reliability, Queueing and Computer Science Applications, Second Edition*. Wiley, 2002.

[TSC09] Jean-Sébastien Tancrez, Pierre Semal, and Philippe Chevalier. Histogram based bounds and approximations for production lines. *European Journal of Operational Research*, 197(3) :1133–1141, 2009.

[TVP92] M. Tremolieres, J.-M. Vincent, and B. Plateau. Determination of the optimal upper bound of a markovian generator. Technical Report 106, LGI-IMAG, Grenoble, France, 1992.

[Var09] R.S. Varga. *Matrix Iterative Analysis*. Springer Series in Computational Mathematics. Springer, 2009.

[Web] Richard R. Weber. The interchangeability of ./m/1 queues in series.

[Whi81] Ward Whitt. Approximating a point process by a renewal process : The view through a queue, an indirect approach. *Management Science*, 27(6) :619–636, 1981.

[Whi83] W. Whitt. The queueing network analyzer. *The Bell System Technical Journal*, 62(9) :2779–2816, 1983.

[Whi85] W. Whitt. The best order for queues in series. **31** :475–487, 1985.

[WVMS96] Willinger Walter, Paxson Vern, and Taqqu Murad S. Self-similarity and heavy tails : Structural modeling of network traffic, 1996.