**UNIVERSITÉ TOULOUSE III - PAUL SABATIER**
**UFR Mathématiques, Informatique, Gestion**

**THÈSE**
Pour obtenir le grade de
DOCTEUR DE L'UNIVERSITÉ TOULOUSE III
*Discipline : Informatique*

par

**Osman SALEM**

Soutenue le 9 Octobre 2006

# Modélisation Algébrique et Évaluation de Performances des Mécanismes de Gestion de la Qualité de Service dans les Réseaux Ad Hoc

**Directeur de thèse :**
M. Abdelmalek BENZEKRI Professeur, IUT-A, Université Toulouse III

**JURY :**

| | | |
|---|---|---|
| Rapporteurs : | Mme. Brigitte PLATEAU | Professeur, INPG-ENSIMAG, Grenoble |
| | M. Guy PUJOLLE | Professeur, Université de Paris VI |
| Examinateurs : | Mme. Annie GRAVEY | Professeur, ENST-Bretagne, Brest |
| | M. Guy JUANOLE | Professeur, Université Toulouse III |
| | M. Yves RAYNAUD | Professeur, Université Toulouse III |
| | M. Abdelmalek BENZEKRI | Professeur, IUT-A, Université Toulouse III |
| Invité : | M. Jean-Michel BRUEL | Maître de Conférences, Université de Pau |

Institut de Recherche en Informatique de Toulouse - UMR UPS - INPT - CNRS 5505
Thème Architecture, Systèmes et Réseaux - Équipe Administration de Réseau Et Interopérabilité de Systèmes
Université Paul Sabatier - 118 Route de Narbonne, 31062 Toulouse Cedex 9

# Remerciements

Je voudrais profiter de ces quelques lignes pour rendre hommage à tous ceux qui ont contribué directement ou indirectement à l'achèvement de ce travail.

En premier lieu, je tiens à remercier chaleureusement Brigitte Plateau et Guy Pujolle pour m'avoir fait l'honneur de rapporter cette thèse, malgré leurs contraintes d'emploi du temps chargés. Je les remercie de leur patience et de l'attention qu'ils ont bien voulu m'accorder. Je remercie ensuite Guy Juanole, Yves Raynaud et Annie Gravey pour avoir accepté de faire partie de mon jury.

Merci à tous ceux qui ont lu le manuscrit de ma thèse, me faisant part de leurs commentaires et suggestions. Je pense spécialement à Abdelmalek Benzekri, qui a fait un travail énorme, pour corriger la première version du manuscrit, à Thierry Desprats pour la lecture attentive de quelques publications, et à François Barrère pour son soutien logistique et pratique.

J'adresse toute ma gratitude à Abdelmalek Benzekri pour m'avoir permis de mener à bien cette thèse, pour ses conseils éclairés, sa grande disponibilité, ses compétences et enfin pour m'avoir guidé dans le monde de la recherche académique. Mes remerciements vont tout naturellement à Jean-Michel Bruel qui m'a encouragé durant le déroulement de cette thèse : merci pour la porte toujours ouverte, les conseils permanents, et les soutiens qu'il m'a portés à l'université de Pau.

Un grand merci à tous les membres du département informatique de l'université de Pau et des Pays de l'Adour. J'ai eu le plaisir de travailler et de vivre avec eux. Les échanges ont toujours été source d'enrichissement et dans un respect mutuel. Merci à toutes et à tous qui ont contribué à mon processus d'intégration dans ce département. Merci à : Alain Teste, les adorables Annig et Laurent Lacayrelle, Bruno Jobard, les ravissants Éric et Sophie Gouarderes, Jean-Michel Bruel, Mauro Gaio, Franck Barbier, Nabil Hameurlain, Nicolas Belloir et Régine Dufaur-Dessus pour les encouragements, les discussions, les dîners et tous les soirées sympathiques passées ensemble, et qui ont rendu mon séjour à PAU très agréable.

Je remercie Eric Cairou, le camarade, l'ami et le frère, d'avoir partagé avec moi une quantité non négligeable de boissons caféinés.

Un grand merci aussi à tous les membres de l'équipe SIERA, je pense spécialement à : André Aoun, Michel Galindo, Michelle Sibilla, Philippe Vidal, Daniel Marquié. Merci également à Martine Labruyère pour sa patience, sa compréhension et son soutien très apprécié dans les démarches administratives. Merci à Maryse Cailloux pour son soutien logistique, et à Jacqueline Arnillas pour sa gentillesse.

Je remercie particulièrement mes camarades d'infortune, les thésards et les autres : Fréderic Grasset, Alia Fourati, Romain Laborde, Patrick Trabé, Zein Ibrahim, Bassem Nasser, Michel Kamel, Fabien Romeo, Cédric Tessier, Laurent Mehats, Nicolas Lamarque, Pierre Lausto, Julien Lacoste, Julien Lesburg, et tous les autres.

Je n'oublie évidemment pas tous mes copains et amis qui ont su, par leur présence, me soutenir. En désordre, je remercie Nancy pour nos chemins parallèles, Nelly pour notre amitié fraternelle et ses sms qui arrivent trop tard dans la nuit, Anne-Laure (elle sait pourquoi), Marie-Hélène pour les tartes, les cafés et tous le reste, Nicolavich et Marianne pour leur bonne humeur, Eliot et son rêve de voyage en Irlande, Baptiste et Patricia pour les chocolats, le foie gras et les fromages pourries. La place me manque pour leur exprimer nominativement ma reconnaissance.

J'adresse un remerciement particulier à Emanuel Lavinal pour ses encouragements et son soutien dans mes jours montagnards, et pour toutes les séances de « remonte-moral ». Merci encore à Michelle, Marion, Fred, Fatou, Nancy-aura, et tous les autres gens dont j'ai eu l'occasion de rencontrer dans les

conférences. Rien ne reste que de si beaux souvenirs, et si beaux moments dans le château de Robin Hood.

Je remercie enfin ma famille pour les soutiens sans limites, pour avoir financé de si longues études, m'avoir soutenu toujours avec amour, et m'avoir donné le goût de savourer ma réussite après si longue et si difficile années de thèse. Merci à maman et à mes sœurs, pour leur aide et leur soutien inconditionnel qui m'a toujours réchauffé le cœur et fait le bonheur des compagnies téléphoniques. Mes pensées vont évidemment à mon père. Papa, ton absence me tue, et tes souvenirs ont été toujours là lorsque j'en avais besoin.

J'ai sûrement oublié de citer certains d'entre vous qui m'ont grandement aidé dans mon travail, et sans doute il y a tant des gens qui n'ont pas aidé directement, mais au fond de mon cœur je vous remercie chaleureusement.

À ma mère,
À la mémoire de mon père.

# Résumé

Les chaînes de Markov sont souvent utilisées pour l'analyse des performances des systèmes dans de nombreux domaines, et elles sont souvent obtenues par le biais d'un formalisme de modélisation de haut niveau. Parmi les formalismes couramment utilisés, on se place dans le cadre des algèbres de processus stochastiques.

La restriction de la description temporelle aux distributions sans mémoire limite les domaines d'applications des algèbres de processus stochastiques, et laisse beaucoup de questions sur l'exactitude des valeurs des performances, obtenues par l'approximation des fonctions de distributions de délais de toutes les activités d'un système par des distributions Markoviennes. Cette thèse propose des méthodes et des algorithmes visant à optimiser le formalisme afin de pouvoir spécifier et évaluer les performances des modèles proches de la réalité. Les méthodes et les algorithmes développés au cours de cette thèse ont été implémentés, et des exemples numériques sont présentés pour illustrer les apports de cette thèse.

Pour cela, nous introduisons tout d'abord le concept des algèbres de processus stochastiques, et les techniques d'analyses et d'agrégation utilisées, ainsi que les nouveautés qu'on peut ramener à ce formalisme. Ensuite, nous nous intéressons aux mécanismes de gestion de la qualité de service dans les réseaux ad hoc, comme nouveau domaine d'application. Après l'identification des questions fondamentales liées aux caractéristiques de ce type de réseaux, nous proposons un nouveau mécanisme de gestion de la qualité de service de bout en bout, puis nous réalisons une modélisation algébrique et des analyses qualitatives et quantitatives sur les diagrammes de transitions sous jacents du modèle algébrique, avant d'utiliser ce dernier pour la construction d'un modèle de simulation. Les analyses quantitatives réalisées sur le modèle de simulation sous différentes conditions, montrent la capacité du modèle proposé à s'adapter au dynamicité des ressources du réseau.

**Mots clés :** Algèbres de processus stochastiques, Distributions phase type, Évaluation des performances, Grand espace d'états, Agrégation, Qualité de service, Réseaux ad hoc.

# Abstract

Markov chains are often used for quantitative analysis of systems in many areas of application, and they are usually obtained by means of high level modeling formalism due to manual difficulty in specifying even a small system. Among widely used high level formalisms, we use stochastic process algebra.

Unfortunately, restriction of temporal delay to memoryless distributions limits the application area of this framework, and leaves a lot of doubt around the precision of the results obtained by approximating distribution delay of all activities by Markovian one. This thesis proposes new methods and algorithms for enhancing the algebraic framework in order to make it able to specify and analyze real models. Methods and algorithms developed during this thesis are implemented, and numerical examples are presented to illustrate our contributions.

We begin by introducing the concept of modeling with stochastic process algebras, and the widely used techniques to reduce and analyze the underlying transition diagram, as well as the novelties that we can bring to this formalism. Afterwards, we turn our attention to quality of service management mechanisms used in ad hoc network, like a new application area for formal design and modeling with algebraic formalism. After the identification of fundamental problems related to the dynamic characteristics of these kind of networks, we propose a new mechanism for providing end to end quality of service, then we realize functional and temporal analysis at the underlying transition diagram from the algebraic model, that we will use for the construction of a simulation model. Simulation results under various network loads show the ability of the proposed model to adapt with resources variations and the dynamic characteristics of ad hoc networks.

**Keywords :** Stochastic Process Algebra, Phase-Type Distribution, Performance Evaluation, Large State Space, Agregation, Quality of Service, Ad Hoc Networks.

# Table des matières

# Table des figures

# Liste des tableaux

# Chapitre 1

# Introduction

Nous n'apprenons rien au lecteur en rappelant que la complexité des mécanismes de gestion des réseaux ne fait qu'augmenter. Le développement rapide de ces mécanismes entraîne des problèmes critiques de performances, avec par exemple des phénomènes de surcharge, de blocage et d'effondrement, ainsi qu'une qualité de service non garantie [Ben03]. Pour garantir les performances de ces systèmes, une analyse qualitative et quantitative est nécessaire dans la phase de conception afin d'identifier les problèmes et de les résoudre.

Le test qui est la technique le plus couramment utilisée pour déterminer si un prototype est suffisamment robuste, permet de trouver de nombreuses erreurs dans les systèmes, mais ne permet en aucun cas de garantir l'absence totale d'erreurs. La conception robuste d'un système demande l'utilisation de méthodes rigoureuses dans la détection des erreurs, comme c'est le cas avec l'utilisation des méthodes formelles permettant la conception et la vérification qualitative d'un modèle de système.

Traditionnellement, l'analyse des performances a été souvent négligée dans la phase de conception jusqu'à ce que les problèmes deviennent apparents dans le prototype. La satisfaction des exigences des performances n'était testée qu'après l'implémentation (ou la fabrication) d'un modèle conçu à l'aide d'un langage de description formelle, permettant de vérifier seulement les aspects qualitatifs du modèle. Ce qui conduit parfois à la reconstruction et au raffinage du modèle lorsque les performances sont jugées médiocres [BDG98]. Une autre stratégie consistait à construire un autre modèle pour l'évaluation quantitative, mais la conception de deux modèles différentes pose des problèmes de comptabilité et de cohérence entre les deux modèles. La nécessité d'intégrer l'analyse des performances dans la phase de conception d'un modèle d'un système, a été largement reconnue [Ber99, Her02, BB98, BDG98].

Pour anticiper les problèmes qualitatifs et quantitatifs avant l'implémentation et la fabrication d'un prototype, il est nécessaire de vérifier les comportements fonctionnels et de prévoir les performances du système dans un souci de gain en coût et en temps lors du développement de ce dernier. Aujourd'hui, des langages de spécifications formelles avec une expressivité temporelle sont mis en œuvre pour fournir un formalisme intégré. Ces derniers offrent différentes méthodes pour la construction de modèles d'un système et pour la réalisation de ces deux types d'analyses, afin de détecter les problèmes potentiels qui peuvent survenir, et répondre aux questions de performance et de surcoût de reconstruction du prototype. De plus, nous pouvons ainsi modifier la vitesse de traitement de certains composants du modèle et dimensionner le système de manière à obtenir les meilleures performances possibles pour une architecture donnée.

Dans les section suivantes, nous faisons une synthèse des méthodes d'évaluation de performances et des formalismes utilisés. Ensuite, nous précisons les objectifs de cette thèse et le plan de cet ouvrage.

## 1.1 Méthodes d'évaluation des performances

L'évaluation des paramètres des performances d'un modèle avant son implémentation en un prototype, peut se réaliser par 3 méthodes : la simulation, les méthodes analytiques et les méthodes numériques.

La simulation permet d'imiter le comportement d'un système en traitant des modèles généralement plus proches de la réalité que les méthodes formelles, et elle sert souvent à valider les modèles de ces derniers. La simulation à événements discrèts (DES) est généralement utilisée pour les modèles stochastiques sans aucune restriction temporelle (distribution générale ou non-Markovienne), où les probabilités d'exécution des actions sont remplacées par des échantillons des lois de distributions associées. La simulation est répétée plusieurs fois (typiquement plusieurs millions de fois) pour que les variables aléatoires représentent bien la distribution de probabilité et son coefficient de variation. En plus, la simulation a l'avantage d'être insensible à la taille de l'espace d'état. Mais, l'utilisation directe de la simulation est difficile et sa complexité temporelle croit avec les détails investis dans le modèle et le degré de précision (ou l'intervalle de confiance) requis. Pour faciliter la mise en place d'une simulation, il sera plus judicieux de construire un modèle formel pour s'assurer du bon fonctionnement du modèle avant sa mise en œuvre dans le simulateur.

Les méthodes analytiques permettent de dériver les paramètres de performances en fonction des paramètres temporels du modèle, tout en utilisant des solutions sous forme de formules mathématiques prédéfinies. Ces méthodes sont très efficaces et beaucoup plus rapides que la simulation, mais la solution n'existe que pour une classe très restreinte des systèmes.

Les méthodes numériques servent de modèles abstraits à partir desquels des résultats exacts peuvent être obtenus, tout en résolvant un ensemble d'équations linéaires dérivées de la chaîne du Markov sous-jacente du modèle. Ces méthodes offrent un compromis entre les deux méthodes précédentes. Elles sont beaucoup plus précises et dans certains cas beaucoup plus rapides que la simulation, mais moins rapides que les méthodes analytiques.

Deux facteurs principaux empêchent l'utilisation des méthodes numériques pour l'analyse d'un modèle réel : la taille de l'espace d'états et la restriction de la fonction de distribution des délais des actions aux distributions sans mémoire.

## 1.2 Formalismes de modélisation

Durant les dernières années, plusieurs techniques de modélisation formelles ont été développées. Toutes sont basées sur le même principe qui consiste à définir :

– les états du système.
– les transitions entre les états.
– les délais des transitions.

Les états sont représentés graphiquement par des nœuds et les transitons par des flèches étiquetées reliant les nœuds entre eux. Les transitions sont étiquetées par des événements dont l'occurrence engendre une transition et un changement d'état du système, après l'expiration de son temporisateur. Le graphe résultant est connu sous le nom de diagramme de transitions représentant le comportement du système.

En supposant que le système étudié a un nombre fini d'états, le changement de l'état du système est engendré par l'exécution d'une action qui déclenche une transition. Mais, il est difficile de réaliser une modélisation manuelle de l'ensemble de toutes les transitions possibles entre les différents états d'un système large et complexe, car souvent le diagramme de transition sous-jacent d'un système réel est de l'ordre de quelques milliers voir même centaines de milliers d'états. Plusieurs formalismes de haut niveau [Hil96, MBC$^+$95, PA91] ont été proposés pour permettre la modélisation d'un système complexe par le biais des formules algébriques compactes, et la dérivation automatique au plus bas niveau, du diagramme de transition et de la chaîne de Markov [Ste94, Wol89] sous jacente, qui ont un très grand nombre d'états et de transitions en utilisant des sémantiques définies en termes de transitions.

Chaque formalisme a sa propre syntaxe et dont la dérivation de la chaîne de Markov n'est autre qu'une application de sa sémantique généralement définie en termes de transitions entre les états. Un logiciel est alors utilisé pour générer l'espace d'états et la matrice « générateur infinitésimal » de la chaîne de Markov, ainsi que pour calculer les solutions stationnaires et transitoires. Quel que soit le formalisme utilisé, l'objectif est le calcul des paramètres de performances du système en question. Certaines méthodes de résolution sont cependant spécifiques à un formalisme, mais des techniques peuvent être

adaptées d'un formalisme à un autre. Parmi ceux largement utilisés dans divers domaines d'application, nous distinguons quatre classes principales de formalismes qui convergent dans les méthodes d'analyse :

- les réseaux de files d'attente.
- les Réseaux de Petri Stochastiques (RdPS).
- les Réseaux d'Automates Stochastiques (RAS).
- les Algèbres des Processus Stochastiques (APS).

Les réseaux de files d'attente [Med02, BDPS04, Kle75] sont un formalisme largement utilisé pour l'évaluation de performance, mais qui est jugé moins expressif que les trois autres formalismes dû au manque de formel. Des algorithmes pour la solution analytique sont extrêmement rapides, et peuvent être utilisés pour évaluer les paramètres de performance du système, en plus de l'évaluation par simulation. Différents outils comme PEPSY [Kir94] et QNAP2 [VP85] sont largement utilisés dans ce domaine.

Les Réseaux de Petri temporisés [GMMP89] ont été les premières approches pour l'introduction de la notion de temps dans ce formalisme. Les Réseaux de Petri Stochastiques (RdPS) [CGL94] avec leurs nombreux dérivés ont ensuite été définis, comme : Réseaux de Pétri Stochastiques Généralisés (ou Generalised Stochastic Petri Nets GSPNs [MBC$^+$95, CMBC93, MBD98, MBC84]), Réseaux de Petri Stochastiques et Déterministes (Deterministic and Stochastic Petri Nets DSPNs [Lin98]), Réseaux de Petri Stochastiques à temps discret (Discret Determinstic Petri Net DDPN [Cia95, ZFH01]), ESPNs (Extended Stochastic Petri Nets [DTGN84]), etc. Ils permettent une description fonctionnelle et temporelle intégrée dans le même modèle, avec une différence au niveau de la distribution du temps d'exécution. Différents outils de réseaux de Petri sont apparus comme : SPNP [HTT00], TimeNET [ZFGH00], GreatSPN [Chi91] et WebSPN [BPST98]. Les Réseaux de Petri n'offrent ni la compositionalité ni l'abstraction.

Les Réseaux d'Automates Stochastiques (RAS [Pla84, D'A99, DKB97, Fou97]) sont un formalisme de haut niveau qui permet la modélisation de chaînes de Markov très grandes et très complexes d'une façon compacte et structurée. Ils ont été construits pour prendre en compte la taille de l'espace d'états lors de la conception d'un modèle. En effet, un système est modélisé à partir de plusieurs composants, qui évoluent souvent en parallèle (d'une façon indépendante) sauf en certaines actions de synchronisation, où les composants interagissent entre eux. Dans ce formalisme, le comportement d'un composant du système est représenté par un ensemble d'états qui constitue un automate (une chaîne de Markov avec un espace d'état raisonnable). Chaque automate est décrit par une matrice locale. L'ensemble de tous les automates ainsi constitué est représenté par une chaîne de Markov multidimensionnelle, dont les états sont ceux de l'espace produit, avec une matrice sous-jacente (appelée descripteur [Pla84]) obtenue en appliquant les opérateurs de l'algèbre tensorielle (ou de Kronecker) sur les matrices locales de chaque automate. La structure de générateur permet d'avoir un gain considérable de l'espace mémoire puisqu'elle évite le stockage de la matrice générateur tout entière. Des techniques de résolution efficaces [DQT98, Ben03] sont alors disponibles pour l'analyse quantitative. Mais, l'utilisation des composants indépendants reliés par l'intermédiaire des fonctions de synchronisations peut produire un espace d'états avec beaucoup d'états inaccessibles.

Dans le cadre de cette thèse, nous nous intéressons particulièrement aux algèbres de processus stochastiques (APS) [Hil96, Ber99, HHK02]. Les APS ne sont qu'une extension des algèbres des processus classiques par l'introduction de la temporisation des actions, ce qui va permettre d'intégrer l'analyse quantitative en plus de l'analyse qualitative, généralement réalisée à l'aide d'outils de model-checking [CGP99]. Le fait de cette extension, signifie qu'ils héritent de tous les avantages de l'ancien formalisme, que ce soit la construction structurée (ou hiérarchique) par la composition d'un modèle complexe à partir des ses petits composants qui interagissent entre eux, ou en faisant une abstraction tout en cachant certains détails du comportement du système. Par contre, les APS, tout comme les réseaux d'automates, n'ont pas la notion de flot, comme c'est le cas dans les réseaux des files d'attente et dans les réseaux de Petri (paquet ou jeton qui circule). Ce qui rend la modélisation des mécanismes de transmission des flots et de la gestion de la qualité de service, un peu plus délicate. Comme cela a été dit, la défense du choix des APS en tant que formalisme de modélisation n'est pas du domaine de cette thèse.

La principale difficulté lors de la construction d'un modèle consiste à déterminer le délai d'exécution de chaque action. Pratiquement, les seules informations qui sont disponibles sur ce délai sont, si on est chanceux, quelques échantillons pour calculer la valeur moyenne, l'écart type, etc. Les plupart des outils des langages de spécification formelle supposent que toutes les exécutions du système sont de type Markovien, donc sans mémoire, c'est-à-dire que l'état futur dépend uniquement de l'état présent, et non pas du passé. Cette restriction dans la représentation temporelle est essentielle afin de pouvoir évaluer les paramètres de performances en résolvant numériquement la chaîne de Markov. Ces chaînes (que ce soit à temps continu ou à temps discret) facilitent l'analyse des performances des modèles [Ste95, Fou97, DQT98], et elles sont particulièrement bien adaptées à l'étude des systèmes parallèles et distribués [SM91, VBO98].

Sûr que cette restriction est un handicap essentiel qui a des conséquences directes sur la précision des paramètres de performances, où l'approximation des délais d'exécution de certaines actions (par exemple Time-Out dans le protocole TCP) par une distribution sans mémoire est nécessaire, pour ne pas compliquer la résolution du modèle avec l'introduction des distributions avec mémoire du temps écoulé dans le passé. Plusieurs solutions à ce problème sont apparues jusqu'à aujourd'hui, que ce soit par l'approximation d'une distribution générale avec une distribution de phase-type [Neu81, Neu75] (états et transitions supplémentaires), ou bien une solution exacte dans certains cas particuliers (méthode de la variable supplémentaire [Ger00], chaîne de Markov semi-régénérative [Ger00], etc.). Nous ne rentrerons pas dans une discussion détaillée sur les différentes techniques existantes. Nous reviendrons sur une discussion un peu plus approfondie dans le chapitre 3, où nous ne sommes pas contenté de l'étude des systèmes Markoviens.

## 1.3 Objectifs de cette thèse

La problématique de cette thèse concerne l'évaluation des performances des systèmes réels, dont les actions ont des délais d'exécution générale. Nous nous intéressons plutôt à la conception et la spécification de nouveaux modèles de gestion de la qualité de service dans les réseaux ad hoc comme domaine d'application, ce qui permet de faciliter la construction d'un modèle de simulation. Ce nouveau type de réseau, qui viendra prendre parfois la place du réseau IP, devra faire face à une demande de ressources très importante liée à l'arrivée massive d'applications multimédias sur des terminaux mobiles. La qualité de service dans ce type de réseaux a attiré notre attention parce qu'elle présente des défis dans la description du temps, des probabilités et des priorités. Par ailleurs, ces réseaux sont mobiles, leur topologie n'est a priori absolument pas maîtrisée et il faut aussi prendre en compte les probabilités d'apparition, et de déplacement des différentes stations.

Pour atteindre une qualité optimale de tels réseaux, il faut affecter les ressources du réseau de la façon la plus judicieuse. Cette affectation doit se faire d'une manière dynamique en agissant sur les mécanismes d'attribution de ressources aux mobiles en fonction des nécessités en qualité de service (QdS). Nous nous concentrons sur le développement d'une nouvelle technique d'allocation de ressource efficace sous différentes distributions de charges dans le réseau, et dans différents contextes de mobilité. Ce vaste problème que nous traitons dans les réseaux ad hoc, où une qualité de service garantie à 100% est difficile à réaliser avec la mobilité, l'interférence, les fluctuations et l'amortissement du signal, ainsi que l'accès distribué au canal de communications, nécessite l'utilisation de plusieurs mécanismes parallèles et distribués, qui sont naturellement très complexes. Sûr que la mise en place d'un modèle de simulation, pour l'évaluation de performances du modèle proposé, n'est pas une tâche évidente en partant d'un organigramme avec une représentation graphique, contenant les composants du modèle et l'interaction entre eux. Pour cela, nous avons utilisé l'algèbre de processus stochastique pour la construction d'un modèle formel, utilisé pour mettre en place un modèle de simulation. Ce modèle algébrique permet de réaliser des analyses qualitatives et quantitatives sur un premier modèle abstrait du système en question.

Cependant la précision des résultats des analyses dépend des détails investis dans le modèle algébrique, le système est parfois compliqué et difficile à modéliser sans s'emparer de certains détails pour rendre l'analyse possible. Le principal problème est la taille de l'espace d'états du système, qui dépasse

souvent le million d'états, et qui nécessite énormément de temps et d'espace mémoire pour l'analyser. On parle du problème de l'explosion de l'espace d'états. Des techniques et des solutions adéquates sont déjà développées et doivent alors être utilisées pour pouvoir analyser le modèle. Par exemple, les techniques développées et utilisées dans le model-checking, visant à analyser certaines propriétés qualitatives des systèmes à grand espace d'états. Comme les diagrammes de décision binaire qui visent à rendre plus compacte la représentation d'états, et qui sont utilisés à la fois en vérification fonctionnelle [Bry92] et en évaluation de performances [HMKS00].

La restriction de la spécification temporelle aux distributions sans mémoire, laisse beaucoup de questions sur l'exactitude des résultats quantitatifs, d'où la nécessité d'un travail sur le formalisme de modélisation, pour pouvoir concevoir et analyser plus facilement et d'une manière plus précise et plus réelle les systèmes étudiés. Cela va nous permettre aussi d'apporter une contribution au domaine de la modélisation des systèmes temps réel, où la modélisation temporelle à l'aide des algèbres de processus stochastique est loin d'être mature.

Donc en récapitulant, l'objectif de cette thèse est double. C'est d'apporter un élément de réponse à l'expressivité temporelle des algèbres de processus stochastiques et à l'explosion de l'espace d'états d'une part, et la conception d'un nouveau mécanisme de gestion de la qualité de service dans les réseaux ad hoc d'autre part.

## 1.4 Plan de l'ouvrage

Cette thèse est organisée en deux parties principales. La première partie regroupe les chapitres 2 et 3. Cette partie se concentre sur les formalismes des algèbres de processus stochastiques, et elle est plutôt consacrée à la modélisation et à l'analyse des performances d'un système réel, ainsi que les nouveautés qu'on peut ramener à ce formalisme, que ce soit au niveau de la modélisation des actions dont les délais suivent une distribution générale, ou au niveau de l'agrégation de l'espace d'états, ainsi que la spécification au niveau algébrique des distributions phase-type et la recherche des paramètres d'un modèle.

La deuxième partie est consacrée à la conception, la spécification et l'analyse de nouveaux mécanismes de gestion de la qualité de service de bout en bout dans les réseaux ad hoc. Dans un premier temps, nous proposons un modèle adaptatif avec la dynamicité du réseau, pour fournir une QdS différenciée et proportionnelle de bout en bout, puis une extension de ce dernier pour fournir une qualité absolue. Nous analysons les performances de ces modèles par simulation sous différentes distributions de trafic et sous différents scénarios de mobilité.

Nous détaillons ici le contenu de ces parties.

### 1.4.1 Les algèbres de processus stochastiques

Le chapitre 2 décrit le contexte de travail, où nous présentons un état d'art sur les algèbres du processus stochastiques. Nous présentons les concepts de base des algèbres de processus classiques et stochastiques, puis nous effectuons un tour d'horizon des différents langages de modélisation algébriques couramment utilisés en évaluation de performances. Nous présentons ensuite une comparaison rapide entre ces différents langages, utilisés pour la modélisation d'un système complexe d'une manière compositionnelle, à partir de sous-systèmes qui interagissent pour accomplir certaines tâches. Nous en donnons ensuite la sémantique opérationnelle, qui permet de dériver un système de transitions à partir d'un modèle algébrique, dans un but de préparer le terrain pour traiter les modèles réels. Puis, nous abordons rapidement les méthodes de model-checking et les méthodes de récompenses, utilisées pour effectuer des analyses qualitatives et quantitatives sur le modèle en question. A la fin de ce chapitre, un modèle algébrique abstrait d'un routeur DiffServ est modélisé, et des analyses qualitatives et quantitatives sont réalisées sur ce modèle.

Le chapitre 3 est consacré à l'introduction des distributions générales dans les formalismes algébrique par le biais des distributions phase-type minimales. L'idée est d'obtenir une extension conservative pour

les algèbres de processus stochastiques en préservant les sémantiques déjà utilisées. Dans ce chapitre, nous mettons l'accent sur les méthodes et algorithmes d'approximation d'une distribution générale avec une distribution phase-type. Nous détaillons les différents algorithmes qui ont été proposés, et les différentes méthodes d'approximations que nous pouvons utiliser. Ensuite, une méthode d'agrégation basée sur les distributions phase type est proposée pour la réduction de l'espace d'états, et une spécification formelle de haut niveau est présentée pour éviter toute ambiguïté dans les modèles. Puis nous précisons l'apport de l'utilisation de la stratégie de présélection avec les distributions phase-type sans écarter la liberté de choix du concepteur.

### 1.4.2 Application à la conception des mécanismes de gestion de la QdS dans les réseaux ad hoc

Dans le chapitre 4, nous commençons par réaliser un état de l'art au sens large sur la qualité de service dans les réseaux ad hoc, et les travaux effectués dans ce domaine. Ensuite, nous traitons les questions fondamentales, que ce soit dans la stratégie de l'allocation de ressources, ou dans l'adaptation à la dynamicité du réseau et aux fluctuations de ressources, en proposant un nouveau mécanisme adaptatif de gestion d'une QdS proportionnelle de bout en bout entre les classes du réseau ad hoc. Puis nous abordons la modélisation algébrique d'une vision abstraite du mécanisme proposé, avant d'utiliser ce dernier pour la construction d'un modèle de simulation. Des analyses quantitatives sous différentes conditions du réseau ont été réalisées, et les résultats ont prouvé la capacité d'adaptation de notre modèle au changement des conditions du réseau.

Dans le chapitre 5, nous proposons une extension du premier modèle pour fournir une QdS absolue de bout en bout plutôt que relative, puis nous réalisons des études quantitatives à l'aide du modèle de simulation.

Pour terminer, le chapitre 6 dresse un bilan des travaux effectués dans cette thèse par rapport à l'état de l'art, et précise les travaux en cours, ainsi que les perspectives de travaux que nous envisageons pour donner suite à cette thèse.

# Algèbres de processus stochastiques

# Chapitre 2

# Les algèbres de processus stochastiques

## 2.1 Introduction

Les algèbres de processus stochastiques [HHK02] (APS en abréviation) sont un formalisme mathématique équipé d'un petit ensemble d'opérateurs puissants, qui permettent de modéliser un système complexe, et de réaliser une étude qualitative et quantitative sur le modèle en question. L'intérêt principal de la modélisation avec les algèbres de processus, c'est qu'elles permettent d'avoir une vision modulaire des composants du système étudié, et de modéliser ensuite les interactions entre ces différents composants, qui fonctionnent en parallèle et coopèrent afin d'accomplir une tâche commune. Différent techniques de vérification fonctionnelle (model-checking [CGP99, ACD93]), et différentes méthodes efficaces pour la résolution numérique de la matrice générateur infinitésimal, représentant la chaîne de Markov [Ste95, Ste94] sous jacente du modèle algébrique, sont alors disponibles à l'utilisation dans ce domaine. Ces techniques permettent de vérifier si la spécification réalisée satisfait les propriétés qualitatives et quantitatives exigées.

Ce chapitre présente les terminologies et les techniques de résolution qui seront utilisées tout au long de cette thèse. Les lecteurs familiers avec les algèbres de processus stochastiques et les notions de : variable aléatoire, distribution de probabilité, distribution sans mémoire, chaîne de Markov, processus stochastiques, model checking, etc., vont trouver utile la lecture de ce chapitre car nos notations sont différents de celles largement utilisées ailleurs.

Nous commençons par rappeler quelques caractéristiques des algèbres de processus stochastiques d'une façon informelle afin de comprendre l'outil de modélisation sans aucune préoccupation théorique, en présentant les syntaxes de trois langages de spécifications largement utilisées dans ce domaine : PEPA [Hil96], TIPP [HHM98, GHR93] et EMPA [Ber99, BG98]. Ensuite, nous donnons la sémantique du langage EMPA, considéré comme une extension de deux premiers pour économiser l'espace. Nous abordons rapidement la chaîne de Markov, qui est le wagon de notre étude pour l'évaluation quantitative, et les différentes méthodes de résolution numériques et les techniques utilisées pour calculer des indices de performances [Ber97a], ainsi que les techniques de model-checking (CTL et $\mu$-calcul) [CGP99, ACD93] utilisées pour analyser les aspects fonctionnels d'un modèle.

À la fin de ce chapitre, nous proposons un modèle algébrique abstrait d'un routeur DiffServ [BBC$^+$98] comme exemple d'application pour la modélisation avec les opérateurs algébriques, et nous exploitons les modèles logiques pour vérifier les propriétés fonctionnelles, ainsi que la méthode algébrique de récompenses pour évaluer les paramètres de performance.

## 2.2 Présentation informelle des Algèbres de processus stochastiques

Les algèbres de processus classiques [BPS01, Mil89, HOA85, BH01] sont un language de spécification formel (SDL [HS97]), utilisé pour spécifier les comportements des composants concurrents d'un système complexe. Ces composants communiquent dans un environnement collaboratif et distribué, où chacun peut effectuer des tâches en parallèle des autres composants [SB03a]. La spécification est réalisée

à l'aide d'un petit ensemble puissant d'opérateurs algébriques. L'opérateur de composition parallèle (qui permet de construire un modèle complexe à partir de ses composants) et l'opérateur d'abstraction (qui permet de cacher les détails internes aux yeux d'un utilisateur à l'extérieur), fournissent un fort pouvoir d'expression.

Plusieurs langages sont apparus dans les dernières années, dont les plus connus sont : Calculus of Communicating Systems (CCS [Mil89]), Communicating Sequential Process (CSP [HOA85]), Algebra of Communicating Processes (ACP [BK85]) et Language of Temporal Ordering Specifications (LOTOS [BB87]).

Ces formalismes permettent d'effectuer seulement l'analyse de la partie fonctionnelle du modèle en question, car ils négligent toute notion de temps quantitatif. Par conséquent, l'évaluation des paramètres de performances d'un modèle, construit à l'aide des algèbre de processus classiques, est souvent réalisée après la conception et l'implémentation d'un modèle [BDG98]. Mais si les performances du système sont jugées médiocres, un nouveau modèle doit être conçu, entraînant une perte de temps et une augmentation du coût des projets. Pour faire face à ces problèmes, l'analyse quantitative est souvent réalisée sur un autre modèle conçu à l'aide d'outils correspondants. Mais la construction de deux modèles indépendants (souvent gérés par deux groupes différents) peut provoquer des problèmes de cohérence et de compatibilité entre les modèles.

Il est donc irréaliste d'exiger que chaque événement d'un système s'exécute d'une manière atomique. Pour pallier ce problème, les algèbres de processus stochastiques associent une variable aléatoire à chaque action, pour représenter son délai d'exécution, afin que ce formalisme réponde aux besoins de l'analyse quantitative. Cette variable est limitée aux distributions sans mémoires pour transformer le diagramme de transitions sous jacent du modèle algébrique en une chaîne de Markov. Ceci simplifie la dérivation des paramètres de performances requis, où différentes méthodes des algèbres linéaires peuvent être exploitées pour la résolution. Aujourd'hui, cette extension est largement utilisée pour la conception, la modélisation, l'analyse qualitative et quantitative de différents types de systèmes dans différents domaines, comme par exemple : la spécification d'une machine industrielle ainsi que les protocoles de communication, le contrôle du trafic aérien et ferroviaire, etc. Plusieurs langages d'algèbres de processus stochastiques sont apparus ces dernières années, comme PEPA [Hil96], TIPP [HHM98] et EMPA [BG98]. Ces langages offrent plusieurs avantages qui sont retenus des algèbres de processus classiques, parmi lesquels nous citons :

- La construction par composition et abstraction apportant des avantages de construction hiérarchique et modulaire pour la spécification d'un modèle, comme étant comme un ensemble des composants qui interagissent entre eux.
- L'utilisation de différentes techniques et lois algébriques pour réduire l'espace d'états lors de l'analyse. Ces lois sont conçues pour s'assurer que le modèle résultant est équivalent à l'original.
- L'expressivité qui permet de décrire d'une manière simple et lisible des comportements qui peuvent être complexes, incluant des notions de parallélisme, de synchronisation, etc.

Nous présentons tout d'abord la notion d'action à la base de la spécification en APS. Comme dans le formalisme classique, les composants sont décrits algébriquement en spécifiant les tâches (actions) de leurs composants. Les composants d'un système sont représentés par des processus, et leurs activités sont spécifiées par des actions, dont chacune est associée à une variable aléatoire qui représente son délai. Chaque fois qu'un processus peut exécuter une action, un échantillon de la fonction de distribution de probabilité associée à l'action, est utilisé pour représenter le délai de son exécution. Un modèle algébrique est constitué tout simplement d'un ensemble de processus liés par des opérateurs algébriques. Ces processus exécutent des actions après des délais aléatoires.

Dans les 3 sous sections, nous décrivons brièvement la syntaxe de PEPA, TIPP et EMPA, puis nous faisons une comparaison entre ces différents langages. Reste à noter, que le langage « EMPA » est le langage le plus représentatif pour nos études car il a été construit récemment en se basant sur PEPA et TIPP. Pour cela, nous nous contentons de donner la sémantique de ce langage qui intègre celles des autres. Son outil contient différents aspects sophistiqués par rapports aux autres outils comme : une méthode algébrique de calcul des indices de performance pour la dérivation automatique de ces paramètres, la ré-

solution par simulation, la modélisation avec la technique de passage des valeurs utilisée dans E-LOTOS, les différentes techniques d'agrégation, les différents outils de model-checking, etc. Mais tous ça n'empêche pas d'utiliser les autres langages, vu leur similitude et l'existence d'outils complémentaires pour la vérification qualitative et l'analyse quantitative.

## 2.3 Présentation formelle des Algèbres de processus stochastiques

Les bases des algèbres de processus ayant été rappelées, nous nous intéressons maintenant aux formalismes stochastiques et nous commençons par une présentation formelle.

### 2.3.1 La syntaxe de PEPA

PEPA (Performance Evaluation Process Algebra) [Hil96] est une algèbre de processus stochastiques dont la variable temporelle associée à chaque action est purement exponentielle. Ce qui résulte en une chaîne de Markov facile à analyser en exploitant les techniques standards. Dans PEPA, comme dans tous les formalismes algébriques Markoviens, chaque action s'exécute après un délai exponentiel. L'activité sera représentée par le couple $(a, \lambda)$, où $a$ est le nom de l'action, et $\lambda | \lambda \in \mathbb{R}^+$ est le taux de la fonction de distribution exponentielle $F(t)$ associée à cet action.

$$F(t) = Pr(délai < t) = 1 - e^{-\lambda t} \tag{2.1}$$

L'ensemble des règles définissant la bonne construction d'une expression, ou bien la syntaxe de PEPA est donnée par l'expression suivante :

$$P = (a, \lambda).P \mid P_1 + P_2 \mid P_1 \underset{S}{\bowtie} P_2 \mid P/L \mid A$$

avec :

$(a, \lambda).P$ est un opérateur de séquence qui signifie une exécution séquentielle de l'action $a$, suivie par le processus $P$. Le délai d'exécution de l'action $a$ est choisi d'une manière aléatoire selon la fonction de distribution exponentielle avec un taux $\lambda$. Le taux d'une action « passive » est noté par le caractère spécial $\top$ pour signifier que cette action s'exécute en synchronisation avec une action « active » (ou temporisée) du même type.

$P_1 + P_2$ est l'opérateur de choix non déterministe, qui permet un choix compétitif entre deux comportements possibles. Le choix compétitif signifie que l'exécution de l'un de ces deux processus, écarte l'exécution de l'autre. Par exemple, Soit $P_1 = (a, \lambda).0$ et $P_2 = (b, \mu).0$, la composition à l'aide de l'opérateur du choix $(P_1 + P_2)$ conduit à observer, soit l'action $a$ avant de s'arrêter, soit l'action $b$ avant de s'arrêter également. La résolution du choix entre ces deux processus est déterminée par la politique de compétition (race condition), où l'action dont le délai d'exécution est le plus petit, gagnera la course et sera considérée comme étant la plus rapide. La probabilité que deux actions soient exécutées au même instant est nulle avec une distribution de probabilité à temps continu, ce qui signifie que la résolution du choix est unique, et qu'un seul processus sera engagé alors que l'autre sera interrompu.

$P_1 \underset{S}{\bowtie} P_2$ est l'opérateur de composition parallèle qui signifie que les deux processus $P_1$ et $P_2$ évoluent en parallèle, d'une manière indépendante pour l'exécution d'actions qui ne sont pas dans la liste $S$ d'un côté, et en se synchronisant sur toutes les actions qui sont dans la liste $S$ de l'autre côté. Cet opérateur de parallélisme permet la construction hiérarchique d'un modèle complexe à partir de petits composants, qui interagissent et se synchronisent sur certaines activités pour réaliser une tâche. La stratégie de synchronisation utilisée dans PEPA est connue sous le nom de la synchronisation patiente, car le taux d'exécution des actions en synchronisation est celui de l'action la plus lente. Lorsque la liste de coopération $S$ est vide, l'opérateur de composition parallèle est noté par $P_1 \parallel P_2$.

$P/L$ est l'opérateur d'abstraction qui transforme toutes les actions listées dans $L$, en actions internes de type $\tau$ (comme l'action $i$ dans LOTOS [BB87]). Ces actions deviennent invisibles du point de vue d'un observateur à l'extérieur du système, et inaccessibles pour les autres composants, ainsi elles ne peuvent pas participer dans la synchronisation de $P$ avec d'autres processus.

$A$ est un processus destiné à l'instanciation et à l'utilisation de la récursivité. Il permet d'assigner des noms aux composants et de décrire un comportement récursif des processus.

L'outil qui supporte le langage PEPA (« PEPA Workbench » [GH03]), est capable de dériver automatiquement à partir d'une description algébrique, la chaîne de Markov et la matrice générateur infinitésimal en plusieurs formats (MatLab, Maple, etc.) pour faciliter l'analyse à l'aide des méthodes et outils de résolution linéaire.

### 2.3.2 La syntaxe de TIPP

L'algèbre de processus stochastiques TIPP [GHR93] est très semblable à PEPA avec quelques différences mineures dans la stratégie de synchronisation entre les activités en parallèle, ainsi que le taux des actions passives. Le taux des actions passives est représenté par 1 à la place de $\top$ en PEPA, et la stratégie de synchronisation est le produit des taux d'exécution des actions en coopération. La syntaxe de TIPP est donnée par l'expression suivante :

$$P = Stop \mid (a, \lambda).P \mid P_1 + P_2 \mid P_1 |[S]| P_2 \mid P \backslash L \mid Rec\ X : P \mid A$$

L'opérateur « *Stop* » sert seulement à marquer la terminaison d'un processus sans pouvoir exécuter aucune action. Les opérateurs : de séquence « . », de choix « + », de composition parallèle « $|[S]|$ » et d'abstraction « $\backslash$ » sont les mêmes que ceux vu dans PEPA, et ils ont les mêmes fonctionnalités. L'opérateur $Rec\ X\ :\ P$ est utilisé pour exprimer un comportement récursif quand $X$ apparaît dans le processus $P$, et l'opérateur constant $A$ pour l'instanciation des processus.

TIPP est implémenté à travers l'outil TIPPtool [HM96] qui est fortement influencé par le langage de spécification LOTOS. Cet outil est capable de dériver automatiquement la chaîne de Markov à partir de la description algébrique, et de calculer les vecteurs de distribution de probabilité dans les deux régimes stationnaire et transitoire pour une chaîne qui ne dépasse pas quelques dizaines de milliers d'états.

### 2.3.3 La syntaxe de EMPA

EMPA (Extended Markovian Process Algebra) [BG98] a été basé sur les langages existants (PEPA et TIPP) et il est considéré aujourd'hui comme une extension de ces deux langages, où les délais d'exécution des actions ne sont plus purement Markoviens. D'une façon similaire aux réseaux de Petri stochastiques généralisés (GSPN) [CMBC93, MBD98], des actions immédiates avec des priorités et des probabilités d'exécution viennent s'ajouter à ce formalisme. La syntaxe d'EMPA est donnée par l'expression suivante :

$$P = 0 \mid (\alpha, \lambda).P \mid (\alpha, \infty_{p,m}).P \mid P_1 + P_2 \mid P_1 ||_S P_2 \mid P/C \mid P \backslash L \mid P[\theta] \mid A$$

Dans l'expression précédente : « 0 » dénote l'opérateur de l'inaction d'un processus. L'opérateur de séquence « . », la fonction de renomage « $\theta$ », l'opérateur d'abstraction « $\backslash$ », l'opérateur de choix « + », l'opérateur de composition parallèle « $||$ » et l'opérateur constant « $A$ » sont ceux vus dans les deux langages précédents. L'opérateur de restriction « $/$ » empêche l'exécution des actions listées dans $C$.

En EMPA, le taux des actions passives est noté par « $*$ », et le taux des actions immédiates est représenté par $\infty$ juste pour pour représenter symboliquement la fausse intuition sur les actions immédiates, comme étant cas particulier de la distribution exponentielle lorsque le taux $\lambda = \infty$.

$$F(t) = Pr(délai < t) = 1 - e^{-\infty.t} = 1 \qquad \forall \text{t} \tag{2.2}$$

Une action immédiate est définie en EMPA par $(a, \infty_{p,m}).P$, où une légère modification de celle de l'exponentielle $(\alpha, \lambda).P$, est introduite pour prendre en compte la résolution de conflit lors de l'exécution de deux actions immédiates au même instant (stratégie de présélection) :

- Une priorité $p$ qui détermine comment agir en cas de conflit entre deux actions immédiates activées simultanément. C'est un entier strictement positif, avec 1 la priorité minimale et $\infty$ la priorité maximale.

– Une masse (ou un poids) de probabilité de transition $m \in [0, 1]$ pour résoudre le conflit entre deux actions ayant la même priorité. Par exemple, le choix dans $(a, \infty_{p,m}).P + (b, \infty_{p,m'}).Q$ est résolu selon la probabilité d'exécution associée à chaque action ($Pr(a) = m/(m + m')$ et $Pr(b) = m'/(m + m')$).

La continuité de la fonction de distribution exponentielle garantit l'exclusivité de l'exécution d'une seule action à un instant donné, et permet l'abstraction de la spécification de ces deux paramètres avec la distribution exponentielle. Pour cela, quand $n$ actions avec des taux d'exécution $\lambda_1, \lambda_2, \ldots, \lambda_n$ sont actives dans un état $s_k$, la probabilité d'exécution de chacune d'entre elles est donnée par :

$$Pr(a_i) = \frac{\lambda_i}{\sum_{j=1}^{n} \lambda_j}$$

Comme dans les autres algèbres de processus Markoviens, la stratégie de course (race condition) est utilisée pour la résolution des conflits entre plusieurs actions actives en même temps, où l'action qui sera exécutée est celle dont la variable aléatoire associée est la plus petite. Les actions immédiates sont alors urgentes et prioritaires par rapport aux actions exponentielles, si elles sont activées simultanément et ne sont pas bloquées par l'environnement (théorème de la progression maximale [BH01]).

**Exemple d'application**

Dans cette section, nous présentons un exemple simple illustrant l'utilisation du formalisme EMPA pour la spécification de la file d'attente $M/M/1/k$ (*cf.* figure 2.1). Le système est modélisé par la composition parallèle de 2 processus, le premier processus $Poisson$ est utilisé pour la génération des paquets, et le deuxième processus $Queue$ représente la file d'attente et le serveur. L'action $arrivée$ représente l'arrivée d'un paquet de l'extérieur vers le système avec un taux $\lambda$. Le gestionnaire de la file sert les clients suivant un taux exponentiel $\mu$ associé à l'action $service$. Mais, comme la file a une capacité finie, les paquets sont perdus si la file est pleine.

$$\begin{aligned}
SYS &\stackrel{def}{=} Poisson ||_S Queue \\
S &\stackrel{def}{=} \{accepté, refusé\} \\
Poisson &\stackrel{def}{=} (arrivée, \lambda).((accepté, *).Poisson + (refusé, *).Poisson) \\
Queue_0 &\stackrel{def}{=} (accepté, \infty_{1,1}).Queue_1 \\
Queue_i &\stackrel{def}{=} (accepté, \infty_{1,1}).Queue_{i+1} + (service, \mu).Queue_{i-1} \quad pour\ 1 \leqslant i \leqslant k-1 \\
Queue_k &\stackrel{def}{=} (refusé, \infty_{1,1}).Queue_k + (service, \mu).Queue_{k-1}
\end{aligned}$$

Figure 2.1 – Modèle algébrique de la file d'attente $M/M/1/k$.

## 2.4 Comparaison entre les stratégies de synchronisation

La différence principale entre les 3 langages (PEPA, TIPP et EMPA) réside dans la stratégie de synchronisation entre les actions de différents composants du modèle [BH01]. Pour illustrer cette différence, nous prenons l'expression suivante :

$$(a, \lambda).P_1 || a || (a, \mu).P_2 = (a, \lambda \star \mu).(B || a || C)$$

En s'interrogeant sur le délai d'exécution de l'action de synchronisation $a$, nous utilisons la fonction « $\star$ » (qu'on va l'appeler loi de coopération) pour représenter le taux résultant pour rendre les deux

termes dans l'équation précédente égaux, nous allons présenter les différentes solutions adoptées par ces langages pour la fonction « $\star$ », afin de déterminer le délai d'exécution de l'action $a$.

Intuitivement, nous pouvons penser à différents opérateurs pour la loi de coopération, comme par exemple : le maximum de deux distributions, le minimum, la moyenne, la convolution, etc. Mais les lois algébriques imposent des restrictions sur le choix de la loi de coopération comme le montrent les équations 2.4 et 2.5.

Pour chercher les contraintes qui permettent de déterminer la loi de coopération, nous considérons l'interaction entre l'opérateur de choix et celui de la synchronisation dans l'expression suivante :

$$((a, X_{F_1}).P_1 + (c, Y_{F_2}).P_2) \,||a||(a, Z_{F_3}).P_3 \tag{2.3}$$

D'un côté, en appliquant la condition de compétition sur l'expression précédente, nous obtenons :

$$\begin{gathered}((a, X_{F_1}).P_1 + (c, Y_{F_2}).P_2) \,||a||(a, Z_{F_3}).P_3 = \\ ((a, min(X_{F_1}, Y_{F_2}) \star Z_{F_3}).(P_1 \oplus_{Pr(X_{F_1} > Y_{F_2})} P_2)||a||P_3\end{gathered} \tag{2.4}$$

avec :

$$P_1 \oplus_{Pr(X_{F_1} > Y_{F_2})} P_2 = Pr(X_{F_1} < Y_{F_2}).P_1 + Pr(X_{F_1} > Y_{F_2}).P_2$$

De l'autre côté, en utilisant la loi de l'associativité de l'opérateur de choix par rapport à celui de la synchronisation, nous obtenons :

$$\begin{gathered}((a, X_{F_1}).P_1 + (c, Y_{F_2}).P_2) \,||a||(a, Z_{F_3}).P_3 = \\ (a, X_{F_1}).P_1||a||(a, Z_{F_3}).P_3 + (c, Y_{F_2}).P_2||a||(a, Z_{F_3}).P_3 = \\ ((a, X_{F_1} \star Z_{F_3}).(P_1||a||P_3) + (a, Y_{F_2} \star Z_{F_3})(P_2||a||P_3)) = \\ (a, min(X_{F_1} \star Z_{F_3}, Y_{F_2} \star Z_{F_3})).\left((P_1||a||P_3) \oplus_{Pr(X_{F_1} \star Z_{F_3} > Y_{F_2} \star Z_{F_3})} (P_2||a||P_3)\right)\end{gathered} \tag{2.5}$$

Mais comme les deux équations 2.4 et 2.5 sont égales, nous pouvons constater les deux restrictions suivantes :

$$min(X_{F_1}, Y_{F_2}) \star Z_{F_3} = min(X_{F_1} \star Z_{F_3}, Y_{F_2} \star Z_{F_3}) \tag{2.6}$$

$$Pr(X_{F_1} > Y_{F_2}) = Pr(X_{F_1} \star Z_{F_3} > Y_{F_2} \star Z_{F_3}) \tag{2.7}$$

Face à ces restrictions (données par les équations 2.6 et 2.7) apparaîtra la différence la plus significative entre ces trois algèbres des processus Markoviens. Le langage PEPA rejette la formule 2.5 et choisit le taux en se basant seulement sur l'idée de la synchronisation patiente (Last To Finish – LTF), où le minimum des deux variables aléatoires distribuées exponentiellement $min(X_{F_\lambda}, Y_{F_\mu})$, est adopté comme solution pour l'opérateur $\star$, afin d'exprimer que le taux de synchronisation entre deux actions est déterminé par l'action la plus lente, ce qui est basée sur une solution tout à fait logique.

$$\begin{gathered}\left((a, X_{F_\lambda}).P_1 + (a, Y_{F_\mu}).P_2\right) ||a||(a, Z_{F_\delta}).P_3 = \\ \left((a, T_{F_{\lambda+\mu}}).(P_1 \oplus_{Pr(X_{F_\lambda} > Y_{F_\mu})} P_2)\right) ||a||(a, Z_{F_\delta}).P_3 = \\ (a, T_{F_{\lambda+\mu}} \star Z_{F_\delta}).\left((P_1||a||.P_3) \oplus_{Pr(X_{F_\lambda} > Y_{F_\mu})} (P_2||a||P_3)\right) = \\ \left(a, (T_{F_{(\lambda+\mu)\times Pr(X_{F_\lambda} < Y_{F_\mu})}} \star Z_{F_\delta})\right).(P_1||a||.P_3) + \left(a, (T_{F_{(\lambda+\mu)\times Pr(X_{F_\lambda} > Y_{F_\mu})}} \star Z_{F_\delta})\right).(P_2||a||P_3)\end{gathered}$$

Sachant que :

$$min(X_{F_\lambda}, Y_{F_\mu}) = T_{F_{\lambda+\mu}}$$

et

$$Pr(X_{F_\lambda} < Y_{F_\mu}) = \frac{\lambda}{\lambda + \mu}$$

*Démonstration.* Le minimum de deux fonctions exponentiellement distribuée avec des taux $\lambda$ et $\mu$ est exponentiellement distribuée avec un taux égal à la somme de deux taux :

$$\begin{aligned} Pr\{min(X_\lambda, Y_\mu) \leqslant t\} &= Pr\{X_\lambda \leqslant t \vee Y_\mu \leqslant t\} \\ &= 1 - Pr\{X_\lambda > t \wedge Y_\mu > t\} \\ &= 1 - (1 - Pr\{X_\lambda \leqslant t\}).(1 - Pr\{Y_\mu \leqslant t\}) \\ &= 1 - \left(1 - (1 - e^{-\lambda.t})\right).\left(1 - (1 - e^{-\mu.t})\right) \\ &= 1 - e^{-(\lambda+\mu).t} \end{aligned}$$

La probabilité d'exécution de l'action $a$ dans l'expression $(a, X_{F_\lambda}).0 + (b, Y_{F_\mu}).0$ est donnée par : $Pr(X_\lambda < Y_\mu) = \frac{\lambda}{\lambda+\mu}$

$$\begin{aligned} Pr\{X_\lambda < X_\mu\} &= \int_0^\infty Pr\{X_\mu > t\} d(Pr\{X_\lambda \leqslant t\}) = \int_0^\infty e^{-\mu.t}.\lambda.e^{-\lambda.t} dt \\ &= \int_0^\infty \lambda.e^{-(\lambda+\mu).t} dt = \left[-\frac{\lambda}{\lambda+\mu}.e^{-(\lambda+\mu).t}\right]_0^\infty \\ &= \frac{\lambda}{\lambda+\mu} \end{aligned}$$

■

En substituant la variable aléatoire par le taux de la fonction correspondante pour simplifier la lecture des formules, on aura la représentation symbolique suivante :

$$\left(a, \frac{\lambda}{\lambda+\mu} \times ((\lambda+\mu) \star \delta)\right).(P_1||a||.P_3) + \left(a, \frac{\mu}{\lambda+\mu} \times ((\lambda+\mu) \star \delta)\right).(P_2||a||P_3)$$

avec :

$$(\lambda+\mu) \star \delta = min(\lambda+\mu, \delta)$$

$$\begin{array}{lll} 1^{ière} \text{ cas :} \quad min(\lambda+\mu, \delta) = \lambda+\mu & \Rightarrow & (a,\lambda).(P_1||a||.P_3) + (a,\mu).(P_2||a||P_3) \\ 2^{ième} \text{ cas :} \quad min(\lambda+\mu, \delta) = \delta & \Rightarrow & (a, \frac{\lambda}{\lambda+\mu} \times \delta).(P_1||a||.P_3) + (a, \frac{\mu}{\lambda+\mu} \times \delta).(P_2||a||P_3) \end{array}$$

Contrairement à la solution logique de PEPA, TIPP adopte une autre solution par intuition pour la loi de coopération $\star$, qui est la multiplication ordinaire de deux taux. Mais il n'y a aucune interprétation stochastique pertinente et utile pour ce choix autre qu'une simplification mathématique, comme le montrent les équations (2.8) et (2.9), obtenues à partir des simplifications des équations (2.6) et (2.7) avec une distribution exponentielle :

$$(\lambda+\mu) \star \delta = (\lambda \star \delta) + (\mu \star \delta) \tag{2.8}$$

$$\frac{\lambda}{\lambda+\mu} = \frac{\lambda \star \delta}{(\lambda+\mu) \star \delta} \tag{2.9}$$

Cette solution intuitive (multiplication) pour la fonction $\star$ préserve l'égalité dans les équations (2.8) et (2.9). Ce choix de la multiplication porte beaucoup d'obscurités autour de l'interprétation architecturale et opérationnelle de la synchronisation entre les composants du modèle.

Finalement, EMPA adopte le modèle maître/esclave ou client/serveur pour la loi de coopération, où le serveur détermine le taux de service et le client joue un rôle passif. C'est-à-dire que les clients peuvent envoyer des requêtes en utilisant une action bien définie, et le serveur seul peut accepter ces requêtes quand il peut les traiter. Sachant que le taux d'une action passive est $\infty$ (action prête à l'exécution dès

que possible), le taux de synchronisation est le taux minimal des actions en considération (l'action la plus lente). Ce choix n'apporte aucune restriction à la généralité de ce formalisme.

$$\lambda \star \infty = min(\lambda, \infty) \tag{2.10}$$

$$(a, \lambda).P_1 + (a, \mu).P_2||a||(a, \infty).P_3 = (a, \lambda).(P_1||a||P_3) + (a, \mu).(P_2||a||P_3) \tag{2.11}$$

Dans le reste de ce manuscrit, nous utilisons le langage EMPA pour modéliser un système et pour réaliser une analyse qualitative et quantitative, tout en profitant de la supériorité des techniques d'analyse offertes par son outil TwoTowers [BG98]. Ce dernier a été récemment augmenté par l'intégration d'outils existants pour les analyses qualitatives et quantitatives, sans la nécessité de convertir le format du diagramme de transitions obtenu, en vue de la résolution.

## 2.5 Sémantique opérationnelle

La sémantique d'une expression est l'ensemble des règles permettant d'associer un sens à cette expression. La sémantique des opérateurs, ou bien la signification d'une expression algébrique, est définie en terme de diagramme de transition entre les états du système. Ces transitions sont définies en utilisant la sémantique opérationnelle structurée (ou SOS [Plo81]) de Plotkin, donnée par l'expression suivante :

$$\frac{Prémisses}{Conclusion} \qquad Condition,$$

Cette expression veut dire que si la $Condition$ est vraie, alors $Prémisses \Rightarrow Conclusion$. L'application de ces règles de sémantique (donné dans le tableau 2.1 et détaillée dans [BH01]) sur un modèle, transforme ses processus en un diagramme de transitions. Ce diagramme est caractérisé par un un ensemble d'états et un un ensemble de transitions étiquetées avec le couple (nom de l'événement, délai de l'exécution) comme le montre la figure 2.2, où les nœuds du graphe correspondent aux états, et les arcs aux transitions. L'exécution d'un événement après l'expiration de son délai d'exécution, déclenche une transition qui va changer l'état actuel du diagramme. Par exemple, la sémantique de l'opérateur de séquence est donnée par :

$$(a, X).P \xrightarrow{a,X} P$$

Intuitivement, $(a, X).P \xrightarrow{(a,X)} P$ signifie que le système, initialement dans l'état $s_0 = (a, X).P$, peut exécuter l'action $a$ après l'expiration de son temporisateur $X$, ce qui déclenche un transition et entraîne un changement d'état en passant à l'état $s_1 = P$. Un nouvel état est donc associé à une expression syntaxique obtenue après l'exécution d'une action.

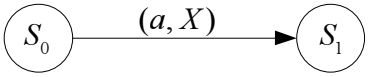



Figure 2.2 – Diagramme de transition de l'opérateur de séquence.

En appliquant les sémantiques au modèle algébrique, nous pouvons dériver automatiquement le diagramme de transitions, qui sera utilisé d'une façon similaire au graphe d'accessibilité sous jacent d'un modèle du réseaux de Petri stochastiques généralisé (GSPN). Ce diagramme de transitions contient en effet des informations fonctionnelles et temporelles. Le diagramme de transitions fonctionnelle est obtenu en supprimant le taux d'exécution des actions de l'étiquette des transitions. Ce dernier est utilisé pour vérifier les propriétés fonctionnelles en exploitant les techniques de model-checking (LTL, CTL, $\mu$-calcul, etc.).

Le diagramme de transitions est très proche de la chaîne de Markov, où les noms des actions s'ajoutent sur l'étiquette des transitions de la chaîne. La suppression de ces informations fonctionnelles réduit ce diagramme en une chaîne semi Markovienne généralisé (GSMP). Ceci est dû à la coexistence des

1. Opérateur de séquence :

$$\frac{}{(a,X).P \xrightarrow{(a,X)} P}$$

2. Opérateur de choix :

$$\frac{(a,X).P \xrightarrow{(a,X)} P \quad \wedge \quad (b,Y).Q \xrightarrow{(b,Y)} Q}{P(X<Y) : (a,X).P + (b,Y).Q \xrightarrow{(a,X)} P} \qquad \frac{(a,X).P \xrightarrow{(a,X)} P \quad \wedge \quad (b,Y).Q \xrightarrow{(b,Y)} Q}{P(X>Y) : (a,X).P + (b,Y).Q \xrightarrow{(b,Y)} Q}$$

3. Opérateur de composition parallèle :

$$\frac{(a,X).P \xrightarrow{(a,X)} P \quad \wedge \quad (b,Y).Q \xrightarrow{(b,Y)} Q}{prob(X<Y) : (a,X).P||_S(b,Y).Q \xrightarrow{(a,X)} P||_S(b,Y).Q} \quad a \wedge b \notin S$$

$$\frac{(a,X).P \xrightarrow{(a,X)} P \quad \wedge \quad (b,Y).Q \xrightarrow{(b,Y)} Q}{prob(X>Y) : (a,X).P||_S(b,Y).Q \xrightarrow{(b,Y)} (a,X).P||_S Q} \quad a \wedge b \notin S$$

$$\frac{(a,X).P \xrightarrow{(a,X)} P \quad (a,Y).Q \xrightarrow{(a,Y)} Q}{(a,X).P||_S(a,Y).Q \xrightarrow{(a,\max(X,Y))} P||_S Q} \quad a \in S$$

4. Opérateur d'abstraction :

$$\frac{P \xrightarrow{a,X} P'}{P/L \xrightarrow{\tau,X} P'} \; a \in L \quad \wedge \quad \frac{P \xrightarrow{a,X} P'}{P/L \xrightarrow{a,X} P'} \; a \notin L$$

5. Opérateur de re-nomination :

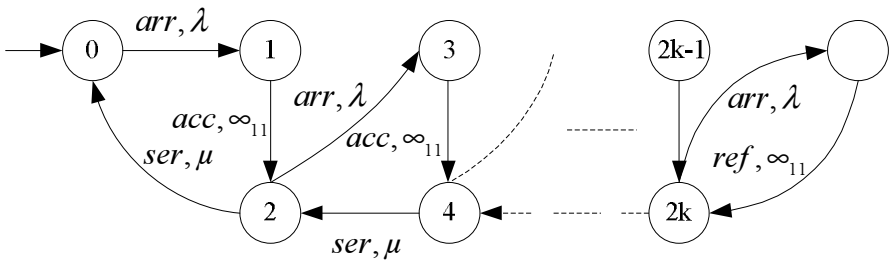
$$\frac{P \xrightarrow{a,X} P'}{P\,[\phi] \xrightarrow{\phi(a),X} P'\,[\phi]}$$

Tableau 2.1 – Sémantique Opérationnelle Structurée (SOS).

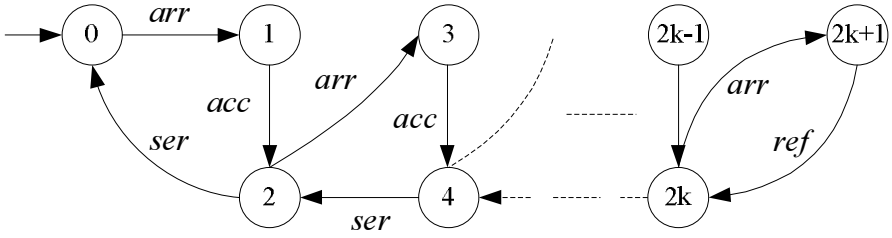
états évanescents qui déclenchent au moins une transition immédiate, avec d'autres tangibles qui ne déclenchent que des transitions exponentielles. La transformation de cette chaîne généralisée (GSMP) en une Chaîne de Markov (CM) [BDG98] consiste à supprimer les états évanescents et à distribuer les probabilités de transitions quittant ces états sur les transitions entrantes comme dans les GSPN [MBD98]. Cette suppression est sans influence sur l'évaluation de performance car le temps de séjour dans les états évanescents est nul. Les méthodes d'algèbre linéaire sont alors utilisées pour analyser le générateur infinitésimal, et dériver les vecteurs des distributions de probabilité dans les deux régimes (stationnaires et transitoires). Les propriétés qualitatives et quantitatives peuvent être exprimées aussi directement en utilisant des formules logiques de CSL [ASSB00] (Continuous Stochastic Logic). Par exemple, le système devrait exécuter une action avant un timeout $t_{out}$ avec un pourcentage de 92%.

**Exemple**

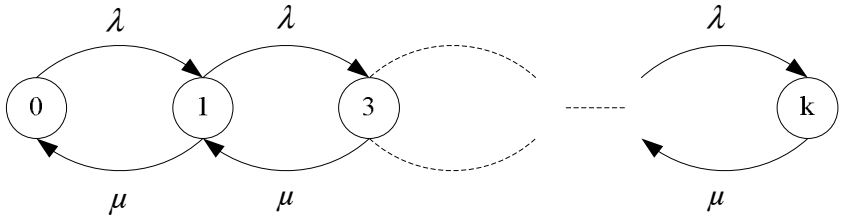
Nous reprenons l'exemple algébrique de la file $M/M/1/k$ pour dériver les 3 diagrammes de transitions, qui sont donnés dans la figure 2.3.




(a) Diagramme de transition intégré.




(b) Diagramme fonctionnel.




(c) Diagramme de performance $MC$.

Figure 2.3 – Diagrammes de transition du modèle de la file $M/M/1/k$.

## 2.6 Chaînes de Markov

### 2.6.1 Définition

Un processus stochastique $X = \{X_t, t \in T\}$ est une famille de variables aléatoires représentant l'observation de $X$ à l'instant $t$. Les valeurs prises par $X_t$ sont les états et $X$ représente l'espace d'état. Si la variable représentant le temps est entière, $t \in \mathbb{N}$, alors on parle d'un processus à temps discret. En revanche, si $t \in \mathbb{R}^+$, le processus est dit à temps continu. Le processus $X$ est une chaîne de Markov si et seulement si, pour $s, t \geqslant 0$ :

$$Prob(X_{t+s} = S_j | X_t = S_i, X_{t-\Delta} = S_{i-1}, \ldots, X_0 = S_0) = Prob(X_{t+s} = S_j | X_t = S_i)$$

La connaissance de l'état présent rend les informations sur le passé non pertinentes pour prévoir l'état futur (propriété Markovienne). Si la probabilité $Prob(X_{t+s} = S_j | X_t = S_i)$ est indépendante de $t$, alors la chaîne est dite homogène :

$$P_{i,j} = Prob(X_{t+s} = S_j | X_t = S_i) = Prob(X_s = S_j | X_0 = S_i)$$

En conséquence, la probabilité que la chaîne visite l'état $j$ à l'instant $t + s$ sachant qu'elle était dans l'état $i$ à l'instant $t$, dépend seulement des deux états $i$ et $j$ et de la différence de temps $s$. Elle est indépendante du temps de séjour dans l'état $i$. Par conséquent, pour satisfaire la propriété Markovienne, le temps de séjour dans un état doit posséder l'absence de mémoire à tout instant, où le temps restant à passer dans l'état actuel doit être indépendant du temps déjà passé dans cet état. Les seules distributions sans mémoires sont : la distribution géométrique parmi les distributions à temps discret et la distribution exponentielle parmi ceux à temps continu.

*Démonstration.* Soit $D$ une variable aléatoire représentant le délai d'exécution d'une action et distribué selon :

1. La fonction de distribution de la probabilité géométrique est $F(t) = Pr(D < t) = 1 - (1-p)^{\frac{t}{\theta}}$ où $p = 1 - q$ et $t \in \mathbb{N}$ :

$$Pr(D > t) = \sum_{i=t+1}^{\infty} Pr(i) = \sum_{i=t+1}^{\infty} p(1-p)^{i-1} = pq^t \sum_{i=0}^{\infty} q^i = pq^t \frac{1}{1-q} = q^t$$

$$Pr(D > t+s | D > t) = \frac{Pr(D > t+s, D > t)}{Pr(D > t)} = \frac{Pr(D > t+s)}{Pr(D > t)} = \frac{1-(1-(1-p)^{t+s})}{1-(1-(1-p)^t)}$$

$$= \frac{q^{t+s}}{q^t} = q^s = Pr(D > s)$$

2. La fonction de distribution de la probabilité exponentielle est $F(t) = Pr(D < t) = 1 - e^{-\lambda t}$ :

$$Pr(D > t) = 1 - F(t) = 1 - (1 - e^{-\lambda t}) = e^{-\lambda t}$$

$$Pr(D > t+s | D > t) = \frac{Pr(D > t+s, D > t)}{Pr(D > t)} = \frac{Pr(D > t+s)}{Pr(D > t)} = \frac{1-(1-e^{-\lambda(t+s)})}{1-(1-e^{-\lambda t})}$$

$$= \frac{e^{-\lambda(t+s)}}{e^{-\lambda t}} = e^{-\lambda s} = Pr(D > s)$$

■

Ce que signifie aussi, qu'il est inutile d'enregistrer le temps écoulé dans les états précédents par une action activée sans qu'elle soit exécutée. Par conséquent, la connaissance du temps écoulé ne donne pas d'information supplémentaire qu'on peut exploiter pour prévoir le temps restant pour l'exécution. Les formules indiquent que la variable aléatoire $X$ reste distribuée selon la même loi de probabilité avec le même taux, comme s'il vient d'être activé dans cet état et à cet instant.

Seules les distributions sans mémoire sont convenables avec la définition des sémantiques des algèbres des processus stochastiques, et l'introduction des distributions générales nécessite une redéfinition de ces dernières. Pour cela, la fonction de distribution de délai d'exécution de toutes les actions d'un modèle algébrique, est limitée soit à la distribution exponentielle soit à la distribution géométrique. La combinaison de deux distributions n'est pas sans mémoire dû au fait que la distribution géométrique est sans mémoire seulement à des intervalles de temps discret multiples de $\theta$. Cette restriction au niveau de la fonction de distribution de délai est essentielle pour l'analyse numérique de performance sans avoir recours aux techniques de simulation.

### 2.6.2 Les méthodes de résolution

Nous sommes intéressés par calculer l'état stationnaire du modèle, i.e., la proportion de temps que la chaîne de Markov reste dans chacun des états. Une chaîne de Markov à temps discret, où les transitions ne peuvent se produire qu'à des instants discrets, est décrite par la matrice de transition $P = \{p_{i,j} | p_{ij} \geqslant 0 \wedge \sum_{j=0}^{n} p_{ij} = 1\}$, alors que celle à temps continu, où les transitions se produisent à des instants arbitraires du temps, est décrite par la matrice générateur infinitésimal $Q = (q_{i,j})$ donnée par ses éléments :

$$\left.\begin{array}{l} q_{ij} = \lambda_{ij} \text{ où } \lambda_{ij} \text{ est le taux de franchissement de } S_i \text{ à } S_j \text{ pour } i \neq j \\ q_{ii} = -\sum_{i \neq j} q_{ij} \end{array}\right\} \Rightarrow \sum_j q_{ij} = 0 \; \forall i \leqslant n$$

$$Q = \begin{bmatrix} -\sum_{i=2}^{n} q_{1i} & q_{12} & \cdots & q_{1n} \\ q_{21} & -\sum_{i=1,i\neq 2}^{n} q_{2i} & \cdots & q_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ q_{n1} & \cdots & \cdots & -\sum_{i=1}^{n-1} q_{ni} \end{bmatrix}$$

La dimension de $Q$ est $n \times n$, $n$ étant le nombre d'états de la chaîne. Dans le cas d'une CMTD, les étiquettes sont les probabilités de transition ($p_{ij}$) de l'état $S_i$ à l'état $S_j$, tandis que dans le cas d'une CMTC les étiquettes sont les taux de transitions ($\lambda_{ij}$) de la fonction de distribution de probabilité exponentielle.

Une fois que la CMTC ou le CMTD est dérivée du modèle algébrique, l'évaluation des paramètres de performances dans les deux régimes (transitoires et stationnaire) se base sur la résolution numérique d'un ensemble d'équations linéaire dérivées à partir des formules données dans le tableau 2.2, pour obtenir les vecteurs de probabilités.

| | *CMTD* | *CMTC* |
|---|---|---|
| *Solution transitoire* | $\pi(t=n) = \pi(0).P^n$ | $\frac{d}{dt}\pi(t) = \pi(t).Q => \pi(0).e^{Qt} = \pi(t)$ |
| *Solution stationnaire* | $\pi = \pi.P$ *avec* $\sum_{i\in S} \pi_i(0) = 1$ | $\pi.Q = 0$ *avec* $\sum_{i\in S} \pi_i(0) = 1$ |

Tableau 2.2 – Solution de la CMTD et CMTC en régime stationnaire et transitoire.

Les méthodes de résolution numérique permettent donc d'obtenir les deux vecteurs de probabilités qui représentent la solution transitoire et stationnaire d'un modèle. Ces méthodes de résolution sont souvent limitées par l'espace mémoire requis pour stocker la matrice $Q$ et le vecteur $\pi$, ainsi que les autres vecteurs utilisés pour la résolution. Les méthodes de résolution directes (élimination de Gauss [LT94], décomposition en LU [Ste95, Ste94]) ont des besoins en espace mémoire trop importants, et une complexité temporelle $O(n^3)$ qui limitent leur capacité de traitement à des modèles avec quelques dizaines de milliers d'état. En revanche, les méthodes itératives [Saa96, Ste95, Ste94] (la méthode de Jacobi, Gauss-Seidel, Sur-Relaxation Successive, Puissance, Runge-Kutta, etc.) sont mieux adaptées pour la résolution d'une grande matrice avec une complexité par itération de $O(n^2)$, sauf que leur convergence est lente et parfois non assurée.

### 2.6.3 Obtention des indices de performances

Une fois obtenus les vecteurs de distribution de probabilité dans les 2 régimes, tous les paramètres de performances peuvent être calculés à partir du vecteur des probabilités stationnaires $\pi_i \in \mathrm{R}$, en utilisant la technique des récompenses [How71]. Mais, afin d'empêcher une analyse manuelle lourde et difficile sur une chaîne de Markov, même celle qui comporte quelques dizaines d'états, une méthode algébrique [Ber97a] a été adoptée et implémentée dans EMPA pour automatiser l'analyse quantitative du modèle. Cette technique permet la spécification des paramètres de performance au niveau algébrique, tout en attachant deux types de récompenses (yield & bonus) à chaque action, dont la spécification prendra la forme suivante : $< a, \lambda, y_i, b_{ij} >$. Une récompense est une valeur réelle qui sera associée à chaque état déclenchant l'exécution de l'action $a$. Ces deux types de récompenses sont détaillés dans [Ber97a]. Plusieurs paramètres de performances pourront être calculés automatiquement selon la formule 2.12 de récompense suivante :

$$R = \sum_{i=1}^{N} y_i.\pi_i + \sum_{i=1}^{N}\sum_{j=1}^{N} b_{ij}.\pi_i.q_{ij} \tag{2.12}$$

Où $\pi_i$ est la probabilité des états qui exécutent l'action correspondante à la récompense associée dans $< a, \lambda, y_i, b_{ij} >$. La caractéristique d'additivité des récompenses, autrement dit la récompense totale gagnée par un état est la somme des récompenses associées aux actions qu'il peut exécuter, rend cette méthode beaucoup plus simple que celle développée pour PEPA par Clark. Clark dans [CH96, CGHR00] a exploité le model-checking temporel de Hennessy-Milner (HML [HM85]) pour fournir une méthode automatique de calcul des indices de performances. Cette méthode consiste à utiliser les formules logiques pour associer une récompense aux états qui vérifient les conditions spécifiées ([formule logique]$\Rightarrow r_i$ ). La syntaxe de la logique HML peut se résumer par l'expression suivante :

$$HML \ni \varphi, \psi ::= \top \mid \bot \mid \neg\varphi \mid \varphi \vee \psi \mid \varphi \wedge \psi \mid [a]\varphi \mid \langle a \rangle \varphi$$

Et les sémantiques des opérateurs sont données par les relations de satisfaction suivantes :

$$S \models \top \Leftrightarrow S \models vrai$$

$$S \models \bot \Leftrightarrow S \not\models \top$$

$$S \models \neg\varphi \Leftrightarrow S \not\models \varphi$$

$$S \models \varphi \wedge \psi \Leftrightarrow S \models \varphi \wedge S \models \psi$$

$$S \models \varphi \vee \psi \Leftrightarrow S \models \varphi \vee S \models \psi$$

$$S \models \langle a \rangle \varphi \Leftrightarrow \exists S / S \xrightarrow{a} S' \wedge S' \models \varphi$$

$$S \models [a]\, \varphi \Leftrightarrow \forall S / S \xrightarrow{a} S' \Rightarrow S' \models \varphi$$

Soit $S$ le diagramme de transitions sous jacent d'un processus, l'expression $S \models \langle a \rangle \varphi$ signifie qu'il est possible de trouver un état $s_i$ qui exécute l'action $a$ puis transiter dans un état $s_j$ vérifiant la propriété $\varphi$. En revanche, l'expression $S \models [a]\, \varphi$ exige la satisfaction de la propriété $\varphi$ par tous les états $s_j$ accessibles après l'exécution de l'action $a$.

Les paramètres de performances sont calculés comme étant la somme de $\sum_i r_i \times \pi_i$, où $\pi_i$ et $r_i$ représentent respectivement la probabilité et l'indice de performance associés aux états vérifiant la formule logique. Mais l'absence de l'additivité oblige le concepteur à spécifier tous les cas possibles, comme par exemple : $\langle a \rangle \langle a \rangle \cdots \langle a \rangle \neg \langle a \rangle$ tt $\Rightarrow n$. Dans ces formules, le concepteur associe une récompense : 1 pour l'état qui exécute l'action $a$ une seule fois, deux s'il exécute $a$ deux fois et ainsi de suite. Le paramètre de performance obtenu par ces formules, est tout simplement calculé en associant le couple de récompenses ($y_i = 1$, $b_{ij} = 0$) à l'action $a$ dans la méthode algébrique, sans se soucier du formalisme du model-checking.

Mais l'évaluation de performance d'un modèle algébrique souffre du problème de l'explosion de l'espace d'états, où une description exhaustive d'un système moderne conduit à un modèle grand et complexe, avec un grand nombre d'états dans la chaîne de Markov. Il est largement connu que le nombre d'états augmente de manière exponentielle avec le nombre de processus en parallèle. Dans un système avec $n$ processus où chacun contient au maximum $m$ événements, l'espace d'état est de l'ordre $O(m^n)$.

Néanmoins, lorsque l'espace d'état est grand, il n'est pas possible d'analyser le modèle. Les techniques de résolution classiques (directes et itératives) qui sont implémentés dans les outils existants, ne peuvent pas résoudre une chaîne de Markov avec une telle taille à cause de grands nombres d'itérations et d'un espace mémoire trop importante requis pour la résolution.

Différentes techniques de simulation (à événement discret [Mit82], à réplication indépendante, à la volée, etc.) peuvent être utilisées comme ultime recours pour la résolution d'un modèle grand et complexe, même avec une description temporelle non Markovienne, comme dans SPADES [DHKK01, SH00] (Stochastic Process Algebra for Discret Event Simulation), mais la précision des résultats obtenus est largement dépendante du nombre d'itérations et du temps de simulation. Les méthodes numériques restent beaucoup plus précises et plus rapides que la résolution par simulation dans la mesure du possible.

Sachant que la matrice générateur de la chaîne de Markov est très creuse (sparse), différentes techniques qui permettent de diminuer les besoins en mémoire pour stocker la matrice, et pour réduire

les nombres d'opérations requises sont apparues, comme l'utilisation de l'algèbre tensorielle [Pla84, Don93], la représentation symbolique (Multi-Terminal Binary Decision Diagram MTBDD [HMKS00]), et les techniques de distribution parallèle de calcul sur plusieurs machines pour augmenter la mémoire et diminuer le temps de calcul d'une grande matrice [Kno00].

La représentation tensorielle de la matrice générateur infinitésimal a été proposé par Plateau [Pla84] en 1984. Kelloul et Hillston dans [HK01] proposent une méthode de transformation de générateur de la chaîne sous jacente d'un modèle algébrique, en une représentation tensorielle tel qu'elle est proposée par Plateau pour le formalisme de modélisation avec les automates stochastiques. Cette technique permet une structuration compacte du générateur de la chaîne associée au modèle, grâce à une formule mathématique appelée descripteur [Fer98, Ben03], afin de réduire les besoins en mémoire. Mais cette technique n'est pas toujours suffisante pour pallier le problème de l'explosion de l'espace d'états.

La représentation symbolique (les diagrammes de décision binaire BDD [Bry86, HMKS00]) utilisée aussi dans les modèles de vérification fonctionnelle, consiste à utiliser des diagrammes de décision pour compresser la représentation de la matrice de la chaîne de Markov généralement creuse. Cette représentation ne garde en mémoire que les éléments distincts non nuls, qui se trouvent sur les feuilles de l'arbre. Cette technique est très intéressante mais son efficacité en mémoire dépend directement du nombre d'éléments distincts de la matrice $Q$. De plus, l'accès à un élément non nul nécessite le parcours du chemin depuis la racine jusqu'aux feuilles (opération lente). Par conséquent, cette représentation essaie de réduire le besoin en mémoire au détriment de l'utilisation de la CPU. Plus de détails à propos de cette technique sont dans [Nik00].

Les différentes techniques d'abstraction de haut niveau, traitent la résolution du problème de l'explosion de l'espace d'états plutôt a priori qu'a posteriori. Ces techniques ont pour but d'ignorer certains détails du système et le modèle résultant n'est qu'une abstraction du système original. Cette méthode permet de réaliser la vérification de certaines propriétés des modèles ayant à l'origine un très grand nombre d'états. Mais, si une erreur est trouvée dans le modèle abstrait alors on peut garantir que celle-ci existe dans le système réel mais l'inverse n'est pas vrai. Par exemple l'absence du blocage dans le modèle abstrait ne garantit pas son absence dans le modèle complet.

Plusieurs autres techniques particulières ont été proposées pour lutter contre l'explosion de l'espace, comme l'agrégation des états en se basant sur certaine notion d'équivalence. Les techniques d'agrégation permettant de simplifier le modèle étudié en réduisant la taille de l'espace d'états. Généralement, l'espace d'états $\{s_0, s_1, \ldots, s_n\}$ est décomposé en plusieurs groupes d'états $\{S_{[0]}, S_{[1]}, \ldots, S_{[N]}\}$, dont chacun regroupe plusieurs états pour former un seul état dans le nouvel espace réduit d'états, avec $N \leqslant n$. Des études sur l'identification des conditions d'agrégation de haut niveau (congruence, isomorphisme, équivalence forte, bisimulation forte et faible, lumpability, etc.), pour grouper les états identiques tout en gardant la propriété de Markov dans la chaîne résultante, sont discutés dans [Hil96, Hil00, GHR01, Rib95].

Beaucoup de travaux se sont intéressés à l'exploitation de la structure compositionnelle du modèle pour dériver les vecteurs de distribution de probabilités du modèle sous forme d'une combinaison de solutions de ses composants isolés, et sans avoir besoin de dériver l'espace d'état global du modèle complet. L'idée de base est l'identification d'une solution particulière sous forme produit en utilisant des propriétés telles que la : réversibilité [HT98, CH02], quasi-réversibilité [HH95], quasi-séparabilité [TG98], etc. Mais ces solutions sont rarement rencontrées lors de la modélisation d'un système réel.

La technique la plus intéressante et la plus efficace pour la résolution de ce problème est la méthode de calcul des bornes stochastiques [FP02]. Cette dernière consiste à déterminer des bornes supérieures et inférieures pour les paramètres de performance plutôt qu'une valeur exacte, tout en cherchant un petit intervalle pour minimiser le taux d'erreur. En effet, la qualité de service est souvent définie en terme de seuil à respecter (par exemple sur la gigue ou le taux de perte), il est donc suffisant d'obtenir une borne supérieure qui vérifie la contrainte prédéfinie.

**Exemple**

Nous reprenons l'exemple algébrique de la file $M/M/1/k$ pour dériver les paramètres de performance comme le débit, le taux d'utilisation, la probabilité de perte, etc. par l'usage de la méthode de récompense algébrique donnée dans (2.12), et une certaine malice dans la détermination des valeurs de $y_i$ et $b_{ij}$. Par exemple, le débit de la file est donné par le nombre des paquets servis par unité de temps :

$$D = \mu.\sum_{i=1}^{N} Pr(service_H) = \sum_{i=0}^{N} y_i.\pi_i + \sum_{i=0}^{N}\sum_{j=0}^{N} b_{ij}.\pi_i.q_{ij} = \sum_{i=1}^{N} \mu.\pi_i$$

Par conséquent, le débit est calculé avec la méthode algébrique dans (2.12), en associant le couple de récompenses $y_i = \mu$ et $b_{ij} = 0$ à l'action de service $service$. Dans cet exemple, cette valeur pourra être obtenue sans recours à la méthode algébrique, avec l'équation suivante :

$$D = \sum_{i=1}^{N} \mu.\pi_i = \mu.(1 - \pi_0) \tag{2.13}$$

Le taux d'utilisation de l'ordonnanceur est définie comme étant la fraction du temps durant laquelle ce composant est utilisé. Ce paramètre est calculé comme étant la somme des probabilités qu'il y a au moins un paquet dans le système :

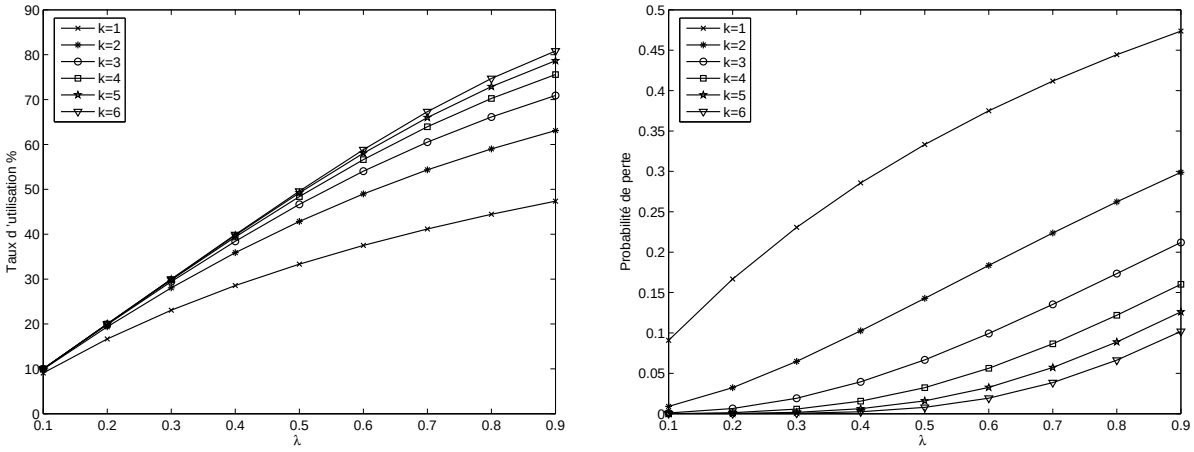
$$U = 1 - \pi_0 = \sum_{i=1}^{N} 1.\pi_i = \sum_{i=0}^{N} y_i.\pi_i + \sum_{i=0}^{N}\sum_{j=0}^{N} b_{ij}.\pi_i.q_{ij} \tag{2.14}$$

Ce paramètre est obtenu en associant le couple de récompenses $y_i = 1$ et $b_{ij} = 0$ à l'action $service$.

La probabilité de perte est calculée comme étant la somme de probabilité de tous les états qui peuvent exécuter l'action $refusé$.

$$L = \sum_{i=0}^{N} Pr(refusé) \tag{2.15}$$

$L$ est obtenu en associant le couple de récompenses $y_i = 1$ et $b_{ij} = 0$ à l'action $refusé$. Les courbes de variation de $U$ et $L$ en fonction du taux d'arrivée $\lambda$, sont présentées dans la figure 2.4, étant donnée que pour $\mu = 1$, on a $D = U = 1 - \pi_0$.




(a) Taux d'utilisation.

(b) Probabilité de perte.

Figure 2.4 – Courbes de variation de $U$ et $L$ en fonction $\lambda$.

## 2.7 Model checking

Une fois le diagramme de transitions sous-jacent du modèle algébrique est dérivé, une tâche importante est la vérification de la satisfaction des propriétés fonctionnelles. Une stratégie naïve serait l'utilisation de la simulation (avec plusieurs répétitions) pour vérifier le bon fonctionnement du système. Le problème posé par l'utilisation de la simulation est qu'elle ne permet pas de couvrir tout l'espace d'états du système à vérifier. Par conséquent, cette méthode ne garantit pas l'absence d'erreurs, car si un état absorbant n'est pas atteint pendant la simulation, ceci ne signifie pas son absence.

Le model-checking est une technique qui permet de s'assurer qu'une propriété, exprimée en une certaine logique formelle, est vérifiée par le diagramme de transition sous-jacent de la spécification initiale, afin de détecter les défauts et d'y remédier. Ce type de vérification permet de spécifier et de vérifier des propriétés en explorant l'espace d'états.

Les logiques temporelles (linéaire ou arborescente) sont utilisées pour énoncer formellement des propriétés propositionnelles sur les exécutions du modèle. Une exécution $\sigma$ de l'espace d'état $S$ est une séquence de transitions $\sigma = s_0 \xrightarrow{a_0} s_1 \ldots s_{n-1} \xrightarrow{a_{n-1}} s_n$. Les outils de model-checking peuvent répondre automatiquement à ces questions avec la réponse : oui ou non, et un avantage de trouver des contre-exemples utiles à la compréhension des situations d'erreur (*cf.* figure 2.5). Les propriétés importantes que l'on souhaite souvent assurer dans un système de transitions sont :

1. L'absence d'états bloquants (c'est-à-dire, un état sans successeurs) et de boucles infinies, qui énonce que le système ne se trouvera jamais dans une situation où il lui est impossible de progresser.
2. L'atteignabilité pour vérifier que certains états peuvent être atteints ou non.
3. La sûreté pour vérifier que sous certaines conditions quelques chose de mauvais ne peut se produire.
4. La vivacité pour s'assurer qu'une action finira par avoir lieu.
5. L'exclusion mutuelle pour garantir l'unicité dans l'utilisation d'une ressource partagée.
6. L'ordonnancement dans l'exécution d'une certaine séquence d'actions.

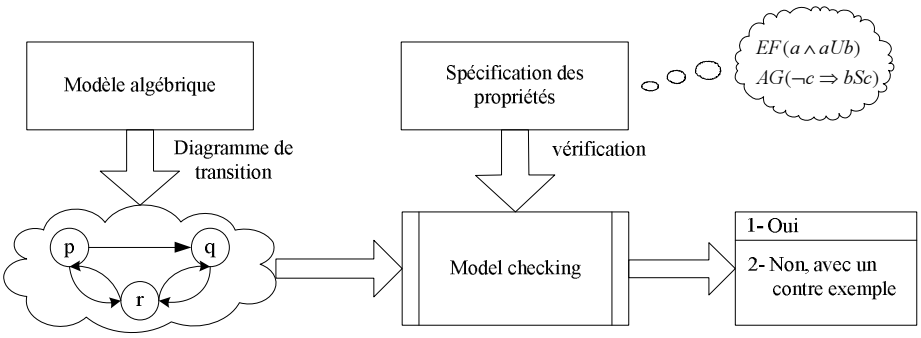



Figure 2.5 – Processus de vérification fonctionnelle.

Nous nous intéressons plus particulièrement aux logiques temporelles qualitatives, comme CTL (Computation Tree logic) [CES86, CGP99] et $\mu$-calcul [Koz83, CGP99]. Dans ces logiques, le temps n'est pas spécifié explicitement et il ne peut être déduit que par la succession d'événements et d'états.

### 2.7.1 CTL

Le model-checking CTL (Computation Tree Logic) est la logique temporelle arborescente définie par Clarke et Emerson [CES86]. Dans la logique CTL, les opérateurs temporels apparaissent uniquement par paire, composés d'un quantificateur de chemin $A$ et $E$, suivies par $G$, $F$, $U$ et $X$. Nous présentons dans ce qui suit la syntaxe et la sémantique de CTL.

### La syntaxe de CTL

Soit $AP$ l'ensemble des propositions atomiques (i.e. sans connecteur logique), et $\varphi$ et $\psi$ étant 2 éléments de $AP$. La syntaxe CTL est définie par l'expression suivante :

$$CTL \ni \varphi, \psi ::= \top \mid \neg\varphi \mid \varphi \vee \psi \mid EX\varphi \mid AX\varphi \mid E(\varphi U \psi) \mid A(\varphi U \psi)$$

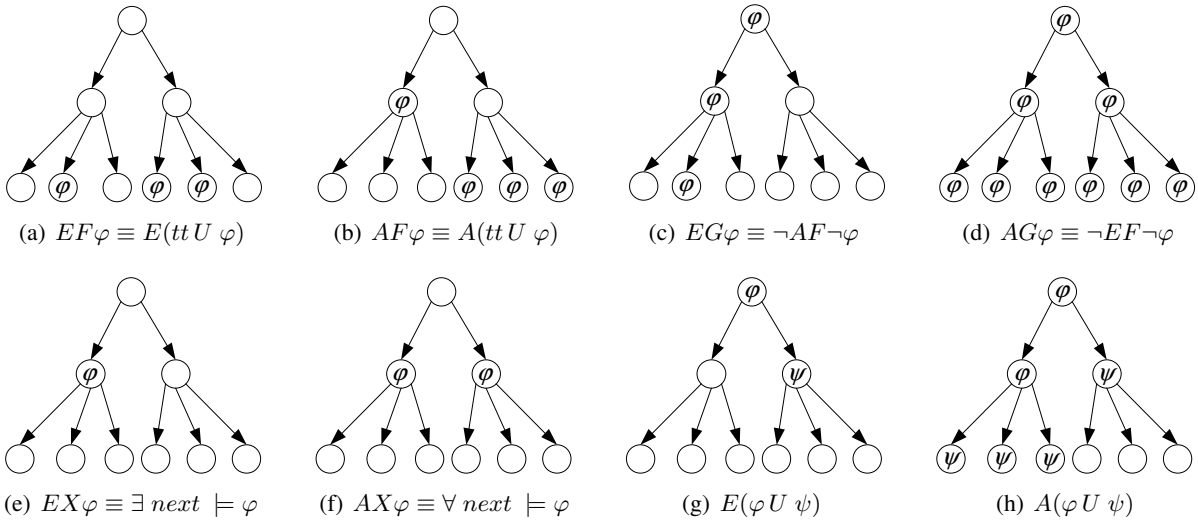
Cette syntaxe autorise l'utilisation des opérateurs booléens $\vee$ et $\neg$. Cependant, plusieurs opérateurs peuvent être dérivés à partir de l'expression précédente :

$$\bot \stackrel{def}{=} \neg\top \qquad \varphi \wedge \psi \stackrel{def}{=} \neg(\neg\varphi \vee \neg\psi)$$

$$AF\varphi \stackrel{def}{=} A(vrai\ U\ \varphi) \qquad EF\varphi \stackrel{def}{=} E(vrai\ U\ \varphi)$$

$$AG\varphi \stackrel{def}{=} \neg E(vrai\ U\ \neg\varphi) = \neg EF\neg\varphi \qquad EG\varphi \stackrel{def}{=} \neg A(vrai\ U\ \neg\varphi) = \neg AF\neg\varphi$$

Sans entrer dans les détails de la sémantique pour le moment, nous précisons que les quantificateurs de chemin d'exécution ($E$ et $A$) permettent de parler du côté arborescent du comportement : $EF\varphi$ signifie qu'il existe au moins un chemin $\sigma$, où $\varphi$ sera vérifiée au moins une fois dans un état appartenant à ce chemin (*cf.* figure 2.6(a)). $AF\varphi$ signifie que $\varphi$ sera sastifaite au moins une fois dans un état futur dans tout les chemins traversant l'état initial (*cf.* figure 2.6(b)). $EG\varphi$ signifie qu'il existe au moins un chemin tout au long duquel, tous les états vérifient la propriété $\varphi$ (*cf.* figure 2.6(c)). $AG\varphi$ énonce que toutes les séquences d'états atteignables à partir de l'état initial vérifient la propriété $\varphi$ (*cf.* figure 2.6(d)). $EX\varphi$ spécifie qu'à partir d'un état $s_i$, il existe au moins un état successeur immédiat, qui satisfait $\varphi$ (*cf.* figure 2.6(e)), et $AX\varphi$ pour dire que tous les états successeurs immédiats, satisfassent $\varphi$ (*cf.* figure 2.6(f)). $E(\varphi\ U\ \psi)$ signifie qu'il existe au moins un chemin, où $\varphi$ sera vrai jusqu'à ce que la propriété $\psi$ le soit (*cf.* figure 2.6(g)), et $A(\varphi\ U\ \psi)$ signifie que la propriété $\varphi U \psi$ est satisfaite sur tous les chemins traversant l'état initial (*cf.* figure 2.6(h)).



(a) $EF\varphi \equiv E(tt\,U\ \varphi)$ (b) $AF\varphi \equiv A(tt\,U\ \varphi)$ (c) $EG\varphi \equiv \neg AF\neg\varphi$ (d) $AG\varphi \equiv \neg EF\neg\varphi$

(e) $EX\varphi \equiv \exists\, next\ \models \varphi$ (f) $AX\varphi \equiv \forall\, next\ \models \varphi$ (g) $E(\varphi\, U\ \psi)$ (h) $A(\varphi\, U\ \psi)$

Figure 2.6 – Représentation graphique des opérateurs de chemin dans CTL.

### La sémantique de CTL

Soit $\{S|s_i \in S\}$ le diagramme de transition, $\varphi$ et $\psi$ sont deux formules de CTL, et $\models$ est la fonction de satisfaction. La notation $s, i \models \varphi$ signifie que la formule $\varphi$ est satisfaite dans l'état $s_i$. Les sémantiques

des expressions CTL sont données par les règles suivantes :

$s, i \models \top\ (vrai)$

$s, i \not\models \bot\ (faux)$

$s, i \models \varphi\ ssi\ s_i \in SAT(\varphi)$

$s, i \models \neg\varphi\ ssi\ s, i \not\models \varphi$

$s, i \models \varphi \vee \psi\ ssi\ s, i \models \varphi \vee s, i \models \psi$

$s, i \models \varphi \wedge \psi\ ssi\ s, i \models \varphi \wedge s, i \models \psi$

$s, i \models AX\varphi\ ssi\ \forall s_{i+1} \Rightarrow s, i+1 \models \varphi$

$s, i \models EX\varphi\ ssi\ \exists s, i+1 \Rightarrow s, i+1 \models \varphi$

$s, i \models A(\varphi\ U\ \psi)\ ssi\ \forall \sigma$ un chemin $(s_i, s_{i+1}, \ldots)$ dans $S\ :\ \exists i | s_i \models \psi \Rightarrow \forall j < i, s_j \models \varphi$

$s, i \models E(\varphi\ U\ \psi)\ ssi\ \exists \sigma$ un chemin $(s_i, s_{i+1}, \ldots)$ dans $S\ :\ \exists i | s_i \models \psi \Rightarrow \forall j < i, s_j \models \varphi$

$Sat(\varphi)$ étant l'ensemble des états où la formule $\varphi$ est vraie.

### 2.7.2 $\mu$-calcul

Le $\mu$-calcul [Koz83, CGP99] est une logique très expressive, vu comme une extension de la logique HML (*cf.* section 2.6.3) avec des opérateurs de point fixe. On va se contenter ici de rappeler les éléments fondamentaux de la logique $\mu$-calcul. Cette présentation reste largement incomplète. Pour une introduction précise et complète au $\mu$-calcul, on pourra se référer à [CGP99].

#### La syntaxe du $\mu$-calcul

La syntaxe du $\mu$-calcul est donnée par l'expression suivante :

$$CTL \ni \varphi, \psi ::= \top \mid \bot \mid \neg\varphi \mid \varphi \vee \psi \mid \varphi \wedge \psi \mid [A].P \mid \langle A \rangle.P \mid \nu Z.\varphi \mid \mu Z.\varphi$$

Où $a \in A$, $A$ une liste d'actions, $Z$ est une variable propositionnelle, $\langle\ \rangle$ est l'opérateur de possibilité, $[]$ est celui de nécessité, $\nu$ est l'opérateur de plus grand point fixe et $\mu$ est l'opérateur de plus petit point fixe.

#### La sémantique du $\mu$-calcul

La sémantique des formules sur actions est identique à celle de HML (Hennessy-Milner Logic), elle est définie inductivement comme suit :

$s, i \models \top\ (vrai)$

$s, i \not\models \bot\ (faux)$

$s, i \models \varphi\ ssi\ s_i \in SAT(\varphi)$

$s, i \models \neg\varphi\ ssi\ s, i \not\models \varphi$

$s, i \models \varphi \vee \psi\ ssi\ s, i \models \varphi \vee s, i \models \psi$

$s, i \models \varphi \wedge \psi\ ssi\ s, i \models \varphi \wedge s, i \models \psi$

$s, i \models [A].\varphi\ ssi\ \forall s' \in \{s \xrightarrow{a} s' \wedge a \in A\} \Rightarrow s' \models \varphi$

$s, i \models \langle A \rangle.\varphi\ ssi\ \exists s' \in \{s \xrightarrow{a} s' \wedge a \in A\} \Rightarrow s' \models \varphi$

$s, i \models \nu Z.\varphi(Z)\ ssi\ \forall s' \in Reach(s) \wedge s' \models \varphi$

$s, i \models \mu Z.\varphi(Z)\ ssi\ \exists s' \in Reach(s)\} \wedge s' \models \varphi$

### 2.7.3 CTL*, PCTL et CSL

la logique CTL* permet d'exprimer des propriétés sur les chemins aussi bien que sur les arbres d'exécution d'un système de transitions, en réconciliant les logiques linéaire (LTL [Pnu77]) et arborescente (CTL), grâce à l'élimination de la restriction sur la composition des opérateurs par paire. Ceci a pour avantage d'augmenter la puissance d'expression du model checking dans la mesure où certaines propriétés de CTL ne peuvent pas être exprimée en LTL et vice versa.

Enfin, reste à noter que la logique CSL [ASSB00] qui n'est autre qu'une extension du PCTL (Probabilistic Computation Tree Logic [HJ94]) pour la CMTC, est beaucoup plus expressive que les logiques utilisées dans ce manuscrit, où certaines propriétés qui sont fausses en CTL, peuvent s'avérer vraies sous restriction d'une certaine probabilité avec les opérateurs du CSL. Mais, pour ne pas trop compliquer la procédure de modélisation et d'analyse pratique, où l'outil EMPA est basé sur l'outil CWB-NC, qui ne supporte pas de vérification stochastique, nous allons utiliser CTL et $\mu$-calcul. Reste à noter que d'autre outils (UltraSAN/Möbius [SOQW95, DCC+02], PRISM [KNP02], etc.) permettent d'appliquer le modèle stochastique CSL sur la CMTC sous jacente d'un modèle algébrique de PEPA.

## 2.8 Modélisation algébrique d'un routeur DiffServ

Cette section présente un exemple de modélisation abstraite d'un routeur *DiffServ*. Cet exemple permet d'aborder les différents concepts des algèbres de processus stochastiques. Nous illustrons à travers cet exemple la composition parallèle et les interactions entre les différents composants du système. Ensuite, nous réalisons une analyse qualitative et quantitative de ce modèle tout en appliquant les techniques expliquées précédemment.

La technologie *DiffServ* permet aux fournisseurs d'accès d'offrir des services différenciés aux trafics de leurs clients selon un contrat *SLA* [MN02] (Service Level Agreement) communément agréé, ou selon le type d'application.

Le routeur de bordure dans *DiffServ* divise le trafic en entrée en un certain nombre de classes, et marque le champ *DSCP* dans l'entête de chaque paquet IP. La conformité au profil convenu, ou *TCA* (Traffic Control Agreement), est vérifiée dans le routeur d'entrée, et le trafic excédentaire peut être rejeté, marqué hors profil, ou se voir affecter une classe de priorité inférieure. Dans le cœur du réseau, les flots ayant des propriétés différents sont regroupés en plusieurs classes de *QdS* ou *BA* (Behavior Aggregates). Les performances perçues par chaque classe dépendent du traitement spécifié de chaque routeur de cœur (Per Hop Behavior–*PHB*). Grosso modo, les routeurs d'entrée associent une priorité (*DSCP*) aux flots, et les routeurs de cœur transmettent les paquets avec un traitement différentié selon cette priorité.

Deux types de comportement du routeur sont définis en plus du service garantie au mieux (best effort) : *EF* (Expedited Forwarding) [JNP99] et *AF* (Assured Forwarding) [HBWW99]. Ces deux types de service ont été proposés pour offrir des traitements différenciés du trafic. Le comportement d'un routeur (ou PHB) avec la classe *EF* est associé aux applications qui ont de fortes contraintes de *QdS* (les applications temps réel comme VoIP et visioconférence), pour fournir un faible taux de perte, une faible latence, une faible gigue, une bande passante assurée et un service de bout en bout assuré à travers les domaines DiffServ. La catégorie *AF* fournit différents niveaux de traitements pour assurer l'acheminement des paquets IP selon la valeur de la priorité, et le contrôle de trafic lors de la congestion du réseau. Dans chaque classe *AF*, un paquet IP est associé à un des trois niveaux de priorités de rejet (Drop Precedence), donc au total, il y a 12 classes de trafic dans *AF*. Généralement, le comportement d'un routeur avec une garantie *AF* est réservé pour le transport des données de TCP, HTTP, FTP, etc.

L'architecture *DiffServ* définit plusieurs éléments qui constituent le chemin emprunté par les paquets lorsqu'ils passent par le routeur. Ces éléments sont représentés dans la figure 2.7, où on voit que les éléments du routeur de bordure, viennent compléter ceux du routeur du cœur.

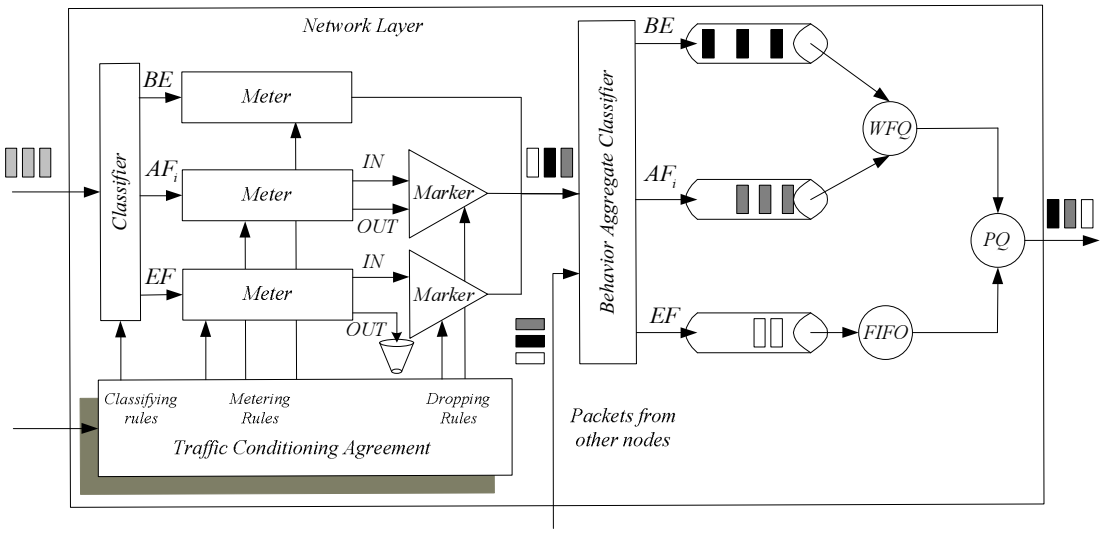



Figure 2.7 – Composants d'un routeur DiffServ.

### 2.8.1 Spécification algébrique

Pour construire un modèle d'un routeur *DiffServ*, une description détaillée semble nécessaire afin de vérifier les exigences qualitatives. Mais, pour pallier le problème de l'explosion de l'espace d'états, nous avons décidé de décrire les actions utiles pour la vérification fonctionnelle comme des actions immédiates, pour qu'elles n'apparaissent que dans le diagramme de fonctionnement et non pas dans le diagramme de performance. A noter que le délai d'exécution de ces actions, qui est utile pour l'évaluation de performances, est agrégé dans d'autres actions du même composant.

Pour éviter une explosion de l'espace d'états et pour des raisons de clarté, notre modèle abstrait d'un routeur de bordure, traite seulement deux classes de paquets « Low et High » (*cf.* figure 2.8). Le modèle abstrait du routeur de cœur est structuré en 3 composants : le classifieur, la file d'attente et l'ordonnanceur.

Les paquets arrivent selon une distribution de Poisson (l'intervalle de temps entre deux arrivées consécutives est exponentiellement distribué avec un taux $\lambda_{i|i\in\{H,L\}}$. Les paquets sont servis dans le même ordre de leur arrivée (FIFO) avec une stratégie de service «non-preemptive». Le temps de service est exponentiellement distribué avec un taux $\mu$. Selon la représentation de Kendall pour les files d'attente, ce modèle s'écrit sous la forme $M/M/1/N$, avec $N$ est la capacité totale du système, étant donné qu'une file d'attente réelle a toujours une capacité finie. Ce modèle est illustré dans la figure 2.8. La spécification

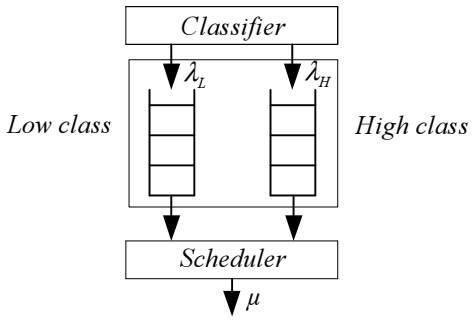



Figure 2.8 – Routeur de bordure.

algébrique est donnée par :

$$DiffServ - BR \stackrel{def}{=} Classifier||_A(Queue_{00}||_D Scheduler)$$

$$A = \{arr_H, arr_L\};$$

$$D = \{deliver_H, deliver_L\};$$

$$Classifier \stackrel{def}{=} (arr_H, \lambda_H).Classifier + (arr_L, \lambda_L).Classifier;$$

$$Queue_{0,0} \stackrel{def}{=} (arr_H, *).Queue_{1,0} + (arr_L, *).Queue_{0,1};$$

$$Queue_{i,0} \stackrel{def}{=} (arr_H, *).Queue_{i+1,0} + (arr_L, *).Queue_{i,1} + (deliver_H, *).Queue_{i-1,0}; \quad Si\ (0 < i < N-1)$$

$$Queue_{0,j} \stackrel{def}{=} (arr_H, *).Queue_{1,j} + (arr_L, *).Queue_{0,j+1} + (deliver_L, *).Queue_{0,j-1}; \quad Si\ (0 < j < N-1)$$

$$Queue_{i,j} \stackrel{def}{=} (arr_H, *).Queue_{i+1,j} + (arr_L, *).Queue_{i,j+1} + (deliver_H, *).Queue_{i-1,j} + (deliver_L, *).Queue_{i,j-1}; \quad Si\ ((i \wedge j) > 0 \wedge (i+j) < N-1)$$

$$Queue_{N-1,0} \stackrel{def}{=} (deliver_H, *).Queue_{N-2,0} + (arr_H, *).(lose_H, \infty_{2,1}).Queue_{N-1,0};$$

$$Queue_{0,N-1} \stackrel{def}{=} (deliver_L, *).Queue_{0,N-2} + (arr_L, *).(lose_L, \infty_{2,1}).Queue_{0,N-1};$$

$$Queue_{i,j} \stackrel{def}{=} (deliver_H, *).Queue_{i-1,j} + (deliver_L, *).Queue_{i,j-1} + (arr_H, *).(lose_H, \infty_{2,1}).Queue_{i,j} + (arr_L, *).(lose_L, \infty_{2,1}).Queue_{i,j}; \quad Si\ (i, j > 0\ \wedge\ (i+j)\ =\ N-1)$$

$$Scheduler \stackrel{def}{=} (deliver_H, \infty_{2,1}).ServeP + (deliver_L, \infty_{1,1}).ServeP;$$

$$ServeP \stackrel{def}{=} (serve, \mu).Scheduler;$$

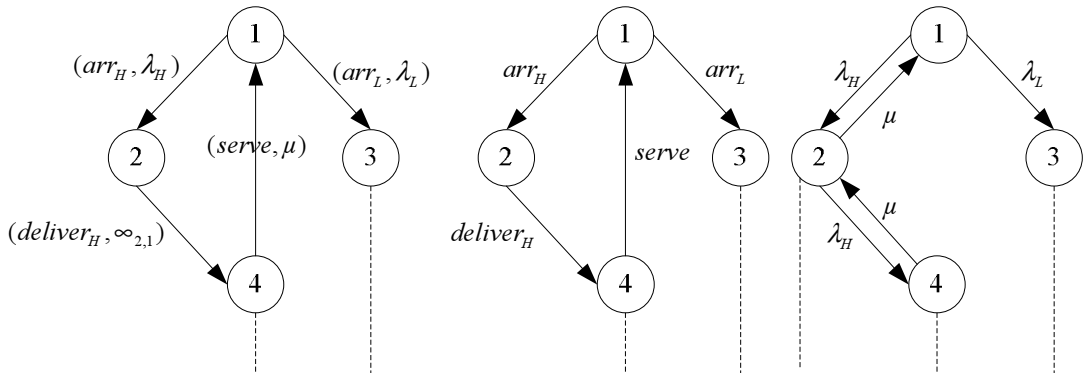
Les arrivées de paquets vers la file sont modélisées par l'événement $arr_i$, et les départs par l'événement $deliver_i$. Les différents événements sont détaillés ci-dessous :

- $arr_i$ : action représentant l'arrivée d'un paquet de la classe $C_i$ dans la file $i$ avec un taux $\lambda_i$.
- $deliver_i$ : action synchronisant le départ d'un paquet de la classe $C_i$ dans la file, et son arrivée dans l'ordonnanceur (ou le gestionnaire de service).
- $serve$ : action représentant le traitement d'un paquet avec un taux $\mu$, et sa transmission vers les couches plus basses (couche liaison).
- $loose_i$ : action représentant la suppression de nouveaux paquets lorsque la file est pleine.

L'outil d'EMPA (*TwoTowers*) génère, après la vérification syntaxique d'une description algébrique d'un modèle, 3 diagrammes de transitions sous forme d'une liste : le diagramme intégré, le diagramme de fonctionnement et le diagramme de performance (CMTC). La figure 2.9 illustre une petite partie de chacun de ces diagrammes.

Le diagramme de transition intégré est dérivé en utilisant les sémantiques des algèbres de processus classiques. Ce diagramme, dont chaque transition est étiquetée avec le nom et le taux de l'action qui la provoque, peut être utilisé pour réaliser une étude intégrée (qualitative et quantitative) comme par exemple la probabilité d'exécution d'une certaine séquence d'actions est supérieur à 90%, en utilisant le modèle de vérification stochastique. Les deux autres diagrammes (le diagramme fonctionnel et la chaîne de Markov) sont dérivés du diagramme intégré, soit en supprimant les informations temporelles qui concernent l'étude quantitative (les taux des actions), soit en supprimant les informations qualitatives (nom des actions). Les vecteurs de distribution de probabilité dans les 2 régimes (transitoire et stationnaire) peuvent être dérivés en utilisant les méthodes de résolution numérique classiques (*cf.* section 2.6.2), et les indices de performances avec la méthode algébrique (*cf.* section 2.6.3).

Le diagramme de transition intégré, sous jacent du modèle algébrique, contient 2632 états (dont 896 tangibles et 1736 évanescents) et 9212 transitions (2688 exponentielles et 6524 immédiates) pour une capacité de la file $N = 4$. La chaîne de Markov résultante contient 896 états et 2688 transitions.




(a) Diagramme intégré. (b) Diagramme fonctionnel. (c) Chaîne de Markov.

Figure 2.9 – Diagrammes de transition.

Suite au fait que l'espace d'état est raisonnable, nous avons commencé à investir un peu plus dans les détails de ce modèle en ajoutant les composants de classification, de marquage et de conditionnement de trafic [SB03c, SB03b]. Notre extension du routeur de bordure pour l'intégration du fonctionnement de routeur de bordure est représentée dans la figure 2.10. Nous avons utilisé la caractéristique de la construction par composition, pour construire le modèle complet en partant du modèle algébrique précédent. La spécification de nouveaux composants du modèle est donnée par :

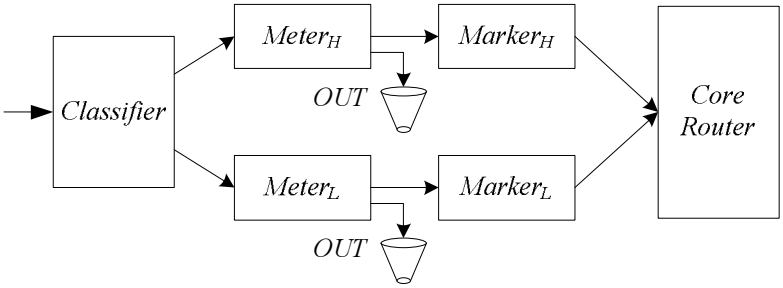



Figure 2.10 – Les composants d'un routeur de bordure DiffServ.

**Classification :** ce composant consiste à déterminer la classe de chaque paquet selon le type de l'application ou selon le SLA, et de le transmettre au composant de mesure/lissage du trafic. Par conséquent, c'est un filtre qui consiste à décomposer le trafic d'entrée sur n sorties, $n$=2 dans notre approche (*cf.* figure 2.11(a)). La spécification algébrique de ce composant est donnée par :

$$\begin{aligned} Classifier\_BR \overset{def}{=}\ & (pck\_arr, \lambda).(ckhdr, \phi). \\ & ((pkt_H, p_1\lambda).Classifier\_BR + (pkt_L, (1-p_1)\lambda).Classifier\_BR); \end{aligned}$$

**Meter$_i$ :** l'élément de mesure se charge de mesurer le trafic qui arrive, et de le comparer avec le profil de la classe correspondante. Le trafic excédentaire sera rejeté (*cf.* figure 2.11(b)), et nous avons écarté le marquage hors profil avec la spécification de 2 classes uniquement. La représentation algébrique de ce composant est donnée par :

$$\begin{aligned} Meter_{i,i\in[H,L]} \overset{def}{=}\ & (pkt_i, *).(verif, \eta). \\ & ((\tau, \infty_{1,1-p}).(to\_mark_i, \infty_{1,1}).Meter_i + (\tau, \infty_{1,p}).(out, \infty_{1,1}).Meter_i) \end{aligned}$$

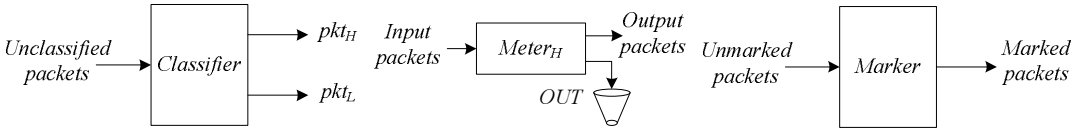
**Marker$_i$ :** Le marqueur se charge d'ajouter la priorité (DSCP) à l'entête de chaque paquet reçu de l'élément de mesure. Ce composant est représenté par un bloc à une entrée et une sortie (*cf.* figure 2.11(c)). La spécification algébrique est donnée par :

$$Marker_{i|i\in[H,L]} \stackrel{def}{=} (to\_mark_i, *).(mark_i, \omega).(arr_i, \lambda_i).Marker_i$$

La composition parallèle de ces composants constitue le modèle complet du routeur DiffServ, donnée par :

$$DiffServ \stackrel{def}{=} Classifier\_BR||_{\{pkt_i\}}(Meter_H||Meter_L)||_{\{to\_mark_i\}}(Marker_H||Marker_L)||_{arr_i} DiffServ - CR$$

Ensuite, nous avons appliqué cette description algébrique à l'outil d'EMPA pour dériver les nouveaux diagrammes de transitions et réaliser une étude qualitative et quantitative.




(a) Classificateur de trafic. (b) Élément de mesure. (c) Marqueur du trafic.

Figure 2.11 – Spécification des composants du routeur DiffServ.

### 2.8.2 Analyse qualitative et quantitative

Le diagramme de transitions intégré, dérivé du modelé algébrique précédent, avec tous les détails qu'on a ajouté pour supprimer les états bloquants, comporte 101532 états (dont 96456 transitoires et 5076 évanescents) et 406652 transitions (392424 exponentielles et 14228 immédiates) avec une capacité $N = 4$.

#### Vérification fonctionnelle

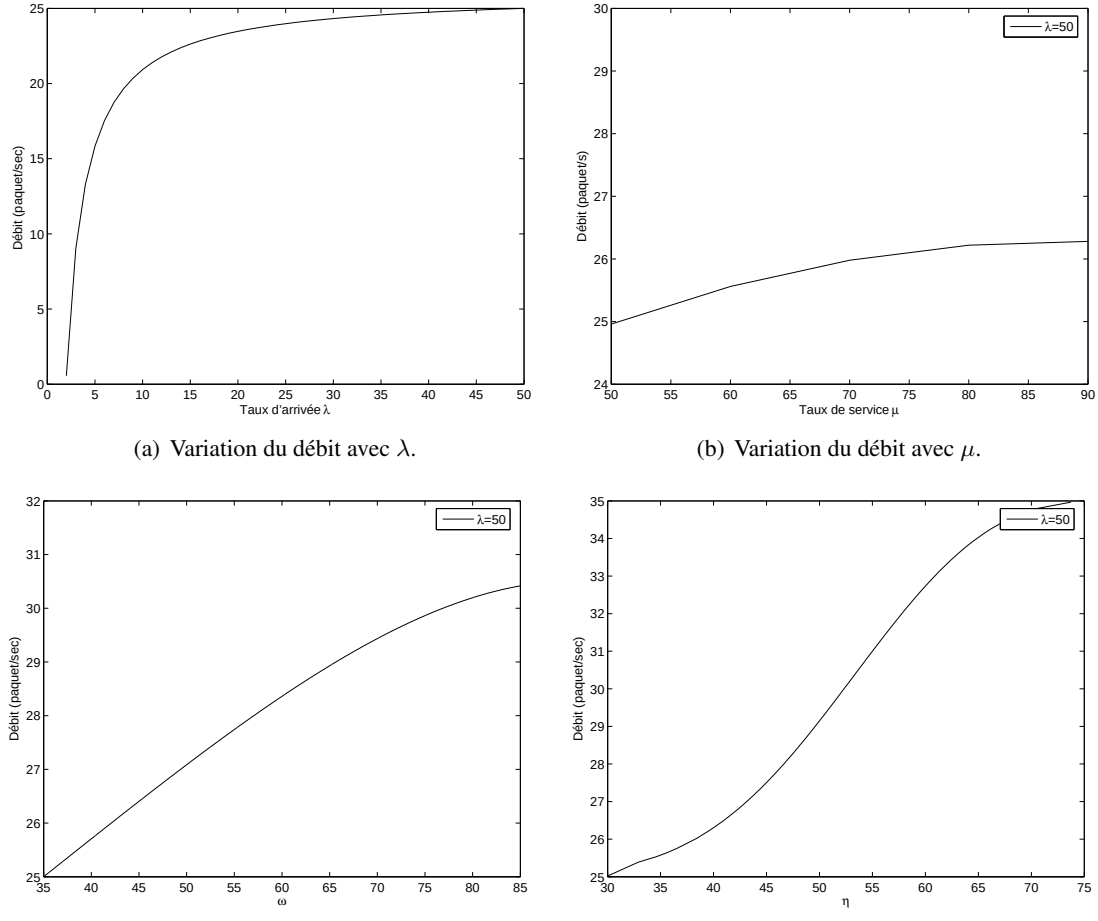
Pour réaliser une étude qualitative, nous avons eu recours au model-checking (CTL et $\mu$-calcul) pour vérifier la satisfaction de certaines contraintes fonctionnelles par les états du modèle. Nous avons essayé de vérifier quelques contraintes en utilisant les formules logiques des expressions suivantes :

- Absence d'états bloquants :
$$AG\langle -\rangle tt \text{ ou } AG\,(EXtt) \text{ ou } \neg\,(\min X = [-]ff \vee \langle -\rangle X)$$
- Il est possible d'atteindre un état où l'action $ckhdr$ sera exécutée :
$$EF(\langle ckhdr\rangle tt) \text{ ou } (\min Y = \langle ckhdr\rangle tt \vee \langle -\rangle Y).$$
- Dépassement de la file d'attente :
$$EF\,(\langle lose_H\rangle tt \vee \langle lose_L\rangle tt) \text{ ou } (\min Z = \langle lose_H\rangle tt \vee \langle lose_L\rangle tt \vee \langle -\rangle Z).$$
- Absence de famine :
$$(AG[deliver_H]AF\langle serve\rangle tt \wedge AG[deliver_L]AF\langle serve\rangle tt)$$
- Service selon la priorité des paquets :
$$AG\,([deliver_H]A\,([deliver_L]ff\ U\ \langle serve\rangle tt))$$

La première propriété est satisfaite par un état s'il exécute au moins une action ($\langle -\rangle tt$) (*cf.* figure 2.6(d) pour la signification de l'opérateur $AG$). La deuxième (resp. la troisième) propriété vérifie l'existence d'un chemin contenant un état, qui peut exécuter l'action $\langle chkhdr\rangle tt$ (resp. $\langle lose_i\rangle tt$). Ces deux propriétés vérifient qu'il est possible d'atteindre un état où l'action de classification (suppression) sera exécutée dans ce modèle. La $4^{ième}$ propriété vérifie une séquence des exécutions des actions, et la $5^{ième}$ propriété signifie qu'une fois l'action $deliver_H$ est exécutée, l'action $deliver_L$ ne peut pas être exécutée avant l'exécution de l'action $serve$ et vice versa.

**Évaluation de performance**

Dans un premier temps, nous nous sommes intéressés au débit du système, en augmentant la valeur du taux d'arrivée $\lambda$ des paquets dans le composant de classification. La figure 2.12(a) montre que le débit du système augmente assez rapidement pour atteindre un seuil de 25 paquets/s. Au-delà d'un taux d'arrivée de 35 paquets/s, le débit ne change quasiment pas. Pour cela, nous avons commencé à nous interroger sur les composants limitant le débit du système. Nous avons commencé à augmenter la vitesse de service de l'ordonnanceur pour un taux d'arrivée de 50 paquets/s. Le débit augmente d'une façon non significative, où il est toujours aux alentours de 25 paquet/s (*cf.* figure 2.12(b)). Cela est dû aux mauvaises valeurs choisies pour la temporisation des actions des autres composants, ou plutôt le dimensionnement de leur vitesses de traitement. A la fin d'une série de tests pour augmenter le débit, on a pu constater la variation du débit avec la vitesse de traitement du composant de marquage et de l'élément de mesure qui ont été très mal choisis au départ, ce qui provoque un blocage lors de la synchronisation avec les autres composants. Les courbes de variation de débit en fonction de la vitesse de traitement sont données dans la figure 2.12.




(a) Variation du débit avec $\lambda$.

(b) Variation du débit avec $\mu$.

(c) Variation du débit avec $\omega$.

(d) Variation du débit avec $\eta$.

Figure 2.12 – Variation du débit en fonction de la vitesse de traitement des composants.

## 2.9 Conclusion

Dans ce chapitre, nous avons présenté la notion de base des trois algèbres de processus stochastiques, ainsi qu'une étude comparative lors de l'utilisation de l'opérateur de composition parallèle. Ensuite, nous

avons abordé les méthodes de vérification qualitatives et quantitatives, utilisées lors de l'analyse d'un modèle algébrique.

Un modèle abstrait de routeur DiffServ est présenté comme un exemple d'application, afin d'illustrer l'intérêt de la modélisation modulaire, où des comportements complexes peuvent être exprimés grâce à des composants qui se synchronisent pour l'exécution d'une tâche commune. Après la description algébrique de ce système, nous avons utilisé les modèles logiques pour détecter les erreurs et les comportements anormaux dans le fonctionnement du modèle algébrique conçu. Puis une étude quantitative a été réalisée en utilisant la méthode algébrique basée sur les récompenses afin d'évaluer les paramètres de performances. Ces deux techniques de vérification permettent une automatisation de l'analyse des diagrammes de transitions.

Un certain nombre de conclusions peuvent être tirées de ce modèle, comme l'étude de l'influence de la vitesse de traitement de chaque composant sur les performances du modèle, pour éviter l'utilisation des composants rapides lorsqu'ils sont placés en synchronisation avec d'autres composants lents, dans le but d'obtenir un système performant avec un coût minimal.

Bien que nous ne prétendions pas que ce chapitre ajoute de nouveaux concepts, il contribue certainement à la clarification des certaines notions utilisées tout au long de ce manuscrit. Dans le chapitre suivant, nous présentons l'approche de la spécification des systèmes dont les délais des activités suivent une distribution générale. Nous détaillons particulièrement les aspects de distributions phase-type, leurs représentations minimale, ainsi que les problèmes inhérents à leur utilisation et les solutions que l'on peut apporter dans ce domaine.

# Chapitre 3

# Modélisation avec les distributions générales

## 3.1 Introduction

Dans les algèbres des processus stochastiques, la spécification temporelle est généralement limitée par l'approximation des délais d'exécution de toutes les actions d'un système, aux distributions Markoviennes (sans mémoire), pour transformer le diagramme temporel sous-jacent en une chaîne de Markov, et ceci pour pouvoir utiliser les méthodes numériques existantes pour évaluer les performances. Cette approximation au niveau de la fonction de distribution de délai d'exécution des actions à une loi exponentielle, laisse beaucoup de questions sur l'exactitude des indices de performance obtenue, parce que les systèmes réels s'engagent dans des activités qui ne sont pas forcement sans mémoire, et limite l'expressivité et les domaines d'application de ces formalismes.

Pour remédier à cet handicap, beaucoup de recherches ont été consacrées à la modélisation temporelle avec des distributions générales, que ce soit au niveau de la sémantique ou au niveau de la résolution numérique de la chaîne sous-jacente. Mais les réponses apportées à ce niveau sont loin d'être satisfaisantes.

## 3.2 Avantages de la distribution exponentielle

La restriction de la fonction de distribution associée aux délais d'exécution des actions à une loi exponentielle, est essentielle pour la sémantique des opérateurs des formalismes algébriques. En effet, les APS tirent profit de l'absence de mémoire de cette distribution, pour traiter le parallélisme par entrelacement, tout en exploitant le théorème de l'expansion de l'algèbre de processus classique :

$$a.0||b.0 = a.(0||b.0) + b.(a.0||0)$$

Ce théorème d'expansion est resté valable dans ce formalisme stochastique grâce à la propriété sans mémoire, qui rendait inutile l'enregistrement du temps écoulé par une action dans les états passés (*cf.* figure 3.1(a)) :

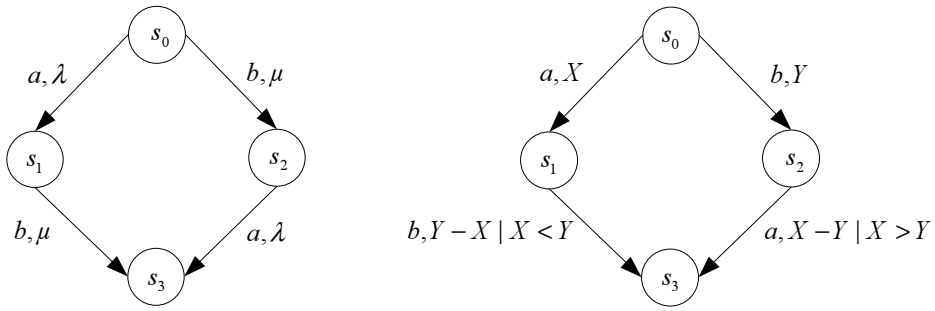
$$(a, \lambda).0||(b, \mu).0 = (a, \lambda).(0||(b, \mu).0) + (b, \mu).((a, \lambda).0||0)$$

Le remplacement de la distribution exponentielle par une distribution générale rend faux le théorème de l'expansion :

$$(a, X).0||(b, Y).0 \neq (a, X).(0||(b, Y).0) + (b, Y).((a, X).0||0)$$

Ceci est dû au fait du temps écoulé lors l'exécution d'une action comme le montre la figure 3.1(b). L'équation précédente doit être corrigée pour prendre en compte une probabilité conditionnelle selon la forme suivante :

$$(a, X).0||(b, Y).0 = (a, X).(0||(b, Y - X|X < Y).0) + (b, Y).((a, X - Y|X > Y).0||0)$$




(a) Avec une distribution exponentielle.

(b) Avec une distribution générale.

Figure 3.1 – Entrelacements des actions $a$ et $b$.

Dans la figure 3.1, dont les états sont donnés dans le tableau 3.1, on veut rappeler la prise en compte de l'écoulement du temps (que l'on qualifie par l'acronyme anglo-saxon « enabling memory ») dans le modèle à distribution générale, où deux actions sont actives simultanément dans l'état $s_0$. L'exécution de l'action dont la variable aléatoire est la plus petite (stratégie de course), après l'expiration de son délai, exige la prise en compte du temps écoulé par l'autre action, dont le délai n'a pas encore expiré (Remaining Firing Time – $RFT > 0$). La chaîne sous-jacente n'est plus une chaîne de Markov, même pas une chaîne semi-Markovienne car les délais d'exécution des actions dépendent non seulement du dernier état visité, mais aussi de tous les états précédents où cette action a continué d'être activée sans expiration de son délai d'exécution. C'est donc une chaîne générale (GSMP [Mat62]). Les distributions de probabilités continues assurent que deux actions ne peuvent pas s'exécuter au même instant, suite au fait que leurs variables aléatoires (leurs délais) sont continues. Pour cela, nous n'avons pas pris en compte la possibilité d'expiration des délais des 2 actions en même temps. En revanche, l'exécution simultanée est possible en cas de distributions discrètes. Par conséquent, l'obstacle principal pour la solution analytique est la présence du $RFT$ dans les états pour les actions qui ont été activées sans exécution.

| États avec des distributions exponentielles | États avec des distributions générales |
|---|---|
| $s_0 = (a, \lambda).0 \|\| (b, \mu).0$ | $s_0 = (a, X).0 \|\| (b, Y).0$ |
| $s_1 = 0 \|\| (b, \mu).0$ | $s_1 = 0 \|\| (b, Y - X).0$ |
| $s_2 = (a, \lambda).0 \|\| 0$ | $s_2 = (a, X - Y).0 \|\| 0$ |
| $s_3 = 0 \|\| 0$ | $s_3 = 0 \|\| 0$ |

Tableau 3.1 – Les états du diagramme de transition.

## 3.3 Processus semi Markovien

Le processus semi Markovien (SMP) est une généralisation de la chaîne de Markov, en éliminant la restriction sur les distributions sans mémoire. Le temps de séjour dans un état est distribué selon une loi générale avec la restriction de réinitialisation de tous les délais après l'occurrence d'une transition (ou bien le changement d'état). L'état futur dépend non seulement de l'état présent, mais aussi du temps de séjour :

$$Pr(X_{n+1} = j, T_{n+1} - T_n \leqslant t | X_n = i, X_{n-1} = i - 1, \ldots, X_0 = 0) =$$
$$Pr(X_{n+1} = j, T_{n+1} - T_n \leqslant t | X_n = i)$$

Avec :

$$Pr(X_{n+1} = j, T_{n+1} - T_n \leqslant t | X_n = i) =$$
$$Pr(X_{n+1} = j | X_n = i).Pr(T_{n+1} - T_n \leqslant t | X_{n+1} = j, X_n = i) = p_{ij}.H_{ij}(t)$$

$H_{ij}(t)$ est le temps de séjour dans l'état $i$ sachant que $j$ est l'état suivant.

Dans un système réel, la distribution des délais d'exécution de toutes les actions n'est pas forcément exponentielle, comme dans la CMTC, ou générale avec une stratégie qui exige la réinitialisation de toutes les variables aléatoires « pre-emptive restrat » comme dans SMP. Quand la distribution générale sans aucune restriction est utilisée, la chaîne sous-jacente est une chaîne semi Markovienne généralisée (GSMP) où la résolution numérique est difficile et complexe.

## 3.4 Processus semi Markovien généralisé

La chaîne GSMP a été introduite dans [Mat62], pour permettre de modéliser plusieurs éléments actifs dans un état, dont la durée de vie de chacune est distribuée selon les lois générales. Cette chaîne est constituée de l'ensemble $S = \{s(t) : t \geqslant 0\}$ décrivant les états d'un système à tout instant $t$, et l'ensemble des événements $E$. À un instant $t$, le processus occupe l'état $s \in S$, où l'ensemble $E_s = \{e_1, e_2, \ldots, e_m\}$ représente les événements actifs dans ce dernier. Chaque événement $e \in E$ est associé à une variable aléatoire réelle $t_e$ distribuée selon une loi de distribution générale $G_e$. Cette variable représente le temps pendant lequel l'événement $e$ doit rester activé avant son exécution. L'événement qui s'exécute est celui qui voit son délai d'exécution expirer en premier. L'exécution de cet événement déclenche une transition vers un état $s' \in S$ sans désactiver les autres événements qui ont été activés dans l'état $s$. Cette chaîne ne possède pas la propriété de Markov et prend en compte les temps écoulés par un événement activé dans tous les états où il est resté actif mais sans exécution.

Généralement, le processus commence dans l'état initial $s_0$ avec un ensemble d'événements actifs $E_{s_0}$. Pour chaque $e \in E_{s_0}$, la valeur de la variable aléatoire associée $(t_e)$ est initialisée selon la distribution $G_e$. En revanche, la variable aléatoire des événements inactifs est initialisée à $t_e = \infty$. Soit $e^*$ l'événement dans $E_{s_0}$ ayant la plus petite v.a., i.e. $e^* = arg_{min_{e \in E_{s_0}}} t_e$. Cet événement $e^*$ déclenche une transition de l'état $s_0$ à un état $s'$ après $t_{e^*}$ unités de temps. Après l'occurrence d'une transition et le changement d'état, chaque événement met à jour son temporisateur $t_e$ dans l'état $s'$ selon les conditions suivantes :

1. Si $e \in E_{s'} \cap (\{e^*\} \cup (E \backslash E_s))$, alors $t'_e$ est initialisé selon la loi de distribution $G_e$.
2. Si $e \in E_{s'} \cap (E_s \backslash \{e^*\})$, alors $t'_e = t_e - t_{e^*}$.
3. Si $e \notin E_{s'}$, alors $t'_e = \infty$.

Le premier point couvre les événements qui sont activés dans $s'$, après leur exécution ou probablement préemption dans $s$. Les événements qui restent activés dans l'état $s'$ sans exécution, mettent à jour leur délai comme suit : $t'_e = t_e - t_{e^*}$ (point 2). Le troisième point concerne l'initialisation des événements non activés dans l'état $s'$. Nous répétons cette procédure après l'occurrence d'une transition avec $s' = s''$ et $t'_e = t''_e$ tant que $E_s \neq \emptyset$. La séquence d'exécution des événements est étiquetée avec le temps et le nom de l'événement qui a déclenché cette transition :

$$\sigma = s_0 \xrightarrow{t_0, e_0} s_1 \xrightarrow{t_1, e_1} s_2 \xrightarrow{t_2, e_2} \cdots$$

Avec $s_i \in S$, $e_i \in E_{s_i}$ et $t_i > 0$ le temps de séjour dans l'état $s_i$. A noter que GSMP est équivalent à une chaîne de Markov à temps continu si toutes les distributions sont exponentielles.

La résolution d'une chaîne généralisée n'est pas toujours possible avec les méthodes numériques. La simulation à événements discrets (DES) est l'ultime recours pour la résolution de cette chaîne. Mais la précision des résultats obtenus est largement dépendante du temps de simulation. Les méthodes numériques basées sur l'analyse du diagramme de transitions sous-jacent restent beaucoup plus précises et plus rapides que la simulation.

L'introduction des distributions générales nécessite un changement au niveau sémantique de tous les opérateurs et plutôt la conception d'un nouveau formalisme algébrique pour exprimer directement un délai distribué selon une loi générale. Mais la distribution générale rend délicate la dérivation du diagramme temporel avec la nécessité de garder une trace de l'état où l'action a été activée jusqu'à son exécution.

Le problème de la spécification et de l'analyse qualitative et quantitative d'un système réel, avec des activités dont les délais suivent une distribution générale, a été étudié dans [Bra02, BBG98, D'A99, Str93, SH00] et fut le sujet de plusieurs formalismes (GSMPA [Bra02], SPADES [SH00], etc.), mais sans apporter une solution numérique applicable en pratique. Récemment, Bravetti dans [BG02] a introduit le formalisme IGSMPA (Interactive Generalised Semi Markov Process Algebra) comme une extension de l'algèbre des processus stochastiques. Ce formalisme inclut tous les opérateurs et permet la spécification de distributions générales directement au haut niveau. Pour garder les traces de chaque action, il a introduit un mécanisme d'identification de l'activation (début) et de l'exécution (fin) de chaque action d'une manière dynamique au niveau de la sémantique. Mais, contrairement aux algèbres de processus stochastiques, le choix entre deux actions est basé sur la politique de présélection, en associant une probabilité à chaque action. Ensuite, il a proposé une méthode pour transformer le diagramme obtenu en une chaîne semi Markovienne généralisée (GSMP) et exploite les techniques de résolution existant pour les cas particuliers afin de la résoudre. Dans [Bra04], il donne un exemple d'application la file $G/G/1/q$, et exploite la notion d'insensitivité, pour résoudre la chaîne GSMP dérivée de la spécification algébrique. La technique d'insensitivité a été exploitée par Clark [Cla99, CH02] pour la résolution d'une chaîne généralisée. Elle consiste à remplacer une distribution générale par une distribution exponentielle de même moyenne, si la chaîne n'est pas sensible à la distribution $G$. Une chaîne GSMP est dite insensitive à la forme de la distribution $G$, si le vecteur de distribution de probabilité stationnaire dépend seulement de la valeur moyenne de $G$. Clark dans [Cla00] est parti encore plus loin en proposant une méthode de construction d'un système qui garantit l'insensitivité de la chaîne résultante, ainsi qu'une solution sous forme produit [CH02]. Mais, les restrictions imposées par cette méthode limitent son domaine d'application à des cas très particuliers.

Aujourd'hui, trois méthodes sont utilisées pour la résolution d'une chaîne avec des distributions générales : la méthode de la variable supplémentaire [Ger00, GL94], la méthode de la chaîne de Markov semi-régénérative [GLT95, CKT94, LTP95] et les distributions de phase-type [Neu81, Neu75].

Dans la première méthode, des variables supplémentaires à valeur continue, représentant l'écoulement du temps, sont associées aux états. Ces dernières permettent la dérivation de séries d'équations différentielles. Cette méthode n'impose pas de restrictions au niveau du nombre d'actions activées dans un état, et dont les délais suivent une distribution générale. Mais sa complexité spatiale et temporelle limite son utilisation à de petits modèles.

La deuxième méthode est basée sur l'identification des instants et des états de régénération, où tous les délais seront réinitialisés. Généralement, l'utilisation de cette méthode nécessite des restrictions au niveau du nombre d'actions actives dans chaque état, leur instant d'activation, ainsi que leur politique d'exécution (préemptive restart, préemptive resume, etc.). Seule une action dont le délai suit une distribution générale peut rester active dans un état donné. Cette méthode a été largement utilisée dans l'analyse de la chaîne sous-jacente des différentes extensions des réseaux de Petri (ESPN [DTGN84], DSPN [Lin98, Ger00, MC87, DTGN84], MR-SPN [Lin98, Ger00, CKT94], CDSPN [PST98, Lin98, LTP95], CGPN [Lin98], etc.). Elle s'applique dans des cas très particuliers, rarement rencontrés lors de la spécification d'un système réel, car dans une chaîne de Markov régénérative, seule une action active dans un état peut être distribuée généralement en concurrence avec les autres qui devaient être exponentiellement distribuées.

La troisième méthode consiste à remplacer les distributions générales par une combinaison de distributions sans mémoire pour transformer la chaîne sous jacente en une chaîne de Markov. Mais contrairement aux deux méthodes précédentes qui permettent une analyse exacte, cette méthode augmente la taille de l'espace d'états et sa précision augmente avec le nombre des états investis.

## 3.5 Modèle symbolique avec horloge

Les premiers travaux que nous avons effectués pour prendre en compte les distributions générales dans les modèles algébriques sont introduits dans l'article [SB04b], qui était dédié à la description de l'utilisation des techniques symbolique et fonctionnelle pour la représentation du temps. La modélisation du protocole de routage DSR [JMH03] (Dynamic Source Routing) utilisé dans les réseaux ad hoc, est présenté comme un exemple d'application dans cet article. Le passage du temps est observé de manière discrète et fonctionnelle en utilisant la méthode de passage de valeurs, qui a été introduite dans EMPA par Bernardo [Ber97b], d'une façon similaire à celle utilisée en CCS [Mil89] et LOTOS [BB87]. Ce type d'extension permet l'association de variables à chaque processus, ainsi que l'échange des valeurs lors d'une synchronisation avec les opérateurs ! et ?, représentant respectivement la transmission et la réception de messages.

La syntaxe de l'extension du langage EMPA avec la technique de passage de valeur est donnée par l'expression suivante :

$$P := (a!(x), \infty_{p,m}).P \mid (a?(x), \infty_{p,m}).P \mid if\ (\beta, prob)\ then\ P\ else\ Q \mid A(loc\_par; loc\_var)$$

Cette extension ajoute à chaque action une information additionnelle représentant un message, et à chaque processus deux informations : paramètres et variables locaux. L'opérateur ! représente la transmission du message $x$ lors d'une synchronisation tandis que l'opérateur ? représente sa réception. L'opérateur conditionnelle $if\ (\beta, prob)\ then\ P\ else\ Q$ est un extension de l'opérateur de garde utilisé en CCS, qui signifie : si la condition $\beta$ est satisfaite avec une probabilité $prob$, alors l'exécution du processus $P$ aura lieu, sinon c'est le processus $Q$ qui sera exécuté. Cette expression conditionnelle est équivalente à l'opérateur de choix probabiliste $\oplus_p$ utilisé dans le chapitre 2.

$$if(\beta, prob)\ then\ P\ else\ Q \equiv (\tau, \infty_{1,prob}).P + (\tau, \infty_{1,1-prob}).Q$$

Le vecteur $loc\_par$ est utilisé pour sauvegarder des données spécifiques au processus, et le vecteur $loc\_var$ pour la transmission et réception des messages au cours d'une synchronisation avec un autre processus.

Pour illustrer cette stratégie, nous considérons le processus $transmetteur$ qui doit retransmettre un paquet si au bout de $n$ unités de temps, aucun accusé de réception n'est reçu, sinon transmettre le paquet suivant. La spécification de ce processus est donné par :

$$\begin{aligned} Trans(to = n, y) = \quad & (ra(?y), *).Transmission(n, y+1) + \\ & if\ (to > 0)\ then\ (tick, \infty_{1,prob}).Trans(to - 1, y) \\ & else\ (\tau, \infty_{1,1-prob}).(reémission, \infty_{1,1}).Trans(n, y); \end{aligned}$$

Reste à noter que l'outil $EMPA_{VP}$ offre la possibilité d'initialiser une variable avec un générateur de nombre aléatoire, selon une fonction de distribution générale. Ce qui permet de modéliser n'importe quel type de délai. La sémantique de cette extension est définie de manière symbolique : la chaîne sous-jacente est une CMTD, et les valeurs sont ajoutées sur l'étiquette d'une manière symbolique. Nous illustrons la représentation de l'expression algébrique dans l'équation (3.1)dans la figure 3.2. L'exécution de l'action $a$ engendre une transition vers le même état, mais avec l'incrémentation de la valeur de la variable symbolique $x$ dans le processus $A$.

$$A(x;) = a!(x).(A(x+1) + A(x+2)) \tag{3.1}$$

L'évaluation de performances ne peut se faire qu'à l'aide des outils de simulation, du fait de cette représentation symbolique, où les arcs sont étiquetés avec des expressions de distributions générales à évaluer. Cette méthode d'annotation symbolique évite une explosion de l'espace d'état. La résolution du modèle se fait à l'aide de la méthode de simulation à réplication indépendante, où à chaque itération d'une simulation, toutes les transitions de l'état courant sont calculées selon les sémantiques formelles,

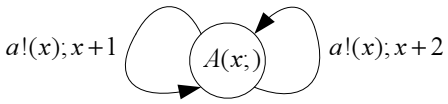



Figure 3.2 – Représentation symbolique.

et seule une seule transition est choisie selon la stratégie de présélection parmi toutes les transitions possibles.

Cependant que les techniques de vérification fonctionnelle continuent à s'appliquer sur le modèle obtenu, la complexité de la description qualitative du temps augmente de façon exponentielle avec le nombre d'actions. Ensuite, le système réel contient généralement des actions dont les délais sont à temps continu et discret. Ce qui limite le domaine d'application ainsi que la précision espérée. Enfin, la simulation à réplication indépendante laisse beaucoup de points d'interrogation sur l'ordre de la complexité temporelle de la simulation et le nombre de répétitions pour obtenir des résultats fiables.

D'autres techniques ont été utilisées pour la solution analytique avec n'importe quelle distribution générale. Il s'agit notamment de la famille de distributions phase-type, utilisées pour remplacer n'importe quelle distribution générale, à travers une combinaison de distributions sans mémoire (exponentielles ou géométriques).

Cette technique transforme la chaîne GSMP en une CMTC ou CMTD, où les techniques numériques classiques peuvent être exploitées pour la résolution. La précision des approximations dépend du nombre de phases (états) investis dans l'approximation et la complexité dépend du nombre de paramètres de la distribution phase-type à rechercher. Cette technique n'a pas suscité beaucoup d'attention dans la modélisation algébrique, suite aux nombres d'états additionnels qu'elle engendre, et au problème d'explosion de l'espace d'état qui augmente de façon non linéaire avec les phases additionnelles.

Le reste de ce chapitre est consacré à l'approximation des distributions non-Markoviennes par des distributions Markoviennes, ainsi que leurs spécifications en haut niveau dans le formalisme algébrique. Les distributions phases ont été étudiées dans les réseaux de files d'attente et les réseaux de Petri, mais notre étude prend une direction complètement différente, envers la représentation avec le plus petit nombre de phases, et une complexité minimale dans la recherche des paramètres, ainsi que les problèmes de la spécification au haut niveau.

## 3.6 Les distributions Phase-type

Les distributions phase-type (à temps continu ou discret) ont été définies dans [Neu81, Neu75], pour l'approximation des distributions non Markoviennes par des distributions Markoviennes. Cette technique permet de modéliser un système avec des activités dont les délais suivent une distribution générale $G$, tout en substituant ces délais par une combinaison adéquate de phases Markoviennes.

Une distribution de phase-type $PH$ est définie par une chaîne de Markov finie, composée de plusieurs états transitoires (les phases), et d'un seul état absorbant. Soit $T$ le temps écoulé avant l'absorption, et $X(t)$ une variable aléatoire représentant l'état du système à l'instant $t$.

$$T = min\{t : X(t) = A\} \text{ où } A \text{ est l'état absorbant}$$

Soit $S$ l'espace d'état représentant la chaîne de Markov dont le $(n+1)^{ième}$ état est un état absorbant. Une distribution de phase-type à temps continu est représentée par $[a, Q]$, où $a$ est le vecteur de probabilité initiale et $Q$ la matrice générateur de dimension $(n \times n)$ de la chaîne sans l'état absorbant. Soit $q$ le vecteur colonne représentant les taux de transition de chaque état vers l'état absorbant, la matrice générateur de la chaîne est donnée par :

$$T = \begin{pmatrix} Q & q \\ 0 & 0 \end{pmatrix} \tag{3.2}$$

où :

$$q = -Qe \quad \text{et} \quad e = (1, \ldots, 1)^T$$

La fonction de distribution est donnée par $F(y) = 1 - ae^{Qy}e$, où $e^{Qy}$ est calculé par le biais de la série de Taylor/MacLaurin donnée par :

$$e^{Qy} = \sum_{n \geq 0} \frac{1}{n!}(Qy)^n \tag{3.3}$$

Soit $f(y)$ la fonction de densité de probabilité et $\mu_k$ le moment d'ordre $k$ donné par :

$$f(y) = ae^{Qy}q \tag{3.4}$$

$$\mu_k = \int_0^\infty y^k dF(y) = (-1)^k k! a Q^{-k} e \tag{3.5}$$

Le nombre de paramètres à rechercher est $(n+1)^2$ paramètres ($n^2 + n$ pour la matrice $T$ et $n+1$ pour le vecteur de probabilité initiale $p$).

D'une manière similaire, une distribution de phase-type à temps discrèt est représentée par $[a, P]$ où $P$ est la matrice de transition de la chaîne sans l'état absorbant. La matrice de transition correspondante est donnée par :

$$B = \begin{pmatrix} P & b \\ 0 & 1 \end{pmatrix} \tag{3.6}$$

$P = [p_{ij}]$ est une matrice de transition de dimension $(n \times n)$, et $b = [b_{i,k}]^T$ est le vecteur de probabilité de transition vers l'état absorbant $n+1$. Soit $F(k)$ la fonction de distribution, $f(k)$ la fonction de densité de probabilité et $\mu_k$ le moment d'ordre $k$ :

$$f(y) = \Pr\{\tau = y\} = aP^{y-1}b \qquad \text{pour } \; y > 0 \tag{3.7}$$

$$F(y) = a \sum_{i=0}^{y-1} P^i b = 1 - aP^y e \tag{3.8}$$

$$\mu_k = k! a (I - P)^{-k} P^{k-1} e \tag{3.9}$$

Les distributions de phase-type les plus utilisées dans la littérature sont : exponentielle, coxian, erlang, hypo-exponentielle, hyper-exponentielle et hyper-erlang (voir figure 3.3).

Mais, l'utilisation des distributions phase-type est souvent regardée comme étant le détonateur de l'explosion de l'espace d'états, suite au nombre d'états additionnels pour représenter les délais d'exécution d'une action. Il est largement reconnu dans les algèbres des processus stochastiques, que le nombre d'états augmente de manière exponentielle avec le nombre des processus en parallèle. Dans un système avec $n$ processus, dont chacun contient $m$ événements, l'espace d'état est de l'ordre $O(m^n)$. Par conséquent, l'approximation des distributions générales avec des distributions phase-type contenant $p$ phases, va augmenter la taille de la chaîne de Markov à l'ordre $O((m \times p)^{n \times p})$, ainsi que le nombre des transitions à cause de l'entrelacement entre les phases représentant les délais des actions en parallèle. Mais, la représentation d'une distribution générale par une distribution $PH$ n'est pas unique [Neu81], où plusieurs combinaisons des distributions phase-type de base peuvent être utilisées pour reproduire une distribution $G$, et différentes techniques d'approximation sont apparues pour la transformation d'une distribution générale $G$ en une distribution de phase-type $PH$. Pour cela, le premier souci est la recherche de la distribution $PH$ avec le nombre des phases minimales pour bien représenter une distribution générale $G$, et pour pallier ce problème de l'explosion de l'espace d'états.

Beaucoup des travaux ont été réalisés pour la recherche de la forme minimale représentant au mieux une distribution générale. Deux approches principales sont utilisées pour la représentation de $G$ par $PH$ minimale : la première est basée sur le principe du maximum de vraisemblance (Maximum Likelihood – ML), et la deuxième méthode est basée sur le principe de l'égalité des moments.

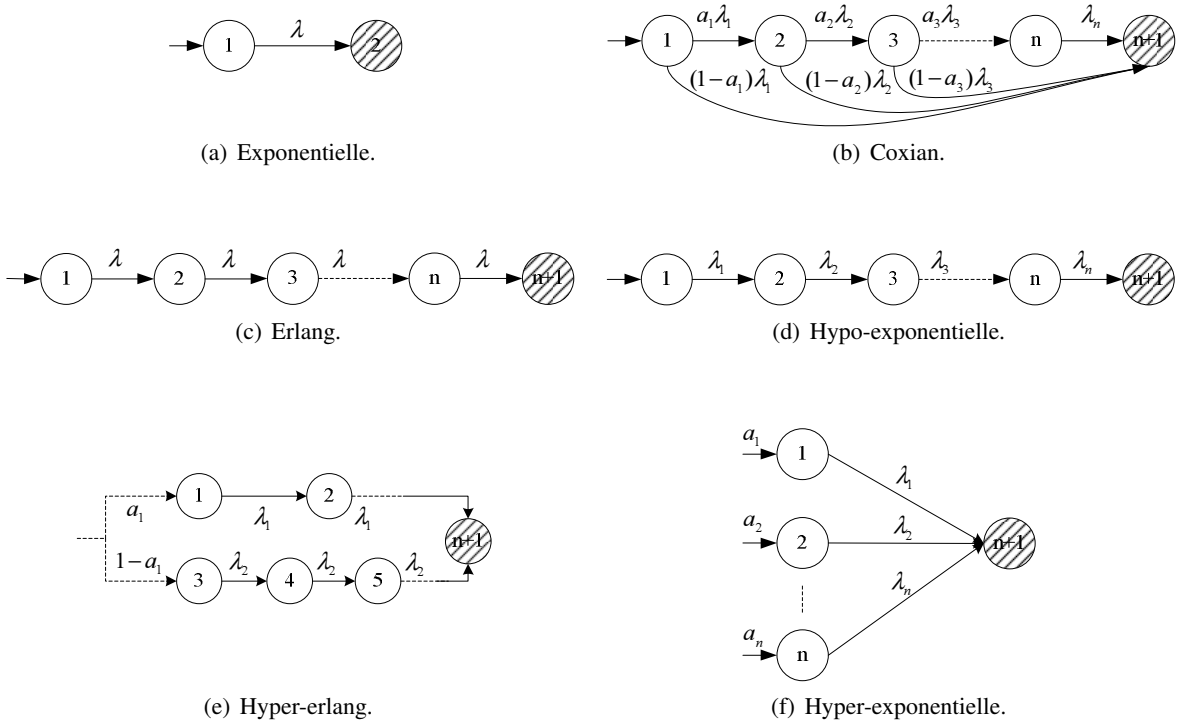



Figure 3.3 – Distributions Phase-type.

## 3.7 Estimation des paramètres avec ML

Quand les distributions phase-type sont utilisées pour remplacer une distribution générale, la première question est l'algorithme de recherche des paramètres des matrices dans (3.2) ou (3.6), ainsi que les éléments du vecteur des probabilités initiales $a$ d'une forme générale. La méthode ML et son extension EM [DLR77, KSH03] (Expectation–Maximization) ont été utilisées pour la recherche de ces paramètres, et implémentées dans l'outil *EMpht* [ANO96], mais la complexité temporelle de convergence de l'algorithme EM avec tous les paramètres de la matrice augmente de façon non linéaire avec le nombre des informations non-observées, qui sont utilisées dans l'algorithme EM, en plus de nombre d'itérations requises dans la méthode Range-Kutta d'ordre 4 utilisée.

Des études [Cum82, Hor03] ont montré que les représentations dans (3.2) et (3.6), ne sont pas les mieux adaptés avec une complexité de recherche minimale, suite au nombre élevé des paramètres. Ceci a mené à l'étude d'une sous-classe de distributions $PH$ appelées les distributions acycliques ($APH$) avec moins des paramètres.

**Définition 3.1.** *une chaîne de Markov avec un seul état absorbant est dite acyclique, si tout état n'est jamais visité plus d'une fois comme montre la figure 3.4, autrement dit il est impossible de retourner dans un état après l'avoir quitté. Les états $i$ peuvent être directement connectés à l'état $j$ si et seulement si $j \geq i$.*

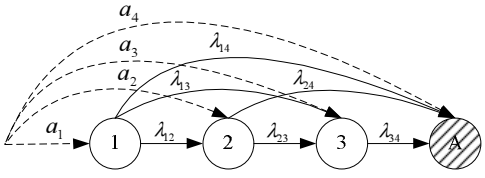
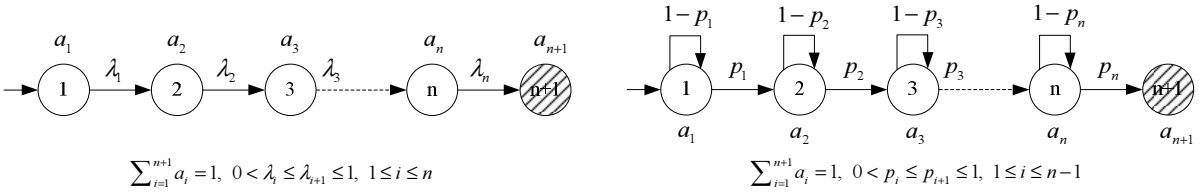



Figure 3.4 – Représentation acyclique.

En utilisant la conjecture que chaque distribution de phase d'ordre $n$ a une représentation acyclique

d'ordre $n$, l'intérêt de l'utilisation d'une chaîne acyclique est la transformation de la matrice en une matrice triangulaire supérieure. Ce qui réduit le nombre des paramètres à rechercher de $N_p = (n+1)^2$ à $N_p = (n^2+3n)/2$ (dont $n(n+1)/2-1$ pour la matrice et $(n+1)$ pour le vecteur de probabilité initiale). On voit clairement que le nombre des paramètres a réduit, mais l'ordre de complexité est resté polynômial $O(n^2)$.

La forme canonique a été introduite par Cumani [Cum82] pour les distributions continues (*cf.* figure 3.5(a)), et utilisée par Horvàth [Hor03] pour les distributions discrètes (*cf.* figure 3.5(a)). Toute chaîne acyclique d'ordre $n$ a une forme canonique de même ordre, ce qui réduit le nombre des paramètres à rechercher relativement à la représentation acyclique, ainsi que la complexité de recherche des paramètres.




(a) CPH. (b) DPH.

Figure 3.5 – Représentation canonique et acyclique.

| | *CPH acyclique* | *CPH canonique* |
|---|---|---|
| *Probabilité initiale* | $a = [a_1, a_2, \ldots, a_{n+1}]$ | $a = [a_1, a_2, \ldots, a_{n+1}]$ |
| *Matrice* | $T = \begin{bmatrix} -\sum_{j=2}^{n+1} \lambda_{1,j} & \lambda_{1,2} & \ldots & \lambda_{1,n+1} \\ 0 & -\sum_{j=3}^{n+1} \lambda_{2,j} & \ldots & \lambda_{2,n+1} \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \ldots & \lambda_{n,n+1} \\ 0 & 0 & \ldots & 0 \end{bmatrix}$ | $T = \begin{bmatrix} -\lambda_1 & \lambda_1 & 0 & \ldots & 0 \\ 0 & -\lambda_2 & \lambda_2 & \ldots & 0 \\ \vdots & \vdots & \ddots & & \vdots \\ 0 & 0 & \ldots & -\lambda_n & \lambda_n \\ 0 & 0 & \ldots & 0 & 0 \end{bmatrix}$ |
| | *DPH acyclique* | *DPH canonique* |
| *Matrice* | $B = \begin{bmatrix} 1-\sum_{i=1}^{n+1} p_{1,i} & p_{1,2} & \ldots & p_{1,n+1} \\ 0 & 1-\sum_{i=2}^{n+1} p_{2,i} & \ldots & p_{2,n+1} \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \ldots & p_{n,n+1} \\ 0 & 0 & \ldots & 1 \end{bmatrix}$ | $B = \begin{bmatrix} 1-p_1 & p_1 & 0 & 0 & \ldots & 0 \\ 0 & 1-p_2 & p_2 & 0 & \ldots & 0 \\ \vdots & \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & 0 & \ldots & p_n \\ 0 & 0 & 0 & 0 & \ldots & 1 \end{bmatrix}$ |

Le nombre de paramètres est réduit à $N_p = 3n+1$ ($2n$ pour la matrice et $(n+1)$ pour le vecteur de probabilité initiale). L'ordre de complexité n'est plus polynômial, mais linéaire $O(n)$. L'algorithme d'estimation des paramètres des distributions phase-type à temps continu est donné dans [BC92], et celui à temps discret dans [Hor03]. Cet algorithme prend en entrée le nombre de phases $n$ souhaité, la distribution générale ou des échantillons, ainsi qu'une certaine valeur $\varepsilon$ de tolérance des erreurs entre une caractéristique de la distribution originale et celle de son approximation. Il retourne les paramètres de la distribution canonique acyclique en utilisant le principe du maximum de vraisemblance (ML) et la méthode d'optimisation linéaire de Nelder-Mead (méthode du simplex). Étant donné $n$ échantillons $(x_1, \cdots, x_n)$ et la forme analytique de la densité de probabilité de la distribution ACPH (*cf.* équation 3.4),

la fonction ML prend la forme suivante :

$$\mathcal{L}(x_1, \ldots, x_n | a_i, \lambda_i) = \prod_{i=1}^{n} f(x_i | a_i, \lambda_i)$$

$$\log \mathcal{L}(X | a_i + \Delta, \lambda_i + \Delta) = \log \mathcal{L}(X | a_i, \lambda_i) + \frac{\partial log \mathcal{L}(X | a_i, \lambda_i)}{\partial a_i} \Delta a_i + \frac{\partial \log \mathcal{L}(X | a_i, \lambda_i)}{\partial \lambda_i} \Delta \lambda_i$$

Le problème de recherche des paramètres $a_i$ et $\lambda_i$ avec les deux conditions suivantes :

1. $0 \leqslant \lambda_1 \leqslant \lambda_2 \leqslant \cdots \leqslant \lambda_n$
2. $a_i \geqslant 0$, et $\sum_{i=1}^{n} a_i = 1$.

Ce problème de l'estimation non linéaire avec les deux contraintes précédentes et des valeurs initiales pour $[a^{(0)}, Q^{(0)}]$, se transforme en un problème d'optimisation linéaire, qui peut se résoudre avec les opérations itératives de la méthode du simplex. Les valeurs initiales sont choisies parmi les 1000 paires de valeurs générées aléatoirement, pour choisir celles qui réduisent au minimum l'erreur relative $\varepsilon$. Les valeurs obtenues dans cette étape sont utilisées comme valeurs initiales à la prochaine itération. La procédure est répétée jusqu'à taux de tolérance requis, ou lorsque le nombre maximum d'itérations est atteint. La tolérance $\varepsilon$ est définie en terme de différence relative entre la fonction originale (fonction de distribution $F(t)$, fonction de densité de probabilité $f(t)$, moment $\mu_i$, etc.) et celle de la distribution $PH$ obtenue ($\hat{F}(t)$, $\hat{f}(t)$, $\hat{\mu}_i$, etc.) comme le montre le tableau 3.2.

| *Tolérance $\varepsilon$* | *Temps continu* | *Temps discret* |
|---|---|---|
| *Entropie* | $\int_0^\infty f(t) \log\left(\frac{f(t)}{\hat{f}(t)}\right) dt$ | $\sum_{i=1}^\infty f_i \log\left(\frac{f_i}{\hat{f}_i}\right)$ |
| *Différence entre les fonctions de densité* | $\int_0^\infty \left\lvert \hat{f}(t) - f(t) \right\rvert dt$ | $\sum_{i=1}^\infty \left\lvert \hat{f}_i - f_i \right\rvert$ |
| *Carrés de Différence entre les fonctions de distribution* | $\int_0^\infty \left(\hat{f}(t) - f(t)\right)^2 dt$ | |

Tableau 3.2 – Erreur relative.

Les trafics du réseau sont normalement caractérisés par le temps d'inter-arrivée et les tailles de paquet, ces paramètres ont été largement modélisés par la fonction de Poisson. Mais des études récentes ont montré que l'arrivée du trafic dans les réseaux n'est pas Poissonienne, mais suit plutôt une distribution à décroissance lente (heavy-tailed distributions). Ces distributions sont définies mathématiquement par une fonction de puissance, donnée par :

$$CCDF = \bar{F}(X) = 1 - F(X) = Pr(X > x) \sim x^{-\alpha} \ \ quand \ \ x \to \infty \tag{3.10}$$

La représentation canonique ne permet pas de bien représenter la phase de décroissance lente (tail) de ce type des distributions, où la représentation de ce type des distributions est réalisée en séparant le corps et la phase de décroissance lente. Conscient de ce problème, Telek et Horvàh dans [HT00] ont utilisé la distribution hyper-exponentielle en parallèle à la distribution canonique (ACPH) pour représenter la phase de décroissance lente (*cf.* à la figure 3.6) tout en la séparant du corps.

Le coefficient de variation est souvent utilisé comme un indicateur du degré de variabilité d'une distribution, ce dernier est donné par :

$$cv^2 = \frac{Var(X)}{E(X)^2} = \frac{\sigma^2}{\mu_1^2} = \frac{\mu_2}{\mu_1^2} - 1 \tag{3.11}$$

Ce coefficient est égal à 1 ($cv^2 = 1$) pour la distribution exponentielle, il est $cv^2 > 1$ pour les distributions à grande variabilité et $cv^2 < 1$ pour les distributions à faible degré de variabilité. Nous allons

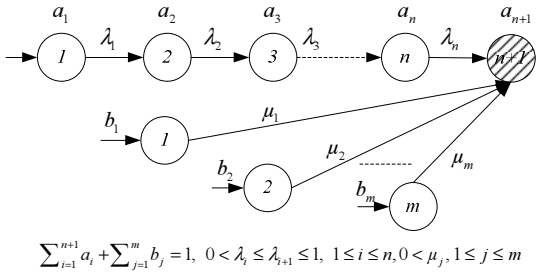



Figure 3.6 – Approximation des distributions à décroissance lente.

voir dans la section 3.8, que la distribution hyper-exponentielle a un $cv^2 > 1$ et la distribution d'Erlang a le plus petit coefficient de variation. Nous allons exploiter ces 2 distributions pour la représentation d'une distribution générale par une distribution Hyper-Erlang (*cf.* figure 3.7) avec $N$ phases, dont chaque branche contient $I_i$ phases.

$$N = \sum_{i=1}^{B} I_i$$

où $B$ est le nombre total des branches, autrement dit le nombre des distributions Erlang en parallèles.

Cette distribution acyclique intègre la distribution d'Erlang pour $B = 1$ (dont le $cv^2 < 1$), hyper-exponentielle pour $I_i = 1$ et $B > 1$ ($cv^2 > 1$), et exponentielle pour $B = I_B = 1$ ($cv^2 = 1$). Son coefficient de variation peut se régler pour varier de $1/N$ à $\infty$, et les résultats obtenus montrent que cette distribution permet de bien représenter les distribuions « heavy-tailed » sans se soucier du nombre de phases utilisées pour représenter la décroissance lente de telles distributions.

### 3.7.1 La distribution Hyper-Erlang

Soit $X$ une variable aléatoire suivant la distribution Hyper-Erlang avec une probabilité initiale $a_i$ de choisir une variable aléatoire $X_i$ suivant une distribution d'Erlang-$I_i$ de paramètre $\lambda_i$. Cette distribution (donnée dans la figure 3.7) n'est autre qu'une composition mixte de $B$ distributions d'Erlang, dont chacune contient $I_i$ phases. Pour la variable $X$, nous utilisant la notation $HEr_{B,I}(a_i, \lambda_i)$ pour représenter cette distribution. La fonction de densité de probabilité (pdf), la fonction de distribution (cdf), et le moment d'ordre $k$ sont données par :

$$f(x) = \sum_{i=1}^{B} a_i \frac{\lambda_i^{I_i}}{(I_i - 1)!} t^{I_i - 1} e^{-\lambda_i x} \tag{3.12}$$

$$F(x) = 1 - \sum_{i=1}^{B} a_i e^{-\lambda_i x} \sum_{j=1}^{I_i - 1} \frac{(\lambda_i x)^j}{(j+1)!} \tag{3.13}$$

$$E(x^k) = \sum_{i=1}^{B} a_i \lambda_i^{-k} \frac{(I_i + k - 1)!}{(I_i - 1)!} \tag{3.14}$$

Où $x \geq 0$, $\lambda_i > 0$, $0 \leq a_i \leq 1$, et $\sum_{i=1}^{B} a_i = 1$.

### 3.7.2 Estimation des paramètres de la distribution $HEr_{B,I_i}$

l'algorithme EM est une méthode itérative utilisée pour la recherche des paramètres d'une fonction de distribution $f(X|\lambda_i)$ étant donné un certain nombre d'observations partielles [KSH03]. Dans notre

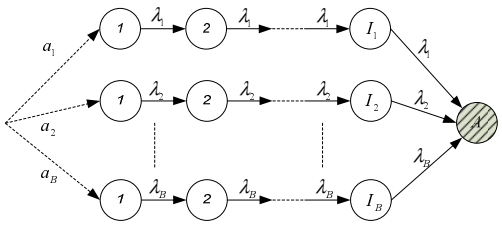


Figure 3.7 – Diagramme de transitions de la distribution $HEr_{B,I_i}$.

scénario, la fonction $f(X|\theta)$ a deux ensembles de paramètres dans $\theta = (a_i, \lambda_i)$, $X$ étant l'ensemble des observations de taille $n$ $(x_1, \ldots, x_n)$. La fonction de densité de la distribution $HEr_{B,I_i}$ est donnée par :

$$f(X|\theta) = \sum_{i=1}^{B} a_i f_i(X|\lambda_i) \tag{3.15}$$

Où $f_i(X|\theta_i)$ est la fonction densité de probabilité (pdf) de la distribution d'Erlang avec un taux $\lambda_i$. La fonction de vraisemblance $\mathcal{L}$ est donnée par :

$$f(X|\theta) = \prod_{j=1}^{n} \sum_{i=1}^{B} a_i f_i(x_j|\lambda_i) = \mathcal{L}(\theta|X) \tag{3.16}$$

L'objectif est la recherche de l'ensemble des paramètres de $\theta$ qui maximise la vraisemblance $\mathcal{L}$, ou plutôt $\hat{\theta} = (\hat{a}_i, \hat{\lambda}_i)$ :

$$\hat{\theta} = \arg\max_{\theta} \log \mathcal{L}(\theta|X) \tag{3.17}$$

$$\log \mathcal{L}(\theta|X) = \log \prod_{j=1}^{n} \sum_{i=1}^{B} a_i f_i(x_j|\lambda_i) = \sum_{j=1}^{n} \log \sum_{i=1}^{B} a_i f_i(x_j|\lambda_i) \tag{3.18}$$

La vraisemblance par calcul direct n'est pas possible dans l'équation précédente avec la fonction logarithme de la somme. L'algorithme EM suppose l'existence de certaines données cachées (ou non observables) $y_k \in \{1, 2, \ldots, B\}$ représentant le chemin emprunté avant l'absorption, pour la simplification du calcul comme le montre l'équation suivante :

$$\log \mathcal{L}(\theta|X, Y) = \log \prod_{j=1}^{n} \sum_{i=1}^{B} a_i f_i(x_j|\lambda_i) = \sum_{j=1}^{n} \log \left(a_{y_j} f_{y_j}(x_j|\lambda_{y_j})\right) \tag{3.19}$$

Soit $g(y)$ la densité de probabilité de $y$, cette fonction est calculée avec les lois de Bayes dans l'algorithme EM. Cet algorithme EM procède en 2 étapes :

**Etape E (Expectation) :** calcul de $\hat{\theta}$ à partir de $Q(\theta, \hat{\theta}) = E(\log \mathcal{L}(\hat{\theta}|X, Y)|X, \theta)$ avec $\mathcal{L}(\theta|X, Y)$ est la fonction de vraisemblance des données complètes $X$ et $Y$.

$$\begin{aligned}
Q(\theta, \hat{\theta}) &= E\left(\log \mathcal{L}(\hat{\theta}|X, Y)|X, \theta\right) = E\left(\log \mathcal{L}(\hat{\theta}|X, Y).g(Y|X, \theta)\right) \\
&= \sum_{j=1}^{B} \left(\log \mathcal{L}(\hat{\theta}|X, Y).g(Y|X, \theta)\right) = \sum_{j=1}^{B} \left(\sum_{i=1}^{n} \log\left(a_j.f_j(x_i|\lambda_j)\right).g(j|x_i, \theta)\right) \\
&= \sum_{j=1}^{B} \left(\sum_{i=1}^{n} \log(a_j) + \log(f_j(x_i|\lambda_j)\right).g(j|x_i, \theta) \\
&= \sum_{j=1}^{B} \sum_{i=1}^{n} \log(a_j).g(j|x_i, \theta) + \sum_{j=1}^{B} \sum_{i=1}^{n} \log(f_j(x_i|\lambda_j).g(j|x_i, \theta)
\end{aligned} \tag{3.20}$$

Où :

$$g(j|x_i,\theta) = \frac{a_j.f_j(x_i|\lambda_j)}{\sum_{i=1}^{B} a_i.f_i(x_j|\lambda_i)} = \frac{a_j.f_j(x_i|\lambda_j)}{f(x_j|\theta)} \tag{3.21}$$

L'algorithme itératif EM utilise la valeur de $\hat{\theta}$ obtenue dans l'itération $k$ ($\hat{\theta}^k$) comme valeur initiale dans l'itération $k+1$ (pour le calcul de $\hat{\theta}^{k+1}$).

**Etape M (Maximisation) :** Maximisation de $Q(\theta,\hat{\theta})$ par rapport à $\theta$ :

$$\begin{aligned} \hat{\theta}^{k+1} &= \arg\max_{\theta} Q(\theta,\hat{\theta}^k) \\ &\Rightarrow \frac{\partial Q}{\partial \hat{\theta}} = 0 \\ &\Rightarrow \frac{\partial Q}{\partial a_i^{k+1}} = 0 \text{ et } \frac{\partial Q}{\partial \lambda_i^{k+1}} = 0 \end{aligned} \tag{3.22}$$

Pour maximiser $Q(\theta,\hat{\theta}^k)$ dans l'équation 3.22, nous devons maximiser le premier terme contenant $a_i$ dans l'équation 3.20, puis le second terme contenant $\lambda_i$ d'une façon indépendante, car ils ne sont pas liés. Dans [KSH03], une extension de l'algorithme EM est présentée, où les paramètres $a_i$ sont calculés via le théorème des multiplicateurs de Lagrange (utilisé pour résoudre les problèmes d'optimisation sous contrainte), et ils sont donnés dans l'équation 3.23 :

$$\frac{\partial Q}{\partial a_i^{k+1}} = 0 \Rightarrow a_i^{k+1} = \frac{1}{n}\sum_{l=1}^{n} \frac{a_i^k f_i(x_l|\lambda_i^k)}{\sum_{j=1}^{B} a_j.f_j(x_l|\lambda_j^k)}, i = 1,\ldots,B \quad \text{où } \sum_{i=1}^{B} a_i^{k+1} = 1 \tag{3.23}$$

Les paramètres $\lambda_i$ sont calculés à travers la dérivé du deuxième terme dans l'équation 3.20 :

$$\frac{\partial Q}{\partial \lambda_i^{k+1}} = 0 \Rightarrow \sum_{i=1}^{n} g(j|x_i,\theta)\frac{\partial}{\partial \lambda_j}\log\left(f_j(x_i|\lambda_j)\right) = 0 \tag{3.24}$$

où $f_j(x_i|\lambda_j)$ est la fonction de densité d'Erlang de la $j^{ième}$ branche, donnée par :

$$f_j(x_i|\lambda_j) = \frac{(\lambda_j^{I_j})}{(I_j-1)} x^{I_j-1} e^{-\lambda_j.x_i} \tag{3.25}$$

Pour simplifier l'écriture de l'équation 3.24, nous utilisons la technique de changement de variable, en supposant que $U = f_j(x_i|\lambda_j)$. L'équation 3.24 prend la forme suivante :

$$\sum_{i=1}^{n} g(j|x_i,\theta)\frac{\partial}{\partial \lambda_j}\log(U) = 0 \Rightarrow \sum_{i=1}^{n} g(j|x_i,\theta)\frac{\partial U/\partial \lambda_j}{U} \tag{3.26}$$

La dérivé de la nouvelle variable $U$ par rapport à $\lambda_j$ est donnée par :

$$\frac{\partial U}{\partial \lambda_j} = \frac{\left(I_j.\lambda_j^{I_j-1} - \lambda_j^{I_j} x_i\right)}{(I_j-1)} x^{I_j-1} e^{-\lambda_j x_i} \tag{3.27}$$

et :

$$\frac{\partial U}{\partial \lambda_j}/U = \left(\frac{I_j}{\lambda_j} - x_i\right) \tag{3.28}$$

En remplaçant les équations 3.27 et 3.28 dans l'équation 3.26, on aura :

$$\sum_{i=1}^{n} g(j|x_i,\theta)\left(\frac{I_j}{\lambda_j} - x_i\right) = 0 \Rightarrow \lambda_j = \frac{I_j.\sum_{i=1}^{n} g(j|x_i,\theta)}{\sum_{i=1}^{n} g(j|x_i,\theta).x_i} \tag{3.29}$$

**ENTRÉES:** Les observations $X$ et le nombre de phases $N$.
**SORTIES:** La liste des paramètres $[a, Q]$.

1. $\theta_0 = \theta$ {$\theta$ est une valeur aléatoire.}
2. $K = maxNB\_Iterations$
3. **pour** $i = 1$ to $K$ **faire**
4. **pour** $j = 0$ to $B$ **faire**
5. $f(x_l|\theta^{i-1}) = \sum_{k=1}^{B} a_k^{i-1} f_k(x_l|\lambda_k^{i-1})$
6. $g(j|x_l,\lambda_j^{i-1}) = \frac{\alpha_j^{i-1} f(x_l|\lambda_j^{i-1})}{f(x_l|\theta^{i-1})}$
7. $a_j^i = \frac{1}{n}\sum_{l=1}^{n} g(j|x_l,\lambda_j^{i-1})$
8. $\lambda_j^i = \frac{I_i \sum_{l=1}^{n} g(j|x_l,\lambda_j^{i-1})}{\sum_{l=1}^{n} g(j|x_l,\lambda_j^{i-1}).x_l}$
9. **si** $|\theta_i - \theta_{i-1}| < 10^{-6}$ **alors**
10. **return**($\theta_i$)
11. **finsi**
12. **fin pour**
13. **fin pour**

**Algorithme 3.1:** Algorithme d'approximation de la distribution $HE_{a_i,\lambda_i}$.

La complexité temporelle de l'algorithme 3.1 d'estimation est de l'ordre $O(n.K.B)$. Dans son implémentation, l'algorithme recherche toutes les combinaisons possibles des phases (1 par branche, 2 par branche, etc.) et retourne celle qui a la plus petite erreur. Les résultats de l'approximation des distributions données dans le tableau 3.3, en utilisant l'algorithme 3.1, sont présentés dans la figure 3.8. L'utilisation de la distribution $HEr_{B,I}$ pour l'approximation des fonctions de distributions, dont la queue ne suit pas une fonction une fonction à décroissance lente (*cf.* figure 3.9), a montré sa supériorité sur la méthode utilisée pour l'approximation par la forme canonique. A titre de comparaison entre les résultats d'approximation obtenus par la forme canonique et Hyper-Erlang, nous présentons dans les figures 3.10(a) et 3.10(b) les résultats d'approximations des distributions Weibull($k = 3$,$\lambda = 1.2$) et Triangulaire($a = 0, b = 1, c = 2$) par la forme canonique, étant donné leurs représentations par la forme $HEr_{B,I}$ dans les figures 3.9(c) et 3.9(b).

Nous avons implémenté l'algorithme 3.1, ainsi que celui utilisé pour la recherche des paramètres de la forme canonique [BC92] sous MatLab, et nous avons crée une interface graphique pour rendre plus conviviale l'utilisation de 2 algorithmes. La capture de l'écran de l'interface graphique est donnée dans la figure 3.11.

L'utilisation du MatLab n'a rien de surprenant, suite aux techniques des stockage utilisées pour les matrices creuses, ainsi qu'une large gamme des méthodes de résolution linéaires adaptées. Conscient de l'utilité de ces options, l'outil de PEPA intègre la possibilité de dériver la matrice générateur de la chaîne sous format compatible avec Maple et MatLab. Par contre, *TwoTowers* génére la chaîne sous forme d'une liste dans un fichier, et nous avons écrit un script pour transformer ce fichier en un autre ayant un format compatible avec l'entrée de MatLab.

| *Fonction de distribution* | *Densité de probabilité pdf* |
|---|---|
| *Pareto I* | $\begin{cases} \alpha B^{-1} e^{-\frac{\alpha}{B}x} & \text{si } x \leq B \\ \alpha B^{\alpha} e^{-\alpha} x^{-(\alpha+1)} & \text{si } x > B \end{cases}$ |
| *Pareto II* | $\frac{b^{\alpha} e^{-\frac{b}{x}}}{\Gamma(\alpha)} x^{-(\alpha+1)}$ |
| *Bounded Pareto* | $\frac{\alpha k^{\alpha}}{1-(k/p)} x^{-\alpha-1}$ |
| *Shifted Pareto* | $\alpha k^{\alpha} (x+k)^{-\alpha-1}$ |
| *Matrix exponential* | $\frac{1}{2} e^{-x} + \frac{1}{2} e^{-(x-1)}$ si $x \geq 1$ |
| *Shifted exponential* | $\left(1 + \frac{1}{(2\pi)^2}\right)(1 - \cos(2\pi x)) e^{-x}$ |

Tableau 3.3 – Densités de probabilité des fonctions considérées.

Dans un recherche sur les nouveaux travaux réalisés autour de l'algorithme EM lors de la rédaction de cette manuscrit, nous avons trouvé des algorithmes et des techniques plus récentes et plus sophistiquées [Gei06], ainsi que des résultats qui parfois dépassent ceux de notre algorithme, parmi lesquels nous citons [WZLX05, TBT05] et la méthode de division et de rassemblement D&C – EM (Divide & Conquer) proposée par Riska dans [RDS04] pour la distribution hyper-exponential. Cette méthode est basée sur la division des échantillons en plusieurs partitions, dont le coefficient de variation de chaque partition est supérieur à une valeur seuil prédéfinie. L'avantage de l'utilisation de la méthode de D&C est l'accélération de l'algorithme EM en réduisant le nombre des échantillons, et l'inconvénient est la limitation aux distributions avec un $cv^2 > 1$ pour pouvoir diviser les échantillons en plusieurs partitions.

## 3.8 La méthode d'égalité des moments

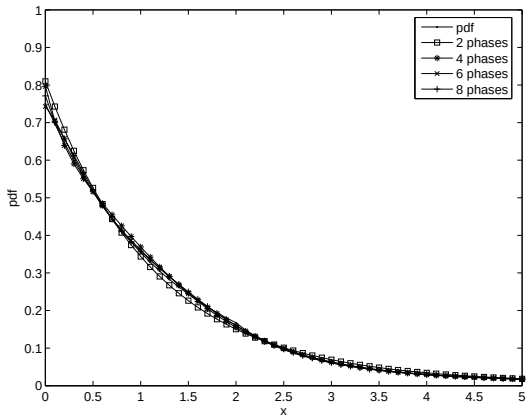
Le moment d'ordre $i$ d'une distribution $G$ est définie par $\mu_i = E\left[X^i\right]$, où $X$ est une variable aléatoire distribuée selon la loi $G$. Cette méthode permet l'approximation d'une distribution $G$ avec une distribution $PH$ prédéfinie qui a les mêmes moments $\mu_k | k = 1, 2 \ldots m$ que $G$, où $m$ est l'ordre de l'approximation. La restriction au niveau de la fonction de distribution $PH$, permet de réduire l'espace de recherche et d'améliorer l'efficacité de l'algorithme de recherche. Par exemple, si $G$ et $PH$ ont le même moment d'ordre 1 ($\mu_1$), alors leurs variables aléatoires ont la même moyenne. Cette approximation d'ordre 1 est importante car elle remplace la distribution $G$ par une autre exponentielle, mais pratiquement cette approximation est imprécise, et elle conduit à des conclusions erronées car elle ignore la variabilité de la distribution $G$, portée par les autre moments qui ont un ordre $k > 1$.

Pour lutter contre la complexité des algorithmes de recherche des paramètres d'une distribution de phase, et pour éviter le problème de l'explosion combinatoire de l'espace d'états, lorsque le nombres de phases nécessaires pour représenter la durée d'une action est grand, l'ordre de l'approximation de $G$ par $PH$ a été souvent limité à 2. De cette façon, les variables aléatoires de $PH$ et $G$ ont la même moyenne et la même variance.

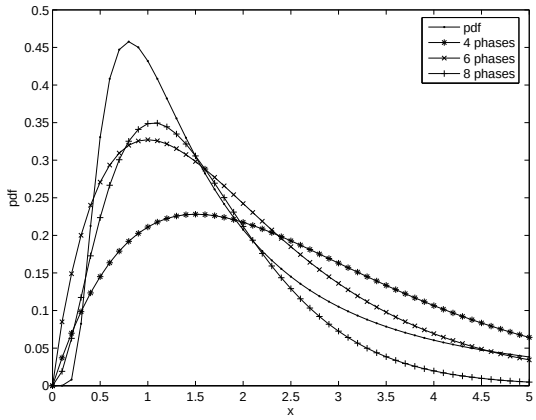
L'approximation avec seulement deux phases de la distribution hyper-exponentielle (*cf.* figure 3.12) nécessite la recherche de trois paramètres : $p$, $\lambda_1$ et $\lambda_2$, comme l'illustre sa fonction de densité de probabilité donnée par :

$$h_2(x) = p\lambda_1 e^{-\lambda_1 x} + (1-p)\lambda_2 e^{-\lambda_2 x}, x \geq 0,\ \lambda_1, \lambda_2 > 0 \tag{3.30}$$

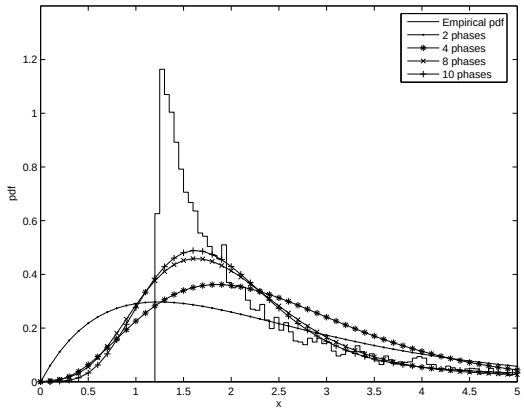
L'equation (3.31) est dérivée du fait que la moyenne et le coefficient de variation de la distribution hyper-exponentielle, doivent être égaux à ceux de la distribution générale $G$ (ou bien des échantillons



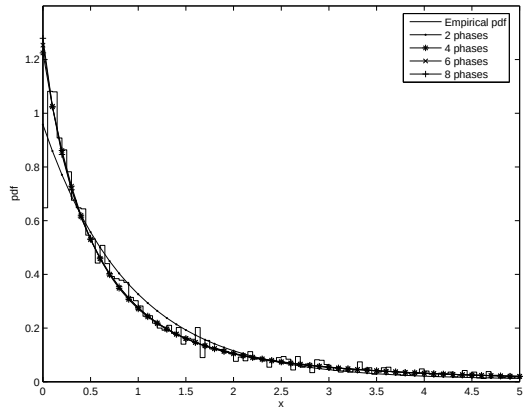

(a) Pareto I ($\alpha = 1.5$, $B = 4$).



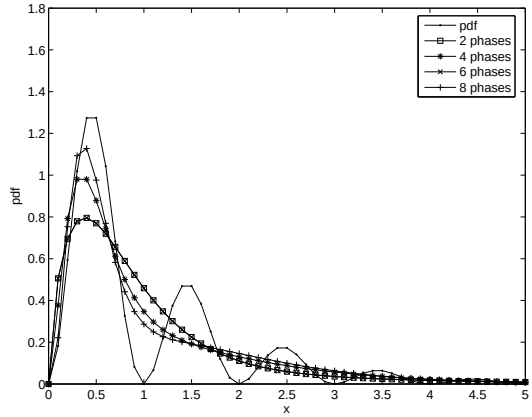

(b) Distribution Pareto II ($\alpha = 1.5$, $b = 4$).



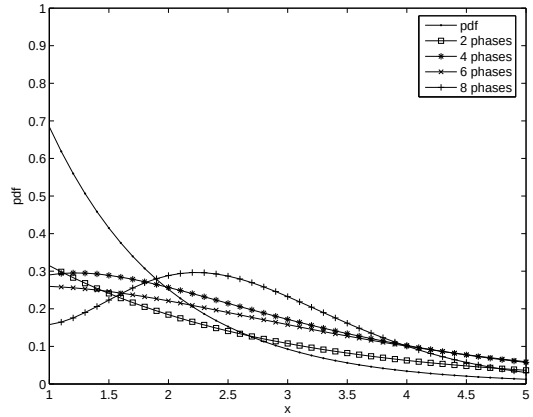

(c) bounded Pareto ($\alpha = 1.5$, $k = 1.2$, $p = 10$).




(d) Shifted Pareto ($\alpha = 1.5$, $k = 1.2$).




(e) Matrix Exponential.



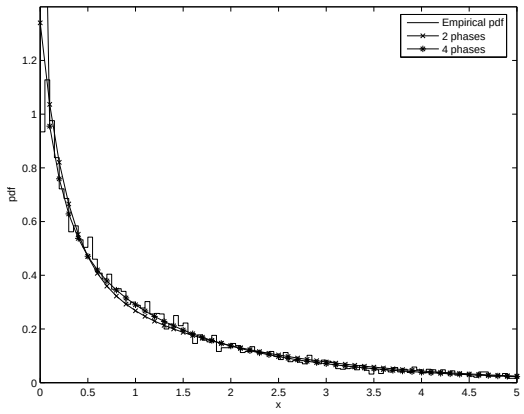

(f) Shifted Exponential.

Figure 3.8 – Approximation par $HEr_{B,I}$.

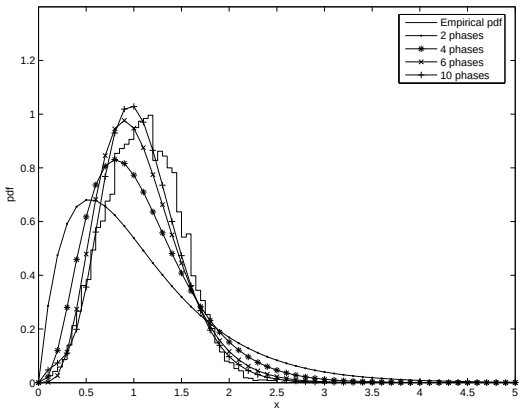
dont on dispose).

$$\mu_{1,G} = \mu_{1,h_2} = \sum_{i=1}^{2} \frac{p_i}{\lambda_i} = \frac{p}{\lambda_1} + \frac{1-p}{\lambda_2} \text{ et } cv_G^2 = cv_{h_2}^2 = \frac{\frac{2p}{\lambda_1^2} + \frac{2(1-p)}{\lambda_2^2}}{\mu_{1,h_2}^2} - 1 \tag{3.31}$$

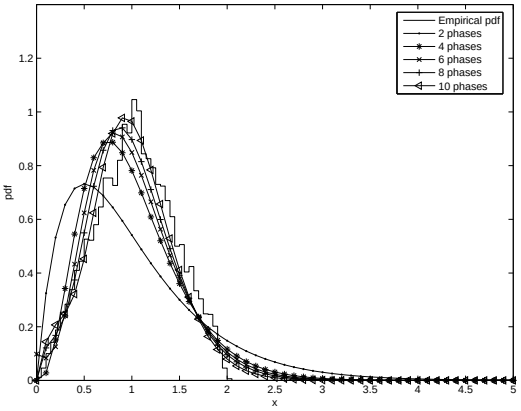
C'est un système linéaire de deux équations à 3 inconnues. Ce système peut se résoudre en supposant qu'une de ces 3 variables est connue. Par exemple, on peut assigner une valeur quelconque à $p$ sous la



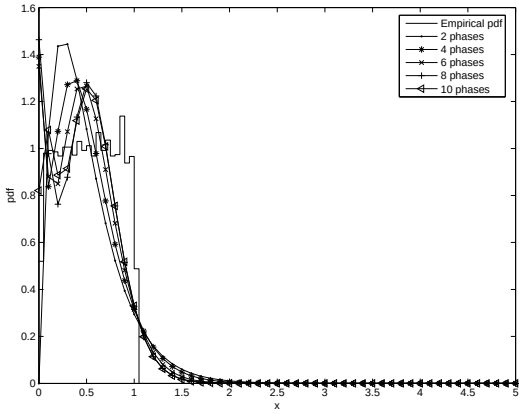

(a) Weibull($k = 0.8$, $\lambda = 1.2$).




(b) Weibull($k = 3$, $\lambda = 1.2$).

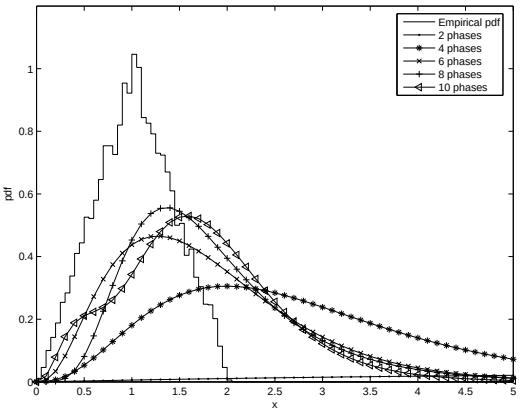



(c) Triangulaire ($a = 0$, $b = 1$, $c = 2$).

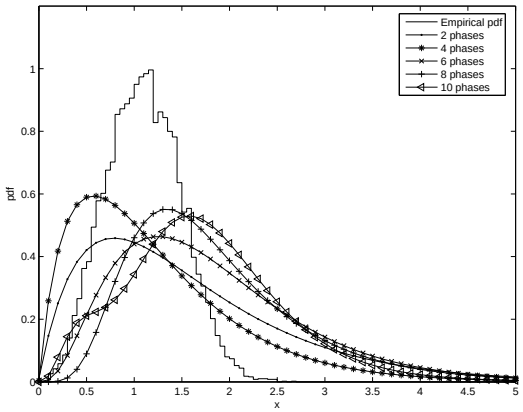



(d) Uniforme ($a = 0$, $b = 1$).

Figure 3.9 – Approximation des distributions Weib, Triang et Unif par $HEr_{B,I}$.




(a) Triangulaire ($a = 0$, $b = 1$, $c = 2$).




(b) Weibull($k = 3$, $\lambda = 1.2$).

Figure 3.10 – Approximation par la forme canonique.

condition que $p \in [0, 1]$. En supposant que $\frac{p}{\lambda_1} = \frac{1-p}{\lambda_2}$ dans (3.31), on aura :

$$p = \frac{1}{2}\left(1 + \sqrt{\frac{cv^2 - 1}{cv^2 + 1}}\right),\ \lambda_1 = \frac{2p}{\mu_{1,G}},\ \lambda_2 = \frac{2(1-p)}{\mu_{1,G}} \tag{3.32}$$

Mais, cette distribution à un coefficient de variation $cv^2 \geq 1$ qui limite son domaine d'application.

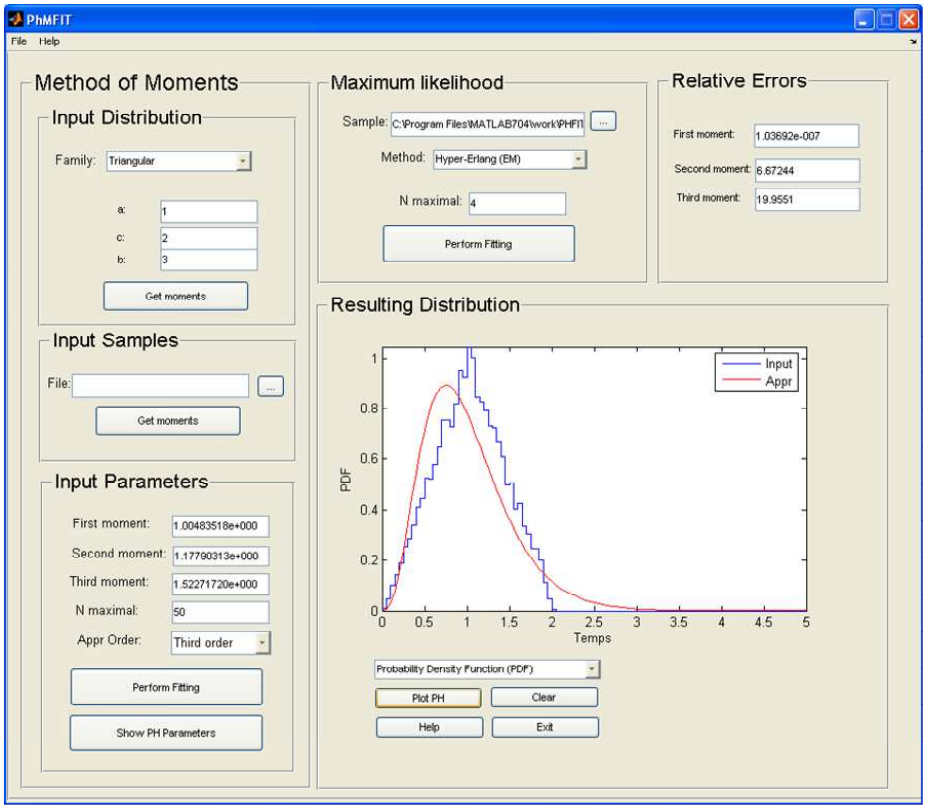



Figure 3.11 – Interface graphique.

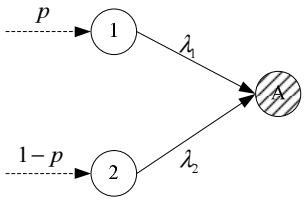



Figure 3.12 – Hyper-exponentielle avec 2 phases.

Cette méthode d'approximation qui consiste à régler les deux premiers moments, a été largement traitée avec d'autre distributions $PH$ pour couvrir toutes les distributions $G$, même avec un $cv^2 < 1$. Sauer et Chandy [SC75] ont proposé l'utilisation d'une distribution généralisée d'*Erlang* (donnée dans la figure 3.13(a)) pour l'approximation des distributions avec $cv^2 < 1$, et la distribution hyper-exponentielle à 2 phases pour l'approximation des distributions générales avec un coefficient $cv^2 \geq 1$. Marie [Mar80] a proposé l'utilisation d'une distribution généralisée d'*Erlang* avec $n$ phases pour les distributions avec $cv^2 < 1$ (donnée dans la figure 3.13(a)), et dont les paramètres de cette dernière sont données par : $n = \lceil \frac{1}{cv^2} \rceil$, $p_{Er} = 1 - \frac{2n.cv^2+n-2-\sqrt{n^2+4-4n.cv^2}}{2(n-1)(cv^2+1)}$, et $\lambda = \frac{1-p_{Er}+np_{Er}}{\mu_1}$), et la distribution *Coxian* avec seulement 2 phases, pour les distributions avec $cv^2 > 1$ (donnée dans la figure 3.13(b)), et dont les paramètres sont données par : $\lambda_1 = \frac{2}{\mu_1}$, $\lambda_2 = \frac{1}{\mu_1.cv^2}$, et $p_C = \frac{1}{2.cv^2}$, pour couvrir tous les domaines de variations comme le montre la figure 3.14.

Ceci est dû au fait que la distribution *coxian* avec deux phases seulement, est capable de bien imiter les distributions avec un coefficient de variation élevée. En revanche, la distribution *coxian* nécessite beaucoup plus de phases pour représenter les distributions avec une petite variabilité en second ordre. Par conséquent, plus de phases implique plus des paramètres à déterminer et plus de complexité de calcul. La distribution d'*Erlang* est beaucoup plus favorable pour l'approximation des distributions à faible variabilité suite au fait que son coefficient de variation diminue lorsque le nombre de phases augmente,


(a) Distribution d'*Erlang-n* généralisée. (b) Distribution *Coxian-2*.

Figure 3.13 – Approximation d'ordre 2.

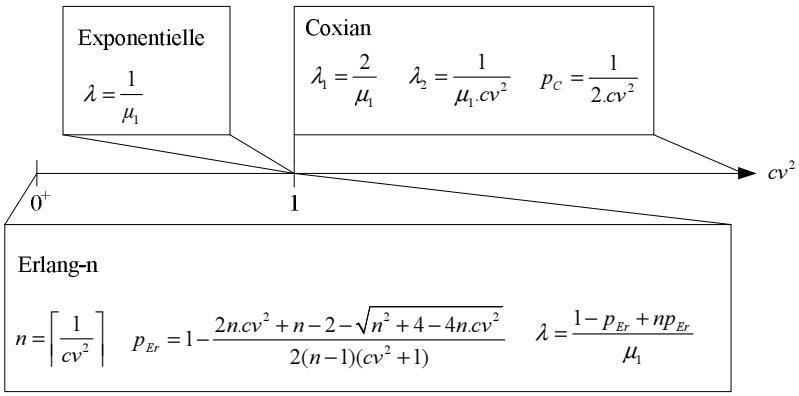



Figure 3.14 – Approche d'approximation par $PH-2$.

comme l'illustrent ses caractéristiques données dans (3.33) :

$$\begin{aligned} pdf_{Er}(t) &= \frac{t^{n-1}\lambda^n}{(n-1)!}e^{-\lambda t} \\ \mu_{1,Er} &= \frac{n}{\lambda},\ \mu_{2,Er} = \frac{n(n+1)}{\lambda^2},\ \mu_{3,Er} = \frac{n(n+1)(n+2)}{\lambda^3} \\ \sigma^2_{Er} &= \frac{\mu_{2,Er}}{\mu^2_{1,Er}} - 1 = \frac{n}{\lambda^2} \Longrightarrow\ cv^2_{Er} = \frac{\sigma^2_{Er}}{\mu^2_{1,Er}} = \frac{1}{n} \underset{n\to\infty}{\longrightarrow} 0 \end{aligned} \tag{3.33}$$

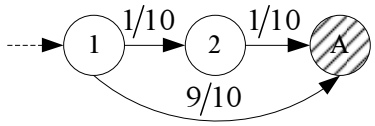
**Théorème 3.1** (Aldous)**.** *Aldous dans [AS87], a prouvé que le coefficient de variation ($cv^2$) d'une distribution phase-type continue d'ordre $n$ est toujours $cv^2 \geq 1/n$, et la valeur minimale qu'on peut représenter est donnée par $cv^2_{\min} = 1/n$. Cette valeur dépend uniquement du nombre de phases $n$ investis dans la représentation, et ne peut être obtenu que par la distribution d'Erlang($\lambda$, $n$).*

Comme exemple d'application à l'approche de Marie, la distribution de phases qui correspond à une distribution Weibull $W(1, 1/2)$ (avec $\mu_1 = 2$ et $cv^2 = 5$) est la distribution coxian avec $p = 1/10$, $\lambda_1 = 1$, et $\lambda_2 = 1/10$ (présentée dans la figure 3.15(a)). Par contre, la distribution de phases qui correspond à une distribution uniforme continue $U(0, 1)$ (avec $\mu_1 = 1/2$ et $cv^2 = 1/3$) est Erlang-3 avec $p = 1$ et $\lambda = 6$ (présentée dans la figure 3.15(b)). En revanche, l'approximation de la même distribution avec un changement de l'intervalle, par exemple $U(1, 2)$ (avec $\mu_1 = 3/2$ et $cv^2 = 1/27$) nécessite 27 phases dans la distribution d'Erlang avec $p = 1$ et $\lambda = 18$, et cela nous semble exagéré pour la représentation de la durée d'une seule action. Par conséquent, un compromis entre précision et nombre d'états doit être trouvé pour l'utilisation de cette méthode. A noter que le coefficient de variation d'une $PH$ (à temps continu) est inversement proportionnelle au nombre de phases $n$ selon l'équation $n = \lceil 1/cv^2 \rceil$, ce qui implique que pour une distribution $G$ avec un petit coefficient de variation, $n$ doit être grand.

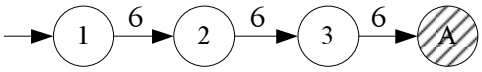
Malheureusement, si le $cv^2 = 0$ (la distribution déterministe), $n = \lceil 1/cv^2 \rceil_{cv^2 \to 0} \to \infty$ pour obtenir une meilleur approximation :

$$cv^2 = 0 \Longrightarrow n \to \infty$$

La figure 3.17 suivante illustre bien cette situation, où plus de 50 phases sont requises pour obtenir une bonne approximation de la distribution déterministe ($d = 1$). Nous ne rentrons pas dans les détails pour le moment de la solution de ce problème, mais nous allons y revenir dans la section 3.11.

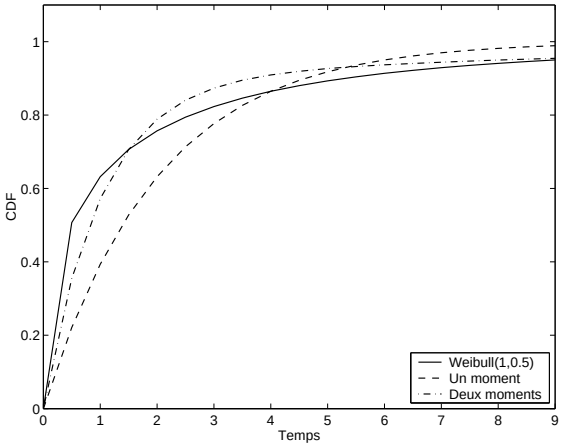



(a) $PH$ correspondante à $W(1, 1/2)$.

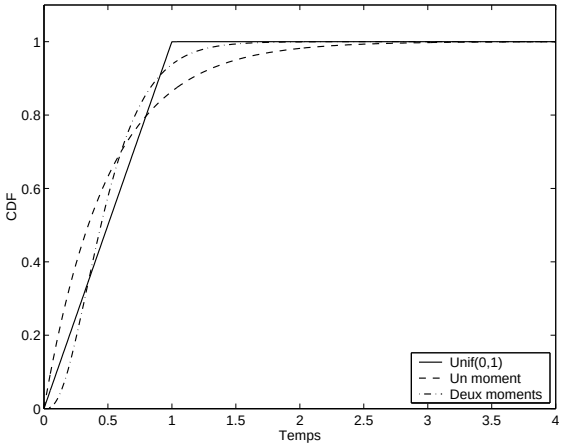



(b) $PH$ correspondante à $U(1, 2)$.

Figure 3.15 – Approximation de $G$ par $PH$.




(a) Weibull(1,1/2)




(b) Uniform(0,1)

Figure 3.16 – Approximations par $PH - 2$.

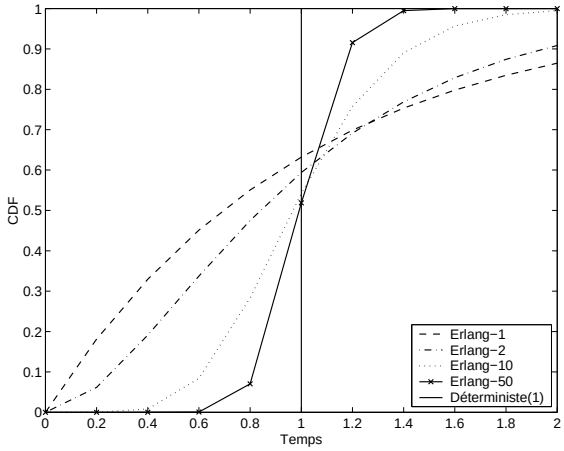



Figure 3.17 – Approximation de la fonction dét(1) par la distribution de $PH$ « Erlang-n ».

### 3.8.1 Travaux relatifs

Johnson et Taaffe dans [JT91] ont montré dans les systèmes des files d'attente, que la méthode d'approximation qui consiste à régler les deux premiers moments, ne suffit pas pour avoir une bonne approximation, où certains paramètres de performances dépendent du moment de troisième ordre ($\mu_3$). Pratiquement, une meilleure précision est obtenue en augmentant l'ordre de l'approximation. Généralement, il est souhaitable d'égaliser autant de moments possibles de $G$ pour avoir une approximation précise, mais ceci a une conséquence directe sur le nombre de phases, et signifie beaucoup plus d'états, dans un environnement où l'évaluation des performances souffre du problème de l'explosion de l'espace d'états.

L'approximation de l'ordre 3 est considérée comme un compromis entre une bonne précision et l'explosion de l'espace d'état. Pour traiter cette approximation, Whitt [Whi82] et Altiok [Alt85] ont considéré les fonctions de distributions avec un coefficient de variation $cv^2 > 1$, et un moment d'ordre 3 suffisamment large ($\mu_3 > 3\mu_2^2/2\mu_1$). Whitt a utilisé la distribution hyper-exponentielle avec 2 phases pour représenter une distribution $G$ par une distribution $PH$, en satisfaisant les 2 conditions précédentes. En revanche, Altiok a utilisé la distribution coxian-2. Telek et Heindl [TH02b, TH02a] ont déterminé les bornes supérieures et inférieuers des 3 premiers moments d'une distribution $PH$ acyclique avec 2 phases seulement ($APH-2$), et ils ont présenté une méthode d'approximation pour les distributions $G$ par une distribution coxian-2 ($APH-2$) dont les bornes des moments inclus ceux de la distribution $G$. Ils n'ont considéré que les fonctions avec une probabilité initiale égale à zéro.

### 3.8.2 Algorithme d'approximation EC-3

Osogami et Harchol-Balter dans [OHB03b] ont traité le problème d'égalité des 3 premiers moments de toutes les distributions générales $G$ (même ceux qui n'appartiennent pas à $APH_2$), pour bien les représenter par une distribution de phase acyclique d'ordre $n$ ($APH_n$). Dans [OHB03a], ils ont déterminé des valeurs approximatives pour les bornes supérieures et inférieures d'une distribution $APH_n$, et ont proposé un algorithme d'approximation dont le nombre de phases dans la chaîne résultante est quasi-minimal, où la distribution de phase résultante contient au plus $min+1$ phases. La chaîne de la distribution phase-type adoptée pour la représentation quasi-minimale est la composition séquentielle de la distribution Erlang(n-2) avec Coxian-2, comme illustre la figure 3.18. Le choix de cette forme de la distribution $PH$ n'a rien de surprenant, où il était connu que la distribution coxian avec 2 phases est bien adaptée pour l'approximation des distributions $G$ avec $cv^2 > 1$, et la distribution d'Erlang pour celles qui ont un $cv^2 < 1$. De cette façon, ils ont couvert toutes les distributions générales, et ils ont réduit la complexité de recherche des paramètres dans l'équation 3.2, de $O(n^2)$ ($n+1 \times n+1$ est la dimension de la matrice $T$) à $O(1)$ , avec seulement 6 paramètres à rechercher ($n$, $\lambda_1$, $\lambda_2$, $p_C$, $p_E$, $\lambda_E$).

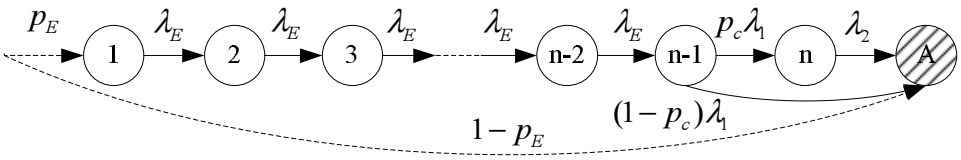



Figure 3.18 – Chaîne Erlang-Coxian résultante.

L'introduction de la notion du moment normalisé a simplifié encore plus la complexité de l'algorithme, et a transformé le problème d'égalité de 3 moments, à l'égalisation de 2 moments normalisés seulement. Le deuxième et le troisième moment normalisés sont donnés par :

$$m_2 = \frac{\mu_2}{\mu_1^2} \quad \text{et} \quad m_3 = \frac{\mu_3}{\mu_1\mu_2} \tag{3.34}$$

La multiplication de la matrice générateur $Q$ d'une distribution $PH$ (définie par $[a, Q]$), par une constante $c$ change la valeur du moment d'ordre $k$ défini dans l'équation 3.5, à $\mu_k = (-1)^k k! p(cQ)^{-k}1 = c^k(-1)^k k! pQ^{-k}1$. Les moments normalisés $m_2$ et $m_3$ sont insensibles à cette multiplication et leurs valeurs sont inchangeables. Cette propriété transforme le problème d'approximation des 3 premiers moments ($\mu_1$, $\mu_2$, et $\mu_3$) d'une distribution générale $G$ en un problème d'approximation de deux moments normalisés ($m_2$ et $m_3$) sans se soucier de la valeur du premier moment qui sera réglée par une simple multiplication. Pour clarifier la simplification des formules en utilisant la forme normalisée, prenons la distribution d'Erlang-n (donnée dans la figure 3.3(c)), et dont les 3 premiers moments sont donnés par :

$$\mu_1 = \frac{n}{\lambda} \quad \mu_2 = \frac{n(n+1)}{\lambda^2} \quad \mu_3 = \frac{n(n+1)(n+2)}{\lambda^3} \tag{3.35}$$

Les moments normalisés prennent la forme suivante :

$$m_2 = \frac{n+1}{n} \quad \text{et} \quad m_3 = \frac{n+2}{n} \tag{3.36}$$

En utilisant la notation du moment normalisé, les résultats de Telek et Heindl [TH02b, TH02a] peuvent se résumer par le théorème suivant :

**Théorème 3.2** (Telek et Heindl)**.** *Une distribution $G \in S^{(2)^*}$, où $S^{(2)^*}$ est l'ensemble des distributions (avec une probabilité initiale égale à zéro) qui peuvent être bien représentées par la distribution coxian-2, si et seulement si $G$ satisfait une des 3 conditions suivantes :*

*i.* $\frac{9m_2^G - 12 + 3\sqrt{2}(2-m_2^G)^{\frac{3}{2}}}{m_2^G} \leq m_3^G \leq \frac{6(m_2^G-1)}{m_2^G} \ \wedge \ \frac{3}{2} \leq m_2^G \leq 2$

*ii.* $m_3^G = 3 \ \wedge \ m_2^G = 2$

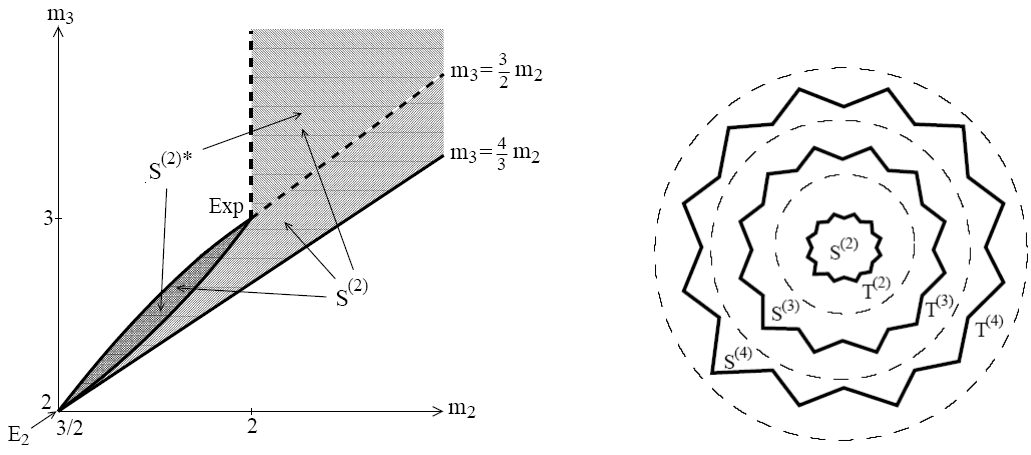
*iii.* $\frac{3}{2}m_2^G < m_3^G \ \wedge \ 2 < m_2^G$

Osogami et Harchol-Balter dans [OHB03a] ont complété l'ensemble $S^{(2)^*}$ établi par Telek et Heindl, en considérant en plus les distributions avec une probabilité initiale non nulle. Par conséquent, $S^{(2)}$ est l'ensemble des distributions générales $G$ qui peuvent être représentées par une distribution $APH_2$ comme montre la figure 3.19(a). Mais l'ensemble complet, qui est représenté par $S^{(n)}$ avec $n = 2, 3, 4$ dans la figure 3.19(b) ($S^{(n)}$ est l'ensemble des distributions $G$ qui sont bien représentées par une distribution $APH_n$, même avec une probabilité initiale non nulle), a semblé avoir une frontière irrégulière et sans relation avec celle de $S^{(n-1)}$ pour les auteurs, mais avec $S^{(n)} \supset S^{(n-1)} \ldots \supset S^{(2)}$. Pour cela, ils ont cherché le plus petit ensemble $T^{(n)} \supset S^{(n)}$ dont les limites sont régulières, et ils ont construit leur algorithme de recherche des paramètres de la chaîne donnée dans la figure 3.18, tout en se basant sur cet ensemble. L'algorithme 3.2 résume leurs travaux. Mais, comme l'ensemble $T^{(n)}$ contient des distributions appartenant à $S^{(n+1)}$, la solution proposée contient au plus $n+1$ états, et est très proche de la solution optimale possédant le nombre minimal de phases $n$ ($APHM_n$).

**Définition 3.2.** *une distribution $G \in T^{(n)}$, si et seulement si les moments normalisés $m_2^G$ et $m_3^G$ satisfont une des conditions suivantes :*

*i.* $m_2^G > \frac{n+1}{n} \quad et \quad m_3^G \geq \frac{n+2}{n+1}m_2^G$

*ii.* $m_2^G = \frac{n+1}{n} \quad et \quad m_3^G = \frac{n+2}{n}$.




(a) L'ensemble $S^{(2)}$ et $S^{(2)^*}$ (b) $S^{(n)} \subset T^{(n)} \subset S^{(n+1)}$

Figure 3.19 – Caractérisation de la distribution $PH$ optimale.

Bibbio et Telek [BHT05] ont récemment utilisé la notion du moment normalisé introduit par Osogami dans [Oso05], pour donner les caractéristiques exactes (représentées dans les figures 3.20(a) et 3.20)

**ENTRÉES:** les trois premiers moments d'une distribution générale $\mu_1$, $\mu_2$ et $\mu_3$
**SORTIES:** la liste des paramètres et le nombre d'états de la représentation Erlang-Coxian

1. $m_2^G = \frac{\mu_2}{(\mu_1)^2}$ et $m_3^G = \frac{\mu_3}{\mu_1\mu_2}$
2. $p = \begin{cases} \frac{(m_2^G)^2+2m_2^G-1}{2(m_2^G)^2} & si\ m_3^G > 2m_2^G - 1 \wedge \frac{1}{m_2^G-1}\ \textit{est un nombre entier} \\ \frac{1}{2m_2^G-m_3^G} & si\ m_3^G < 2m_2^G - 1 \\ 1 & sinon \end{cases}$
3. $\mu_1^W = \frac{\mu_1}{p_E}$ ; $m_2^W = p_E m_2^G$ ; $m_3^W = p_E m_3^G$.
4. $n = \begin{cases} \left\lceil \frac{m_2^W}{m_2^W-1} \right\rceil - 1 & si\ m_3^W = 2m_2^W - 1\ et\ m_2^W \leq 2, \\ \left\lfloor \frac{m_2^W}{m_2^W-1} + 1 \right\rfloor & sinon \end{cases}$
5. **si** $n == 1$ **alors**
6. $p_c = 0,\ \lambda_E = \frac{1}{\mu_1^W}$ *et exit*
7. **sinon**
8. $m_2^C = \frac{(n-3)m_2^W-(n-2)}{(n-2)m_2^W-(n-1)}$ ;
9. $\mu_1^C = \frac{\mu_1^W}{(n-2)m_2^C-(n-3)}$
10. $\alpha = (n-2)(m_2^C-1)\left(n\,(n-1)\,(m_2^C)^2 - n(2n-5)m_2^C + (n-1)(n-3)\right)$
11. $\beta = \left((n-1)m_2^C - (n-2)\right)\left((n-2)m_2^C - (n-3)\right)^2$
12. $m_3^C = \frac{\beta m_3^W-\alpha}{m_2^C}$
13. **finsi**
14. $u = \begin{cases} 1 & si\ 3m_2^C = 2m_3^C \\ \frac{6-2m_2^C}{3m_2^C-2m_3^C} & autrement \end{cases}$ ; $v = \begin{cases} 0 & si\ 3m_2^C = 2m_3^C \\ \frac{16-6m_2^C}{m_2^C(3m_2^C-2m_3^C)} & autrement \end{cases}$
15. $\lambda_1 = \frac{u+\sqrt{u^2-4v}}{2\mu_1^C}$; $\lambda_2 = \frac{u-\sqrt{u^2-4v}}{2\mu_1^C}$; $p_C = \frac{\lambda_2(\lambda_1\mu_1^C-1)}{\lambda_1}$; $\lambda_E = \frac{1}{(m_2^C-1)\mu_1^C}$

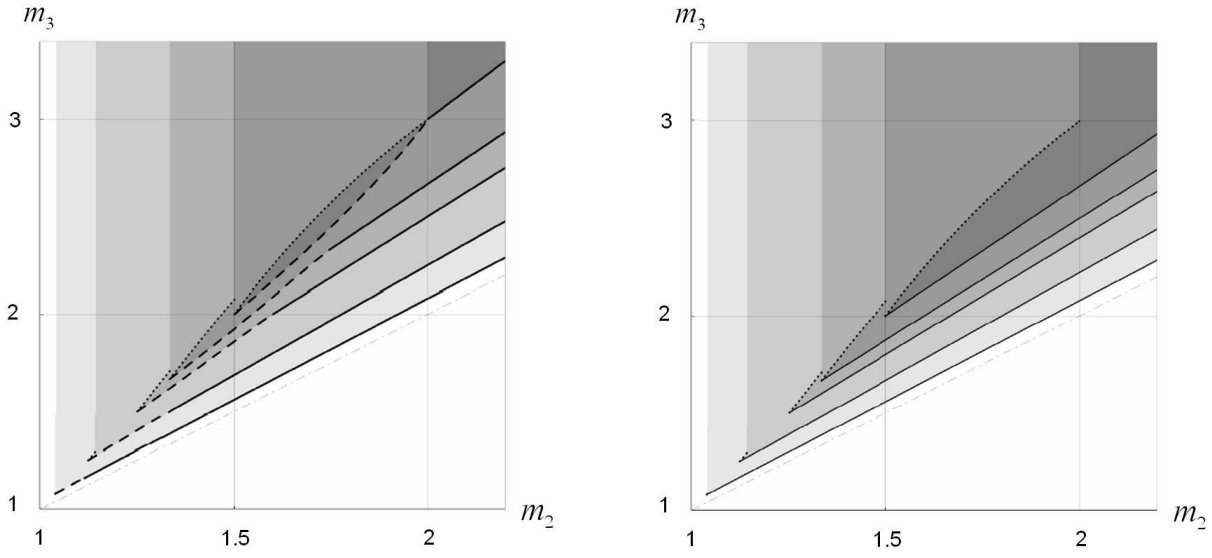
**Algorithme 3.2:** $(n, p_E, p_c, \lambda_E, \lambda_1, \lambda_2) = Erlang - Coxian(\mu_1, \mu_2, \mu_3)$

des ensembles $S^{(n)}$ et $S^{(n)^*}$ ($S^{(n)^*}$ est l'ensemble des distributions générales avec une probabilité initiale nulle, et qui sont bien représentées par une distribution $APHM_n$). Ils ont utilisé les bornes exactes qu'ils ont obtenues pour déterminer la distribution acyclique minimale sous forme d'une combinaison séquentielle Erlang-Exponentielle (figure 3.21(a)) ou Exponentielle-Erlang (figure 3.21(b)). Partons de l'idée que la distribution $APH_n$ n'est autre qu'une composition séquentielle d'une distribution $APH-(n-1)$ donnée par les paramètres $[a_1, Q_1]$, avec une phase additionnelle avec un taux de franchissement $\lambda$ et une probabilité initiale $1-p$ (*cf.* figure 3.22). L'addition de cette phase additionnelle à la distribution $APH_{n-1}$ modifie les moments de la de la façon suivante :

$$\begin{aligned} \mu_{1,APH_n} &= p\left(\mu_{1,APH_{n-1}} + \frac{1}{\lambda}\right) + (1-p)\frac{1}{\lambda} \\ \mu_{2,APH_n} &= p\left(\mu_{2,APH_{n-1}} + \frac{2\mu_{1,APH_{n-1}}}{\lambda} + \frac{2}{\lambda^2}\right) + (1-p)\frac{2}{\lambda^2} \\ \mu_{3,APH_n} &= p\left(\mu_{3,APH_{n-1}} + \frac{3\mu_{2,APH_{n-1}}}{\lambda} + \frac{6\mu_{1,APH_{n-1}}}{\lambda} + \frac{6}{\lambda^3}\right) + (1-p)\frac{6}{\lambda^3} \end{aligned} \tag{3.37}$$

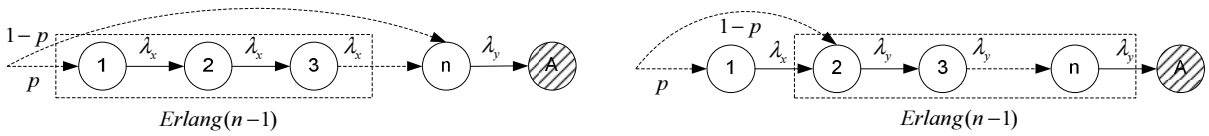
Les moments normalisés ($m_2$ et $m_3$) de la distribution $APH_n$, étant donnés les moments normalisés de la distribution $APH_{n-1}$ ($\bar{m}_2$ et $\bar{m}_3$), prennent la forme suivante :

$$m_2 = \frac{2(1+p\lambda\bar{m}_1) + p(\lambda\bar{m}_1)^2\bar{m}_2}{(1+p\lambda\bar{m}_1)^2} \qquad m_3 = \frac{6(1+p\lambda\bar{m}_1) + p\lambda^2\bar{m}_1^2\bar{m}_2(\lambda\bar{m}_3\bar{m}_1+3)}{2(1+p\lambda\bar{m}_1)^2 + p\lambda^2\bar{m}_1^2\bar{m}_2(1+p\lambda\bar{m}_1)^2} \tag{3.38}$$




(a) $APH_n$. (b) $APHM_n$.

Figure 3.20 – Limites des moments pour $n = 2, 3, 4, 8, 25$.




(a) Erlang-Exp. (b) Exp-Erlang.

Figure 3.21 – Chaîne de Markov pour la distribution $PH$ minimale.

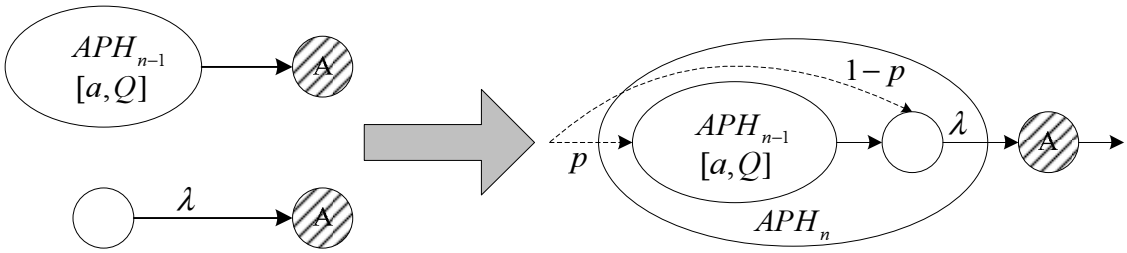



Figure 3.22 – Forme minimale.

**Théorème 3.3** (Bibbio et Telek)**.** *Une distribution $G \in S^{(n)^*}$ si et seulement si, $m_2^G$ et $m_3^G$ satisfont les conditions suivantes :*

$$m_2^G \geq \frac{n+1}{n} \qquad et$$

$$m_3^G \; limite \; inférieure : \begin{cases} Inférieure_{APH(n)} \leq m_3^G & si \; \frac{n+1}{n} \leq m_2^G \leq \frac{n+4}{n+1} \\ \frac{n+1}{n} m_2^G < m_3^G & si \; \frac{n+4}{n+1} \leq m_2^G \end{cases}$$

$$m_3^G \; limite \; supérieure : \begin{cases} m_3^G \leq supérieure_{APH(n)} & si \; \frac{n+1}{n} \leq m_2^G \leq \frac{n}{n-1} \\ m_3^G < \infty & si \; \frac{n}{n-1} < m_2^G \end{cases}$$

*Où :*

$$
\begin{aligned}
Inférieure_{APH(n)} &= \frac{(3+a_n)(n-1)+2a_n}{(n-1)(1+a_np_n)} - \frac{2a_n(n+1)}{2(n-1)+a_np_n(na_a+2n-2)} \\
supérieure_{APH(n)} &= \frac{1}{n^2m_2^G}\left(2(n-2)(nm_2^G-n-1)\sqrt{1+\frac{n(m_2^G-2)}{n-1}}+(n+2)(3nm_2^G-2n-2)\right) \\
p_n &= \frac{(n+1)(m_2^G-2)}{2m_2^G(n-1)}\left(\frac{-2\sqrt{n+1}}{\sqrt{4(n+1)-3nm_2^G}}-1\right) \\
a_n &= \frac{m_2^G-2}{p_n(1-m_2^G)+\sqrt{p_n^2+\frac{p_nn(m_2^G-2)}{n-1}}}
\end{aligned}
$$

**Théorème 3.4** (Bobbio et Telek)**.** *Une distribution $G \in S^{(n)}$ si et seulement si, $m_2^G$ et $m_3^G$ satisfont les conditions suivantes :*

$$
\begin{aligned}
m_2^G \geq \tfrac{n+1}{n} \quad & et \\
m_3^G\ limite\ inférieure : \quad & \tfrac{n+2}{n+1}m_2^G < m_3^G \\
m_3^G\ limite\ supérieure : \quad & \begin{cases} m_3^G \leq supérieure_{APH(n)} & if \frac{n+1}{n} \leq m_2^G \leq \frac{n}{n-1} \\ m_3^G < \infty & if \frac{n}{n-1} < m_2^G \end{cases}
\end{aligned}
$$

avec supérieure$_{APH_n}$ a la même valeur qu'au théorème précédent.

**Théorème 3.5** (Bobbio et Telek)**.** *Soit la distribution $G \in S^{(n)^*} \backslash S^{(n-1)^*}$. $G$ est bien représentée par Erlang-Exp si et seulement si, $m_2^G \leq \frac{n}{n-1} \vee m_3^G \leq 2m_2^G - 1$. Les paramètres de la séquence des distributions Erlang-Exp sont données par :*

$$
\begin{aligned}
\lambda_x &= \frac{ap+1}{\mu_1^G} \\
\lambda_y &= \frac{\lambda_x(n-1)}{a} \\
a &= \frac{(bm_2^G-2)(n-1)b}{(b-1)n} \\
p &= \frac{b-1}{a} \\
b &= \frac{2(4-n(3m_2^G-4))}{m_2^G(4+n-nm_3^G)+\sqrt{nm_2^Gc}} \\
c &= 12(m_2^G)^2(n+1)+16m_3^G(n+1)+m_2^G(n(m_3^G-15)(m_3^G+1)-8(m_3^G+3))
\end{aligned}
$$

*Par contre, si $m_2^G > \frac{n}{n-1} \wedge m_3^G > supérieure_{APH_{n-1}}$, alors $G$ sera bien représentée par la séquence de distributions Exp-Erlang avec les paramètres suivants :*

$$
\begin{aligned}
\lambda_x &= \frac{ap+1}{\mu_1^G} \\
\lambda_y &= \frac{\lambda_x(n-1)}{a} \\
a &= \frac{2(f-1)(n-1)}{(n-1)(m_2^Gf^2-2f+2)-n} \\
p &= (f-1)a
\end{aligned}
$$

*Où $f$ est la solution d'une équation d'ordre 4 donné par $c_4f^4+c_3f^3+c_2f^2+c_1f+c_0=0$*

$$
\begin{aligned}
c_4 &= m_2^G(3m_2^G-2m_3^G)(n-1)^2 \\
c_3 &= 2m_2^G(m_3^G-3)(n-1)^2 \\
c_2 &= 6(n-1)(n-m_2^G) \\
c_1 &= 4n(2-n) \\
c_0 &= n(n-2)
\end{aligned}
$$

**Théorème 3.6** (Bobbio et Telek)**.** *Soit G une distribution de probabilité générale avec une probabilité initiale non nulle, et des moments normalisés $m_2^G$ et $m_3^G$ appartenant aux limites de $APHM(0)$, donnés dans le théorème 3.4. Deux cas sont possibles pour la représentation par la distribution phase-type minimale :*

1. *Si $\{m_2^G, m_3^G\} \in APH_n \wedge m_3^G \neq \frac{3}{2}m_2^G$, alors la représentation est la même que celle donnée par le théorème 3.5.*
2. *Si $\{m_2^G, m_3^G\} \in APHM_n \backslash APH_n$ , alors la représentation est donné dans la figure 3.23 avec les paramètres suivants : $q = \frac{1}{2m_2^G - m_3^G}$, $a$ et $Q$ sont les paramètres de la distribution de phase donnés par le théorème 3.5 pour la distribution G avec des moments $m_2^{G^*} = \frac{m_2^G}{2m_2^G - m_3^G}$ et $m_3^{G^*} = \frac{m_3^G}{2m_2^G - m_3^G}$.*

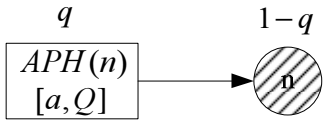



Figure 3.23 – $APHM_n$.

L'avantage de cette méthode analytique est le gain apporté au niveau de la complexité spatiale et temporelle pour les recherches des paramètres de la distribution $APH_n$. Pour aller un peu plus loin, on peut remplacer les petites valeurs de $m_2$ et $m_3$ avec d'autres $m'_2$ et $m'_3$, et on aura une bonne approximation, sans l'utilisation de l'algorithme EM ou la méthode itérative de simplex. Nous avons programmé et intégré dans notre outil (*cf.* figure 3.11) toutes les méthodes de moments pour leur utilisation dans la suite de ce chapitre.

### 3.8.3 Exemple d'application

Cette section présente un exemple de modélisation d'un processus à l'aide des algèbres de processus stochastiques et des distributions phases-types. Nous illustrons à travers la modélisation algébrique le comportement du passager d'un vol possédant un mobile et cherchant à se connecter au réseau dans un terminal d'aéroport (*cf.* figure 3.24), l'utilisation des distributions phase-type dans la modélisation algébrique.

La zone d'émission de la station de base est limitée, et ne couvre pas tous les endroits du terminal. Les utilisateurs qui sont assis en dehors de la zone de couverture, se connectent d'une façon ad hoc en comptant sur les stations intermédiaires pour router leurs informations. Au cas où le passager trouve une place où il se connecte, il reste à sa place jusqu'à l'appel pour l'embarquement ou il se déplace à la recherche d'une connexion après une coupure causée par le mouvement d'une station intermédiaire. Dans le cas où il ne trouve pas de place libre, le passager se déplace vers un autre terminal en attendant l'appel. Mais, conscients du fait que les informations d'affichage peuvent changer, certains passagers préfèrent aller vérifier s'il y a un changement sur l'affichage concernant leur vol, ou peut être rechercher une place qui se soit libérée dans leur terminal d'embarquement. En revanche, d'autres passagers préfèrent jouer l'indifférence et une fois connectés au réseau, préfèrent rester à la même place jusqu'à l'embarquement. La spécification algébrique est donnée par :

$$
\begin{aligned}
P_1 &\stackrel{def}{=} (arr\_ter_A, \lambda).P_2 \\
P_2 &\stackrel{def}{=} (stay\_bord\_ter_A, \mu_1).P_3 + (move\_ter_B, \mu_2).P_4 \\
P_3 &\stackrel{def}{=} (change\_place, \alpha).P_2 + (bording, Y_G).P_1 \\
P_4 &\stackrel{def}{=} (stay\_bord\_ter_B, X_G).P_5 \\
P_5 &\stackrel{def}{=} (check\_display, \mu).P_2 + (bording, Y_G).P_1
\end{aligned}
$$

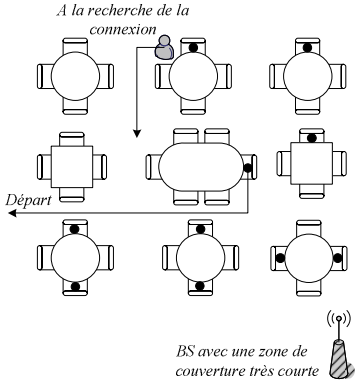



Figure 3.24 – Scénario de modélisation.

L'arrivée d'un passager au terminal A est représentée par l'action $arr\_ter_A$, et l'action $stay\_bord\_ter_A$ représente le fait que le passager a trouvé une place où il reste jusqu'à l'embarquement ou la coupure de la connexion. L'action $move\_ter_B$ est utilisée pour spécifier le déplacement de l'utilisateur envers un autre terminal, où il va rester un certain temps dans ce dernier, avant de vouloir soit vérifier l'affichage, soit partir pour l'embarquement. La chaîne sous jacente de la spécification algébrique est représentée par la figure 3.25.

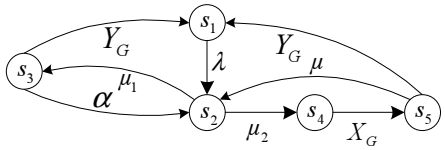



Figure 3.25 – Diagramme de transitions de $P_1$.

Nous nous intéressons à la variation de pourcentage du temps où le passager est connecté, ainsi que le taux de vérification de l'affichage et la fréquence du changement de place en fonction de $X_G$. Les valeurs des paramètres sont données dans le tableau 3.4, et les courbes de variation de ces indices de performances en fonction des différentes valeurs de $X_G$ sont données dans la figure 3.26.

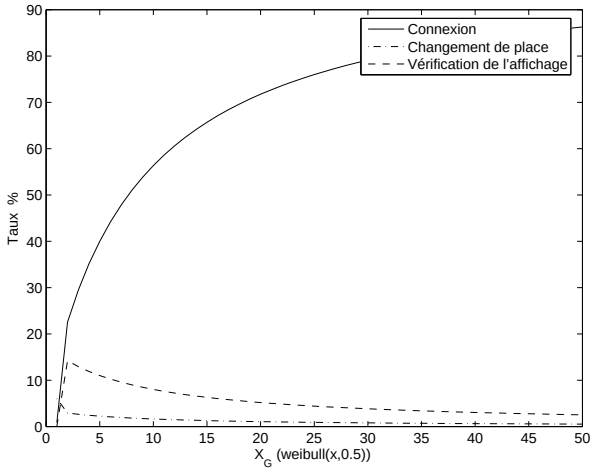



Figure 3.26 – Analyses quantitatives.

| Taux | Valeur | Taux | Valeur |
|---|---|---|---|
| $\lambda$ | 2 | $\alpha$ | 0.5 |
| $\mu_1$ | 10 | $\mu_2$ | 1 |
| $X_G$ | $Weibull(30, 1)$ | $\mu$ | 10 |
| $Y_G$ | | $Weibull(a, 0.5)$ $a \in [1, 50]$ | |

Tableau 3.4 – Valeurs numériques.

## 3.9 Agrégation de l'espace d'état

Dans cette section, nous essayons d'utiliser les distributions phase-type pour réduire l'espace d'états même dans des cas particuliers.

Étant donné que l'utilisation des distributions $PH$ augmente l'espace d'état, nous avons essayé d'utiliser la stratégie de « lumpability » pour réduire la taille de l'espace d'états et accéléré le calcul des vecteurs de probabilités, ainsi que les mesures de performances.

Soit $S' = \{S_1, S_2, \ldots, S_n\}$ le nouvel espace d'états obtenu après le regroupement des états de $S$ en $n$ partitions, avec $\|S'\| << \|S\|$ ($\|\|$ est la représentation de l'opérateur cardinal d'un ensemble). Le regroupement des états en partitions choisies au hasard, nous ramène à la case de départ lors de la construction du générateur infinitésimal, car si on note par $q(s_i, s_j)$ le taux de transition dans la chaîne originale, alors le taux de transition envers une partition $S_i$ est donné par :

$$q(s_i, S_j) = \sum_{k \in S_j} q(s_i, s_k) \tag{3.39}$$

La question la plus importante est le taux de transitions entre les partitions, généralement donné par :

$$q(S_i, S_j) = \sum_{k \in S_i} \pi_k q(s_k, S_j) \tag{3.40}$$

Par conséquent, le calcul exact du vecteur de probabilités dans les nouvel espace d'états agrégés, nécessite le calcul de probabilité de chaque état dans chaque partition, en d'autre terme la résolution de l'espace d'état original.

Kemeny et Snell dans [KS60] ont identifié les conditions d'agrégation permettant de conserver la propriété de Markov, et de calculer les valeurs des paramètres du vecteur de probabilité par la résolution de la chaîne agrégée, puis par décomposition des partitions. Ces conditions sont regroupées sous le terme lumpability où différentes classification des conditions d'agrégation existent (strong, ordinarily, strict, exactly, etc.).

**Théorème 3.7** (Kemeny et Snell)**.** *Une chaîne de Markov est agrégable (strong lumpable) par rapport à une partition $S_i$, si et seulement si :*

$$\forall S_k, S_l \in S' \wedge \forall s_h, s_l \in S \quad \textit{on a} \sum_{s_h \in S_i} q(s_h, S_k) = \sum_{s_l \in S_i} q(s_l, S_k)$$

Donc après le regroupement des états en partition, on peut procéder à la construction du nouvel espace d'états, et à la résolution du système linéaire sous-jacent, pour le calcul des probabilités de $S_i$. Le vecteur des probabilités des états $s_i$ dans la chaîne originale est calculé en utilisant la probabilité de séjour dans le macro-état :

$$\pi(S_i) = \sum_{s_k \in S_i} \pi'(s_k)$$

La première question est la recherche des états agrégables en partitions. Étant donnée la matrice générateur $Q$ de la chaîne de Markov et un ensemble de partitions $S_j, \ldots, S_i$, la construction de la matrice d'indication $V$, dont les éléments sont données dans (3.41), facilite la vérification de la condition d'agrégation avec cet ensemble de partitions.

$$V_{i,j} = \begin{cases} 1 & \text{si } s_i \in S_i. \\ 0 & \text{autrement.} \end{cases} \tag{3.41}$$

Soit $U = (V^T V)^{-1} V^T$, alors l'agrégation de $Q$ selon une partition donnée est possible si et seulement si $VUQV = QV$, et la matrice de la chaîne agrégée est donnée par $Q_L = UQV$. Dans un souci de

clarification, nous prenons l'exemple suivant :

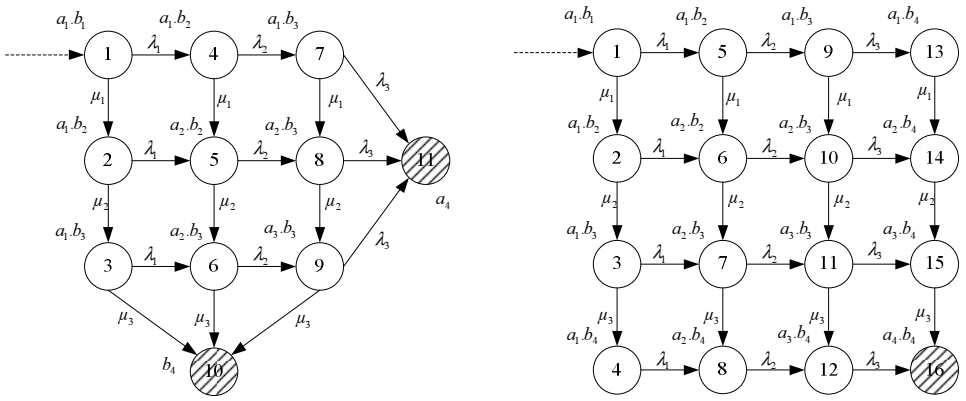
$$Q = \left(\begin{array}{cc|cc} -8 & 2 & 5 & 1 \\ 4 & -10 & 1 & 5 \\ \hline 5 & 3 & -10 & 2 \\ 3 & 5 & 8 & -16 \end{array}\right)$$

Soit $S$ un ensemble de partitions données composé de $S = s_1, s_2, s_3, s_4$, la vérification de cet ensemble est donné par :

$$V = \begin{array}{c} \\ s_1 \\ s_2 \\ s_3 \\ s_4 \end{array} \begin{array}{c} \begin{array}{cc} S_1 & S_2 \end{array} \\ \left(\begin{array}{cc} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{array}\right) \end{array} \quad U = \left(\begin{array}{cccc} 0.5 & 0.5 & 0 & 0 \\ 0 & 0 & 0.5 & 0.5 \end{array}\right) \quad VUQV = \left(\begin{array}{cc} -6 & 6 \\ -6 & 6 \\ 8 & -8 \\ 8 & -8 \end{array}\right) \quad QV = \left(\begin{array}{cc} -6 & 6 \\ -6 & 6 \\ 8 & -8 \\ 8 & -8 \end{array}\right)$$

Nous considérons les deux processus $R = (b_1, X_G).P + (b_2, Y_G).Q$ et $T = (b, X_G).0 \underset{b}{\bowtie} (b, Y_G).0$, où $X_G$ et $Y_G$ sont deux variables aléatoires distribuées selon une certaine loi générale $G$. L'approximation de $G$ par plus de 4 phases n'est pas convenable avec l'explosion de l'espace d'états. Seules les distributions avec un petit coefficient de variation $cv^2 < 0.5$ nécessitent plus de phases et peuvent être représentées par les distributions $PH$ à temps discret avec un petit nombre des phases comme le montre la section 3.11. Nous nous contentons pour le moment de représenter $G$ par une distribution $PH$ canonique (donnée par la figure 3.5(a)) avec $n+1 = 4$ phases seulement, dont $a$ est le vecteur des probabilités initiales.

Les diagrammes de transitions pour les processus $R$ et $T$ avec la distribution $ACPH_4$ sont donnés dans la figure 3.27. La technique d'agrégation au sens de strong lumpability, ne peut pas jouer un grand rôle sur l'agrégation de ces diagrammes.




(a) Diagramme de transition de $R$. (b) Diagramme de transition de $T$.

Figure 3.27 – Diagrammes de transition de $R$ et de $T$.

En pratique, la complexité de recherche des partitions augmente de façon non linéaire avec la taille du générateur. De plus cette condition d'agrégation est rarement rencontrée dans la modélisation d'un système réel, qui a rarement un comportement symétrique, mais plutôt une asymétrie locale. Beaucoup de travaux sur la recherche d'un algorithme d'agrégation même avec un certain taux d'erreur (e.g. Near-lumpability, etc.) ont été réalisés. Nous avons commencé par nous interroger sur une méthode d'agréga-

tion de l'espace d'états, même dans un cas particulier, car l'utilisation des distributions phase-type, vient ajouter de l'huile sur le feu, en augmentant la taille du générateur de la chaîne.

En supposant que le problème de l'explosion de l'espace d'états ne réside pas dans la dérivation de la matrice, mais plutôt dans la résolution du système linéaire, pour l'obtention des vecteurs de probabilités, les distributions phases-type peuvent servir pour réduire l'espace l'état. Normalement, la dérivation du vecteur de distribution de probabilités n'est pas un objectif en lui-même, mais il est utilisé pour le calcul des indices de performance.

Généralement des récompenses sont ajoutées à un certain nombre d'états selon une certaine condition d'exécution des actions, et seules les probabilités de ces états sont significatives dans le calcul des indices de performances en question. En résumé, on est juste intéressé par les probabilités de séjour dans les états qui satisfont la propriété $\phi_{perf}$ :

$$s \in SAT \text{ avec } SAT = \{s \in LTS | s \models \phi_{perf}\}$$

Les récompenses sont associées aux états de la façon suivante :

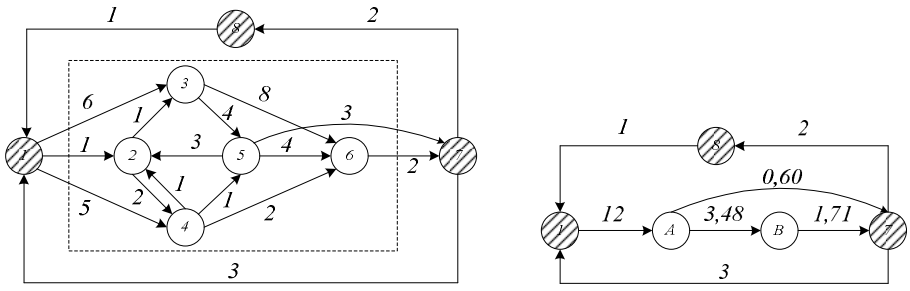
$$r = \begin{cases} 0 & \text{si } s_i \in LTS \backslash SAT. \\ r_i & \text{si } s_i \in SAT. \end{cases}$$

L'indice de performance est calculé en utilisant la formule suivante :

$$Perf = \sum_{i=1}^{\|SAT\|} \pi_{s_i} \times r_{s_i} \tag{3.42}$$

Nous prenons un exemple pour clarifier la situation d'agrégation en utilisant les distributions phase type. Considérons la CMTC dans la figure 3.28(a). En supposant que les récompenses sont attachées aux états 1, 7 et 8, l'indice de performance nécessite seulement les probabilités $\pi_1, \pi_7, \pi_8$. Nous avons calculé le vecteur des probabilités de la chaîne originale :

$$\pi = (0.0585, 0.0527, 0.0336, 0.0995, 0.0334, 0.3009, 0.1404, 0.2809)$$




(a) Avant l'agrégation. (b) Après l'agrégation.

Figure 3.28 – Agrégation en utilisant les distributions $PH$.

Pour réduire l'espace d'états nous avons cherché à remplacer la chaîne dans la boîte par la distribution de phase appropriée, en utilisant notre programme en MatLab pour la recherche de la distribution $PH$ avec la méthode d'égalité des moments. Les 3 premiers moments sont calculés avec l'équation 3.5. Étant donné que le $cv^2 = 1,74 > 1$, la distribution $PH$ obtenue est une distribution coxian à deux phases, et la chaîne agrégée est donnée dans la figure 3.28(b). Le vecteur de probabilités obtenu est :

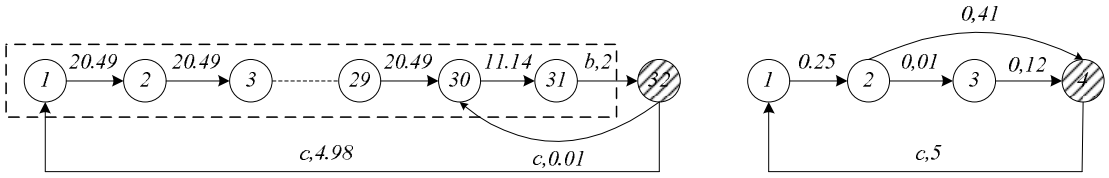
$$\pi = (\pi_1 = 0.0585, \pi_A = 0.1716, \pi_B = 0.3486, \pi_7 = 0.1404, \pi_8 = 0.2809)$$

Les taux de transitions de la chaîne originale ont été choisis au hasard, et les résultats obtenus sont identiques en considérant les états la chaîne originale et ceux de la chaîne agrégée. Si le coefficient de variation de la matrice à agréger est $cv^2 < 0.5$, alors un certain taux d'erreur dans l'approximation peut avoir lieu en limitant le nombre de phases.

Cette technique va s'avérer utile avec l'utilisation des distributions phase-type, pour représenter une séquence d'états qui n'interviennent pas dans l'analyse de performances par un petit nombre de phases (2 ou 3 phases). Suite à l'équation 3.37, la séquence des distributions acycliques avec une phase exponentielle va augmenter le coefficient de variation, et par conséquent réduire le nombre de phases nécessaires pour la représentation de cette séquence. Considérons la modélisation algébrique suivante comme un exemple d'application :

$$\begin{aligned} P_1 &\stackrel{def}{=} (a, U(1,2)).(b,\lambda).(c,\mu).P_1 \\ P_2 &\stackrel{def}{=} (c,\top).P_2 \\ Système_1 &\stackrel{def}{=} P_1 \underset{c}{\bowtie} P_2 \end{aligned}$$

L'approximation de la distribution $U(1,2)$ avec la méthode de moments, nécessite 27 phases pour une approximation d'ordre-2, et 30 phases pour une approximation d'ordre-3. Mais dans le système algébrique, l'exécution de $a$ est toujours suivie par l'action exponentielle $b$, ce qui réduit le nombre de phases nécessaires lors d'une approximation d'ordre-3 de 30 à une distribution $ACPH_3$, constaté après l'application de la matrice au programme. Les chaînes originale et agrégée sont présentées dans la figure 3.29, et les vecteurs de probabilités sont donnés dans le tableau 3.5, où on peut constater l'égalité des probabilités dans les états correspondants à l'exécution de l'action $c$ dans les 2 chaînes (32 et 4). Cette




(a) Diagramme de transitions du $système$.

(b) Diagramme de transitions agrégé.

Figure 3.29 – La chaîne de Markov sous-jacente du système.

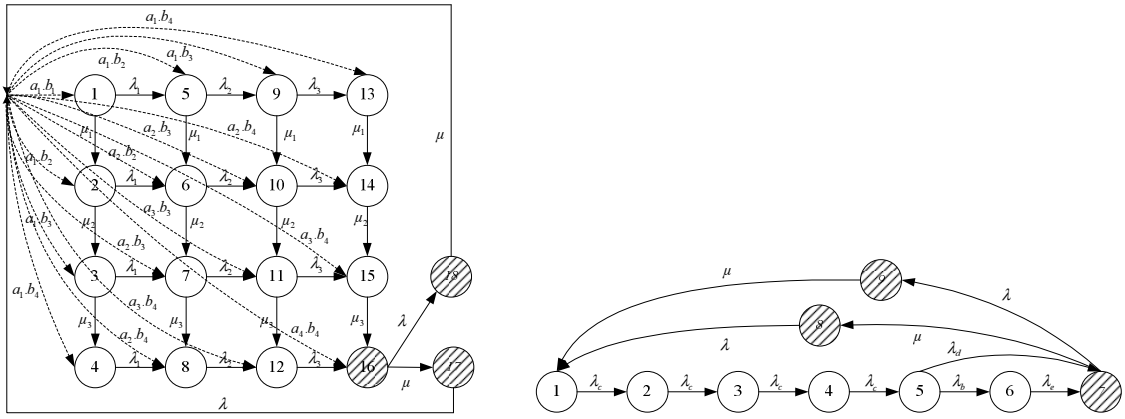
| Chaîne de Markov | Vecteur de probabilités | | | |
|---|---|---|---|---|
| Originale | $\pi_{i\|i\in\{1,29\}} = 0.0072583$ | $\pi_{30} = 0.01339$ | $\pi_{31} = 0.74627$ | $\pi_{32} = 0.029851$ |
| Compressée | $\pi_1^c = 0.57597$ | $\pi_2^c = 0.34807$ | $\pi_3^c = 0.046112$ | $\pi_4^c = 0.029851$ |

Tableau 3.5 – Distribution de probabilités de 2 chaînes.

technique de réduction est très efficace avec l'opérateur synchronisation de PEPA, car ce formalisme utilise la stratégie de synchronisation patiente, qui a une représentation correspondante à l'exécution de deux actions sans entrelacement possible envers les phases intermédiaires. Pour illustrer la situation, nous prenons l'exemple suivant :

$$\begin{aligned} P_1 &\stackrel{def}{=} (a, X_G).(b,\lambda).P_1 \\ P_2 &\stackrel{def}{=} (a, Y_G).(c,\mu).P_2 \\ Système_2 &\stackrel{def}{=} P_1 \underset{a}{\bowtie} P_2 \end{aligned}$$

La variable aléatoire $X_G$ est distribuée selon la loi uniform $U(0.5, 1.5)$, et $Y_G$ est Weibull($k = 2, \lambda = 4$). Ces deux variables sont représentées par $APH_4$. La chaîne sous jacente et son agrégation sont données dans la figure 3.30. Les vecteurs de probabilités obtenues sont $\pi_1 = (0.026574, 0.014496, 0.0049805, 0.14787, 0.10872, 0.052844, 0.10919, 0.036398, 0.21839)$ et $\pi_2 = (0.1578, 0.1578, 0.1578, 0.1578, 0.0047, 0.00012, 0.1092, 0.036398, 0.21839)$. Une fois encore, la chaîne est réduite de moitié dans le pire des cas, car dû au faible coefficient de variation correspondant aux états agrégés. Mais, le parallélisme représenté par 15 phases, peut se réduire à 2 phases seulement.



(a) Diagramme de transitions du $système$.

(b) Diagramme de transitions agrégé.

Figure 3.30 – La chaîne de Markov sous-jacente du système.

## 3.10 Spécification de haut niveau

Nous commençons par rappeler quelques définitions de l'algèbre tensorielle, utilisées pour dériver le générateur de la chaîne associée à la composition de plusieurs processus. Ensuite, nous décrivons la spécification algébrique à haut niveau de la représentation d'une distribution générale par une distribution phase-type acyclique et canonique de 4 phases maximum (présentée dans la figure 3.5), ainsi que les modifications nécessaires pour éviter l'explosion de l'espace d'états.

Avant de traiter le problème de la spécification de haut niveau, nous rappelons la définition de 2 opérateurs de base de l'algèbre tensorielle (somme et produit), ainsi que leurs propriétés.

### 3.10.1 Algèbre tensorielle

L'algèbre tensorielle est définie par deux opérateurs matriciels : le produit tensoriel (appelé produit de Kronecker) et la somme tensorielle.

#### Produit tensoriel

**Définition 3.3.** *Le produit tensoriel de deux matrices $D \otimes E$, de dimensions $(n_1 \times m_1)$ et $(n_2 \times m_2)$ respectivement, est une matrice de dimensions $(n_1 n_2 \times m_1 m_2)$ donnée par :*

$$C = D \otimes E = \begin{pmatrix} d_{1,1}E & d_{1,2}E & \dots & d_{1,m_1}E \\ d_{2,1}E & d_{2,2}E & \dots & d_{2,m_1}E \\ \vdots & \vdots & \ddots & \vdots \\ d_{n_1,1}E & d_{n_1,2}E & \dots & d_{n_1,m_1}E \end{pmatrix}$$

**Lemme 3.1.** *Étant donné 4 matrices A, B, C et D. Le produit tensoriel possède la propriété suivante :*

$$(A \otimes B)(C \otimes D) = (AC \otimes BD) \tag{3.43}$$

*Les deux propriétés suivantes ne sont qu'une déduction du lemme précédent :*

$$r(A \otimes B) = (rA) \otimes B = A \otimes (rB) \tag{3.44}$$

$$(A \otimes B)^n = A^n \otimes B^n \tag{3.45}$$

**Somme tensorielle**

La somme tensorielle de deux matrices carrées $A = (a_{ij})_{n\times n}$ et $B = (b_{ij})_{m\times m}$, $S = A \oplus B$ est définie selon la formule suivante :

$$S = A \oplus B = (A \otimes I_{mm}) + (I_{nn} \otimes B) \tag{3.46}$$

Prenons par exemple $n = 2$ et $m = 3$, nous aurons :

$$S = \left(\begin{array}{ccc|ccc} a_{11} & 0 & 0 & a_{12} & 0 & 0 \\ 0 & a_{11} & 0 & 0 & a_{12} & 0 \\ 0 & 0 & a_{11} & 0 & 0 & a_{12} \\ \hline a_{21} & 0 & 0 & a_{22} & 0 & 0 \\ 0 & a_{21} & 0 & 0 & a_{22} & 0 \\ 0 & 0 & a_{21} & 0 & 0 & a_{22} \end{array}\right) + \left(\begin{array}{ccc|ccc} b_{11} & b_{12} & b_{13} & 0 & 0 & 0 \\ b_{21} & b_{22} & b_{23} & 0 & 0 & 0 \\ b_{31} & b_{32} & b_{33} & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & b_{11} & b_{12} & b_{13} \\ 0 & 0 & 0 & b_{21} & b_{22} & b_{23} \\ 0 & 0 & 0 & b_{31} & b_{32} & b_{33} \end{array}\right)$$

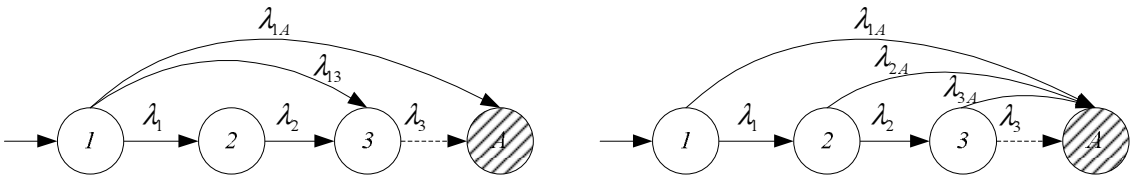
$$S = \left(\begin{array}{ccc|ccc} a_{11}+b_{11} & b_{12} & b_{13} & a_{12} & 0 & 0 \\ b_{21} & a_{11}+b_{22} & b_{23} & 0 & a_{12} & 0 \\ b_{31} & b_{32} & a_{11}+b_{33} & 0 & 0 & a_{12} \\ \hline a_{21} & 0 & 0 & a_{22}+b_{11} & b_{12} & b_{13} \\ 0 & a_{21} & 0 & b_{21} & a_{22}+b_{22} & b_{23} \\ 0 & 0 & a_{21} & b_{31} & b_{32} & a_{22}+b_{33} \end{array}\right)$$

La somme tensorielle $S = A \oplus B$ décrit une CMTC sous-jacente de la composition parallèle de deux processus, décrits à travers leurs matrices générateur $A$ et $B$. Ces deux processus se comportent d'une manière concurrente et indépendante les uns des autres.

### 3.10.2 Description algébrique

La spécification des distributions $PH$ au niveau algébrique semble délicate avec l'existence des probabilités initiales dans les distributions $PH$. L'existence de deux types d'actions (immédiates et exponentielles) dans le formalisme algébrique « EMPA », permet de modéliser facilement les distributions $PH$, mais le théorème de progression maximale et la condition de course entre les actions, rendent le résultat du choix entre deux actions $c$ et $b$, dont une ($c$ par exemple) est distribuée selon $APH_2$ et l'autre ($b$) exponentielle (ou temporisée), est l'engagement dans l'exécution de l'action $PH$ sans jamais exécuter l'action exponentielle.

Les représentations canonique d'ordres 2 et 3 (*cf.* figure 3.31) permettent de remplacer une distribution canonique par une autre acyclique de même ordre, mais avec une probabilité initiale égale à 1. Les valeurs des paramètres dans ces distributions sont fonctions de ceux de la forme canonique $ACPH_n$, et




(a) CF-2. (b) CF-3.

Figure 3.31 – Entrelacements des actions $a$ et $b$.

ils sont détaillés dans [Hor03]. Ce qui permet d'avoir l'illusion d'éviter le piège du théorème de progression maximale lors du choix entre deux actions dont une est temporisée et l'autre est $PH$. La spécification algébrique d'une action avec une distribution $PH$ représentée sous sa forme $CF-1$ et $CF-2$ avec 4 phases seulement, est donnée par les expressions suivantes :

$$\begin{aligned} CF-1 \ : \ (b, X_G).P &\stackrel{def}{=} (\tau, \infty_{1,a_1}).(\tau, \lambda_1).(\tau, \lambda_2).(b, \lambda_3).P + \\ &\quad (\tau, \infty_{1,a_2}).(\tau, \lambda_2).(b, \lambda_3).P + (\tau, \infty_{1,a_3}).(b, \lambda_3).P \\ CF-2 \ : \ (b, X_G).P &\stackrel{def}{=} (\tau, \lambda_1).(\tau, \lambda_2).(b, \lambda_3).P + (\tau, \lambda_{13}).(b, \lambda_3).P + \\ &\quad (b, \lambda_{14}).P \end{aligned} \tag{3.47}$$

Pour illustrer le problème de la spécification avec la représentation $CF-1$, nous considérons le choix dans l'expression suivante :

$$\begin{aligned} Q &= (b, X_G).0 \ + \ (c, \mu).0 \\ &= (\tau, \infty_{1,a_1}).(\tau, \lambda_1).(\tau, \lambda_2).(b, \lambda_3).0 \ + \ (\tau, \infty_{1,a_2}).(\tau, \lambda_2).(b, \lambda_3).0 + \\ &\quad (\tau, \infty_{1,a_3}).(b, \lambda_3).0 \ + \ (c, \mu).0 \end{aligned}$$

Dans cette expression, l'action $c$ ne sera jamais exécutée suite au théorème de la progression maximale, où l'action immédiate est plus prioritaire que l'action temporisée.
Les 2 formes de spécifications dans (3.47) permettent de modéliser facilement les distributions $PH$ pour les opérateurs de : séquence, re-nommage, abstraction, récursivité. Mais, la spécification avec les opérateurs de choix et de composition parallèle, semble difficile à haut niveau suite aux entrelacements entre les phases additionnelles.
Le choix dans l'expression $(b, X_G).P + (c, \mu).Q$ après le remplacement de $(b, X_G)$ par sa forme $CF-2$, ne représente pas le minimum de 2 variables aléatoires, où l'engagement dans l'exécution d'un chemin de l'action $b$ écarte l'exécution de $c$. En revanche, si $X_G$ et $Y_G$ sont deux variables aléatoires représentant les délais des actions $b$ et $c$, et qui sont distribuées selon $PH$ où $X_G \sim PH[a, Q_1]$ et $Y_G \sim PH[b, Q_2]$, le minimum de ces 2 variables suit une distribution $PH$ donnée par $[\gamma, Q]$, avec :

$$\gamma = a \otimes b \quad Q = Q_1 \oplus Q_2 \tag{3.48}$$

Par conséquent, la spécification du choix revient à la spécification de la matrice sous-jacente du produit tensoriel, pour prendre en compte les entrelacements entre les états intermédiaires. Cette tâche a été automatisée par le biais d'un convertisseur créé sous MatLab, par la création d'un fichier qui transforme la matrice résultante du produit tensoriel en une description algébrique donnée dans un fichier.
Pour plus de clarification, nous considérons les deux processus $R$ et $T$, utilisés dans la section 3.9. Les diagrammes de transitions pour les processus $R$ et $T$ avec la distribution $ACPH_4$ sont donnés dans la figure 3.27. La matrice générateur de la chaîne absorbante, sous jacente de la composition parallèle de 2

variables $PH$ ($X$ et $Y$) avec la stratégie de synchronisation patiente est donnée par :

$$\gamma = [a \otimes b, b_{n+1}a, a_{n+1}b] \quad Q = \begin{pmatrix} Q_1 \oplus Q_2 & I \otimes (Q_2 \times 1) & (Q_1 \times 1) \otimes I \\ 0 & Q_1 & 0 \\ 0 & 0 & Q_2 \end{pmatrix} \tag{3.49}$$

Mais, la stratégie de synchronisation dans le formalisme « EMPA », rend le diagramme de transition beaucoup plus simple en cas de synchronisation, car elle ne peut avoir lieu qu'entre une action active et d'autres passives $((b,\lambda).0||_b(b,*).0) = (b,\lambda).(0||0)$. Donc, le problème d'entrelacement entre les phases n'existe pas pour les actions en synchronisation appliquant la stratégie maître-esclave. Par contre, si la liste de synchronisation est vide, le diagramme de transition est celui donné dans la figure 3.27(b), et qui est sous-jacent à la composition parallèle, de la spécification algébrique avec la forme canonique.
En résumé, l'opérateur de choix est le seul opérateur où le concepteur doit prendre des précautions lors de son utilisation avec les distributions phase-type, en ajoutant les entrelacements entre les phases additionnelles.

### 3.10.3 Stratégie de présélection

Contrairement à la stratégie de course, la stratégie de présélection base son choix entre deux actions, d'une manière indépendante de la valeur de la variable aléatoire, mais tout simplement en associant un poids à chaque action, de façon similaire à la politique de résolution de conflit entre deux actions immédiates en EMPA.

Bravetti et Gorrieri dans [BG99], ont reconnu la nécessité de l'utilisation de cette stratégie lors de la conception d'un nouveau formalisme de l'algèbre des processus avec des distributions générales (IG-SMP). L'implémentation de la stratégie de présélection ne nécessite aucune modification à l'outil existant, mais juste un changement dans la description algébrique fournie par le concepteur, pour déterminer comment agir en cas du choix entre deux actions. Pour illustrer cette modification, nous prenons le processus $S$ :

$$S = (b,\lambda_{1,a_1}).0 + (c,\mu_{1,1-a_1}).0 \Leftrightarrow S = (\tau,\infty_{1,a_1}).(b,\lambda).0 + (\tau,\infty_{1,a_1}).(c,\mu).0$$

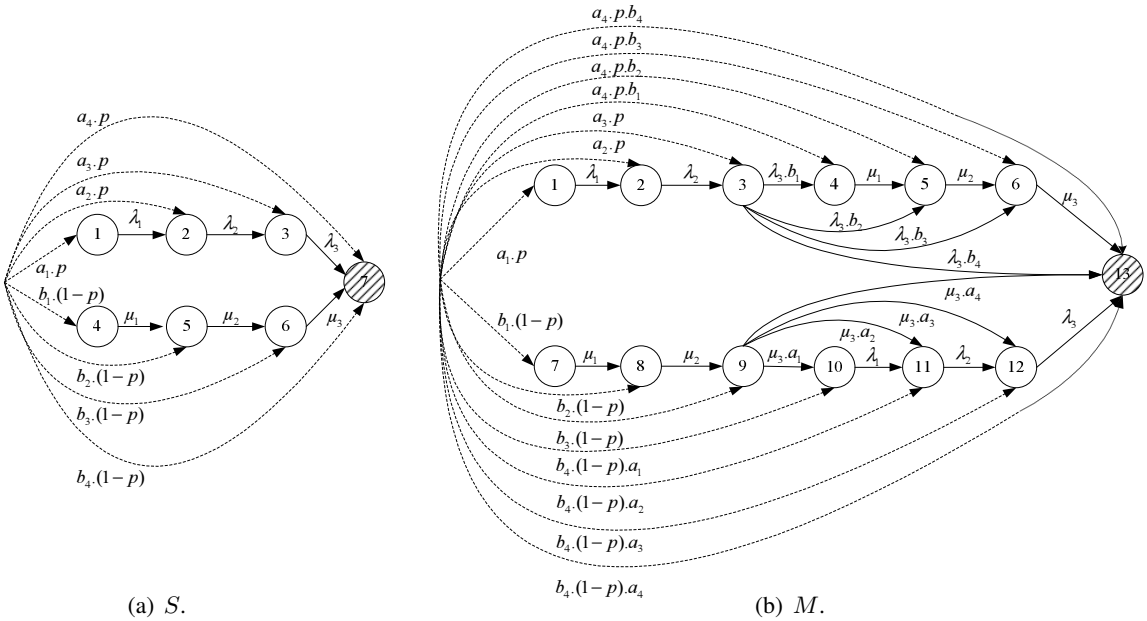
L'avantage de cette stratégie est le choix primitif de l'action à activer avec les opérateurs de choix et de composition parallèle, ce qui écarte tout entrelacement possible entre les phases additionnelles avec l'utilisation des distributions $PH$, où une seule action est activée dans un état donné. Le choix est résolu par l'activation d'une seule action, selon la probabilité associée, et la composition parallèle se ramène à un choix entre deux exécutions séquentielles des actions en question, car selon le théorème de l'expansion (*cf.* théorème 3.8), l'opérateur de composition parallèle se transforme en un opérateur de choix, comme l'illustre l'expression suivante :

$$\begin{aligned} M &= (b,X_{1,a_1}).0||(c,Y_{1,1-a_1}).0 \\ &= (b,X_{1,a_1}).(0||(c,Y_{1,1-a_1}).0) + (c,Y_{1,1-a_1}).((b,X_{1,a_1}).0||0) \end{aligned}$$

**Théorème 3.8.** *Soient $P$ et $Q$ deux processus donnés par $P = \sum a_j.P_j$ et $Q = \sum b_l.Q_l$. Le théorème de l'expansion permet d'exprimer l'entrelacement de l'exécution des actions lors de la composition parallèle de ces deux processus en utilisant l'opérateur de choix :*

$$\begin{aligned} P||_{a_1\cdots a_n}Q = &\sum_{a_j \notin \{a_1\cdots a_n\}} a_j.(P_j||_{a_1\cdots a_n}Q) + \sum_{b_l \notin \{a_1\cdots a_n\}} b_l.(P||_{a_1\cdots a_n}Q_l) + \\ &\sum_{a_j = b_l \in \{a_1\cdots a_n\}} a_j.(P_j||_{a_1\cdots a_n}Q_l) \end{aligned}$$

La figure 3.32 montre les diagrammes de transitions sous-jacents du processus $S$ et $M$, en supposant que les actions $b$ et $c$ dans ces processus, suivent une distribution générale. Ces diagrammes sont obtenus en remplaçant les distributions générales $G$ par une distribution $ACPH_4$ et tout en utilisant la politique de présélection. Cette stratégie a pour avantage de permettre plus facilement la réduction de la chaîne sous-jacente du modèle algébrique avec la technique utilisée précédemment.




(a) $S$. (b) $M$.

Figure 3.32 – Diagrammes de transitions de $S$ et $M$.

En terme de représentation matricielle, la matrice générateur infinitésimal de l'opérateur de choix entre deux actions $(b, X_G)$ et $(c, Y_G)$, étant données les 2 matrices générateur de la forme $ACPH_4$ correspondantes à leurs distributions ($Q_1$ et $Q_2$), ainsi que les vecteurs de probabilités initiales ($a$ et $b$), est calculée de la fonction suivante :

*Fonction choix* $[c, Q] =$ *Entrée*$(p, a, b, Q_1, Q_2)$

$$c = [p \times a, (1-p) \times b]$$

$$et$$

$$Q = \begin{bmatrix} Q_1 & 0 \\ 0 & Q_2 \end{bmatrix}$$

Selon la figure 3.32(b), la recherche de la composition parallèle se ramène dans un premier temps à la recherche de la représentation matricielle de l'opérateur de séquence de deux actions :

*Séquence* $[c, Q] =$ *Entrée*$(a, b, Q_1, Q_2)$

$$c = [a, a_{n+1} \times b]$$

$$et$$

$$Q = \begin{bmatrix} Q_1 & -(Q_1 \times 1) \times b \\ 0 & Q_2 \end{bmatrix}$$

La matrice de l'opérateur du parallélisme est calculée selon la fonction suivante :

*Fonction parallèle* $[c, Q] =$ *Entrée*$(p, a, b, Q_1, Q_2)$

$$c = [p. * (a_1, a_2, a_3, a_4.b_1, a_4.b_2, a_4.b_3, a_4.b_4), (1-p). * (b_1, b_2, b_3, b_4.a_1, b_4.a_2, b_4.a_3, b_4.a_4)]$$

$$et$$

$$Q = \begin{bmatrix} Q_1 & (-Q_1 * 1) * b & 0 & 0 \\ 0 & Q_2 & 0 & 0 \\ 0 & 0 & Q_2 & (-Q_2 * 1) * a \\ 0 & 0 & 0 & Q_1 \end{bmatrix}$$

### 3.10.4 Exemple de modélisation

Nous considérons la file $M/G/1/2q/2q$ avec 2 classes de priorités, et $q$ clients par classe, avec un service prioritaire préemptif, comme exemple d'application à la modélisation avec les stratégies de course et de présélection. Nous avons spécifié ce système de manière compositionnelle à partir de trois processus. La spécification algébrique avec la stratégie de course est présentée dans la figure 3.33.

$$QS_{M/G/1/2q/2q} = A||_{Arr}(Q_{0,0}||_{Del}S)$$

$$Arr = a_H, a_L$$

$$Del = s_H, s_L$$

$$\begin{aligned}
A &\stackrel{def}{=} (a_H, \lambda_H).A + (a_L, \lambda_{L.}).A \\
Q_{0,0} &\stackrel{def}{=} (a_H, *).Q_{1,0} + (a_L, *).Q_{0,1} \\
Q_{i,0} &\stackrel{def}{=} (a_H, *).Q_{i+1,0} + (a_L, *).Q_{i,1} + (s_H, *).Q_{i-1,0} \qquad 0 < i < q-1 \\
Q_{0,j} &\stackrel{def}{=} (a_H, *).Q_{1,j} + (a_L, *).Q_{0,j+1} + (s_L, *).Q_{0,j-1} \qquad 0 < j < q-1 \\
Q_{q,0} &\stackrel{def}{=} (a_H, *).Q_{q,0} + (a_L, *).Q_{q,1} + (s_H, *).Q_{q-1,0} \\
Q_{0,q} &\stackrel{def}{=} (a_H, *).Q_{1,q} + (a_L, *).Q_{0,q} + (s_L, *).Q_{0,q-1} \\
Q_{i,j} &\stackrel{def}{=} (a_H, *).Q_{i+1,0} + (a_L, *).Q_{i,j+1} + (s_H, *).Q_{i-1,j} \qquad 0 < i < q \wedge 0 < j < q \\
S &\stackrel{def}{=} (s_H, \mu).S + (s_L, X_G).S
\end{aligned}$$

Figure 3.33 – Modélisation algébrique de la file $M/G/1/2q/2q$.

Les arrivées des paquets sont modélisées par l'action $a_i$, les départs du système par l'action $s_i$, synchronisant le départ d'un paquet de la file vers l'extérieur. Les paquets arrivent selon la distribution exponentielle avec un taux $\lambda_H = 0.6$ pour la classe $H$, et $\lambda_L = 0.4$ pour la classe $L$. Le temps de service de la classe $H$ est exponentiellement distribué avec un taux $\mu = 1$, et celui de la classe $L$ suit une distribution générale $G$. Nous avons utilisé les distributions $Weibull(\alpha = 1/2, \beta = 1)$ et Pareto bornée $BP(k = 0.1, P = 150, \alpha = 1.2)$ dont les fonctions de distribution sont données dans (3.50) et (3.51). Plus de détails sur les parametres de la distribution $BP$ sont données dans la section 4.4.2 du chapitre 4.

$$F_W(x) = 1 - e^{-(x/\beta)^\alpha} \tag{3.50}$$

$$F_{BP}(x) = \frac{1}{1-(k/P)^\alpha}\left[1-(k/x)^\alpha\right] \tag{3.51}$$

La chaîne de Markov sous-jacente du modèle algébrique avec une capacité du système égale à 2, est présentée dans la figure 3.34. Les approximations des fonctions des distributions $Weibull(\alpha = 1/2, \beta =$

1) et $BP(k = 0.1, P = 150, \alpha = 1.2)$ par les distributions $PH$ correspondantes sont données dans les figures 3.35(a) et 3.35(b). Soit $\pi$ le vecteur de distribution de probabilités obtenu avec l'approximation de

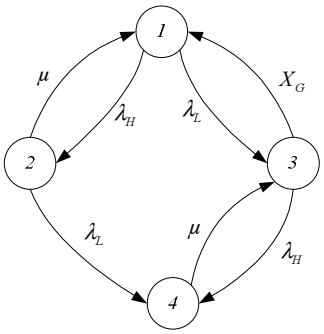
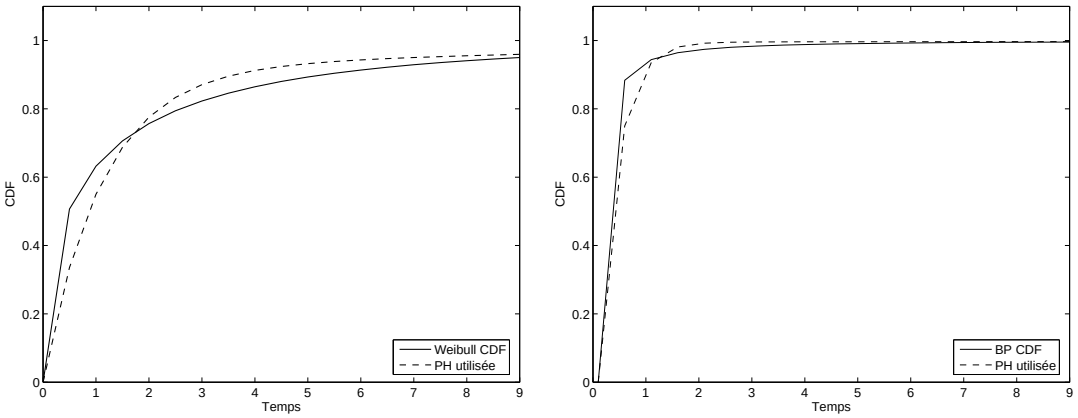



Figure 3.34 – Chaîne de Markov de $M/G/1/2/2$




(a) $Weibull(\alpha = 1/2, \beta = 1)$. (b) $BP(k = 0.1, P = 150, \alpha = 1.2)$.

Figure 3.35 – Approximations par des distributions $PH$.

distribution $G$ par une distribution $PH$ correspondante, tous les indices de performances seront calculés à partir de ce vecteur. Nous nous intéressons au taux d'utilisation du système, ainsi que la probabilité d'avoir deux clients dans le système (probabilité de blocage). Ces deux paramètres sont calculés en utilisant les formules suivantes :

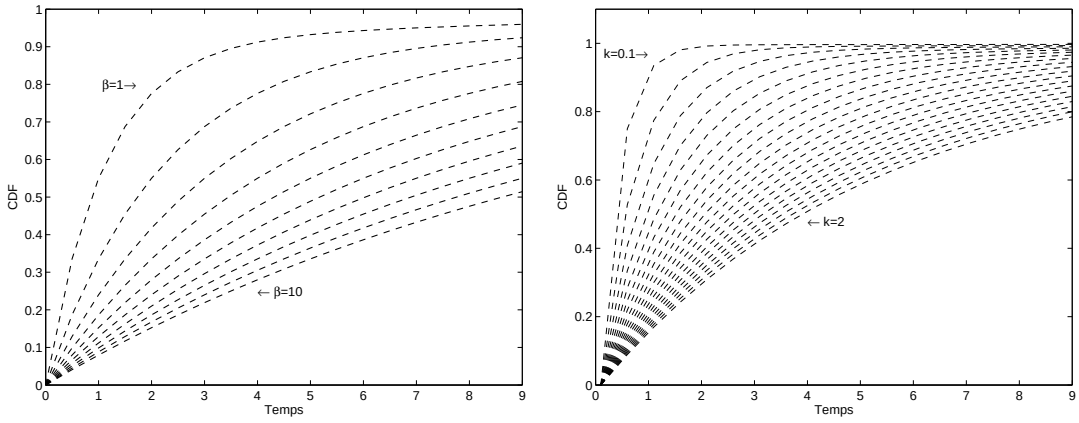
$$Utilisation = 1 - \pi_1 \tag{3.52}$$

$$P(2\ clients) = \pi_4 \tag{3.53}$$

Les courbes de variations de ces indices en fonction des paramètres des distributions utilisées (*cf.* figure 3.36) sont présentées dans les figures 3.37 et 3.38 avec une stratégie de course, et dans les figures 3.39 et 3.40 avec une stratégie de présélection.

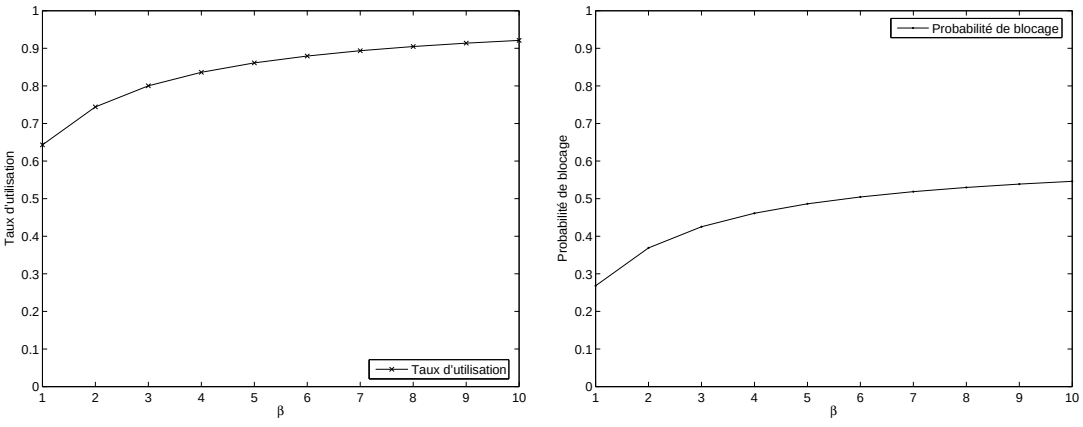
## 3.11 Un petit mot sur les distributions $PH$ à temps discret

La plupart des recherches sur l'évaluation de performances ont supposé une échelle de temps continu. Moins d'attention a été portée aux systèmes à temps discret et les distributions phase-type discrètes. Toutefois, ceci est dû au fait que les modèles à temps discret sont plus difficiles à modéliser, car plusieurs événements peuvent avoir lieu pendant une même unité de temps, et il faut choisir quelle transition



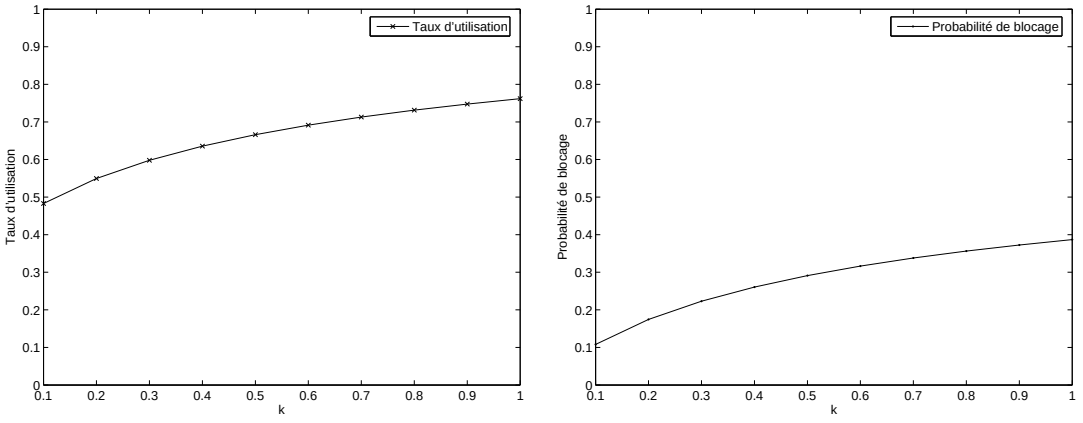

(a) $Weibull(\alpha = 1/2, \beta = 1 \dots 10)$. (b) $BP(k = 0.1 \dots 2, P = 150, \alpha = 1.2)$.

Figure 3.36 – Variation des distributions avec $\beta$ et $k$.



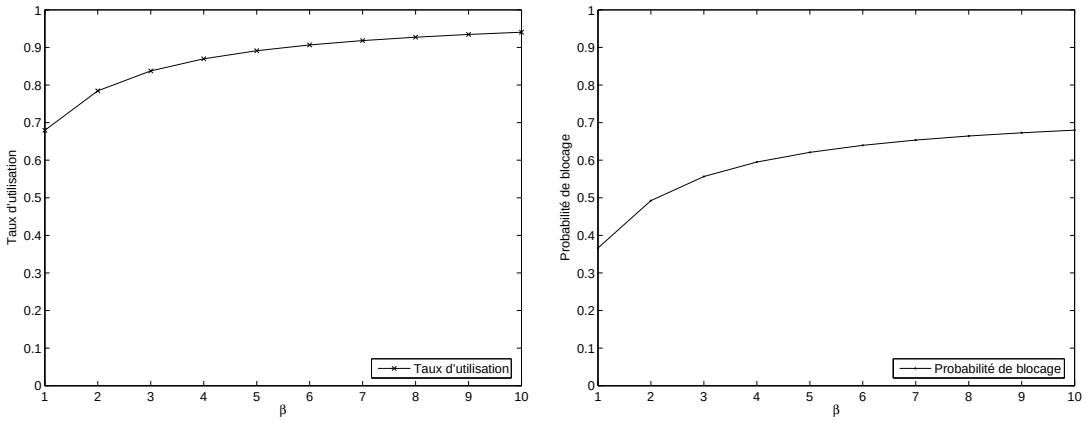

(a) Taux d'utilisation. (b) Probabilité de blocage.

Figure 3.37 – Variation des indices de performances avec $Weib(\alpha = 1/2, \beta = 1 \dots 10)$.



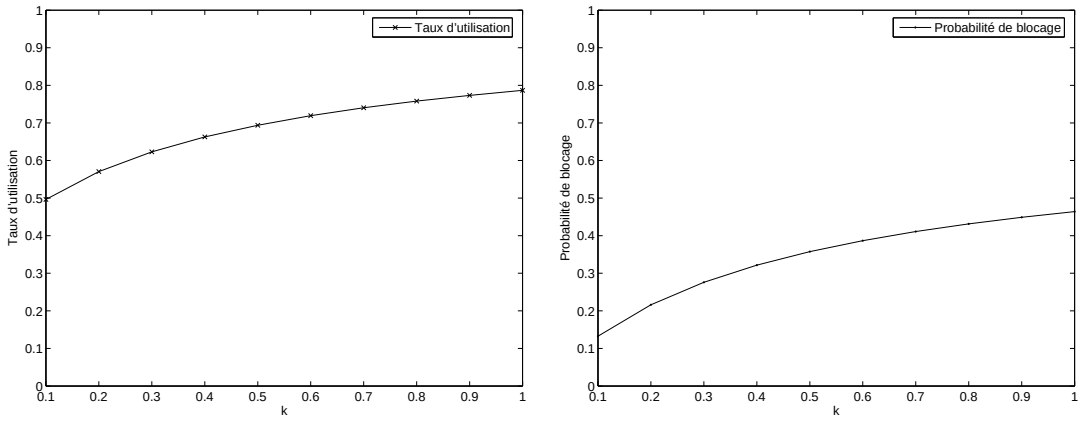

(a) Taux d'utilisation. (b) Probabilité de blocage.

Figure 3.38 – Variation des indices de performances avec $BP(k = 0.1 \dots 1, P = 150, \alpha = 1.2)$.




(a) Taux d'utilisation. (b) Probabilité de blocage.

Figure 3.39 – Variation des indices de performances avec $Weib(\alpha = 1/2, \beta = 1 \ldots 10)$.




(a) Taux d'utilisation. (b) Probabilité de blocage.

Figure 3.40 – Variation des indices de performances avec $BP(k = 0.1 \ldots 1, P = 150, \alpha = 1.2)$.

effectuer. La sémantique de la stratégie de présélection résout ce problème en précisant, en cas de conflit, l'événement ayant lieu en premier selon une priorité et une probabilité d'exécution.

Les distributions phase-type discrètes peuvent être utilisées pour représenter à l'aide d'un petit nombre de phases, une distribution générale dont le coefficient de variation est faible, même avec $cv^2 = 0$, contrairement aux distributions à temps continu, dont le coefficient de variation minimal est donné par $cv^2_{\min} = 1/n$ (théorème d'Aldous [AS87]). Ces distributions sont capables de bien représenter une distribution déterministe ($d = \tau$) avec un nombre d'états $n = \tau/\sigma$, $sigma$ étant l'unité d'échantillonnage. Par exemple, la représentation de la distribution déterministe $d$ par une distribution phase-type à temps discret $[a, P]$ et un intervalle $\sigma = 1$, nécessite un nombre de phases $n = d/\sigma = d$ (comme le montrent la figure 3.41 et l'équation 3.54). La distribution déterministe $d = 4$ par exemple, peut être représentée par 4 phases comme l'illustre la figure 3.42(a). Par contre, si $n = d/\sigma$ n'est pas un nombre entier, alors la distribution phase-type ne pourra être qu'une approximation ($n = \lfloor d/\sigma \rfloor$ pour une sous-estimation, et $n = \lceil d/\sigma \rceil$ pour une sur-estimation).

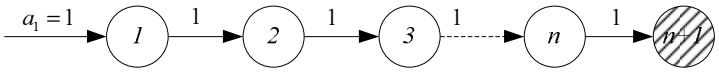
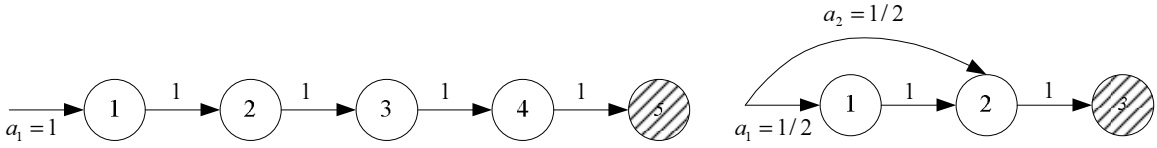



Figure 3.41 – $ADPH_n$ pour représenter une distribution déterministe $d$.

$$a = [1, 0, \ldots, 0] \quad et \quad P = \begin{bmatrix} 0 & 1 & 0 & 0 & \ldots & 0 \\ 0 & 0 & 1 & 0 & \ldots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \ldots & 0 \end{bmatrix} \tag{3.54}$$

La distribution uniforme discrète $U(1,2)$ peut être représentée par 3 phases avec les distributions $PH$ à temps discret, comme l'illustre la figure 3.42(b). Mais, les distributions à temps continu peuvent être bien représentées par les distributions phase-type à temps discret ($DPH$) avec un petit nombre de phases, et un intervalle de discrétisation $\sigma$ suffisamment petit. Généralement, une seule phase suffit pour la représentation de la distribution exponentielle, comme l'illustre la figure 3.43 pour l'approximation de la distribution exponentielle avec un taux $\lambda = 0.5$. La largeur du pas de discrétisation joue un rôle très important dans la précision des résultats des analyses numériques appliquées sur la chaîne de Markov discrète obtenue. Horváth dans [Hor03] a déterminé les valeurs supérieure et inférieure de l'intervalle du pas de discrétisation, données par l'équation 3.55. Pour plus de clarification, les représentations de la distribution continue uniforme $U(1,2)$ par $DPH$ avec $n = 1, 2, 4, 8$ phases et un pas de discrétisation $\sigma = 0.1$ dans la figure 3.44(a), et pour une valeur de $\sigma = \lceil (1/n - cv^2) . \mu_1 \rceil$ sont présentées dans la figure 3.44(b).

$$\left[\frac{1}{n} - cv^2(X)\right] E(X) < \delta \leqslant \frac{E(X)}{n-1} \tag{3.55}$$




(a) $ADPH$ pour $d = 4$. (b) $ADPH$ pour $U(1,2)$.

Figure 3.42 – Représentation des distributions déterministe(4) et Uniforme(1,2) par les distributions $PH$ à temps discret.

Sachant qu'un système réel contient des actions stochastiques et déterministes, la combinaison de deux distributions (géométrique et exponentielle) ne peut pas produire une chaîne sans mémoire à tout instant. Étant donné qu'une distribution phase-type à temps discret n'est autre qu'une combinaison de distributions géométriques (sans mémoire), elle est sans mémoire mais à des instants discrets qui sont multiples de $\sigma$, contrairement aux distributions $PH$ à temps continu, qui sont sans mémoire à tout instant. Pour pallier ce problème, la méthode de l'uniformisation permet de transformer une CMTC en une CMTD.

### 3.11.1 Méthode de l'uniformisation

L'uniformisation associe à une CMTC donnée par la matrice générateur $Q$, une CMTD gouvernée par la matrice de transition $P$ donnée par :

$$P = I + \frac{Q}{q} \qquad avec \qquad q \geqslant \max_i |\lambda_{ii}| \tag{3.56}$$

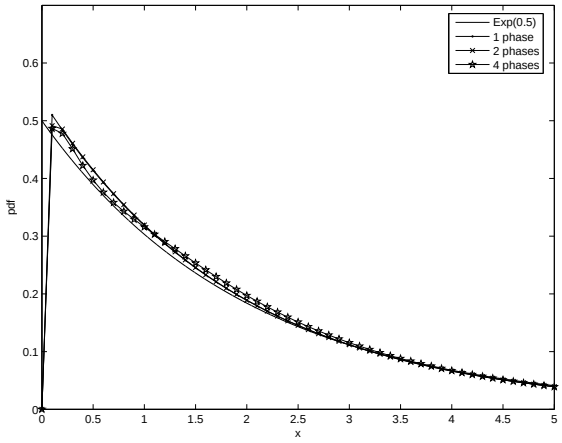
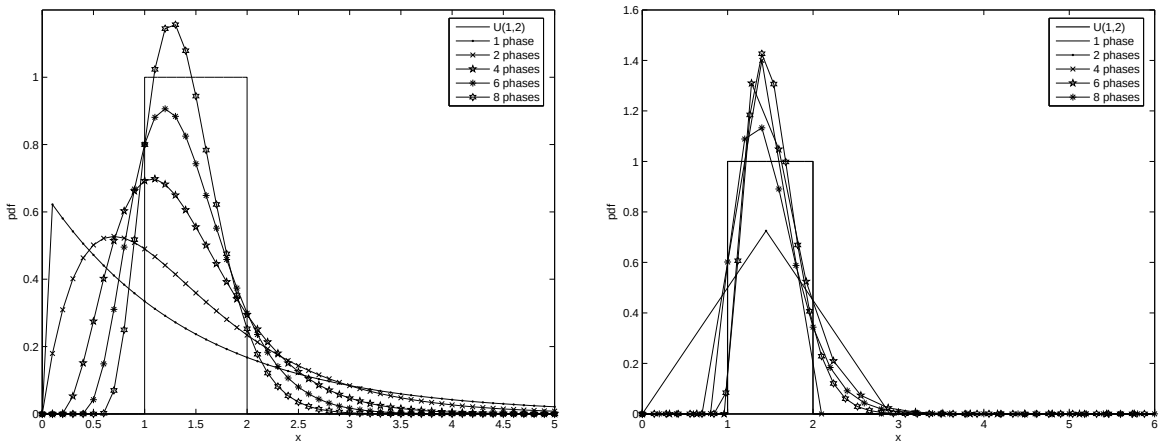



Figure 3.43 – Approximation de la distribution $EXP(0.5)$ avec $DPH$.




(a) $U(1,2)$ avec $\sigma = 0.1$.

(b) $U(1,2)$ avec $\sigma = 1.45, 0.7, 0.32, 0.2, 0.14$.

Figure 3.44 – Représentation de la distribution $U(1,2)$ par $DPH$.

Où $I$ est la matrice identité d'ordre $n$. Il en résulte que $Q = q(P - I)$ et que :

$$\pi(t) = \pi(0)e^{Qt} = \pi(0)e^{q(P-I)t} = \pi(0)e^{-qIt}e^{qPt} = \pi(0)e^{-qt}e^{qPt} \tag{3.57}$$

Le calcul de l'exponentiel d'une matrice est obtenu en utilisant la série de Taylor/MacLaurin comme étant :

$$e^{Qt} = \sum_{i=0}^{\infty} \frac{(Qt)^i}{i!} \tag{3.58}$$

En utilisant l'expansion à travers une série de Taylor, l'équation 3.57 prend la forme suivante :

$$\begin{aligned} \pi(t) &= \pi(0)e^{-qt} \sum_{n=0}^{\infty} \frac{(qt)^n P^n}{n!} = \pi(0) \sum_{n=0}^{\infty} \psi(qt,n)P^n = \sum_{n=0}^{\infty} \psi(qt,n)\left(\pi(0).P^n\right) \\ &= \sum_{n=0}^{\infty} \psi(qt,n).\hat{\pi}(n) \end{aligned} \tag{3.59}$$

*où :*

$$\psi(qt,n) = e^{-qt}\frac{(qt)^n}{n!} \quad et \quad n \in \mathbb{N} \tag{3.60}$$

$\psi(qt, n)$ dans l'équation 3.60 n'est autre que la distribution de Poisson avec une valeur moyenne $qt$, c'est-à-dire la probabilité de la transition avec un taux $q$ dans l'intervalle $[0, t[$. $\hat{\pi}(n)$ est le vecteur de distribution de probabilité de la CMTD avec une matrice de transition $P$. Le vecteur $\hat{\pi}(n)$ est calculé de la façon suivante :

$$\hat{\pi}(0) = \pi(0) \quad et \quad \hat{\pi}(n) = \hat{\pi}(n-1).P \quad et \quad n \in \mathbb{N}^+ \tag{3.61}$$

Pratiquement, on peut calculer une valeur approximative de $\pi_t$ en utilisant l'équation 3.61 et en arrêtant la somme à un certain terme $k_a$, de telle sorte que l'erreur d'approximation soit bornée.

$$\pi(t) \approx \tilde{\pi}(t) = \sum_{n=0}^{k_a} \psi(qt, n)\hat{\pi}(n)$$

$$et \tag{3.62}$$

$$\|\pi(t) - \tilde{\pi}(t)\| \leqslant 1 - \sum_{n=0}^{k_a} e^{-qt}\frac{(qt)^n}{n!} \leqslant \varepsilon$$

Où $\varepsilon$ est la tolérance spécifiée par l'utilisateur. On choisit $k_a$ tel que :

$$1 - \sum_{n=0}^{k_a} e^{-qt}\frac{(qt)^n}{n!} \leqslant \varepsilon \Rightarrow \sum_{n=0}^{k_a} \frac{(qt)^n}{n!} \geqslant \frac{1-\varepsilon}{e^{-qt}} = (1-\varepsilon)e^{qt} \tag{3.63}$$

## 3.12 Conclusion

Dans ce chapitre, nous avons rappelé les notions de base et les travaux principaux dans le domaine de la recherche des distributions phases-type minimales, et nous avons proposé un nouvel algorithme pour l'approximation des distributions ayant une grande variabilité (heavy-tailed) avec un faible nombre de phases. Ensuite, nous avons cherché une méthode d'agrégation lors de l'utilisation de ces distributions avec les opérateurs algébriques, pour éviter l'explosion de l'espace d'états, et nous avons identifié dans la suite, les problèmes inhérents à la spécification de haut niveau ainsi que les solutions appropriées pour contourner ces problèmes.

L'objectif de ce travail était, d'une part, l'étude de la spécification de systèmes réels par le biais des distributions phase type ayant un nombre de phases minimales, et d'autre part, la recherche d'une technique de réduction de l'espace d'états.

Nous croyons que les résultats et les observations mentionnées dans ce chapitre sont tout à fait appropriées pour modéliser et analyser les systèmes réels. Pour cela, après ces deux premiers chapitres qui ont présenté des techniques de modélisation à l'aide des algèbres de processus stochastiques, la suite de cette thèse se concentre sur la conception de nouveaux modèles pour fournir une qualité de service dans les réseaux ad hoc.

# Qualité de service dans les réseaux Ad Hoc

# Chapitre 4

# Un nouveau mécanisme de gestion de la QdS dans les réseaux ad hoc

Durant les dernières années, l'émergence des technologies multimédia avec les équipements sans fil, a renforcer les exigences des applications en terme de la Qualité de Service (*QdS*) dans les réseaux ad hoc. La qualité de service attendue du réseau ad hoc est équivalente (sinon mieux) à celle offerte par les équipements de télécommunications sans fil.

Dans les réseaux filaires, de nombreux travaux ont conduit à la définition de solutions de QdS et l'étude de ces solutions a permis de déterminer leurs avantages et leurs limites. Les réseaux ad hoc introduisent de nombreuses contraintes liées essentiellement à l'utilisation du médium radio d'une façon distribuée, à la mobilité et à l'absence de coordinateur centrale, compliquant les solutions de QdS.

Dans ce chapitre, nous présentons le fruit de premiers travaux dans la conception d'un mécanisme de gestion de la qualité de service de bout en bout dans les réseaux ad hoc. Nous proposons une solution adaptative avec la dynamicité du réseau ad hoc, ensuite nous réalisons une modélisation algébrique avant d'utiliser la simulation, pour analyser l'influence de la mobilité sur les performances du modèle en question. Nous présentons à la fin du chapitre les résultats des simulations mettant en évidence une différenciation proportionnelle entre les délais des différentes applications.

Ce chapitre débute par l'analyse des problèmes posés par les réseaux ad hoc, puis nous présentons un état de l'art des solutions de la qualité de service au sens large du terme, qui ont été proposées dans le contexte des réseaux ad hoc. Un modèle de gestion de la QdS est présenté dans la suite, puis modélisé algébriquement et évalué par simulation, sous différents scénarios et conditions du réseau.

## 4.1 Définition et caractéristiques du réseau ad hoc

Un réseau ad hoc [CM99] est un ensemble des terminaux mobiles, qui communiquent à travers leurs cartes sans fil tout en formant un réseau temporaire, sans l'existence d'une infrastructure pré-déployée, ni l'intervention d'un administrateur. Ce type de réseau se forme généralement d'une manière spontanée et dynamique avec l'apparition et le mouvement de nœuds. Chaque nœud (par nœud on veut dire terminal) communique directement avec les autres nœuds situés dans la portée de la zone d'émission et de réception de son interface radio, et indirectement avec les autres nœuds situés à l'extérieur de sa zone d'émission, en exploitant la coopération de nœuds intermédiaires jouant le rôle de routeurs, en relayant les informations vers les autres. Dans ce type de réseau, chaque nœud joue le rôle d'une station émettrice et d'un routeur.

Le fonctionnement des réseaux ad hoc n'est pas proche de celui des réseaux cellulaires, car ces derniers reposent entièrement sur la présence de stations de base assurant la connexion et le routage par le biais d'un réseau filaire.

Les réseaux ad hoc sont une alternative aux réseaux IP et la facilité de leur déploiement est un avantage indéniable dans certaines circonstances où l'on ne dispose pas (ou plus) d'infrastructure. Les participants d'une réunion, les intervenants des opérations de secours menées sur des zones touchées

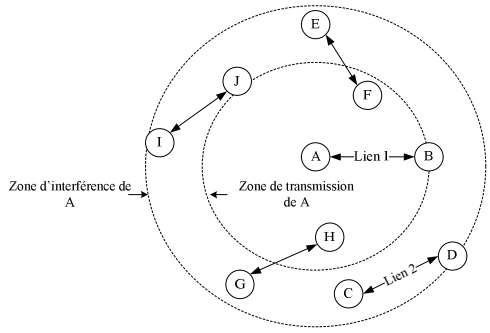



Figure 4.1 – Effet de l'interférence dans un réseau ad hoc.

par une catastrophe naturelle qui a détruit toute l'infrastructure (un tremblement de terre, un tsunami, un volcan, ou un attentat), les éléments engagés sur un champ de bataille, peuvent tous tirer profit des caractéristiques de tels réseaux pour échanger des informations. Ainsi que les zones montagneuses et rurales qui ne disposent pas d'une infrastructure câblée couvrant les zones habitées, où sans aucun doute l'investigation dans des infrastructures dans ces zones n'est pas favorable pour des raisons de coût d'investissement et d'amortissement.

A partir de cette définition et de ces applications générales, il est intéressant de mettre en avant les caractéristiques principales qui différencient un réseau ad hoc d'un réseau IP classique. D'abord, il s'agit d'un canal radio où le signal contenant l'information est soumis à toutes les contraintes d'affaiblissement du signal lors de la propagation dans l'air. En comparaison aux réseaux filaires, la bande passante est faible avec une large fluctuation dû à la moindre source d'interférence, et tout en restant dépendant de la variation des conditions climatiques qui engendrent l'évanouissement du signal. La bande passante du canal radio est rare à cause des régulations qui empêchent l'utilisation de la totalité du spectre. Shannon a fixé les limites du débit d'informations transmissibles sur un canal de bande passante donnée. Par conséquent, il ne sera pas possible d'accroître indéfiniment la bande passante allouée aux communications radio, comme c'est le cas dans le réseau filaire.

L'accès distribué au médium partagé oblige les nœuds voisins de rentrer dans une phase de conflit pour accéder à ce dernier. Pour empêcher le conflit, des mécanismes de gestion de transmission centralisée et décentralisée sont apparue [ER00, ST05, WG99, AC01, DC99, BCV01, KLS$^+$01, WG05, SK96, VBG00, RNT03, NABT03], où une station centrale synchronise l'accès d'un groupe de nœuds dans la méthode d'accès centralisée, et un algorithme décentralisé qui permet à chaque nœud de lutter contre les autres pour avoir son opportunité d'accéder au médium.

L'interférence entre les liens voisins dans la figure 4.1, force tous les nœuds à se taire durant la transmission de $A \mapsto B$, où la zone d'interférence de $A$ dépasse le lien 1. Différentes solutions à ce problème pour s'assurer que les transmissions n'interfèrent pas, comme l'utilisation d'une antenne directionnelle pour limiter la zone d'interférence, ou la transmission de chaque nœud dans un intervalle de temps (TDMA) [ZC02], ou encore l'utilisation de la modulation de code (CDMA) au-dessus d'une infrastructure TDMA [Hay00, ER00, Lin01, ST05], etc.

Ensuite, la mobilité des nœuds constitue à l'évidence une caractéristique très spécifique aux réseaux ad hoc, et engendre la formation d'une topologie non maîtrisée a priori. Dans cette vision, les nœuds peuvent joindre une zone de couverture d'un réseau, ce qui peut engendrer la formation de nouveaux liens, ou encore des coupures de liens à tout moment lorsqu'un nœud jouant le rôle de routeur, quitte la zone de couverture du réseau, ainsi que la redistribution du trafic traversant ce dernier. Sachant que les flots traversant un chemin, contenant un lien qui vient d'être coupé, doivent être interrompus et réacheminés sur un autre chemin quand ce dernier existe, les paquets en cours de transmission en avant du lien coupé seront inévitablement perdus. Par conséquent, la mobilité des nœuds va provoquer un changement fréquent de la topologie du réseau de manière aléatoire. Les protocoles usuels d'Internet ne sont pas toujours adaptés à des changements fréquents de topologie, ce qui rend les techniques de routage

du réseau IP non convenable à ce type de réseau.

Contrairement au réseau IP, où il y a une distinction nette entre les stations émettrices et les routeurs intermédiaires, prenant en charge l'acheminement des données, cette différence n'existe pas dans les réseaux ad hoc car tous les nœuds peuvent être amenés à assurer des fonctions de routage.

Enfin, la consommation d'énergie constitue un problème important pour des équipements fonctionnant grâce à une alimentation électrique autonome. Ces équipements intègrent des modes de gestion d'énergie et il est important que les protocoles mis en place dans les réseaux ad hoc prennent en compte ce problème, tout en réduisant le nombre de paquets de contrôle. Beaucoup de recherches ont été investies au niveau du routage car la gestion de la mobilité génère davantage de flux de contrôle et de signalisation lorsqu'un changement de topologie est détecté. Nous allons revenir sur ces protocoles de routage dans la section 4.2.4.

Différentes études ont proposé des méthodes pour fournir une qualité de service face à ces problèmes, que ce soit au niveau liaison ou au niveau routage. Plusieurs mécanismes d'accès pour faire de la réservation de la bande passante, ou de la différenciation fonctionnelle entre les trames pour assurer que les plus prioritaires soient transmises avant les autres, et divers protocoles de routage qui cherchent une ou plusieurs routes, satisfaisant un certain paramètre de *QdS*, sont apparus. Ces mécanismes sont insuffisants pour garantir une *QdS* de bout en bout durant la durée de vie d'une session. Il est donc certain qu'un mécanisme de *QdS* performant et adapté à la mobilité et à la variation des ressources dans ce type de réseaux est plus que nécessaire.

## 4.2 Qualité de Service et réseaux ad hoc

Le terme qualité de service regroupe une multitude de concepts distincts. Dans nos propres sens, il désigne tout mécanisme permettant d'adapter le comportement du réseau aux besoins des applications. Cette notion englobe les mécanismes permettant d'allouer une portion des ressources du réseau à un flot, ou bien de garantir un délai de bout en bout borné, et un taux de perte limité, etc.

Sûr, que c'est un travail qui s'étale sur plusieurs couches de protocoles, et qui nécessite parfois une collaboration entre elles, et même la création d'une sous-couche intermédiaire pour gérer la communication. Pour cela, avant de faire un état de l'art sur les différents mécanismes de gestion de la *QdS* dans ce type de réseaux, nous allons rappeler les différents mécanismes utilisés dans chaque couche, en commençant du bas vers le haut, et tout en identifiant les problèmes inhérentes à ces derniers, et les éléments de réponse qu'on peut apporter à chaque niveau.

### 4.2.1 Mécanismes d'accès au médium

Dans les réseaux ad hoc, le support de communication est partagé entre les nœuds situés dans la même zone de couverture de leurs cartes réseau. Pour gérer l'accès au medium, différents mécanismes de contrôle d'accès ont mené à la norme IEEE 802.11b [WG99], où les accès concurrents sont gérés par le protocole de détection de la porteuse CSMA, qui est bien approprié aux medias de transmission par diffusion. Ce protocole repose sur une écoute du canal de transmission, couplée à une attente aléatoire avant émission, afin de réduire la fréquence des émissions simultanées des trames rendant souvent la réception de ces trames impossible à cause de collisions.

### 4.2.2 Le protocole d'accès au médium IEEE 802.11b

Dans les réseaux ad hoc, les nœuds partagent le canal d'une manière distribuée en utilisant la fonction DCF (Distributed Coordination Function) de la norme IEEE 802.11b. Cette fonction met en œuvre un certain nombre de mécanismes visant non pas à détecter les collisions mais à les éviter, étant donné l'impossibilité de l'émetteur de mesurer la qualité du signal au niveau du récepteur, et l'évanouissement du signal lors de la propagation dans l'air. La fonction DCF est basée sur le mécanisme CSMA/CA, et l'algorithme de rétention binaire (Binary Exponentiel Backoff – BEB), pour gérer les collisions.

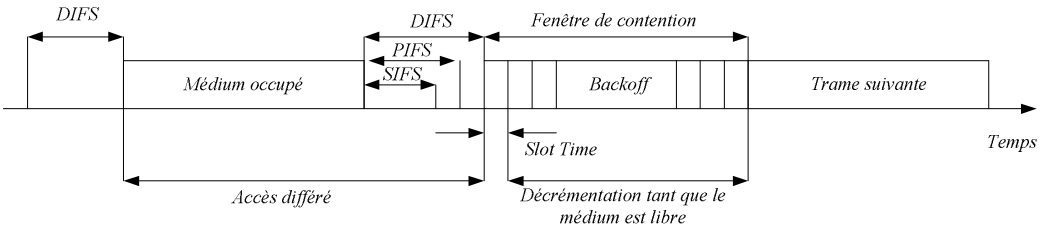



Figure 4.2 – Méthode d'accès *DCF*.

L'algorithme de DCF est exécuté localement sur chaque station afin de déterminer les périodes d'accès au médium. Avant chaque émission, la station émettrice teste l'inactivité du canal. S'il est libre durant un certain laps de temps défini par DIFS (Distributed Inter Frame Space), la transmission est alors possible. Par contre, s'il est occupé, une procédure de *backoff* est enclenchée. La station choisit alors un nombre aléatoire $BT$ (appelé backoff Timer) dans l'intervalle $[0, CW_{min}]$, selon la fonction de distribution uniforme discrète, et reporte sa transmission jusqu'à la libération du médium (*cf.* figure 4.2). Lorsque le support devient libre pour une période DIFS, la variable $BT$ est décrémentée d'une valeur de 1 ($BT_{nouvelle} = BT_{ancienne} - 1$) chaque unité de temps que le médium est resté libre. Cette unité du temps est appelé « SlotTime » dans la suite de ce document. La première station à atteindre la valeur 0 émet alors sa trame. Les autres stations suspendront le processus qui sera repris dés la fin de la transmission actuelle.

Après la transmission, l'émetteur attend l'acquittement correspondant de la part du récepteur. Ce dernier procède à une détection d'erreurs au moyen d'un CRC standard, pour vérifier que la trame ne contient pas d'erreur avant d'envoyer un acquittement. Cet acquittement indique à l'émetteur qu'aucune collision n'a eu lieu. L'absence de réception de cet acquittement provoque la retransmission de la trame et ce processus sera répété jusqu'au succès de l'opération, ou jusqu'à atteindre le nombre maximal de retransmissions autorisées ($R = 7$). Au-delà de $R$ retransmissions, la trame est détruite.

Ce mécanisme ne permet évidemment pas de supprimer les collisions entre trames. Si deux émetteurs tirent la même valeur aléatoire $BT$, ils émettront au même instant. A l'image d'Ethernet, l'attente aléatoire sera tout d'abord tirée dans un intervalle de valeurs faible $[0, CW_{min}]$. En cas de collision, la taille de cet intervalle sera doublée par les stations qui sont impliquées, afin de réduire le risque d'une nouvelle collision. Si encore une collision se reproduit, cette valeur sera encore doublée comme le montre l'équation 4.1, jusqu'à atteindre une limite maximale définie par la norme ($CW_{max}$). Après une transmission avec succès, la station émettrice réinitialise sa fenêtre de contention à $CW_{min}$. Les valeurs de tous ces paramètres sont donnés dans le tableau 4.1.

$$CW = \begin{cases} 2^j CW_{min} & 0 \leqslant j \leqslant m \\ 2^m CW_{min} & m \leqslant j \leqslant R \end{cases} \tag{4.1}$$

| *Paramètres* | *Slot Time* | $SIFS$ | $DIFS = SIFS + 2 \times aSlotTime$ | $CW_{min}$ | $CW_{max}$ | $R$ |
|---|---|---|---|---|---|---|
| 802.11b | 20$\mu$s | 10$\mu$s | 50 $\mu$s | 32 | 1024 | 7 |

Tableau 4.1 – Paramètres du standard IEEE 802.11b.

**Modélisation algébrique du mécanisme CSMA/CA**

Pour évaluer les performances du mécanisme CSMA/CA, lors de son utilisation dans une topologie composée de $n$ stations voisines, comme le montre la figure 4.3, nous avons modélisé algébriquement

chaque station par le processus $S_n$, et le médium partagé par $CH$. La description algébrique est donnée par :

$$
\begin{aligned}
CSMA/CA &\stackrel{def}{=} (S_1||\ldots||S_n)||_C CH \\
C &= \{idle, busy, tm_i, ra_i\} \\
S_i &\stackrel{def}{=} \langle gen\_msg, \mu\rangle .Sensor_i \\
Sensor_i &\stackrel{def}{=} \langle idle, \infty_{11}\rangle .Propagation_i + \langle busy, \infty_{11}\rangle .BACKOFF_i \\
Propagation_i &\stackrel{def}{=} \langle tm_i, \infty_{11}\rangle .WAIT\_ACK_i \\
BACKOFF_i &\stackrel{def}{=} \langle standby_i, \nu\rangle .Sensor_i \\
WAIT\_ACK_i &\stackrel{def}{=} \langle ra_i, *\rangle .S_i + \langle to, \theta\rangle .BACKOFF_i \\
CH &\stackrel{def}{=} CH_{tx}||_{toggle} CH_{State} \\
CH_{State,i} &\stackrel{def}{=} \langle idle, *\rangle .CH_{State,i} + \langle toggle, *\rangle .CH_{State,b} \\
CH_{State,b} &\stackrel{def}{=} \langle busy, *\rangle .CH_{State,b} + \langle toggle, *\rangle .CH_{State,i} \\
CH_{tx} &\stackrel{def}{=} \sum_{i=1}^{n} \langle tm_i, *\rangle .Trans \\
Trans &\stackrel{def}{=} \langle elapse\_prop\_delay, \eta\rangle . \langle toggle, \infty_{11}\rangle .Success + \\
&\qquad \sum_{j=1, j\neq i}^{n} \langle tm_j, *\rangle .Collision \\
Success &\stackrel{def}{=} \langle propagate, \lambda\rangle . \langle ra_i, \vartheta\rangle . \langle toggle, \infty_{11}\rangle .CH_{tx} \\
Collision &\stackrel{def}{=} \langle toggle, \infty_{11}\rangle . \langle elapse\_prop, \eta\rangle . \langle toggle, \infty_{11}\rangle .CH_{tx}
\end{aligned}
$$

Où $tm_i$, $ra_i$, $standby_i$, et $to$ sont les actions représentant la transmission d'une trame, la réception d'une accusée, la procédure de backoff, et le timeout.

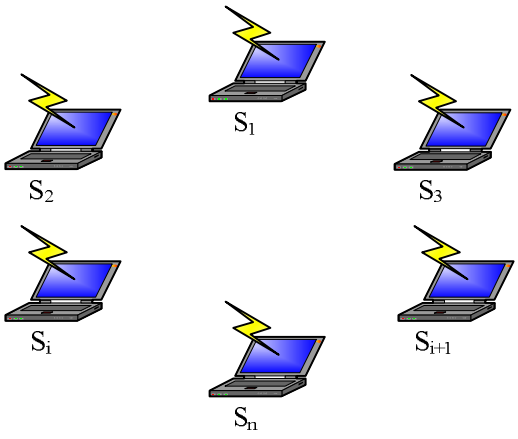



Figure 4.3 – Topologie.

La station $S_i$ commence par se renseigner sur l'état du médium (libre ou occupé) avant de transmettre ($tm_i$), et elle rentre dans une phase de backoff ($BACKOFF_i$) s'il est occupé. Avant l'expiration du temps de propagation, le médium peut sembler encore libre pour une autre station $j$, dont la transmission ($tm_j$ pour $j \neq i$) provoque une collision.

La méthode algébrique de récompense a été utilisée pour calculer le taux d'utilisation du médium et la probabilité de collision, tout en variant le nombre de stations et le temps de transmission (ou bien la taille de la trame). Les valeurs de $1/\lambda$ pour les différentes tailles des trames utilisées sont données dans

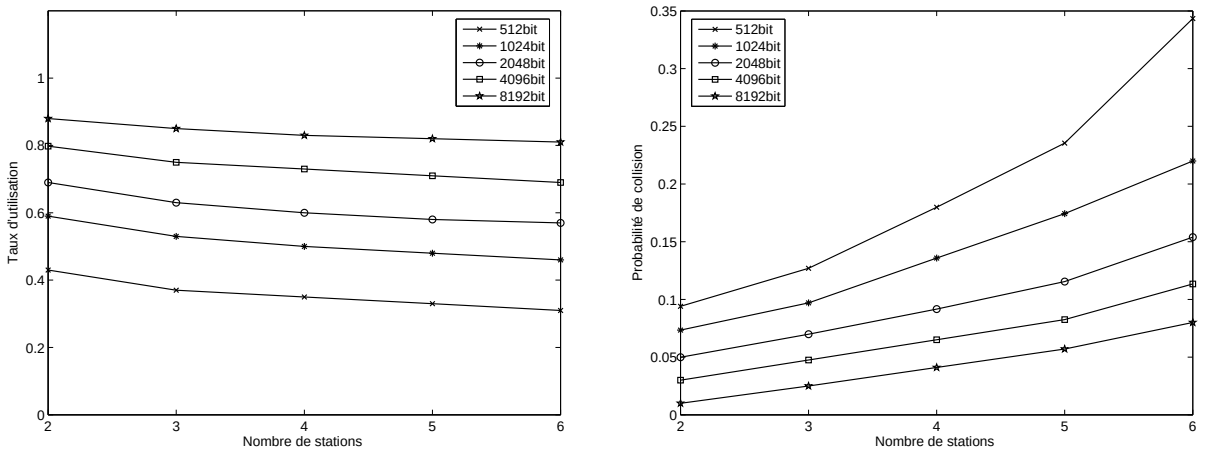
le tableau 4.2, et la variation de la taille de la chaîne de Markov, en fonction du nombre des stations, est donnée dans le tableau 4.3. Les courbes de variation de ces 2 paramètres sont données dans les figures 4.4(a) et 4.4(b), qui montrent que le taux d'utilisation augmente et la probabilité de collision diminue avec la taille d'une trame.

| Taille de la trame (bit) | Temps de transmission $1/\lambda$ ($\mu$s) |
|---|---|
| 512 | 46.54 |
| 1024 | 93.09 |
| 2048 | 186.18 |
| 4096 | 186.18 |
| 8192 | 744.72 |

Tableau 4.2 – Temps de transmission.

| Station | États | Transitions |
|---|---|---|
| 2 | 54 | 144 |
| 3 | 243 | 810 |
| 4 | 972 | 3888 |
| 5 | 3645 | 17010 |
| 6 | 13122 | 69984 |

Tableau 4.3 – Taille de la chaîne de Markov.



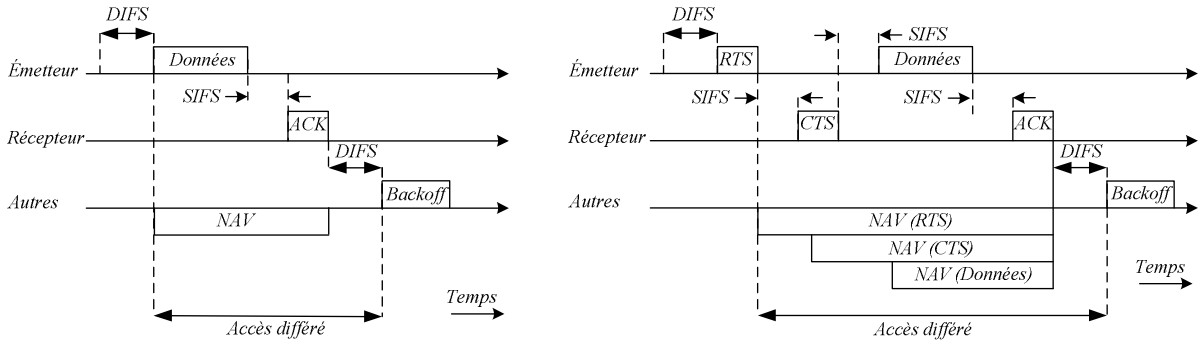

(a) Taux d'utilisation.

(b) Probabilité de collision.

Figure 4.4 – Performances du modèle algébrique CSMA/CA.

Toutefois, une collision ne pouvant être détectée, l'envoi d'une trame ne sera jamais interrompu et occupera le canal radio jusqu'à la fin de la transmission. Pour réduire l'impact des collisions sur les messages de données longs, il est possible de faire précéder l'envoi de chaque trame de données par un échange de messages courts comme montre la figure 4.5, alors les collisions arriveront essentiellement sur ces trames courtes. L'émetteur envoie au récepteur une requête d'émission (Request To Send – RTS). Le récepteur, si son canal radio est disponible, autorise l'émetteur à transmettre par une confirmation (Clear To Send – CTS). À la réception de l'autorisation, l'émetteur transmet la trame de données. L'intervalle de temps séparant la réception d'une des trames de cet échange (RTS, CTS, données et acquittement) et l'émission de la suivante est égal à une valeur constante SIFS (plus petit que DIFS), afin d'empêcher l'interruption de ce mécanisme par une autre trame. Tout mobile à portée radio de l'émetteur ou du récepteur captera l'une de ces trames contenant la durée de l'envoi de la trame de données correspondante. Ces voisins s'abstiendront alors de transmettre jusqu'à la fin de cette trame afin de ne pas provoquer de collision. Un mécanisme de détection virtuelle de la porteuse a été ajouté. Chaque station tient à jour un vecteur d'allocation réseau NAV (Network Allocation Vector) contenant la durée des émissions et permettant ainsi de prévoir l'état d'occupation du support physique. Lorsqu'une station

souhaite émettre, elle vérifie auprès du NAV que le support physique est libre.




(a) Données – ACK. (b) RTS – CTS.

Figure 4.5 – Mécanismes de transmission avec et sans RTS/CTS.

Dans un souci d'identification des problèmes de ce mécanisme, qui dégradent ses performances, et afin de définir les mécanismes permettant d'y remédier, le problème de non équité en termes de nombres d'accès au médium a été identifié dans [GW05, Sha04].

**Problème d'iniquité d'accès**

Le standard IEEE 802.11b a été conçu pour permettre un accès équitable au médium, à toutes les stations en compétition. Dans un réseau à un saut, le protocole permet à chaque émetteur d'accéder au médium avec la même probabilité, et par conséquent une certaine équité en terme de nombre de trames émises (sans prendre en compte le volume de données transmis). Beaucoup de travaux ont montré que dans un contexte multi-sauts, des différences dans la topologie, telles que l'existence des zones plus denses que d'autres, ou encore la présence d'interférences entre différents trafics, peuvent avoir un impact important sur la probabilité de transmission avec succès des différents émetteurs. Ce problème vient du grand nombre de contentions et de collisions entre les nœuds qui se trouvent au cœur du réseau, relativement à ceux des frontières. Nous illustrons ce problème d'équité dans l'accès au médium à travers un scénario simple représenté dans la figure 4.6.

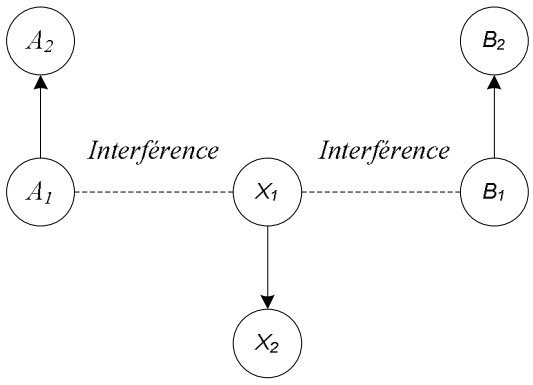



Figure 4.6 – Trois couples émetteur-récepteur sont en zone de détection de porteuse de leurs voisins directs et émettent continuellement des trames.

Les transmissions de $A_1 \mapsto A_2$ et $B_1 \mapsto B_2$ interférent avec celles de $X_1 \mapsto X_2$, sachant que les transmissions $A_1 \mapsto A_2$ et $B1 \mapsto B_2$ n'interfèrent pas entre elles. Le lien $X_1 \mapsto X_2$ est confronté à plus de contention que les liens voisins. Les résultats de la simulation, réalisée sur cette topologie dans QNAP, sont données dans le tableau 4.4, tout en supposant un canal parfait avec un taux de perte égal à zéro.

IEEE 802.11b avec un débit de lien à 11Mbps a été utilisé, et les paramètres de la couche MAC sont donnés dans les tableaux 4.1 et 4.5. Chaque paire a toujours une trame à émettre, et afin de simplifier la modélisation, les trois émetteurs utilisent des trames de longueur égales et constantes.

| Liens | Débit reçu par $A_2$ | Débit reçu par $B_2$ | Débit reçu par $X_2$ |
|---|---|---|---|
| $A_1 \mapsto A_2$ | $\approx 8$ | | |
| $A_1 \mapsto A_2$ et $B_1 \mapsto B_2$ | $\approx 8$ | $\approx 8$ | |
| $A_1 \mapsto A_2$ et $X_1 \mapsto X_2$ | $\approx 4$ | | $\approx 4$ |
| $A_1 \mapsto A_2$, $B_1 \mapsto B_2$ et $X_1 \mapsto X_2$ | $\approx 4$ | $\approx 4$ | $\approx 2$ |

Tableau 4.4 – Problème d'équité dans l'accès au médium avec IEEE 802.11b.

| Entête Mac | 272 bits | Entête physique | 128 bits |
|---|---|---|---|
| ACK & RTS | 112 bits+ entête PHY | CTS | 160 bits + entête PHY |
| Temps de préambule | 144 $\mu$s | Délai de propagation | 1 $\mu$s |
| Timeout de CTS & RTS | 300 $\mu$s | Taille de la trame | 1543 octets (1500 + 34) |

Tableau 4.5 – Paramètres de simulation.

Dans ce scénario, les transmissions entre les nœuds $A_i$ (et entre $B_i$) peuvent exploiter la bande passante au maximum (le 8 Mbps qui reste dû aux entêtes MAC). Même quand les liens entre les $A_i$ et $X_i$ sont actifs, ils partagent équitablement la bande passante (4 Mbps). De l'autre coté, lorsque $A_1$, $B_1$ et $X_1$ sont actifs simultanément, le débit du lien $X_1 \mapsto X_2$ diminue en chute libre. L'émetteur $X_1$ doit patienter en moyenne plus longtemps que $A_1$ et $B_1$ avec lesquels il est en concurrence. Le nombre de collisions qu'il subit est double de celui ressenti par $A_1$ ou $B_1$. En général, dans le cas où les émetteurs en concurrence tirent avant chaque émission, une valeur aléatoire pour le BT (backoff), dans un intervalle croissant avec le nombre de collisions subies, et dont la taille de cet intervalle est réduite à une valeur minimale après une transmission réussie, les différents émetteurs auront un comportement par rafale. En effet, transmettre une trame avec succès implique que le prochain backoff sera tiré aléatoirement dans un intervalle de taille plus faible statistiquement que les émetteurs concurrents. Pour cela, $A_1$ et $B_1$ pourront accéder au médium plus souvent et dans un délai plus court, bloquant le mécanisme de détection de porteuse de $X_1$.

Ce scénario met en évidence une grave inégalité dans l'accès au médium pour les nœuds du cœur, suite aux mécanismes mis en œuvre dans la norme IEEE 802.11b, et qui accentuent par ailleurs le déséquilibre constaté. Ce problème à des conséquences directes sur les couches supérieures (niveau application), où les applications de voix et de vidéo ayant des restrictions en termes de délai, gigue et bande passante, et avec ce problème, se trouvent dans l'incapacité d'accéder au médium.

Ce problème aura aussi une conséquence sur les protocoles de routage et risque de provoquer un effondrement du réseau. Beaucoup de travaux ont été réalisés pour fournir un accès équitable [VBG00, QS02, GW05]. Un autre problème sous-jacent de cette philosophie d'accès équitable, oppose l'équité en terme de paquets et l'équité en terme de flux. Mais même un accès équitable n'est ni désirable, ni souhaitable pour les stations ayant différents niveaux de priorités. Deux mobiles ayant la même probabilité d'accès au médium ne constituent pas forcément un scénario équitable lorsque l'un des deux doit transmettre plus de flux que l'autre. De notre point de vue, il n'est pas équitable de pénaliser les nœuds relayant plus de paquets sous prétexte d'équité.

En plus, étant donné que la fonction DCF permet un accès au médium avec contention, où il s'agit donc d'un service de type au mieux (best effort), cette fonction ne permet pas de faire une différenciation entre les trames de différentes priorités, non seulement entre les nœuds voisins, mais aussi au sein du même nœud.

### 4.2.3 Une extension de la fonction DCF pour la qualité de service

DCF est une fonction d'accès libre n'effectuant aucune gestion de la QdS. En effet, elle est efficace quand la charge du réseau est faible puisque les stations peuvent émettre sans délai, mais elle ne fournit aucune différenciation entre les flux de données de priorités différentes. Elle est conçue pour permettre un accès distribué au médium pour toutes les stations d'une façon équitable, ce qui n'est pas le cas. La décision de transmission est prise localement dans chaque nœud, sans prendre en compte les niveaux de priorité des paquets sur les nœuds voisins. Avec l'absence d'une station de base, il n'y a aucun contrôleur centralisé qui peut distribuer l'accès aux nœuds selon les priorités de leurs paquets. Par conséquence, le nœud possédant réellement le paquet qui a la priorité la plus élevée, et le nœud possédant le paquet qui a la plus faible priorité, ne se rendent pas compte de la situation et rentrent dans une phase de contention pour accéder au médium. Ce qui rend fragile tous les mécanismes de différenciation au niveau de la couche réseau, car lorsqu'il s'agit de définir des priorités d'accès entre différents terminaux ou entre différents flux routés par des terminaux différents, il n'est pas possible de n'utiliser que des files, aussi évoluées soient elles au niveau réseau. Il est nécessaire de modifier le comportement du protocole d'accès au médium.

Durant la phase de contention, le temps d'attente des nœuds détermine leur priorité de transmission, c'est-à-dire le nœud qui aura le plus petit temps d'attente sera prioritaire par rapport aux autres. Un grand nombre de mécanismes de différenciation de services en terme de délai d'accès ont été proposés et analysés [AC01, DC99, KLS$^+$01, MCM$^+$02, NABT03]. La plupart de ces mécanismes sont basés sur la réduction de la fenêtre de contention, le délai entre trame (IFS), et tous les autres paramètres affectant le temps d'attente. D'une façon générale, le nœud qui aura le plus petit délai (DIFS) et qui tirera la plus petite valeur de $BT$ gagnera le droit de transmission par rapport aux autres. Ce qui aura une conséquence directe sur l'augmentation de la bande passante allouée et la réduction des délais de ses paquets. Ces deux paramètres ont été exploités dans l'extension IEEE 802.11e [WG05] pour incorporer et consolider une méthode de différenciation de service distribuée entre les nœuds.

#### IEEE 802.11e

IEEE 802.11e définit deux méthodes d'accès au médium. La première, appelée HCF (Hybrid Coordination Function), est utilisée pour l'accès centralisé, et elle est destinée à l'utilisation d'un point d'accès. L'autre, totalement distribuée, appelée EDCF (Enhanced Distributed Coordination Function), n'est autre qu'une extension de la fonction DCF, en incorporant un mécanisme de différenciation de service. La présentation et l'analyse de cette fonction, sont présentées dans [MCM$^+$02].

Une station EDCF inclut 4 DCF dont chacune est utilisée pour une classe de trafic. Nous allons exploiter cette extension dans la suite de ce manuscrit, pour fournir une QdS (proportionnelle ou absolue) dans le réseau ad hoc.

Une station EDCF contient 4 niveaux de priorité en utilisant 4 stations DCF, ou 4 catégories d'accès (Access Categories – $AC_i | i \in [0, 3]$) selon la taxonomie de la norme. Il ne s'agit plus d'avoir une seule file d'attente pour toutes les trames quelle que soit leur priorité, mais plutôt 4 files d'attente différentes comme le montre la figure 4.7. Il est envisagé de fournir des priorités d'accès en utilisant différents temps d'attente pour chaque catégorie d'accès. Plus le temps d'attente est court, plus la priorité d'accès pour la transmission est grande. Mais comme le temps d'attente d'une station DCF est donné par la somme de la valeur de DIFS (valeur constante), et de la variable aléatoire BT (backoff) choisie dans l'intervalle $[0, CW_{min} - 1]$, l'association de différentes priorités aux 4 catégories d'accès, revient à l'association de différentes valeurs de ces paramètres pour chaque classe. Pour cela, EDCF utilise différentes valeurs pour : le délai entre 2 trames (Arbitration Inter Frame Spacing – $AIFS(AC_i)$), la fenêtre

de contention minimale ($CW_{min}[AC_i]$) et la fenêtre maximale ($CW_{max}[AC_i]$) pour chaque $AC_i$, afin de réaliser une différenciation d'accès entre les trames appartenant aux différentes classes, à la place d'une seule variable $DIFS$, $CW_{min}$, et $CW_{max}$ dans le 802.11 DCF.

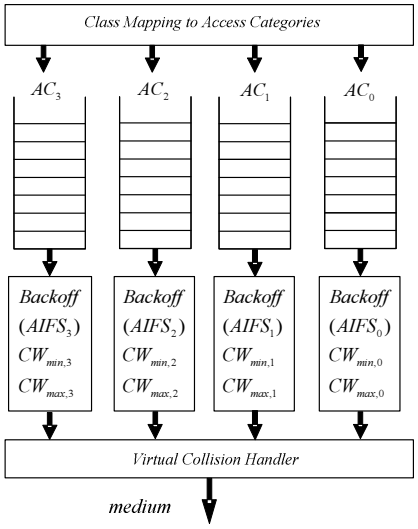



Figure 4.7 – Les 4 catégories d'accès (ACs) d'une station EDCF.

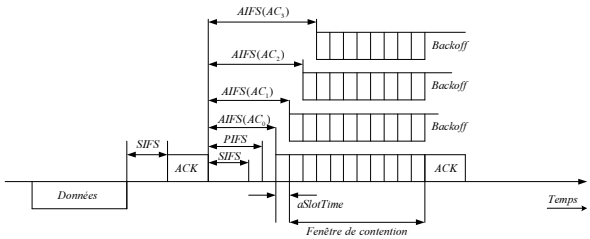



Figure 4.8 – Mécanisme d'accès EDCF.

Donc plusieurs délais inter-trames sont introduits de la façon suivante : $AIFS(AC_0) < AIFS(AC_1) < AIFS(AC_2) < AIFS(AC_3)$. La durée de ces paramètres pour une catégorie d'accès $i$ est calculée de la façon suivante :

$$AIFS(AC_i) = SIFS + AIFSN(AC_i).aSlotTime \tag{4.2}$$

Où $AIFSN(AC_i)$ est un nombre entier strictement positif, et la limite inférieure de $AIFS(AC_i)$ fixée à $DIFS$.

Chaque $AC_i$ a son propre $AIFS(AC_i)$ et son propre temporisateur de *Backoff* ($BT(AC_i)$). Pour qu'une catégorie d'accès $AC_i$ puisse accéder au canal, il est nécessaire que le médium soit libre durant une période correspondante à $AIFS(AC_i)$ (*cf.* figure 4.8). Après cette période, un mécanisme similaire à celui de l'accès décentralisé dans la norme 802.11b est utilisé, où une durée de *backoff* est générée après chaque collision selon l'équation suivante :

$$BT(AC_i) = random(0, CW_{i,j} - 1) \times aSlotTime \tag{4.3}$$

où $random()$ est une fonction génératrice d'un nombre aléatoire dans l'intervalle $[0, CW_{i,j} - 1]$, selon la loi de distribution uniforme discrète, et $aSlotTime$ est l'unité de temps. La valeur initiale de la fenêtre de contention est $CW_{i,j} = CW_{min}(AC_i) = 2^{i+3}$ lors de la première tentative de transmission, et après chaque collision, $CW_{ij}$ augmente d'un facteur exponentiel de 2 jusqu'à une certaine valeur maximale $CW_{max}(AC_i)$, comme illustré dans l'équation 4.4. Il est possible d'utiliser différentes valeurs d'augmentation de la fenêtre de contention, où cette dernière est multipliée par un facteur de persistance $PF_i$ modifiable, mais cette variable a été supprimée dans les derniers drafts.

$$CW_{i,j} = \begin{cases} 2^j CW_{i,min} & 0 \leq j \leq m \\ 2^m CW_{i,\min} & m \leq j \leq R \end{cases} \tag{4.4}$$

Dans 4.4, $CW_{ij}$ sera réinitialisée après une tentative réussie à $CW_{i,min}$. $R$ est le nombre maximal de retransmissions au niveau MAC, il est égal à 7 dans $DCF$ et $EDCF$, et $j$ est le nombre de collisions. Le temporisateur $BT(AC_i)$ est décrémenté d'une unité ($BT^{AC_i}_{nouvelle} = BT^{AC_i}_{ancienne} - 1$) durant les périodes d'inactivité du canal. Si deux catégories d'accès $AC_i$ sur la même station ont tiré la même valeur de

$BT(AC_i)$, la collision sera résolue de manière virtuelle. La trame avec la priorité la plus élevée sera transmise, et l'autre trame sera traitée comme étant victime d'une collision en déclenchant le mécanisme de rétention exponentielle.

La fonction EDCF permet donc de définir pour chaque trame un niveau de priorité au sein du même nœud, ainsi qu'un niveau de priorité pour acquérir l'accès au médium. L'évaluation de ce protocole par simulation indique que la différenciation s'opère bien, mais les flots de faible priorité peuvent être victimes de famine dans un environnement surchargé. Un mécanisme de limitation des débits des différents flux pourrait être nécessaire pour pallier ce problème.

Lindgren, Almquist et Schelén dans [LAS03] évaluent les performances de différentes solutions permettant d'effectuer une différenciation de services dans les réseaux locaux sans fil. Ils comparent essentiellement la fonction EDCF de la norme $IEEE$ 802.11$e$, avec Blackburst [SK96] et DFS (Distributed Fair Scheduling) [VBG00]. En terme d'utilisation des ressources radio, Blackburst semble obtenir les meilleures performances, puisqu'il n'engendre aucune collision. Ce protocole permet par ailleurs d'obtenir la différenciation la plus nette entre deux classes de trafic distinctes. Même si Blackburst engendre un surcoût important lorsque la charge du réseau augmente, cette solution reste la meilleure. En termes de délais de transmission, la période de brouillage précédant l'émission de chaque trame dans Blackburst, est compensée par l'absence de collisions dans le réseau, et donc par l'absence d'accroissement du backoff et l'absence de retransmissions. Mais, les auteurs ont oublié de notifier que la différenciation offerte par Blackburst est cependant très marquée, et peut parfois conduire à une famine touchant les flux de faible priorité. La fonction EDCF conduit elle aussi à une différenciation efficace entre deux classes de trafic, mais la présence de nombreuses collisions limite les performances globales du protocole. Toutefois, les performances de l'EDCF dans un réseau ad hoc multi-sauts seront soumises aux mêmes contraintes que la fonction DCF en termes d'inégalité d'accès et de probabilité de collision.

### 4.2.4 Protocoles de Routage dans les réseaux ad hoc

En 1995, l'IETF crée le groupe de travail MANET afin de définir un protocole de routage standard pour les réseaux ad hoc. Généralement ces protocoles sont classés en 3 catégories selon la méthode utilisée pour découvrir le chemin entre la source et la destination :

1. Les protocoles proactifs : qui cherchent à maintenir leur table de routage à jour, en échangeant des paquets « hello » avec les voisins, à intervalles réguliers. Parmi ces protocoles, nous citons par exemple DSDV [PB94] (Destination-Sequenced Distance-Vector) , TBRPF [OTL04](Topology Dissemination Based on Reverse-Path Forwarding), etc.
2. Les protocoles réactifs : qui entreprennent la recherche d'une route uniquement s'il y a des paquets à émettre. Parmi ces protocoles, nous citons par exemple : DSR [JMH03] (Dynamic Source Routing), AODV [PRD03] (Ad Hoc On Demand Distance Vector), TORA [PC97] (Temporally Ordered Routing Algorithm), etc.
3. Les protocoles hybrides : ils forment une combinaison des deux catégories précédentes, comme par exemple le protocole ZRP [HPS02] (Zone Routing Protocol), etc.

Dans les protocoles proactifs, chaque routeur construit un graphe représentant la cartographie complète du réseau. Il se charge de calculer au moyen de l'algorithme de Dijkstra [Dij59], l'arbre des plus courts chemins, qui le relie à chacune des destinations potentielles du réseau relativement à la métrique considérée. Cependant, les routes sont disponibles immédiatement pour la transmission avec les protocoles proactifs, mais le trafic induit par les messages de contrôle fréquents dans le cas d'une forte mobilité est souvent pénalisant. En plus, la taille des tables de routage augmente linéairement avec le nombre de nœuds, et le mécanisme de détection de coupure d'un lien converge lentement. Des études comparatives dans [BMJ$^+$98] montrent que certains protocoles réactifs sont beaucoup plus performants que les protocoles proactifs dans les réseaux ad hoc, bien que la technique de recherche d'une route par inondation puisse être coûteuse, et provoquer des délais importants. Par conséquent, les protocoles réactifs semblent convenir à des réseaux à forte mobilité, tandis que les protocoles proactifs sont mieux adaptés à des réseaux plus statiques et sans contraintes d'énergie.

Contrairement aux protocoles proactifs, qui se contentent de chercher le chemin le plus court vers une destination, les protocoles réactifs diffusent une requête à une destination prédéfinie (Route Request – RREQ) uniquement s'il y a des paquets à envoyer, et ils se basent sur le routage par la source en utilisant le chemin inclus dans le premier paquet de réponse reçu (Route Reply – RREP). Ceci est avantageux pour l'économie de l'énergie et pour la QdS, où le choix du premier chemin peut s'expliquer par le fait que la réponse a traversé le chemin qui a le plus court délai de bout en bout. D'autre part, un routage à travers le plus court chemin pourrait conduire à une mauvaise répartition de la charge. L'utilisation d'autres paramètres pour la recherche (stabilité du signal, durée de vie du lien, bande passante, etc.) peuvent améliorer la qualité et la pertinence du chemin, et conduire à un meilleur équilibrage de la charge. Mais ces techniques ne sont pas suffisantes pour fournir une qualité du service pendant toute la durée de vie d'une session. Plusieurs protocoles de routage qui prétendent fournir une solution partielle ou complète pour satisfaire la QdS de bout en bout sont apparues, parmi lesquels nous citons : OLSR [CJ03] (Optimized Link State Routing Protocol), AODV-QoS [PRD01], MP-DSR [LLP+01] (Multi-Path Dynamic Source Routing), ASAP [XSA03](Adaptive reServation and pre-Allocation Protocol), CEDAR [SSB99] (Core Extraction Distributed Ad hoc Routing algorithm), TBP [CN99] (Ticket Based Probing), etc. La majorité de ces protocoles sont basés sur la recherche d'une route qui satisfait une ou plusieurs contraintes de qualité de service, ainsi que la réservation tout au long de ce chemin. Mais, Chen dans [Che99] a montré que le problème de recherche d'une route avec plus d'un paramètre de QdS est un problème NP-complet, et en supposant que $NP \not\subset P$, les protocoles de routage qui cherchent un chemin selon plusieurs critères de QdS sont loin d'être pratiques.

Les protocoles hybrides essaient de tirer profit des caractéristiques des deux types de protocole précédents, tout en essayant de remédier à leurs inconvénients, tant au niveau de délai d'établissement du chemin dans les protocoles réactifs, qu'au niveau de la surcharge du réseau avec les messages du contrôle et la capacité de stockage nécessaires dans les protocoles proactifs. Les protocoles hybrides utilisent les protocoles proactifs pour le routage à destination de stations proches et les protocoles réactifs pour les longues distances.

Récemment, quelques protocoles ont été retenus (DSR, AODV, OLSR et TBRPF) en vue de normalisation, et ont gagné le statut expérimental de RFC. Dans le reste de ce chapitre, nous allons nous intéresser plutôt aux mécanismes de gestion de la QdS tout en supposant que ces protocoles sont assez matures pour être confrontés assez rapidement à un changement dynamique de la topologie avec un nombre minimum de messages de contrôle.

## 4.3 Architectures de la qualité de service

Deux écoles travaillent à la définition de mécanismes de gestion de la qualité de service dans le réseau IP : l'intégration de service (IntServ) [BCS94] et la différenciation de service (DiffServ) [BBC+98].

La conception de la première architecture de QdS par l'IETF date de la fin des années 1980 ; il s'agit de l'architecture Integrated Services (IntServ) [BCS94]. Ce modèle assure aux différents flux des garanties sur le délai de bout en bout, le débit, etc. Tout flux de données peut bénéficier de garanties individuelles pour que le réseau puisse assurer le niveau de service demandé. Deux types de services sont définis. Guaranteed Service [SPG97] offre des garanties de délai strictes pour des applications temps réel telles que la voix sur IP ou la vidéo. Les routeurs sont alors chargés de borner le délai passé par les paquets de tels flux dans les files d'attente. Controlled Load [Wro97] offre aux applications sensibles à une montée en charge du réseau, un service similaire à celui d'un réseau non chargé, c'est-a-dire qu'une proportion importante de paquets est transmise et un délai faible est garanti. IntServ est utilisé conjointement avec le protocole de réservation RSVP [BZB+97]. L'architecture IntServ est orientée vers le traitement flux par flux. Elle se base sur les mécanismes de contrôle d'admission et de réservation de ressources pour chaque flux afin de garantir une QdS de bout en bout. Dans le réseau IP, cette architecture souffre du problème du passage à l'échelle engendré par le stockage massif des informations d'état de chaque flux sur chaque routeur.

En revanche, l'architecture Differentiated Services (DiffServ) [BBC+98] ne fait pas de réservation de

ressources et se base sur un traitement différencié léger par classe (ou par agrégation des flux) sur chaque routeur, à la place d'un traitement par flux dans IntServ. Donc, on ne parle plus alors de garanties par flux mais de garanties par agrégat de flux. DiffServ définit plusieurs classes de trafic (ou plusieurs niveaux de priorité), où les différents flux doivent s'intégrer à une de ces classes afin de bénéficier des garanties correspondantes. Par exemple, la classe Expedited Forwarding (EF) [JNP99] assure un délai, une gigue et un taux de perte faible ainsi qu'une bande passante minimale garantie. Elle est destinée aux applications temps réel telles que la vidéo ou la voix sur IP. La classe Assured Forwarding (AF) [HBWW99] permet de définir plusieurs taux de pertes, où quatre sous-classes sont définies, dont chacune est séparée en trois niveaux de priorité ($AF_{ij}|i \in [1,4] \wedge j \in [1,3]$). A chaque sous classe une proportion de la bande passante est allouée, et lorsque le débit à émettre dans une classe dépasse la portion allouée, les paquets sont détruits par ordre de priorité.

L'idée de base de DiffServ consiste à reporter le maximum de traitements en bordure de réseau. Chaque paquet d'un flux appartenant à une classe est marqué par le routeur d'entrée au moyen d'un champ particulier de l'entête IP (DSCP) [NBBB98]. Les routeurs traversés par le paquet appliquent alors une politique de gestion de file d'attente (ou Per Hop Behavior – PHB) particulières et dépendantes de la classe de trafic associée au paquet. Une multitude de politiques de gestion de files d'attentes ont été proposées (FIFO, Priority Queueing, Weighted Round Robin [KSC91], Fair Queueing [DKS89], Random Early Dectection [FJ93], etc.). Cette architecture distribuée a été construite pour résoudre le problème de passage à l'échelle associé à IntServ. Mais en réalité elle n'offre aucune garantie en termes de QdS de bout en bout.

La plupart des approches visant à offrir une QdS dans les réseaux ad hoc sont basées sur IntServ et DiffServ. La migration de l'architecture IntServ dans le réseau ad hoc est jugée inadéquate et trop lourde vu les caractéristiques de ce type de réseaux, données en terme de :

- une capacité de stockage limitée.
- une estimation difficile de la bande passante disponible qui varie tout le temps avec la mobilité et les conditions climatiques.
- la contention du message de signalisation de RSVP avec les paquets de données.
- la méthode de réservation en 2 phases (aller-retour) avec le protocole de signalisation RSVP.
- la difficulté de la réservation des ressources dans un environnement distribué et dont l'accès est basé sur la contention.

La méthode de réservation de RSVP, consiste à déterminer les ressources disponibles tout au long d'un chemin dans un premier temps, puis de les réserver dans le sens inverse dans un deuxième temps. Cette méthode est trop lente pour s'adapter à la mobilité des nœuds, qui engendre des coupures de liens fréquents dans le chemin où les ressources ont été réservées, et qui nécessite la mise en place et la configuration d'une nouvelle réservation sur le nouveau chemin. Par conséquent, le contrôle d'admission nécessite l'échange de messages de contrôle avec la mobilité et la coupure des liens. En plus, sachant que la libération des ressources sur l'ancien chemin peut se réaliser après l'expiration d'un temporisateur, ces ressources vont rester non disponibles jusqu'à l'expiration de ce dernier. Ceci laisse beaucoup de points d'interrogation sur les mécanismes de réservation des ressources et leur utilité pour garantir une certaine qualité de service, sachant qu'on ne peut garantir ni la durée de vie, ni l'existence des ressources réservées au cours d'une session.

De l'autre côté, DiffServ est basée sur l'agrégation des flux en un petit nombre prédéfini de classes. Les routeurs d'entrée associent une priorité (DSCP) aux flux, et les routeurs de cœur transmettent les paquets avec un traitement différencié selon cette priorité. Mais, la notion des routeurs de bordure et des cœur, ainsi que la notion de SLA [MN02] n'existe pas dans les réseaux ad hoc. Pour contourner ce problème, chaque hôte joue le rôle d'un routeur de bordure lors de la transmission de ses propres données, et d'un routeur de cœur en relayant les paquets des autres. Mais, la classe Expedited Forwarding (EF) de DiffServ garantit un faible délai et un faible taux de perte dans la file d'attente, si et seulement si le taux de service est supérieur au taux d'arrivée des paquets. En plus, Assured Forwarding (AF) offre un traitement différencié seulement au niveau d'un routeur et n'offre aucune garantie de bout en bout. Par conséquent, le fonctionnement de DiffServ est basé essentiellement sur le provisionnement de ressources,

qui n'est pas possible dans les réseaux ad hoc.

Ni IntServ, ni DiffServ ne représentent la solution parfaite en matière de qualité de service pour les réseaux ad hoc [BN02, CCL03, AS02]. Durant ces dernières années, un très grand nombre de travaux de recherche se sont intéressés à la conception des protocoles et architectures de gestion de QdS pour les réseaux ad hoc. La plupart sont basées sur une modification d'IntServ et de DiffServ, comme : Flexible Quality of Service Model for Mobile Ad Hoc Networks (FQMM) [XSLC00], Two-Layered Quality of Service Model for Reactive Routing Protocols for Mobile Ad Hoc Networks (2LQoS) [NBMR02], IN-band SIGNAling (INSIGNA) [LAZC00], et Service differentiation in stateless Wireless Ad Hoc Networks (SWAN) [ACVS02]. Mais les performances de ces modifications ne sont pas toujours à la hauteur des exigences des applications courantes.

Le modèle FQMM définit une architecture hybride (combinaison d'IntServ et de DiffServ), en intégrant la réservation explicite des ressources pour la classe contenant les applications temps réel, avec la différenciation de service pour les autres classes. Ce modèle souffre du problème de passage à l'échelle d'IntServ, ce qui le rend bien adapté pour des réseaux de petite taille, soit de quelques dizaines de mobiles. Il souffre aussi des problèmes d'inefficacité de DiffServ avec la variation de la bande passante et le mécanisme d'accès distribué dans ce type de réseaux.

Le modèle 2LQoS considère deux types de métriques de qualité de service afin de définir des classes de trafic. Les métriques en rapport avec le bon fonctionnement du réseau, telles que le nombre de sauts des routes, le niveau de batterie des mobiles routeurs, ou encore la stabilité des routes sont utilisées lors de la découverte de chemin. L'utilisation de ce type de métriques permet de ne pas considérer certaines routes lors de l'exploration du réseau afin, par exemple, d'effectuer un équilibrage de charge. La recherche d'une route vers une destination s'effectue de façon réactive et ce processus sélectionne a priori plusieurs routes vers une même destination. Les applications spécifient alors un certain nombre de critères portant sur le délai, la gigue ou encore le débit de la route demandée, permettant de sélectionner la route la plus convenable parmi l'ensemble des routes découvertes. Les différentes classes définies par IntServ ou DiffServ peuvent alors être traduites en combinaison de ces métriques.

Le protocole INSIGNIA définit un ensemble de messages destinés à offrir aux réseaux ad hoc un mécanisme de réservation de bande passante léger et réactif aux variations de ressources. Afin de ne pas surcharger le réseau avec les paquets de contrôle, la plupart des informations nécessaires pour l'établissement des routes avec réservation sont contenues dans les paquets de données, sous la forme d'une option IP. Une application désirant réserver un certain volume de bande passante, spécifie dans l'entête des paquets de données deux niveaux de bande passante, le niveau souhaité dans le meilleur cas et un niveau dégradé ($\max_{BW}$ et $\min_{BW}$). Un algorithme de recherche de route répondant aux critères spécifiés est utilisé, mais tout en supposant l'existence d'un mécanisme d'évaluation de la bande passante disponible pour chaque station. Chaque paquet appartenant à un flux privilégié contiendra cet entête spécifique qui pourra être modifié afin de déterminer une route admissible, et avertir la destination de la disponibilité des ressources, à la manière du mécanisme Explicit Congestion Notification (ECN) d'IP [RFB01]. Ainsi, lorsque le débit d'un flux ne peut plus être assuré, la destination est chargée d'avertir la source afin qu'elle prenne les mesures adéquates. INSIGNIA est indépendante du protocole de routage, et des mécanismes de contrôle d'admission et de réservation de bande passante.

D'autres protocoles de réservation de ressources ont été proposés, comme dRSVP [MST01] qui n'est autre qu'une version adaptée de RSVP aux caractéristiques des réseaux ad hoc, tout en essayant de remédier à quelques problèmes parmi ceux cités auparavant, ou encore plus le protocole de réservation AQOR (Ad hoc QoS On-demand Routing) proposé dans [XG03]. Mais nous ne voulons pas rentrer dans un état de l'art sur les mécanismes et les protocoles de réservation, car nous sommes convaincus de l'inefficacité de ces mécanismes de réservation des ressources pour ce type de réseaux. Dans la suite de ce chapitre, nous allons montrer qu'un bon niveau de QdS sera obtenu, si l'application essaie de s'adapter à la dynamicité du réseau, plutôt que d'essayer de faire des réservations non garanties d'une bande passante qui varie tout le temps. Sachant que ces réservations nécessitent une synchronisation globale entre tous les nœuds du réseau afin d'empêcher de mettre en péril les communications en cours.

Le modèle SWAN propose un nouveau mécanisme situé entre les couches réseau et MAC, permettant

de différencier deux classes de trafic : le trafic temps réel et le trafic best effort (au mieux). Le débit du trafic au mieux varie en fonction des besoins du trafic temps réel, c.-à-d. en occupant une partie pré-spécifiée de la bande passante, et la partie libre de celle qui est consacrée à la classe temps réel. De cette façon, une portion fixe de la bande passante disponible est garantie pour le trafic au mieux afin d'éviter un état de famine. La transmission des flots temps réel passe à travers un mécanisme de contrôle d'admission (CAC), chargé d'envoyer une requête intégrée dans le paquet de routage *Route Request*, généralement diffusé par les protocoles de routage réactifs lors de la recherche d'un chemin. Cette requête a pour but de collecter la bande passante minimale disponible tout au long d'un chemin, et la destination est chargée d'envoyer cette valeur à la source en l'intégrant dans le paquet *Route Reply*. Le CAC base sa décision, d'accepter ou de refuser le flot d'une application comme étant membre de la classe temps réel, sur la requête de QdS reçue de l'application et la réponse reçue du réseau.

L'objectif du modèle SWAN, en plus de la différenciation de service, est de maintenir le délai de transmission le plus bas possible tout en conservant un débit élevé. Conscient du fait que la bande passante varie après l'ouverture d'une session (spécialement avec le changement fréquent de la topologie dûe à la mobilité et à la redistribution de charge). Par ailleurs le mécanisme de contrôle d'admission au niveau de la source, ne prend pas en considération la dégradation de la QdS des flots anciens par l'admission de nouveaux. Les routeurs sont alors chargés d'évaluer le délai d'accès au médium, en observant les instants de mise en file d'attente de chaque trame et les instants de réception des acquittements au niveau MAC correspondants. Lorsque ce délai s'allonge au-delà d'un seuil fixé, les routeurs positionnent le bit de notification explicite de congestion (ECN) [RFB01] dans les trames suivantes afin d'avertir les destinations de l'augmentation de la charge dans une zone du réseau. Les destinations sont alors chargées d'avertir les sources correspondantes afin qu'elles prennent les mesures appropriées. Divers mécanismes sont mis en place afin d'éviter les phénomènes d'oscillations qui peuvent survenir lorsque toutes les sources limitent leur trafic simultanément ou retransmettent les paquets à travers un chemin alternatif par la suite.

Le modèle SWAN et son principe de fonctionnement se résument par la figure 4.9. Ce modèle a prouvé son efficacité par rapport à DiffServ dans la comparaison par simulations sous NS-2 (Network Simulator) réalisée dans [AGRN03].

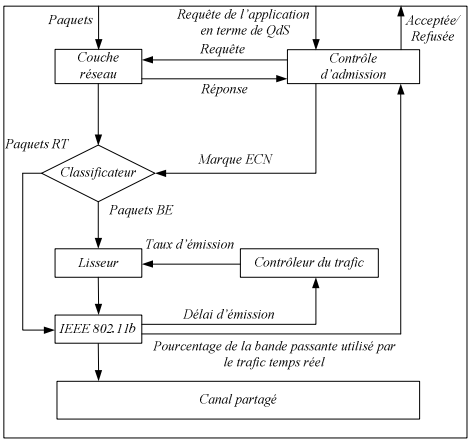



Figure 4.9 – Le modèle SWAN.

Dans un état de l'art de la recherche en matière de QdS pour les réseaux ad hoc, Wu et Harms dans [WH01], introduisent une classification des solutions de qualité de service pour les réseaux ad hoc. Cette classification sépare les solutions proposées dans ce domaine en 4 catégories. Les modèles de QdS regroupent les définitions d'architectures destinées à assurer une certaine qualité de service. Les protocoles de signalisation définissent un ensemble de messages de contrôle, destinés par exemple à provoquer la réservation de ressources dans les routeurs. Les protocoles de routage avec QdS sont chargés de la recherche de routes répondant à certains critères. Enfin, les protocoles d'accès au médium

avec QdS fournissent un ensemble d'outils permettant de mettre en œuvre certaines mécanismes de qualité de service. Ces différentes catégories ne sont évidemment pas disjointes et il est souvent difficile de classer les solutions proposées dans une catégorie particulière. La qualité de service nécessite plutôt un travail de coordination de toutes les couches, et la gestion des interactions entre ces couches par le biais d'une sous couche, comme c'est le cas avec SWAN.

Beaucoup de questions restent soulevées dans SWAN, comme le mécanisme de calcul de la bande passante disponible. En fait, le mécanisme de contrôle d'admission utilisé dans SWAN, compte sur le protocole de routage pour trouver la bande passante disponible dans un chemin, mais sans préciser un mécanisme d'évaluation de cette valeur. En plus, sachant que les besoins des applications temps réel en terme de QdS ne sont pas les mêmes, où contrairement aux applications du VoIP, qui imposent des restrictions strictes en terme de délai et de la variation du délai (gigue), avec une faible bande passante, les applications vidéo nécessitent une large bande passante mais avec une certaine tolérance au niveau de la gigue. SWAN est basé sur 2 classes seulement (temps réel et best effort), et il ne fait pas la différence entre les trafics des applications temps réel. Par conséquent, comme ce modèle sert les paquets des applications temps réel avec la même priorité, il rejette ces paquets avec la même probabilité lors de l'occurrence d'une congestion.

Plusieurs niveaux de priorité sont nécessaires dans la différenciation, spécialement lors de l'occurrence d'une congestion, comme les différents niveaux offerts par DiffServ dans le réseau IP. En plus, SWAN ne précise pas la priorité des paquets de routage, et prétend être distribué et ne pas nécessiter le stockage d'informations liées aux états des flots sur les nœuds intermédiaires. Ce qui semble contradictoire avec le mécanisme de contrôle de congestion utilisé, où les routeurs intermédiaires doivent se rappeler si un flot est nouveau ou ancien avant de l'arrêter pour réduire la congestion. Nous allons essayer d'apporter un élément de réponse à cette contradiction.

DiffServ spécifie plusieurs classes de trafic et tire profit des mécanismes de contrôle de congestion dans TCP (*slow start*, *congestion avoidance*, *fast retransmit* et *fast recovery*), qui réduit la fenêtre de transmission (*Sliding Window*) après la détection d'une perte. L'utilisation de DiffServ garantit une interopérabilité avec le réseau filaire. Pour cela, il nous semble assez intéressant d'utiliser DiffServ plutôt que SWAN pour les réseaux ad hoc.

A la différence du réseau IP, l'accès au medium de transmission est partagé dans les réseaux ad hoc, et la décision de transmission est prise localement dans chaque nœud, sans prendre en compte le niveau des priorités des paquets sur les nœuds voisins, ce qui rend fragile l'idée de base de DiffServ. En plus, l'architecture DiffServ ne définit aucune action de contrôle que ce soit pour prévoir et empêcher la congestion, ni pour s'y adapter le cas échéant. La simulation dans [AGRN03] a montré les difficultés de l'architecture DiffServ de s'adapter au mécanisme d'accès distribué et à la mobilité, ainsi que la supériorité de SWAN par rapport à DiffServ en termes de débit et de délai de bout en bout. Il est évident d'après ces résultats que la technique de différenciation statique est inadéquate. C'est pourquoi un certain dynamisme basé sur un retour de l'état du réseau doit être ajouté pour anticiper et empêcher une surcharge du réseau, et pour régler dynamiquement le trafic selon l'état du réseau.

Les tous premiers travaux que nous avons effectués pour fournir une QdS dans les réseaux ad hoc sont introduits dans l'article [SB04a], qui est dédié à la proposition d'une extension adaptative de l'architecture DiffServ, pour améliorer ses performances et sa flexibilité d'adaptation à la variation de la bande passante dans les réseaux ad hoc. La modélisation algébrique et les vérifications fonctionnelles ont été réalisées dans cet article, mais nous avons préféré présenter dans ce chapitre, la dernière extension de cette proposition, qui est basée sur la différenciation relative plutôt qu'absolue. Une QdS garantie à 100% ne peut pas être assurée avec la mobilité dans le réseau ad hoc. Pour cela, un mécanisme de garantie absolue peut nous sembler exagéré dans un premier temps et une dégradation pour une petite période de temps est inévitable.

Récemment, Dovrolis dans [DSR99, LLY01] a proposé une version relative de DiffServ pour le réseau IP. Dans ce contexte, les flux du réseau sont groupés en $N$ classes, sachant a priori que les paramètres de la QdS (temps d'attente de la file, taux de perte, etc.) reçue par la classe $i$ est toujours supérieure à celle reçue par la classe $i-1$, pour $1 \leqslant i \leqslant N$. Ce modèle est relatif, dans la mesure où on ne peut pas garantir

de valeur spécifique pour un paramètre de QdS dans une classe, mais on peut garantir une valeur proportionnelle à celle d'une autre classe, d'où le nom de cette architecture PDS (Proportional Differentiation Service [DSR99]) donné par Dovrolis. Par conséquent, l'application peut régler la priorité associée à son trafic d'une manière adaptative pour satisfaire ses contraintes de QdS, avec les fluctuations de la charge des réseaux. La proportionnalité dans ce modèle peut se réaliser de manière distribuée, et indépendamment de la valeur exacte de la bande passante disponible, et de la fonction de distribution de probabilité de l'arrivée des paquets.

Ce modèle résout le problème des deux niveaux de priorités identifiés dans SWAN, et semble être suffisamment souple, dynamique et réactif pour s'adapter au changement fréquent de topologie et à la redistribution de charge dans les réseaux ad hoc. Dans un but de fournir une différenciation relative dans le réseau ad hoc, nous allons dans un premier temps donner un petit rappel de cette architecture, ainsi que les différents mécanismes mis en œuvre pour réaliser une proportionnalité entre les paramètres de la QdS, ensuite nous allons identifier les pathologies empêchant la réalisation d'une telle proportionnalité entre les différentes classes.

### 4.3.1 Différenciation de service proportionnelle

Les premiers travaux réalisés sur la différentiation relative étaient consacrés à la différentiation de délais entre les classes (Proportional Delay Differentiation – PDD). D'autres paramètres de QdS ont été étudiés, comme le taux de perte [DR00], la gigue, etc.

Soit $\bar{d_i}(t, t+\tau)$, $\bar{d_j}(t, t+\tau)$ les délais moyen des classes $i$ et $j$ durant l'intervalle du temps $[t,\ t+\tau]$, l'objectif du modèle PDD est de garantir un rapport de proportionnalité fixé entre le délai des paquets d'une classe $i$ par rapport à ceux d'une classe $j$ selon la formule suivante :

$$\frac{\bar{d_i}(t, t+\tau)}{\bar{d_j}(t, t+\tau)} = \frac{\delta_i}{\delta_j}, \quad \forall\, i \neq j \text{ et } i, j \in \{1, 2, \ldots, N\} \tag{4.5}$$

Où $\delta_i$, $\delta_j$ les paramètres de différenciation pour les classes $i$ et $j$. Ces paramètres sont associés aux classes d'une telle manière que la classe avec la plus grande priorité aura le plus petit délai, i.e. $1 = \delta_1 > \delta_2 > \ldots > \delta_N > 0$. Pour simplifier l'équation 4.5, Dovrolis a introduit la notion de délai normalisé comme étant :

$$\hat{d_i} = \frac{\bar{d_i}(t, t+\tau)}{\delta_i} = \frac{\bar{d_j}(t, t+\tau)}{\delta_j} = \hat{d_j} \qquad 1 \leqslant i, j \leqslant N \tag{4.6}$$

L'équation 4.5 montre que le délai aperçu par la classe $i$ est non seulement proportionnel à celui de la classe $i-1$, mais aussi plus grand. Le rapport de proportionnalité aux paramètres de différenciation prédéfini dans l'équation 4.5 doit toujours être satisfait, et ce de manière indépendante de la charge du réseau et de sa variation. Cette proportionnalité a été obtenue dans le réseau IP, par la conception des nouveaux mécanismes de gestion des files d'attente. Une multitude d'ordonnanceurs assurant la proportionnalité entre les classes ont été utilisés, par exemple WTP (Waiting Time Priority), PAD (Proporitional Average Delay) et HPD (Hybrid Proportional Delay) sont présentés dans [DSR02]. La différence entre ces mécanismes de service est liée à la vitesse de convergence quand la charge varie, mais tous ces mécanismes associent une priorité dépendante du temps d'attente de chaque paquet.

Il est désormais possible de configurer le délai d'une classe $N$ de telle manière qu'elle soit 80% de celui de la classe 1. La principale idée de cette architecture se base sur le fait que, même si la QdS de chaque classe varie avec la charge du réseau, la distance entre les paramètres considérés des différentes classes reste constante. Pour fournir une QdS absolue de bout en bout, Dovrolis dans [DR01] a utilisé un mécanisme de priorité dynamique, où l'application peut augmenter/diminuer la priorité associée à ses paquets selon un rapport d'écho envoyé par le récepteur sur la QdS perçue.

Dans le reste de ce manuscrit, nous allons utiliser le gestionnaire WTP pour réaliser une proportionnalité entre les classes dans le réseau ad hoc, mais n'importe quel autre gestionnaire proportionnel pourrait être utilisé à la place.

**L'ordonnanceur Waiting time priority**

Ce gestionnaire a été introduit en 1964 dans [Kle64] sous le nom de « Time dependent priority », où la priorité associée à chaque paquet augmente de manière proportionnelle au temps d'attente dans la file. Si un paquet avec une classe $i$ arrive à l'instant $\tau$, sa priorité augmente avec le temps de la façon suivante :

$$p_i(t) = \frac{1}{\delta_i}(t - \tau) = \frac{T_{attente}}{\delta_i} \tag{4.7}$$

A noter que $(t - \tau)$ n'est autre que le temps d'attente $T_{attente}$ d'un paquet dans la file. Le classificateur de la file d'attente ajoute l'instant d'arrivée $t_{arrivée}$ à l'entête de chaque paquet avant de l'envoyer dans la file qui correspond à sa classe, et le gestionnaire sert le paquet de la classe $i$ qui a la plus haute priorité $p_i(t)$ à l'instant $t$, et selon le mécanisme du premier arrivé premier servi dans une classe.

$$serve\_p(t) = \arg_{i=1,\ldots,N} \max(p_i(t)) \tag{4.8}$$

Si deux paquets ont la même priorité à l'instant $t$, ils seront transmis dans un ordre aléatoire, mais normalement le processus d'arrivée des paquets dans le réseau suit une loi de probabilité continue (Poisson, Pareto, etc.) où la probabilité que 2 paquets arrivent au moment instant est nulle.

L'objectif de ce gestionnaire est d'égaliser la priorité de toutes les classes sur le même nœud, et implicitement d'égaliser le temps d'attente normalisé des paquets sur le même nœud. Pour clarification, les paquets transmis des classes $i$ et $j$ doivent vérifier l'égalité suivante :

$$\frac{T_{attente,i}}{\delta_i} = \frac{T_{attente,j}}{\delta_j} \quad \forall i, j \in \{1, 2, \ldots, N\} \tag{4.9}$$

Sachant que ce mécanisme a prouvé son efficacité dans le réseau IP, nous voudrions l'utiliser dans le réseau ad hoc. Mais, le gestionnaire WTP doit être centralisé, afin de connaître le temps d'attente de tous les paquets pour décider lequel est le plus prioritaire avant de le transmettre. Cet environnement du travail centralisé est garanti dans le réseau IP, où tous les paquets transmis sur une ligne sont envoyés par le même routeur.

Par contre, dans les réseaux ad hoc, le médium de communication est partagé entre tous les nœuds voisins, et l'accès à ce dernier est distribué en utilisant le mécanisme de contention (CSMA/CA). Comme indiqué précédemment, l'accès au médium n'est pas équitable, ce qui rend les délais d'attente entre les classes d'un même nœud différents. Même si la fonction EDCF a été conçue pour réaliser une certaine différentiation qualitative de service entre les classes, où les paquets de haute priorité seront transmis en majorité avant les autres, cette fonction telle qu'elle est définie ne fournit aucun moyen pour régler quantitativement la degré de proportionnalité entre les différentes classes.

Ce mécanisme d'accès empêche le gestionnaire WTP de réaliser la proportionnalité entre les classes d'un même nœud. En fait, la couche réseau envoie le paquet le plus prioritaire à la couche liaison de façon irréversible. Mais comme l'accès au canal consiste à attendre un temps additionnel aléatoire au niveau MAC, avant de transmettre la trame englobant le paquet qui était le plus prioritaire selon WTP, une inversion de priorité peut avoir lieu où la trame en cours de transmission ne correspond plus au paquet le plus prioritaire. Pour plus de clarification, supposons que le paquet $p_n$ est reçu par la couche MAC à l'instant $t_n$, la trame correspondante ne sera pas envoyée immédiatement, mais après un temps aléatoire distribué selon la loi de distribution discrète uniforme si le médium est occupé. A l'instant de transmission $t_{tx}$, le paquet qui se trouve dans la couche MAC peut ne plus avoir la plus grande priorité $p_i(t)$ comme le montre la figure 4.10.

Par conséquent, le modèle PDD ne peut pas atteindre ses objectifs en fournissant une QdS proportionnelle entre les différentes classes dans les réseaux ad hoc. Pour remédier à ce problème et rectifier la proportionnalité entre les différentes classes, nous allons exploiter cette architecture de différenciation relative, et les mécanismes de contrôle de congestion et d'admission utilisés dans SWAN, ainsi que la fonction d'accès EDCF pour proposer une nouvelle architecture de gestion de la QdS dans les réseaux ad hoc.

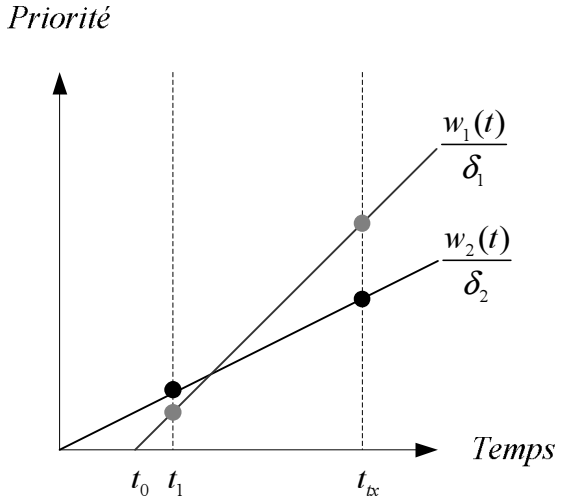



Figure 4.10 – Inversement de priorité.

### 4.3.2 Architecture proposée

La solution proposée pour fournir une QdS de bout en bout est illustrée dans la figure 4.11. Elle est basée sur l'extension de PDS avec les mécanismes de : contrôle d'admission (CAC), contrôle de congestion (CC), estimation de la bande passante, adaptateur dynamique de priorité et une légère modification du standard IEEE 802.11e, à travers un composant de régulation de la valeur initiale de la fenêtre de contention, situé entre les deux couches réseau et MAC. La solution proposée se base sur l'ordonnaceur WTP et l'extension du IEEE 802.11e pour fournir une proportionnalité entre les classes comme nous allons le montrer dans la section suivante. Les autres composants sont utilisés pour empêcher la saturation du réseau et pour fournir une méthode dynamique d'adaptation de la charge avec la variation des conditions du réseau.

Ce modèle [SB06a] fonctionne de la manière suivante : l'application temps réel spécifie le délai de bout en bout maximal et la gigue tolérée au mécanisme d'adaptation de la priorité (DCS). Ce dernier composant se charge de communiquer ces valeurs au mécanisme de contrôle d'admission après l'avoir stocké pour une utilisation ultérieure. Le mécanisme de contrôle d'admission se charge de demander au protocole de routage réactif de chercher une route envers une destination pré-spécifiée, et déclenche un temporisateur pour déterminer le délai de bout en bout après la réception du paquet *RREP* ($D_{e2e} = Temporisateur/2$). Le paquet du protocole de routage *RREQ* est envoyé avec la priorité la plus élevée $N$ pour déterminer la capacité du réseau à fournir ce délai dans le pire cas. En général, tous les paquets de routage et de contrôle dans cette architecture sont envoyés avec la priorité la plus élevée.

Si le délai estimé est plus large que le délai requis par l'application ($D_{e2e} > D_{app}$), alors le CAC refuse la connexion en demandant à l'application soit de réduire sa contrainte de délai ou bien de réessayer plus tard. De cette manière, nous essayons de garantir la QdS des flux existant en évitant leur dégradation, et la transmission inutile de flots temps réel qui engendrerait un gaspillage dans l'utilisation de la bande passante, car même si ces paquets arrivent tard à leur destination, ils seront sans utilité et seront rejetés par le récepteur. Par contre si le délai $D_{e2e} < D_{app}$, le CAC envoie un message de notification au composant d'ajustement de priorité contenant le délai estimé $D_{e2e}$.

Cette valeur sera utilisée pour déterminer directement la priorité correspondante dans cet environnement où les délais des classes sont proportionnels. Mais ce mécanisme de contrôle d'admission est réalisé avant l'ouverture d'une session, et son impact sur les flux existants n'est pas pris en compte dû à la difficulté d'obtenir une estimation exacte avec l'absence d'informations d'états des flux sur les autres nœuds, et suite au surcoût du nombre des messages de contrôle à échanger. Pour cela, un mécanisme d'ajustement de priorité est utilisé pour minimiser l'impact des nouveaux flux et pour fournir une méthode d'adaptation dynamique des flux temps réel préexistants avec la variation de la charge du réseau,

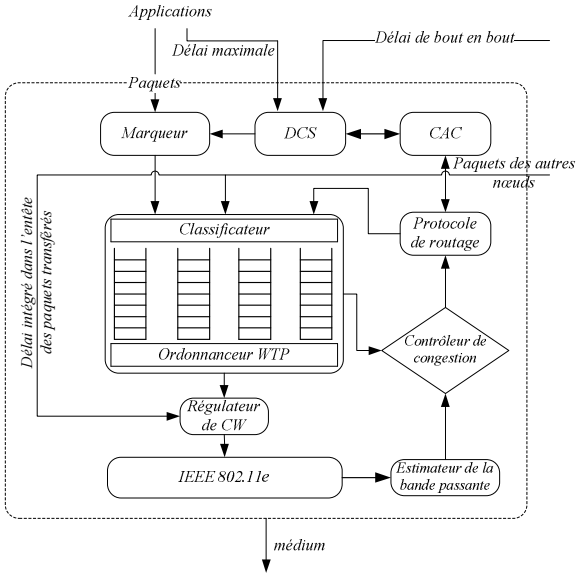



Figure 4.11 – Architecture proposée.

en augmentant la priorité de ces flux selon les contraintes de délai de bout en bout requises et reçues. Cependant, même si les nouveaux flux ont passé le contrôle d'admission, et le mécanisme d'ajustement de priorité augmentant dynamiquement le niveau de priorité, aucun de ces mécanismes ne garantit l'absence d'une surcharge du réseau. Lorsque la charge dépasse la capacité de réseau, une congestion aura inévitablement lieu, et certain flux doit être arrêtés pour éviter que ce problème ne persiste. Pour cela, un mécanisme d'estimation de la bande passante doit être utilisé pour, non seulement estimer la valeur disponible, mais aussi signaler l'occurrence d'une congestion au mécanisme de contrôle de congestion, qui doit alors prendre en charge le contrôle de trafic et arrêter un certain nombre de flots.

**Modélisation algébrique.**

Dans le but de construire un modèle algébrique pour clarifier le fonctionnement du mécanisme proposé, et qui sert comme modèle de base à celui de la simulation, nous proposons une spécification algébrique abstraite du modèle étudié dans la figure 4.11, tout en mettant l'accent sur l'interaction entre les composants. Pour cela, la modélisation a été réalisée à travers la syntaxe de ÆMILA [BCD02] (PADL[1] & EMPA), une extension du formalisme EMPA, qui est dirigé plutôt pour une conception architecturale. Nous avons réalisé la spécification de tous les composants en parallèle comme étant des boîtes noires, avec seulement les interactions entre eux, tout en essayant d'éviter l'explosion de l'espace d'état, en écartant au maximum la spécification des actions internes de chaque processus. Dans un souci de cohérence et de simplification, nous avons préféré donner la spécification des composants sous sa forme algébrique :

$$QoS\_model \stackrel{def}{=} DCS||_D CAC||_C ROUT||_R CC||_N Marker||_M PDS||_P CW\_REG||_Q MAC\_SD$$

$$\begin{aligned} D &= \{notify\_cac, notify\_dcs, send\_prio\} \\ C &= \{request\_rp, wait\_rep\} \\ R &= \{send\_RREQ, wait\_RREP, congestion\_signal, send\_RRER\} \\ N &= \{congestion\_signal, stop\_flow\} \end{aligned}$$

[1] Performance Architecture Description Language

$$
\begin{aligned}
M &= \{tx\_pds\} \\
P &= \{tx\_pkt\_REG\} \\
Q &= \{tx\_to\_MAC, send\_delay\_MACsch\} \\
DCS &\stackrel{def}{=} (app\_req, \lambda)(notify\_cac, \partial)(notify\_dcs, *).DCS_1 + \\
&\quad (receiver\_report, \beta).DCS_1 \\
DCS_1 &\stackrel{def}{=} (send\_prio, \varphi).DCS \\
CAC &\stackrel{def}{=} (notify\_cac, *).(request\_rp, \gamma).(wait\_rep, *).(notify\_dcs, \tau).CAC \\
ROUT &\stackrel{def}{=} (request\_rp, *).R_1 + \\
&\quad (congestion\_signal, *).(send\_RRER, \upsilon).ROUT \\
R_1 &\stackrel{def}{=} (send\_RREQ, \psi).(wait\_RREP, *).ROUT \\
CC &\stackrel{def}{=} (rec\_avBW, PH_{normal}).CC_1 \\
CC_1 &\stackrel{def}{=} (\tau, \infty_{1,0.7}).(calcul, \iota).CC + \\
&\quad (\tau, \infty_{1,0.3}).(congestion\_signal, \iota).(stop\_flow, \infty_{11}).CC \\
Marker &\stackrel{def}{=} (rec\_pkt, \lambda_1).(mark, \omega).(tx\_pds, \vartheta).Marker + \\
&\quad (send\_prio, *).(update\_cache, \iota).Marker \\
PDS &\stackrel{def}{=} (send\_RREQ, *).(chk\_head\_tx\_mac, \varepsilon).(wait\_RREP, PH_{Weibull}).PDS + \\
&\quad (send\_RRER, *).PDS + \\
&\quad (stop\_flow, *).PDS + \\
&\quad (tx\_pds, *).PDS_1 \\
PDS_1 &\stackrel{def}{=} (\tau, \infty_{1,0.9}).(tx\_pkt\_REG, \sigma).PDS + \\
&\quad (\tau, \infty_{1,0.1}).(drop, \infty_{1,1}).PDS \\
CW\_REG &\stackrel{def}{=} (rec\_pkt\_from\_neig, PH_{normal}).REG_1 + \\
&\quad (tx\_pkt\_MAC, *).REG_3 \\
REG_1 &\stackrel{def}{=} (update\_cache, \iota).REG_2 \\
REG_2 &\stackrel{def}{=} (send\_delay\_MACsch, \infty_{1,1}).CW\_REG \\
REG_3 &\stackrel{def}{=} (update\_cache, \iota).((\tau, \infty_{1,0.9}).REG_4 + \\
&\quad (\tau, \infty_{1,0.1}).CW\_REG) \\
REG4 &\stackrel{def}{=} (tx\_to\_MAC, \infty_{11}).CW\_REG \\
MAC\_SD &\stackrel{def}{=} (send\_delay\_MACsch, *).(update\_cache, \iota).MAC\_SD + \\
&\quad (tx\_to\_MAC, *).MAC\_SD_1 \\
MAC\_SD_1 &\stackrel{def}{=} (select\_CW\_\min, \infty_{1,1}). \\
&\quad (csma\_ca\_procedure, PH_{Uniform}).MAC\_SD_2 \\
MAC\_SD_2 &\stackrel{def}{=} (tx\_frame, PH_{BP}).MAC\_SD
\end{aligned}
$$

La figure 4.12 illustre les composants, les interactions et les actions utilisées dans la modélisation algébrique. Après la dérivation des 3 diagrammes de transition et la suppression de tous les états avec

blocage, nous avons réalisé des analyses qualitatives et quantitatives, tout en s'intéressant aux séquences d'exécution des actions (*cf.* equation 4.10), et au débit du système (*cf.* equation 4.13). Dans un souci de cohérence, nous avons préféré mettre l'accent sur les résultats obtenus par le modèle de simulation, construit à partir du modèle algébrique. Reste à noter que la taille du diagramme de transition est de 180582 états et de 325147 transitions, et que la vérification de certaines propriétés fonctionnelles a duré 4 jours sur une machine avec un processeur AMD Athlon XP 2200 et 1 Go de RAM. Pour éviter ce problème, nous avons essayé de re-formuler et optimiser les propriétés de CTL utilisées, ainsi que la puissance de la machine (Intel Pentium 1.73 GHz avec 1 Go de RAM), mais le gain apporté est minime ($\simeq$ de quelques heures). Les résultats de la vérification de certaines propriétés étaient « segmentation fault », sachant qu'en théorie l'outil est supposé être capable de traiter un espace de 1 million d'états, donc 5.54 fois plus grande.

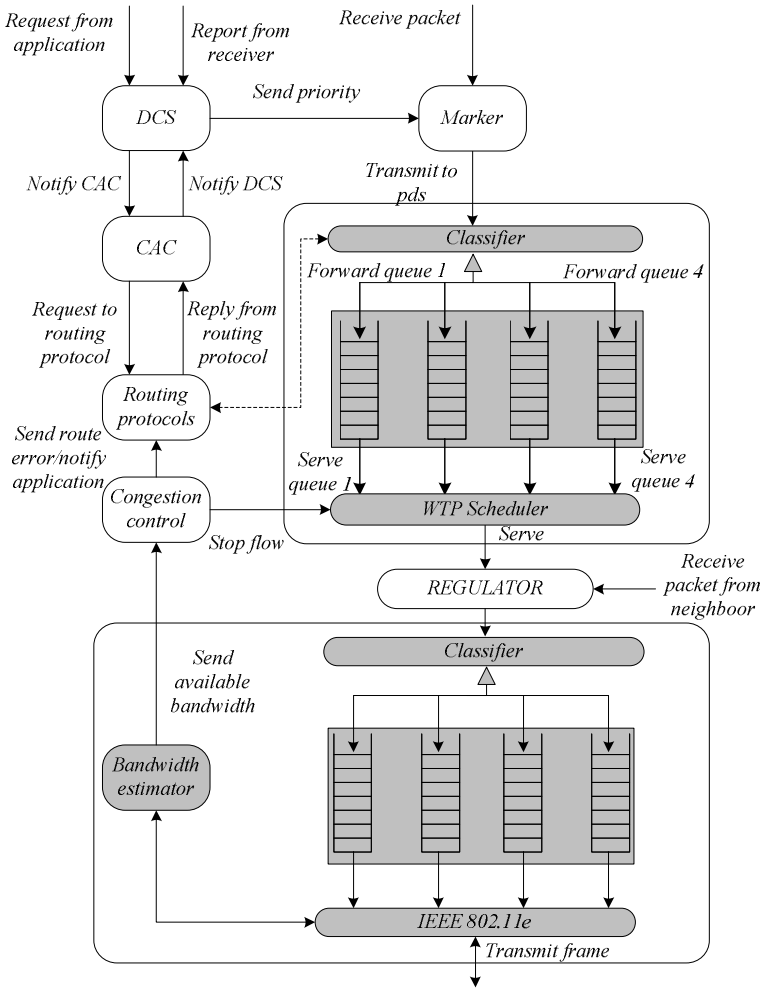



Figure 4.12 – Représentation graphique.

$$
\begin{aligned}
Prop1 &= (minX = [-]ffou\langle-\rangle X) \\
Prop2 &= AG([notify\_cac]\, ff \vee \\
&\qquad \langle notify\_cac\rangle A\,(([app\_req]\, ff \wedge [send\_prio]\, ff)\ \mathrm{W}\ \langle notify\_dcs\rangle tt)) \\
Prop3 &= AG([rec\_pkt] A\,([rec\_pkt]\, ff\ \mathrm{W}\ \langle tx\_pdss\rangle tt) \wedge \\
&\qquad [tx\_frame]\, A\,([tx\_frame]\, ff\ \mathrm{W}\ \langle tx\_to\_MAC\rangle tt))
\end{aligned}
\tag{4.10}
$$

Dans le reste de ce chapitre, nous allons nous concentrer sur les tâches de chaque composant, ainsi

que l'analyse du modèle complet par simulation.

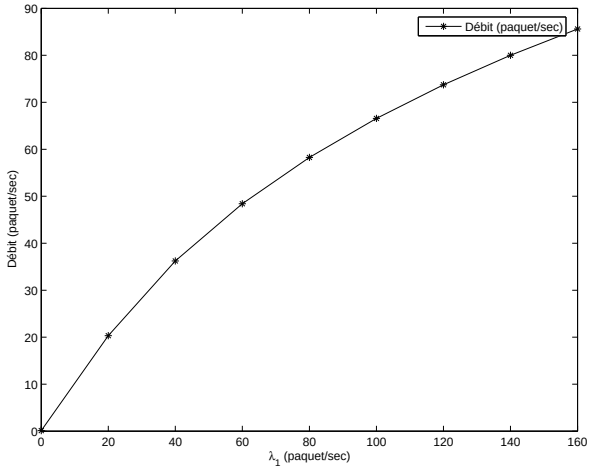



Figure 4.13 – Variation du débit avec $\lambda_1$.

### 4.3.3 IEEE 802.11e et le régulateur dynamique de la fenêtre de contention.

Un des principaux inconvénients de l'EDCF est l'algorithme utilisé pour ajuster la fenêtre de contention qui est peu efficace lorsque le réseau est chargé. Nous proposons une légère modification prenant en compte les besoins de l'application et la charge du réseau afin d'initialiser la fenêtre de contention. Cette modification est introduite dans le but de rétablir la proportionnalité entre les différentes classes dans le réseau ad hoc.

A l'image de la fonction de coordination distribuée (DCF) de la norme IEEE 802.11, les paramètres $AIFS(AC_i)$, $CW_{min}(AC_i)$ et $CW_{max}(AC_i)$ dans EDCF sont définis de manière statique. De ce fait, les fenêtres de contention $CW_{min}(AC_i)$ des différents flux sont initialisées de manière statique et ne prennent pas en compte la dynamicité du réseau. Une taille de la fenêtre trop élevée engendre un surcoût inutile et une sous-utilisation de la bande passante, mais aussi l'utilisation d'une taille trop faible accroît le nombre de collisions dans le réseau. Lorsqu'une collision s'est produite sur une trame, la taille de cette fenêtre est doublée pour DCF et EDCF (selon le Draft 13 de l'IEEE 802.11e [WG05], $CW_{min}(AC_i)$ est multiplié par 2 (le facteur de persistance $PF_i$ est supprimé dans les derniers drafts). Lorsqu'une transmission est réussie, cette valeur est réinitialisée à $CW_{min}(AC_i)$.

Or, transmettre une trame avec succès ne signifie pas que la situation de congestion ayant provoqué l'accroissement de la fenêtre de contention soit résolue. Lorsque le réseau est chargé, on constate alors des oscillations dans la taille des fenêtres qui ne sont pas souhaitables. Ainsi, quand la charge du réseau augmente, en d'autres termes quand le nombre de stations augmente, la différenciation de service avec EDCF devient inefficace comme le montrent les résultats des simulations effectuées par Romdhani et Malli dans [RNT03, MNTB04, NAT04]. Dans ce cas, aucune QdS n'est garantie, et ceci est dû au nombre élevé de collisions et au temps d'attente important, provoqué par le backoff dans chaque cycle de contention, et qui engendre un gaspillage dans l'utilisation des ressources disponibles. Qiang dans [NABT03] a présenté 3 mécanismes de décrémentation linéaire, qui proposent de ne plus réinitialiser statiquement la fenêtre de contention après une transmission réussie, mais de réduire sa taille d'un certain facteur. Romdhani [RNT03] a proposé AEDCF (Adaptive Enhanced Distributed Function), où l'initialisation de la fenêtre de contention prend en compte le taux de collision et le degré de congestion du médium. Cette technique semble conduire à de meilleurs résultats en termes de débit, de délai, de taux de perte et d'utilisation de la capacité du canal, que la fonction EDCF telle qu'elle est définie dans des réseaux saturés.

La différenciation de service dans IEEE 802.11e est qualitative, et elle ne fournit aucune garantie

ni en terme de valeur spécifique du délai d'accès pour une classe, ni en terme de valeur proportionnelle entre les délais des différentes classes. Dans ce contexte, nous allons étudier l'influence des paramètres de contrôle ($CW_{min}[AC_i]$, $CW_{max}[AC_i]$, $AIFS(AC_i)$, etc.) sur les délais d'accès de chaque catégorie $AC_i$, afin de fournir une valeur proportionnelle.

Bianchi [Bia00] a proposé un modèle analytique du mécanisme d'attente dans une station 802.11b sous forme d'une chaîne de Markov à temps discret, afin de déterminer le débit en saturation d'une station DCF. Pour un réseau comportant $n$ stations en concurrence, chacune ayant toujours une trame à émettre, ce modèle suppose que la probabilité de collision est constante pour chaque trame, et elle est indépendante du nombre de retransmissions. L'auteur a modélisé tout d'abord le comportement d'une station isolée afin de déterminer la probabilité qu'un terminal émette une trame dans un intervalle de temps donné, ainsi que la probabilité de collision. Le débit de saturation maximal peut, dans cette situation être approché par $1 \Big/ \left(n.\sqrt{T_C/2}\right)$ où $T_C$ est la durée moyenne d'une collision exprimée en nombre d'unités de temps « Slot Time ». L'efficacité d'un réseau ainsi déterminée est fonction du nombre de mobiles en concurrence et de la probabilité de transmission, et approche les 85% dans le meilleur cas.

Dans les articles [WPL+02] et [VL02], les auteurs ont amélioré ce modèle en prenant en compte le fait que, lorsque de nombreuses collisions surviennent dans un réseau, c'est-à-dire dès que le réseau est saturé, les tailles des fenêtres de contention des différents émetteurs concurrents augmentent. Chatzimisios dans [CVB02] a utilisé le dernier modèle pour dériver le délai. Robinson dans [RR04] a réalisé la même étude réalisée par Bianchi pour la fonction EDCF. Xiao dans [Xia04] a continué l'étude précédente pour analyser l'influence de la taille de la fenêtre de contention sur les performances du mécanisme de différenciation. Dans [SB06b], nous avons utilisé ces modèles pour dériver le délai de transmission d'une catégorie d'accès.

Soit $\tau_i$ la probabilité de transmission d'une classe $i$ par une station dans une unité de temps :

$$\tau_i = \frac{2(1-2p_i)}{(1-2p_i)(CW_{i,min}+1) + p_i CW_{i,min}(1-(2p_i)^R)} \tag{4.11}$$

La probabilité de collision $p_i$ est donnée par :

$$p_i = 1 - (1-\tau_i)^{n_i-1} \prod_{j=0, j\neq i}^{L} (1-\tau_j)^{n_j} \tag{4.12}$$

Où $L$ est le nombre de $AC_i$ par station.

La collision d'une trame transmis par la catégorie d'accès $AC_i$ peut se produire, si au moins une de $AC_i$ de $n-1$ nœuds transmet dans le même unité de temps, ou encore au moins une de $AC_j$ de $n$ nœuds transmet.

Les 2 équations 4.11 et 4.12 sont obtenues à partir de l'analyse de la chaîne de Markov d'une façon similaire à celle de Bianchi dans [Bia00]. $p_i$ et $\tau_i$ sont les solutions d'un système de deux équations à 2 inconnues.

La probabilité de transmission $P_{tr}$ d'une trame dans une unité de temps est donnée par :

$$P_{tr} = 1 - \prod_{j=0}^{L} (1-\tau_j)^{n_j} \tag{4.13}$$

La probabilité $P_S$ d'avoir une seule transmission sur le canal dans cette unité de temps :

$$P_S = \frac{\sum_{j=0}^{L} \left( n_j \tau_j (1-\tau_j)^{n_j-1} . \prod_{k=0, k\neq i}^{L} (1-\tau_k)^{n_k} \right)}{P_{tr}} \tag{4.14}$$

La probabilité $P_{Str,i}$ d'avoir une transmission réussite d'une trame de classe $i$ sur le canal dans cette unité de temps :

$$P_{i,Str} = n_i \tau_i (1-\tau_i)^{n_i-1} . \prod_{k=0, k\neq i}^{L} (1-\tau_k)^{n_k} \tag{4.15}$$

Le débit normalisé est donné par :

$$S = \sum_{i=0}^{L} S_i = \frac{\sum_{i=0}^{L} P_{i,str}.E[P_{Len,i}]}{(1-P_{tr}).T_{i,e} + \sum_{i=0}^{L} P_{i,str}T_{i,S} + (P_{tr} - \sum_{i=0}^{L} P_{i,str})T_{i,c}} \tag{4.16}$$

Où $E[P_{Len,i}]$ est la longueur moyenne des trames transmis par $AC_i$, $S_i$ est le débit normalisé, $p_i$ est la probabilité de collision, et $T_{i,c}$ est le temps d'une collision. Tous ces paramètres sont détaillés dans [Es05] avec :

$$\begin{aligned}
T_{i,S} &= AIFS_i + (T_{PHY} + T_{RTC-MAC}) + SIFS + (T_{PHY} + T_{CTS-MAC}) + SIFS + \\
&\quad (T_{PHY} + T_{MAC} + T_{i,MSDU}) + SIFS + (T_{PHY} + T_{ACK-MAC}) \\
T_{i,c} &= AIFS_i + (T_{PHY} + T_{RTC-MAC}) \\
T_{i,e} &= aSlotTime \\
AIFS_i &= DIFS + AIFSN_i \times aSlotTime \\
E[P_{Len,i}] &= T_{PHY} + T_{MAC} + T_{i,MSDU}
\end{aligned}$$

Le délai $T_{i,D}$ est défini comme étant la différence entre l'instant où la période de backoff d'une classe $i$ commence et l'instant où la trame pourra être transmise sans collision :

$$T_{i,totale} = T_{i,S} + T_{i,D} \tag{4.17}$$

Le débit d'une station de classe $i$ est donné par :

$$s_i = \frac{S_i}{n_i} = \frac{E[P_{Len,i}]}{T_{i,S} + T_{i,D}} \tag{4.18}$$

Le délai d'attente avant une transmission réussie n'est autre que :

$$T_{i,D} = \frac{E[P_{Len,i}]}{s_i} - T_{i,S} \tag{4.19}$$

**Analyse approximative.**

En partant de l'équation 4.12 :

$$(1-p_i)(1-\tau_i) = (1-p_j)(1-\tau_j) = \prod_{i=0}^{L} (1-\tau_j)^{n_i} \qquad 1 \leqslant i,j \leqslant L \tag{4.20}$$

De l'équation 4.20, nous pouvons constater que si $\tau_i \neq \tau_j$, alors $p_i \neq p_j$. Mais, généralement la fenêtre $CW_{i,\min} \gg 1$ et $CW_{j,\min} \gg 1$, et les probabilités $\tau_i \ll 1$ et $\tau_j \ll 1$. Par conséquent, nous dérivons l'approximation $p_i \approx p_j$ à partir de l'équation 4.20. En appliquant les considérations sur l'équation 4.11, nous obtiendrons :

$$\frac{\tau_i}{\tau_j} \approx \frac{CW_{j,min}}{CW_{i,min}} \tag{4.21}$$

Partons des équations 4.12 et **??**, nous aurons :

$$\frac{S_i}{S_j} = \frac{P_{i,str}E[P_{Len,i}]}{P_{j,str}E[P_{Len,j}]} = \frac{n_i\tau_i(1-\tau_j)E[P_{Len,i}]}{n_j\tau_j(1-\tau_i)E[P_{Len,j}]} \approx \frac{n_i\tau_i E[P_{Len,i}]}{n_j\tau_j E[P_{Len,j}]} \approx \frac{n_i CW_{j,min}E[P_{Len,i}]}{n_j CW_{i,min}E[P_{Len,j}]} \tag{4.22}$$

Par conséquent, on aura :

$$\frac{s_i}{s_j} \approx \frac{\frac{E[P_{Len,i}]}{CW_{i,min}}}{\frac{E[P_{Len,j}]}{CW_{j,min}}} \tag{4.23}$$

Donc la proportion de débit de deux classes est essentiellement proportionnelle à leurs fenêtres de contention initiales, ainsi que la taille du paquet. En plus, dans les équations 4.19 et 4.23, on a $\frac{E[P_{Len,i}]}{s_i} \gg T_{i,S}$ et $\frac{E[P_{Len,j}]}{s_j} \gg T_{j,S}$ pour $n_i > 1$ et $n_j > 1$. Le délai entre deux classes adjacentes prend alors la forme suivante :

$$\frac{T_{i,D}}{T_{i+1,D}} \approx \frac{\frac{E[P_{Len,i}]}{s_i}}{\frac{E[P_{Len,i+1}]}{s_{i+1}}} \approx \frac{CW_{i,min}}{CW_{i+1,min}} \tag{4.24}$$

La différenciation entre les classes est essentiellement liée aux valeurs initiales de leur fenêtre de contention. Par conséquent, la proportionnalité entre les classes du même nœud sera rétablie en ajustant dynamiquement la fenêtre de contention initiale de la façon suivante :

$$\frac{CW_{i+1,min}}{CW_{i,min}} = \frac{\delta_{i+1}}{\delta_i} \tag{4.25}$$

Le délai de bout en bout n'est autre que la somme des délais sur les nœuds intermédiaires. Mais le délai sur chaque nœud intermédiaire dépend de sa charge, et avec l'objectif du mécanisme d'accès de fournir un accès équitable, ce délai augmente de façon exponentielle sur les nœuds constituant une artère principale dans le chemin des plusieurs communications. Les délais de bout en bout pour les différentes classes traversant différents nœuds adjacents ne seront pas proportionnels.

Sachant que ce type d'équité n'est pas favorable, nous voulons adapter le mécanisme d'accès pour offrir un délai égal sur tous les nœuds, en partant de nos convictions qu'il sera plus équitable de ne pas pénaliser les nœuds transférant plus des données. Pour cela, nous voulons offrir un délai égal sur tous les nœuds actifs des chemins adjacents comme le montre l'équation 4.26, où $a$ et $b$ sont les identifiants de deux nœuds voisins actifs.

$$\frac{\bar{d}_i^a(t, t+\tau)}{\bar{d}_j^b(t, t+\tau)} = \frac{\delta_i}{\delta_j}, \quad \forall\, i \neq j \quad et \quad \forall\, a \neq b \tag{4.26}$$

Comme la proportionnalité entre les différentes classes sur le même nœud tient grâce à une initialisation proportionnelle de la fenêtre de contention, l'égalité de délai d'une seule classe entre deux nœuds voisins, pousse les autres classes à suivre. Pour gérer l'échange des délais des paquets entre les nœuds, chaque paquet porte dans son entête deux valeurs : le délai du paquet en cours, et l'estimation du délai d'attente sur les nœuds voisins de la station émettrice. Ces deux valeurs seront utilisées par le nœud du prochain saut, pour calculer le délai du paquet à émettre, ainsi que la nouvelle fenêtre de contention selon les formules suivantes :

$$CW^a_{C\max}(t_k) = CW^a_{C\max}(t_{k-1}) \times \left(1 + \eta \frac{\hat{d}^a_R(t_{k-1}) - \hat{d}^a_{C\max}(t_{k-1})}{\hat{d}^a_R(t_{k-1})}\right) \tag{4.27}$$

Avec :

$$\hat{T}^a_{att,i}(t_k) = \frac{t^a_{tx,i} - t^a_{rx,i}}{\delta_i}$$

$$\hat{d}^a_i(t_k) = \alpha \hat{T}^a_{att,i}(t_k) + (1-\alpha)\hat{T}^a_{att,i}(t_{k-1})$$

$$\hat{d}^a_R(t_k) = \delta \hat{d}^a_i(t_k) + \beta \hat{d}^b_R(t_k) + (1-\delta-\beta)\hat{d}^a_R(t_{k-1})$$

Où $\hat{d}^b_i(t_k)$ est le délai normalisé du paquet de la classe $i$ sur les nœuds voisins de $a$, et $\hat{d}^a_R(t_k)$ est la valeur estimée du délai des réseaux sur le nœud $a$, et $\alpha, \beta, \eta \in [0, 1]$. En effet, chaque nœud estime

son délai moyen après la transmission d'un paquet en utilisant la formule de RTT, ainsi que le délai estimé pour accéder au réseau $\hat{d}^a_R(t_k)$ en utilisant les valeurs intégrées dans les entêtes des paquets reçus. Ensuite, chaque nœud règle la fenêtre de contention de la classe maximale selon la formule 4.27, tout en faisant la comparaison entre son propre délai de transmission et le délai moyen de transmission de ses voisins. Si le délai $\hat{d}^a_R(t_{k-1}) > \hat{d}^a_{C_{max}}(t_{k-1})$, alors il augmente sa fenêtre de contention pour égaliser le délai et vice versa. Il est sûr, que la couche liaison n'a pas accès à l'entête des paquets, c'est donc le mécanisme de régulation de priorités, situé entre les deux couches qui se charge de lui communiquer toutes ces informations. L'initialisation des fenêtres de contention des autres classes s'effectuent selon l'équation 4.25.

### 4.3.4 Mécanisme de contrôle d'admission.

Si les flots dépassent la capacité du réseau, il n'est pas possible de n'utiliser que des files, aussi évoluées soient elles pour gérer la congestion. Il est nécessaire à ce moment de limiter le nombre de flots acceptés pour anticiper le problème de surcharge. Par conséquent, un mécanisme de contrôle d'admission avant l'ouverture d'une session est nécessaire. Le rôle de ce mécanisme est de s'assurer de l'existence d'un chemin, capable de satisfaire le délai de bout en bout requis par l'application, mais seulement avant l'ouverture d'une session, et de notifier l'application si le réseau est incapable de satisfaire cette contrainte.

Ce mécanisme de contrôle d'admission à la source a été utilisé dans le modèle SWAN, étant donné que la mobilité et la variation dynamique de la bande passante, rendent l'évaluation de l'impact des nouveaux flots sur les contraintes de QdS des flots existant très dérisoire, et même trop compliquée avec un surcoût des paquets de contrôle à échanger. Pour cela, ce mécanisme n'est pas capable de garantir la satisfaction des exigences durant toute la durée d'une session. Il se charge seulement de déclencher un temporisateur, lors de la recherche d'une route (diffusion du paquet RREQ) et de déterminer le délai de chaque chemin après la réception du RREP comme étant $D_{e2e} = (Temporisateur)/2$. Conscient de la variation dynamique de la bande passante, et l'ignorance de l'impact des flots sur les autres, ce mécanisme ne peut pas empêcher la surcharge du réseau. Pour cela un mécanisme de contrôle de congestion doit être activé pour prendre les mesures nécessaires, en arrêtant un certain nombre de flots.

### 4.3.5 L'ordonnanceur WTP à la couche réseau.

Le fonctionnement de cet ordonnanceur est adopté à la couche réseau sans aucune modification, où le classificateur traite les paquets reçu avant de l'expédier à la file correspondante, et le gestionnaire transmet les paquets des différentes files de manière proportionnelle, en sélectionnant le paquet en tête de file qui a la plus haute priorité comme le montre l'équation (4.8).

### 4.3.6 Ajustement dynamique de la priorité.

Ce mécanisme est chargé de déterminer la priorité minimale pour satisfaire les contraintes des applications en terme de délai. Il commence par régler la priorité $p_i(t)$ à une valeur donnée par :

$$p_i(t) = \arg_{\min_{\delta_i}} \frac{D_{APP}}{D_{e2e}} \times \delta_N \text{ pour } i \in \{1, 2, \ldots, N\} \tag{4.28}$$

Ensuite, il essaie de régler dynamiquement cette priorité en fonction d'un rapport de QdS envoyé par le récepteur, de façon similaire à l'utilisation du protocole RTCP [SCFJ03]. Si la QdS n'est pas satisfaite durant une période $T$, ce mécanisme augmente la priorité tant qu'elle est inférieure à $N$. Dans le cas contraire, il se charge de réduire la priorité à la valeur minimale capable de satisfaire les contraintes de bout en bout. Le fonctionnement de ce mécanisme est donné par l'algorithme 4.1. Nous ne prétendons pas que ce mécanisme est capable de fournir une QdS stricte de bout en bout, il est basé sur la capacité du réseau à fournir ce délai, ce qui n'est pas possible au cas où les ressources disponibles sont insuffisantes,

1. $P_i^0 = p_i(t)$
2. Wait{$K \ \ SlotTime$}
3. **si** {$P_i^{kT} = 1 \wedge D_{e2e} \in SAT_f$ } **alors**
4. $\quad P_i^{k(T+1)} = P_i^{kT}$
5. **sinon si** {$P_i^{kT} > 1 \wedge D_{e2e} \in SAT_f$ } **alors**
6. $\quad P_i^{k(T+1)} = P_i^{kT} - 1$
7. **sinon si** {$P_i^{kT} < N \wedge D_{e2e} \notin SAT_f$} **alors**
8. $\quad P_i^{k(T+1)} \leftarrow P_i^{kT} + 1$
9. **sinon si** {$P_i^{kT} = N \wedge D_{e2e} \notin SAT_f$} **alors**
10. $\quad P_i^{k(T+1)} = P_i^{kT}$
11. **finsi**

**Algorithme 4.1:** Algorithme d'ajustement dynamique de la priorité.

lorsque la charge dépasse la capacité du réseau. Dans ce cas, la seule solution est de réduire la charge à travers un mécanisme de contrôle de congestion.

Dans l'algorithme 4.1, $i$ est l'indicateur d'un flot à la station source, $k$ est l'indice de la période précédente, $P_i^{kT}$ est la priorité de la du flot $i$ à la période $kT$, et $SAT_f$ est l'ensemble de satisfaction des délai de bout en bout d'une application $f$, donné par $SAT = \{D_{e2e} | D_{e2e} \leqslant D_{max,app}\}$.

### 4.3.7 Mécanisme de contrôle de congestion.

Quand un nouveau flot arrive, l'ordonnanceur et le mécanisme d'ajustement dynamique de la priorité sur la station émettrice, essaient d'adapter le nouveau flot avec la variation de la charge du réseau. Mais, si la charge dépasse la capacité du réseau, il n'est pas suffisant d'augmenter la priorité des flots pour gérer la congestion. Dans ce cas, les flots exigeants en terme de délai, voient leurs priorités augmenter jusqu'à $N$. Par conséquent, le taux de collision augmente et le débit va diminuer à zéro. Il est nécessaire à ce moment de limiter le nombre des flots acceptés, en utilisant un mécanisme de contrôle de congestion, chargé de rejeter un ou plusieurs flots, pour régler et plutôt anticiper le problème de congestion du réseau.

En se basant sur ce principe, après que la transmission d'un nouveau flot commence, un mécanisme de détection de congestion est toujours en route pour évaluer l'influence des nouveaux flots sur la bande passante, et déclenche l'alarme en cas de congestion. Le mécanisme de contrôle de congestion prend les mesures nécessaires, en simulant une coupure de lien pour forcer le re-routage de ces flux. Ce mécanisme de contrôle a été utilisé dans le modèle SWAN [ACVS02], et prétend être distribué et sans état des flots (*cf.* section 4.3, page 96), mais avec la nécessité de repérer si un flot est nouveau ou ancien avant de l'arrêter. Pour cela, nous allons essayer dans la suite, d'apporter un élément de réponse à la contradiction dans ce mécanisme, après avoir clarifié la procédure de détection d'une congestion.

La procédure de détection d'une congestion est beaucoup plus difficile dans les réseaux ad hoc que dans le réseau IP, où le dépassement de la capacité de la file signifie l'occurrence d'une congestion. Par contre, dans les réseaux ad hoc, cette métrique semble cependant être faussée. En effet, la taille de la file d'attente n'est plus une indication valide, où la couche MAC retransmet la trame 7 fois ($R = 7$) avant de notifier la congestion aux couches supérieures, c'est-à-dire que la congestion aura lieu avant même que la file d'attente ne commence à se remplir. Pour cela, un mécanisme d'estimation de la bande passante disponible est utilisé comme indicateur de congestion.

Pour estimer la bande passante disponible, nous allons nous baser sur l'état du médium de transmission tout en considérant les deux états possibles du canal : occupé et libre. Soit $T_{total}$ la période de mesure (temps total de mesure), et $T_{tx}$ le temps durant lequel un nœud sent le canal occupé, que ce soit par les transmissions des autres, ou par sa propre transmission. Le canal est considéré comme étant occupé non seulement en cas de transmission, de collision et d'interférence, mais aussi après la réception d'au moins

une des trames RTS ou CTS. Le taux d'utilisation du canal est donné par :

$$U = \frac{T_{tx}}{T_{total}} \tag{4.29}$$

La bande passante disponible est calculée de la façon suivante :

$$B_{disp} = (1-U).C = \left(1 - \frac{T_{tx}}{T_{total}}\right).C = \left(\frac{T_{total} - T_{tx}}{T_{total}}\right).C = \frac{T_{idle}}{T_{total}}.C \tag{4.30}$$

Où $C$ est la capacité du réseau (2Mbps en 802.11, 11Mbps en 802.11b, etc.). D'après l'équation (4.30), la seule variable est $T_{idle}$. Pour cela, déterminer la bande passante disponible revient à déterminer le nombre d'unités de temps ($aSlotTime$) où le médium est libre, entre deux transmissions consécutives. Mais lorsqu'une station gagne le droit d'émettre, elle transmet à la capacité du canal, et à ce moment le nombre d'unité de temps libre durant la transmission ou même parfois entre deux transmissions consécutives est quasi nul. Pour cela, nous avons utilisé une certaine causalité dans l'estimation de $T_{idle}$ en utilisant la formule de RTT :

$$T_{idle}^{n} = \omega T_{idle}^{n} + (1-\omega) T_{idle}^{n-1} \tag{4.31}$$

En effet, en cas de surcharge du réseau, ce nombre tombe à zéro, et la congestion pourra être anticipée par l'estimateur de la bande passante disponible, qui se charge de notifier le mécanisme de contrôle d'admission de cette valeur afin de prendre les mesures nécessaires, pour arrêter les nouveaux flots engendrant la congestion. Notre mécanisme distribué pour la discrimination selon l'ancienneté, consiste à consacrer un octet (256 valeurs) dans l'entête de paquet durant la simulation pour déterminer si un flot est nouveau ou ancien. Ce mécanisme peut se résumer par :

$$\begin{cases} A_i^0 = 0 \\ A_i^{(k+1)T} = A_i^{kT} + 1 & si\ A_i^{kT} < A^{max} \\ A_i^{(k+1)T} = A_i^{kT} & si\ A_i^{kT} = A^{max} \end{cases} \tag{4.32}$$

$A_i^{(k+1)T}$ est l'âge du flot à l'instant $(k+1)T$ et $A^{max} = 255$ est la valeur maximale de cette variable. Quand la valeur de $T_{idle}$ (exprimée en terme d'unité de temps libre) est inférieure à la valeur du seuil $T_{idle,seuil}$, une congestion est notifiée localement au contrôleur de congestion dans tous les nœuds partageant la bande passante. Ce dernier, se charge de déterminer le flot le moins prioritaire (en fonction de son âge). Mais afin d'empêcher tous les nœuds de réagir en réduisant leur transmission, et en engendrant une sous-utilisation de la bande passante, le contrôleur de congestion déclenche un temporisateur dont la valeur d'expiration est un nombre aléatoire choisi selon la fonction de distribution discrète uniforme de l'intervalle $[t_1, t_2]$, avec $t_1 = C_i^{kT} \times \theta$, $t_2 = (A_i^{kT}+1) \times \theta$, où $\theta$ est la longueur de l'intervalle d'attente. Le nœud qui voit son temporisateur expirer en premier, vérifie le nombre d'unités de temps libre pour voir si la congestion persiste. Si oui, il arrête le flot le moins prioritaire en terme d'âge, tout en demandant au protocole de routage de simuler une coupure de lien en envoyant le paquet de signalisation de coupure du lien (Route Error - RERR) vers la source (technique utilisée durant la simulation). Sinon il arrête la procédure de réduction de charge transmise, en considérant qu'un des voisins est le responsable de cette dégradation et qu'il a assumé son entière responsabilité.

## 4.4 Simulations et Résultats.

### 4.4.1 Modèle de simulation.

Afin de valider la modélisation algébrique, le modèle à été implémenté dans un premier temps dans le simulateur QNAP-2 [VP85] sous forme d'un réseau de files d'attente. Ceci est dû à la facilité d'ajouter et d'accéder aux entêtes des paquets. Ensuite, nous avons implémenté le modèle dans le simulateur du réseau NS-2 [Inf05].

Dans un premier temps, nous avons évalué la précision de notre approche pour fournir la proportionnalité entre les 4 classes utilisées dans la topologie de la figure 4.14 à travers QNAP et NS. Ensuite, nous étudions l'impact de différentes lois d'inter-arrivées, et l'impact de la charge du réseau sur les paramètres de différenciations. La zone de couverture d'un nœud est de 250 m, et tous les liens sont supposés bi-directionnelle. Chaque nœud écoute les transmissions de ses voisins. La distance (verticale ou horizontale) entre deux nœuds voisins est de 150 m. La mobilité des nœuds est linéaire dans les 4 directions. Le sens du déplacement est choisi de manière aléatoire avec une vitesse de 2 m/s, sauf pour les nœuds de la première et de la dernière colonne, qui sont supposés fixes durant toute la simulation. Le temps de simulation est de 300 s. Les nœuds $A$, $F$ et $K$ sont les stations émettrices et les nœuds $E$, $J$ et $O$ les destinations. Les paramètres des différentes catégories d'accès du standard 802.11 e, utilisés durant

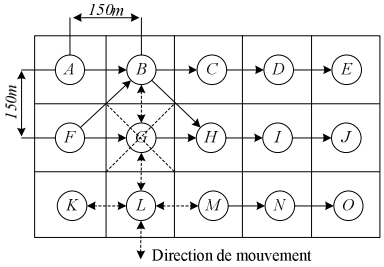



Figure 4.14 – Topologie de la simulation

la simulation sont donnés dans le tableau 4.6. Les valeurs numériques pour les paramètres du modèle utilisés durant la simulation sont données dans le tableau 4.7, et les caractéristiques des différents flots sont données dans le tableau 4.8.

| $AC_i$ | $CW_{min}(AC_i)$ | $CW_{max}(AC_i)$ | $AIFSN_i$ | $AIFS(AC_i)$ | $R$ |
|---|---|---|---|---|---|
| $AC_0$ | 8 | 16 | 2 | $DIFS$ | 7 |
| $AC_1$ | 16 | 32 | 3 | $DIFS + 1 \times aSoltTime$ | 7 |
| $AC_2$ | 32 | 1024 | 4 | $DIFS + 2 \times aSoltTime$ | 7 |
| $AC_3$ | 64 | 1024 | 5 | $DIFS + 3 \times aSoltTime$ | 7 |

Tableau 4.6 – Paramètres des différents catégories d'accès de la fonction EDCF.

### 4.4.2 Résultats des simulations.

Dans un premier temps, nous étudions la degré de proportionnalité entre les 4 flots de $F$ à $J$, qui sont injectés dans le réseau avec les mêmes types de trafic de $A$ et $K$. Les figures 4.15(a) et 4.15(b) montrent la variation de la valeur moyenne sur 3 secondes de délai de bout en bout, ainsi que le rapport entre ces valeurs obtenues via QNAP, et les figures 4.17(a) et 4.17(b) sont les résultats obtenus par le simulateur NS-2. Les résultats de simulation à travers les deux simulateurs sont très semblables mise à part des petites différences liées aux détails des implémentations. Les figures 4.16 et 4.18 montrent la valeur moyenne des délais de différentes classes, ainsi que le rapport de différenciation obtenus via les deux simulateurs. On a constaté que la différenciation est atteinte sous une charge modérée du réseau qui est aux alentours de 21%.

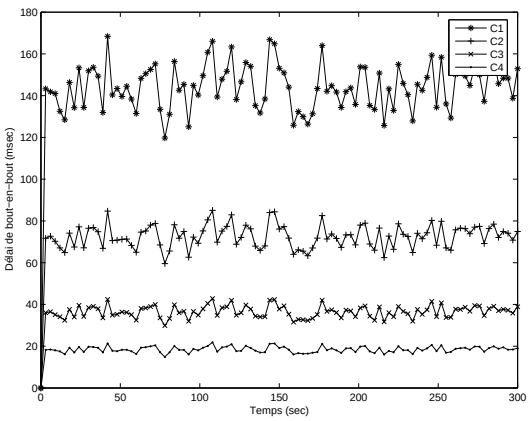
Pour cela, nous avons commencé par nous interroger sur l'influence de la charge du réseau sur les paramètres de différenciation et l'efficacité de notre approche. La charge d'un lien est calculée par l'équa-

| *Paramètres* | *Valeur* | *Paramètres* | *Valeur* |
|---|---|---|---|
| $aSlotTime$ | 9 $\mu$s | $SIFS$ | 16 $\mu$s |
| $AIFS_4 = DIFS = SIFS + 2 \times SlotTime$ | 34 $\mu$s | $AIFS_3$, $AIFS_2$, $AIFS_1$ | 43 $\mu$s, 52 $\mu$s, 61 $\mu$s |
| PLCP Preamble | 16 $\mu$s | PLCP_SIG | 4 $\mu$S |
| tSymbol | 4 $\mu$S | Entête MAC | 28 octets |
| $RTS$ | 20 octets | $CTS$ | 14 octets |
| $ACK$ | 14 octets | 802.11 | 802.11e au dessus de 802.11a |
| Modulation | 16-QAM | Débit | 36 Mbps |
| ACK timeout | 314 $\mu$s | Délai de propagation | 1 $\mu$s |
| Capacité max de la file d'attente à la couche MAC | 50 | Nombre des classes dans la couche réseau | 4 |
| $\delta_i, i \in [1, 4]$ | $\delta_1 = 1/8, \delta_2 = 1/4, \delta_3 = 1/2, \delta_4 = 1$ | $\alpha$, $\delta$ , $\beta$ | 0.9, 0.1, 0.1 |
| Période de l'adaptateur de priorités $T$ | 1 transmission | Capacité de la file d'attente par classe (paquets) | 50 paquets |

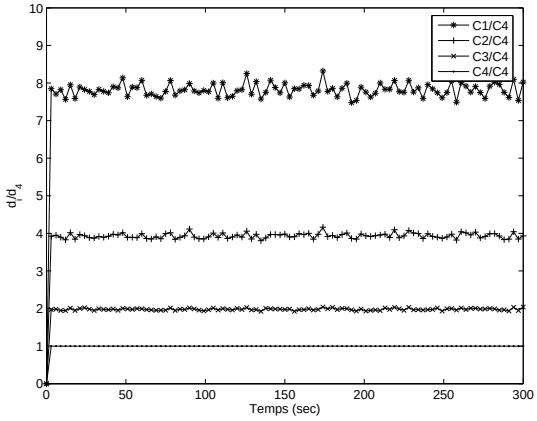
Tableau 4.7 – Paramètres associés au mode de fonctionnement PHY IEEE 802.11a OFDM, MAC, et du modèle.

| Type | Conversa-tionnel | Streaming | Interactive | Best effort |
|---|---|---|---|---|
| | Voix (G.711) | Vidéo (MPEG) | Navigation Web | FTP |
| Taille du paquet ($octet$) | 160 | 1280 | 1100 | 1500 |
| Temps inter-arrivées (ms) | 20 | 10 | 32 | 12.5 |
| Débit (koctet/s) | 8 | 128 | 60 | 120 |

Tableau 4.8 – Les caractéristiques des trafics.

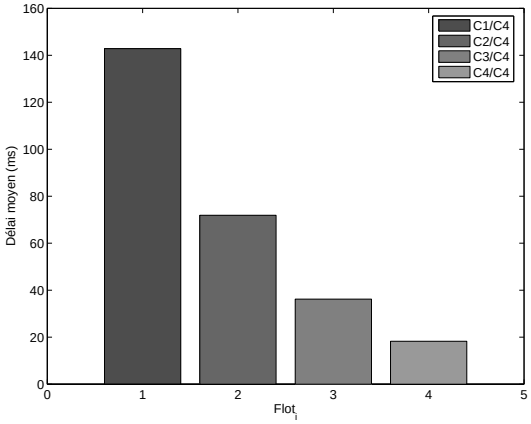



(a) Délai de bout en bout.

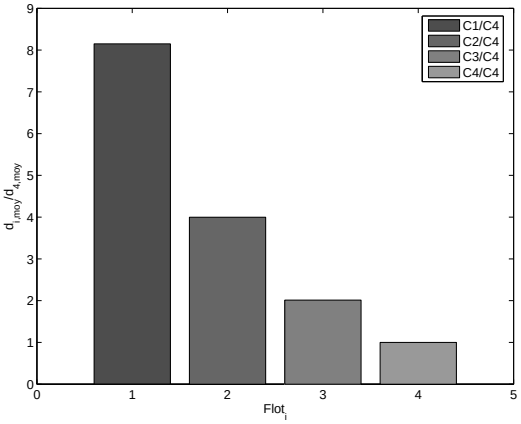



(b) Paramètres de différenciation.

Figure 4.15 – Variation des $d_i$ et des $\delta_i$ avec le temps.

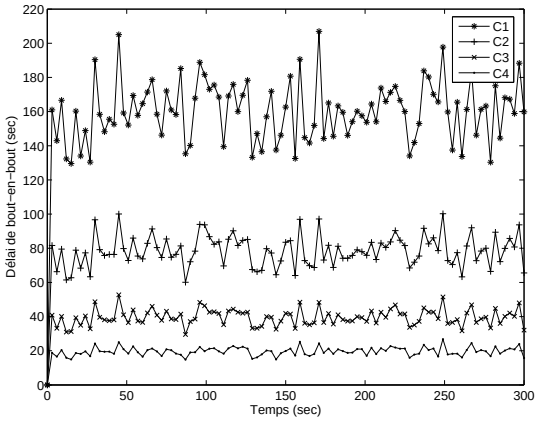



(a) Délai moyen de bout en bout.

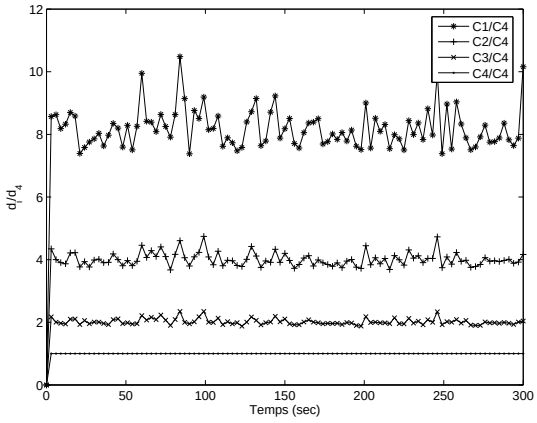



(b) Les valeurs moyennes des $\delta_i$.

Figure 4.16 – Valeurs moyennes des $d_i$ et des $\delta_i$.

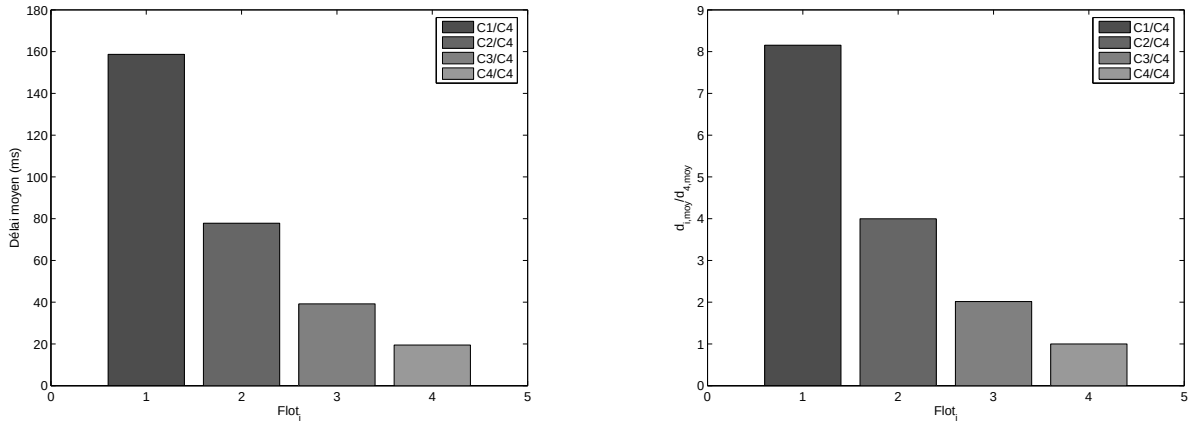



(a) Délai de bout en bout.




(b) Paramètres de différenciation.

Figure 4.17 – Variation des $d_i$ et des $\delta_i$ avec le temps.




(a) Délai moyen de bout en bout.

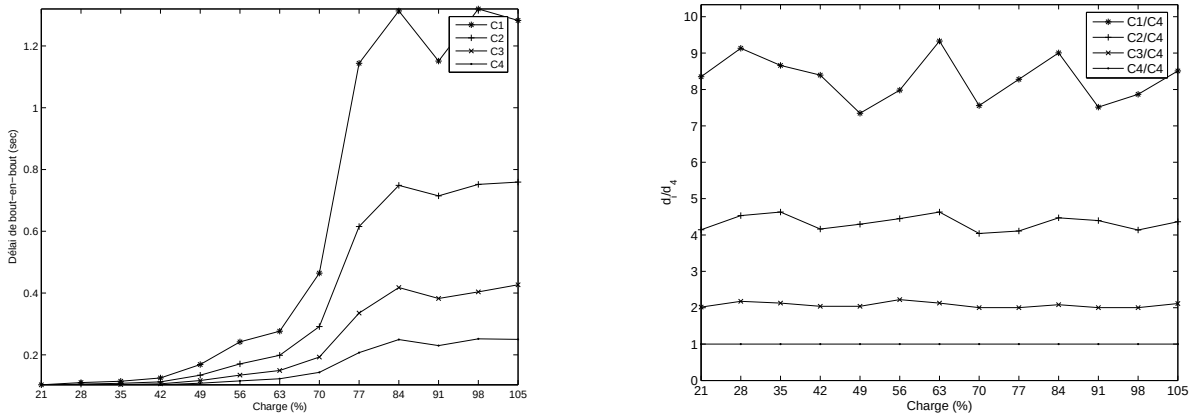
(b) Les valeurs moyennes des $\delta_i$.

Figure 4.18 – Valeurs moyennes des $d_i$ et des $\delta_i$.

tion suivante :

$$Charge = \frac{Débit}{Capacité} \tag{4.33}$$

Nous avons fait varier la charge de 21% à 105% en augmentant le nombre des flux injectés dans le réseau de 12 à 60. Les figures 4.19(a) et 4.19(b) présentent la variation des valeurs moyennes de délai de bout en bout et des paramètres de différenciation avec la variation de la charge.




(a) Délais moyen de bout en bout.

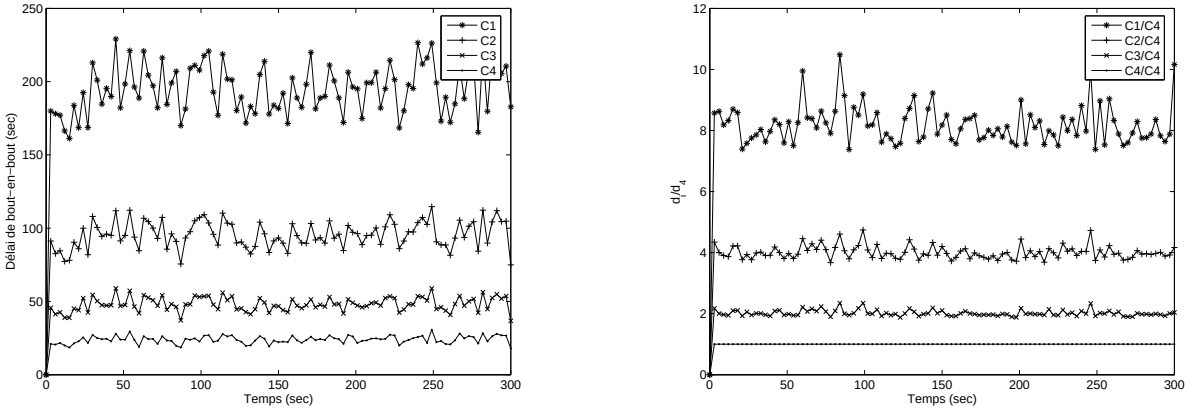
(b) Les valeurs moyennes des $\delta_i$.

Figure 4.19 – Variation des $d_i$ et des $\delta_i$ avec la charge.

Ensuite, nous étudions l'influence de différentes lois d'arrivées de trafic sur les performances du modèle. Dans les tests précédents, nous avons supposé une loi d'arrivée constante (Constant Bit rate – CBR) ou une inter-arrivée déterministe pour les 4 flots. Généralement, l'arrivée des paquets a été largement modélisée dans beaucoup de travaux, par le biais de la distribution de Poisson, où le temps entre deux arrivées successives est distribué selon la fonction exponentielle avec une moyenne égale à $\lambda$ paquet/sec dans l'équation 2.1. Le nombre $n$ des paquets qui arrivent dans un intervalle de longueur $t$ est donné par la distribution de Poisson :

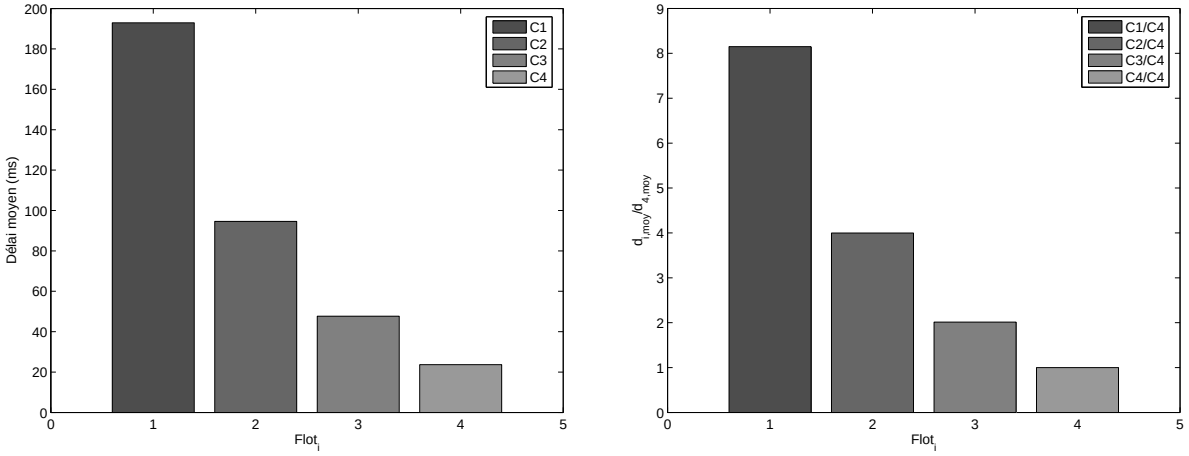
$$Pr(n\ arrivées \in [0,t]) = \frac{(\lambda t)^n}{n!} e^{-\lambda t} \tag{4.34}$$

Les courbes de variation des délais et des paramètres de différenciation des 4 classes sont présentées dans la figure 4.20, et leurs valeurs moyennes dans la figure 4.21.




(a) Délai de bout en bout. (b) Variation des Paramètres de différenciation.

Figure 4.20 – Distribution inter-arrivées exponentielle.




(a) Délai moyen de bout en bout. (b) Les moyennes des $\delta_i$.

Figure 4.21 – Valeurs moyennes.

Mais, des études récentes ont montré que les caractéristiques du trafic du réseau (taille des paquets, temps entre arrivées, etc.), suivent plutôt une distribution à décroissance lente («heavy-tailed») [OOS03]. Ces distributions sont définies mathématiquement par une fonction de puissance, donnée dans l'équation 3.10.

La loi de probabilité utilisée pour modéliser un tel comportement est la distribution de Pareto, puisque sa fonction de distribution cumulative complémentaire ($CCDF$) est proportionnelle à $x^{-\alpha}$ avec $\alpha > 0$.

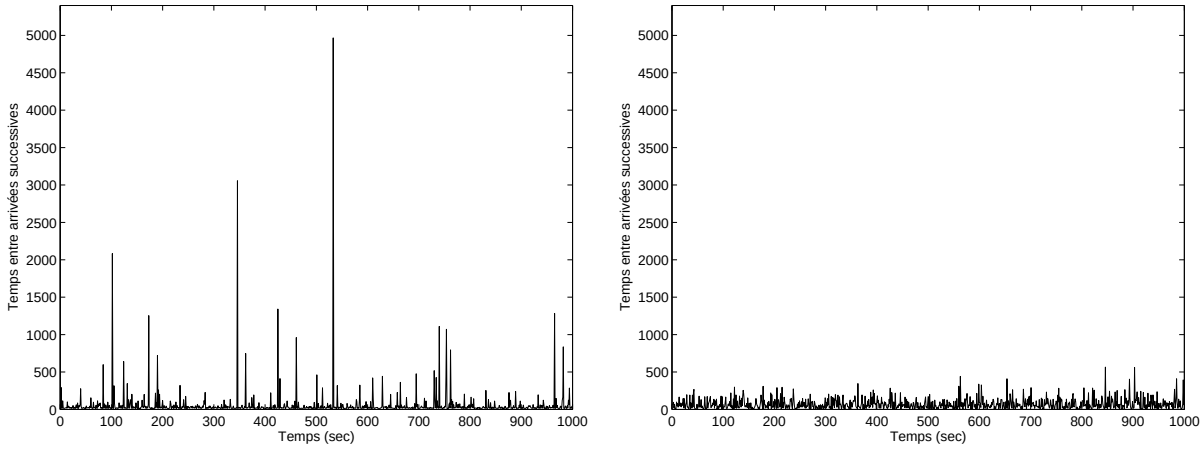
$$\begin{aligned} F(t) &= 0 && pour \;\; t < \kappa \\ F(t) &= 1 - \left(\tfrac{\kappa}{t}\right)^{\alpha} && \forall t \geqslant \kappa, \;\; \alpha > 0 \end{aligned} \tag{4.35}$$

Si le paramètre de puissance $\alpha$ se situe entre $1 < \alpha \leq 2$, la valeur moyenne est finie (*cf.* équation 4.36), mais la variance est infinie. Ce qui signifie une grande variabilité de la variable aléatoire

représentant le temps inter-arrivées.

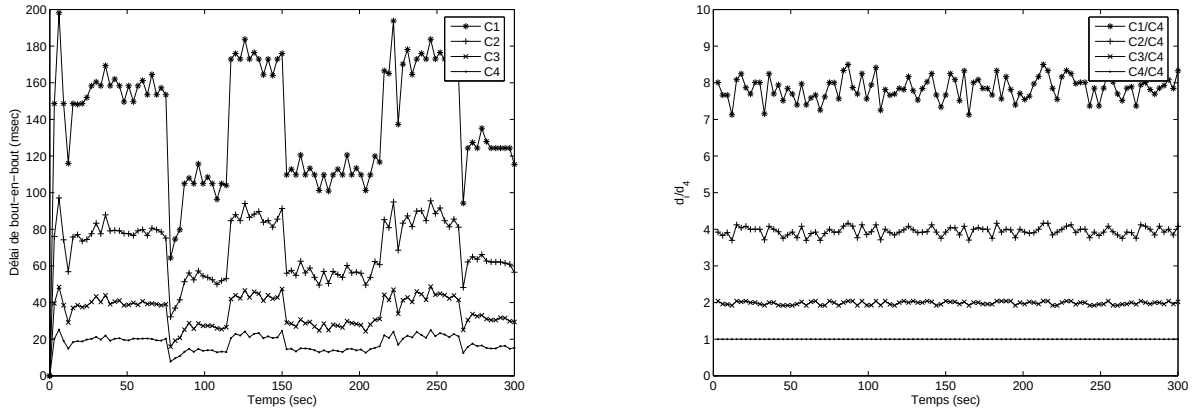
$$E(t) = \frac{\alpha}{\alpha - 1}\kappa \quad pour\ \alpha > 1 \tag{4.36}$$

Juste à titre de comparaison, nous prenons $\alpha = 1.2$ et $\kappa = 12\ \mu s$, $\kappa$ étant la valeur minimale que la variable aléatoire peut prendre, c'est-à-dire la valeur minimale du temps entre 2 arrivées consécutives des paquets du flot best effort. La figure 4.22 montre clairement la différence entre une loi d'inter-arrivée Markovienne avec un taux $\lambda = 13.9$ ms, et une distribution de Pareto, ayant la même moyenne ($E(t) = 72\ \mu s$).




(a) Pareto bornée. (b) Exponentielle.

Figure 4.22 – Variation des échantillons avec le temps.

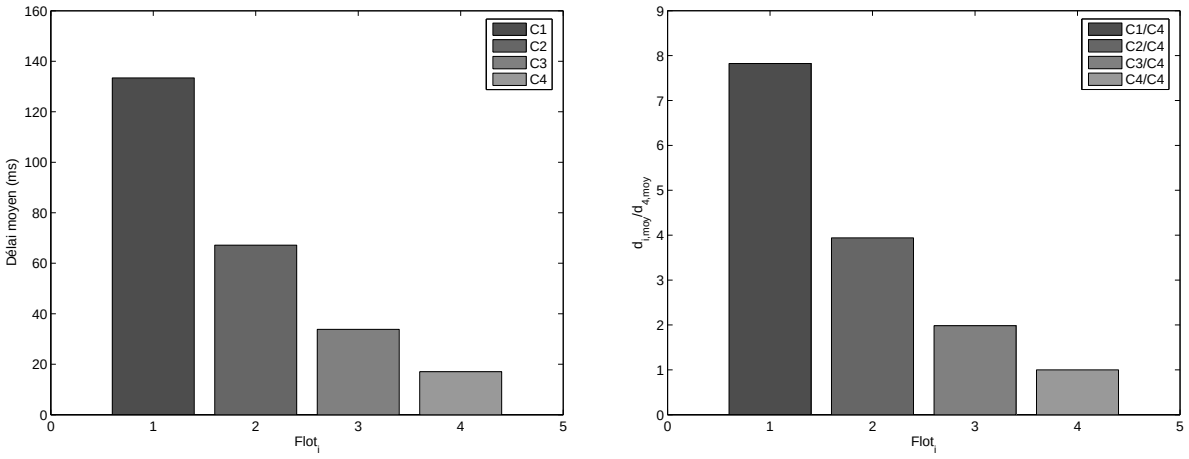



(a) Délai de bout en bout. (b) Variation des Paramètres de différenciation.

Figure 4.23 – Distribution entre arrivées Pareto bornée.

Mais, comme la variable aléatoire représentant le temps entre deux arrivées consécutives peut prendre une valeur infinie, nous avons utilisé une variante de cette fonction (Pareto borné – BP), dont la valeur maximale est limitée à $P = 12\ ms$. La fonction de distribution $BP$, ainsi que la fonction de génération des moments sont données dans l'équation 4.37.

$$\begin{aligned} F(t) &= \frac{1}{1 - (\kappa/P)^{\alpha}} \left[1 - (\kappa/t)^{\alpha}\right], \quad \kappa \leqslant t \leqslant P,\ 0 < \alpha \leqslant 2 \\ \mu_k &= \frac{\alpha}{(k - \alpha)(P^{\alpha} - k^{\alpha})} \left(P^n k^{\alpha} - k^n P^{\alpha}\right) \end{aligned} \tag{4.37}$$




(a) Délai moyen de bout en bout.

(b) Les moyennes des $\delta_i$.

Figure 4.24 – Valeurs moyennes.

Les figures 4.23 et 4.24 présentent la variation des valeurs moyennes de délai de bout en bout et des paramètres de différenciation avec un temps entre arrivées distribué selon la loi de distribuion BP. On peut voir que même avec différentes distributions de trafic, et même avec différents délais pour les classes, la proportionnalité entre les différentes classes est peu perturbée.

## 4.5 Conclusion.

La principale contribution de ce chapitre est l'extension du modèle PDS (Proportional Differentiated Service) pour les réseaux ad hoc, à travers la conception d'un nouveau modèle de gestion de la QdS, basé sur le contrôle dynamique de plusieurs paramètres. Tous les mécanismes présentés dans ce chapitre adoptent un fonctionnement distribué et sans états concernant les flux sur les nœuds intermédiaires. L'efficacité du modèle proposé a été vérifiée, et analysée ensuite par simulation à travers QNAP et NS-2, et les résultats obtenus ont montré l'efficacité et la capacité d'adaptation dynamique de ce modèle face à la variation des conditions du réseau.

Un problème n'a cependant pas été abordé dans ce chapitre. En effet, certaines applications imposent des restrictions strictes au niveau des paramètres de la QdS, et un mécanisme proportionnel ne peut pas répondre à ces besoins. En plus, mettre en œuvre EF avec une garantie proportionnelle plutôt qu'une garantie absolue, n'est pas conforme à la définition de la classe EF. Pour cela, nous allons essayer dans le chapitre suivant, d'améliorer le modèle proposé dans la figure 4.11, afin de fournir une QdS absolue de bout en bout dans les réseaux Ad Hoc.

# Chapitre 5

# Différenciation de service absolue de bout en bout dans les réseaux ad hoc

Ce chapitre traite le problème de gestion d'une QdS absolue plutôt que relative dans les réseaux ad hoc. La fonction *EDCF* du standard *IEEE 802.11e*, garantit une différenciation qualitative intra et inter-nœud, mais ne répond pas aux contraintes des applications en termes de paramètres de QdS de bout en bout. Nous présentons une extension de l'approche précédente pour améliorer le support des applications temps réel interactives dans les réseaux ad hoc, tout en se basant sur un contrôle dynamique des paramètres de différenciation dans *IEEE 802.11e*, selon la charge du réseau et les besoins des applications, pour essayer de satisfaire les contraintes de délai maximal et de débit minimal sur chaque nœud. Pour éviter que la charge dépasse la capacité du réseau, un certain nombre de mécanismes de mesure et de contrôle distribués sont mis en œuvre. Nous présentons des résultats des simulations mettant en évidence la différenciation de service selon les besoins des applications.

## 5.1 Les fondements de cette extension

Dans les réseaux ad hoc, beaucoup de travaux se sont intéressés à la QdS, partant de la conception des nouveaux mécanismes d'accès au médium jusqu'à la couche application. Néanmoins, rares sont les travaux qui ont traité de la QdS de bout en bout, et de l'interopérabilité avec le réseau filaire.

Nous proposons donc de baser l'architecture de QoS pour les réseaux ad hoc sur le modèle DiffServ, dans le but d'étendre le modèle Diffserv à un environnement sans-fil au niveau IP, et ceci pour fournir une QdS cohérente lors de la communication avec un réseau filaire.

Le standard *802.11e* [WG05] permet de garantir une différenciation entre les catégories d'accès au sein de la même station et entre les stations voisines, mais il ne fournit aucune garantie de bout en bout en terme des paramètres de la QdS de chaque flot, et ses performances se réduisent à celui de son prédécesseur lorsque le réseau est surchargé. Dans le chapitre précédent, nous avons prouvé que :

$$\frac{d_{i+1}}{d_i} = \frac{CW_{i+1,min}}{CW_{i,min}} \tag{5.1}$$

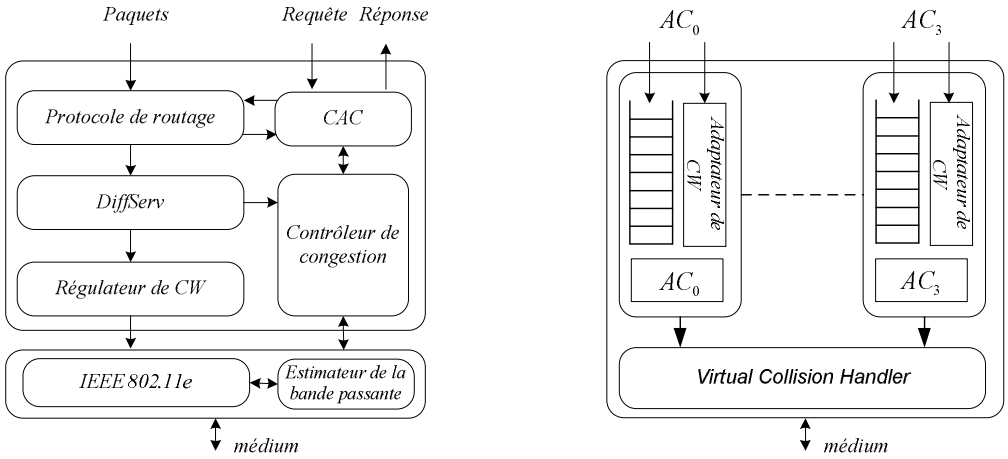
Où $d_i$ et $CW_i$ représentent respectivement le délai et la fenêtre de contention minimale de la classe $AC_i$. Par conséquent, un des principaux inconvénients de la fonction *EDCF* est l'algorithme utilisé pour initialiser la fenêtre de contention de manière statique. Cet algorithme est peu efficace lorsque le réseau est chargé. Nous proposons une légère modification de ce mécanisme pour prendre en compte les besoins de différentes classes, ainsi que la charge du réseau avant d'initialiser la fenêtre de contention, et afin de fournir une qualité de bout en bout plutôt absolue que relative.

Étant donnée que chaque classe de services est caractérisée par ses paramètres de QdS, nous allons nous baser sur ces paramètres, et le modèle de QdS pour le 802.11b proposé dans [YK04], pour assurer une QdS cohérente de bout en bout. Généralement DiffServ [BBC$^+$98] classifie les différents flux en 3 classes : la classe EF [JNP99] regroupe les flux exigeants en terme de délais, la clasee AF [HBWW99]

pour les flux sensibles à la variation de la bande passante, et la classe BE (Best Effort) qui peut s'adapter aux variations de la bande passante et du délai. Dû à ces différentes contraintes, on va associer à chaque classe son propre mécanisme d'ajustement de la fenêtre de contention.

## 5.2 Architecture de la QdS

L'architecture proposée est présentée dans la figure 5.1. À la couche réseau, l'architecture *DiffServ* (*cf.* chapitre 2, figure 2.7) est utilisée pour marquer les flots des applications, ainsi que pour assurer une interopérabilité avec le réseau IP. Beaucoup de travaux sur le couplage de nombreuses priorités de *DiffServ* avec les 4 classes du standard *IEEE 802.11e* ont été réalisés. Nous avons utilisé l'étude réalisée dans [NAS04] pour le couplage statique : $EF \rightarrow AC_0$, $(AF_4, AF_3) \rightarrow (AC_1, AC_2)$, $(AF_2, AF_1, BE) \rightarrow AC_3$. Dans la suite, nous décrivons les fonctions de chaque composant de notre modèle.




(a) Modèle proposé.

(b) IEEE 802.11e avec un adaptateur.

Figure 5.1 – Architecture proposée.

### 5.2.1 Service Différencié et couplage avec EDCF

Généralement, *DiffServ* définit trois classes de services :

**La classe EF (Expedited Forwarding) :** cette classe est destinée aux applications exigeantes en termes de délai de bout en bout. Elle garantit un délai faible dans la file d'attente, et une gigue réduite, ainsi qu'un taux de perte minimal. Les paquets de la classe *EF* ont une priorité supérieure à ceux des autres classes et sont couplés à la classe $AC_0$ au niveau MAC.

La fonction *EDCF* permet d'assurer une différenciation de service qualitative entre les 4 classes prédéfinies, en modifiant les paramètres de temps d'attente de chaque classe ($AIFS(AC_i)$, $CW_{i,min}$, etc.), c'est-à-dire la classe qui aura le plus petit temps d'attente sera prioritaire par rapport aux autres. D'une façon générale, la classe qui aura le plus petit délai entre les trames ($AIFS(AC_i)$) et la plus petite fenêtre de contention $CW_{i,min}$, tirera la plus petite valeur de backoff (donnée dans l'équation 4.3), et augmentera sa chance de gagner le droit de transmission par rapport aux autres. Ce qui aura une conséquence directe sur l'augmentation de la bande passante allouée et la réduction des délais de ses paquets.

La probabilité d'attente est minimisée dans la classe $AC_0$, si et seulement si, le taux de service est supérieur au taux d'arrivée des paquets. L'évaluation de ce protocole par simulation indique que la différenciation s'opère bien lorsque le réseau est non chargé. Par contre, dans un environnement surchargé, 70% des paquets reçus sont détruits par le récepteur, suite à leur arrivée tardive, où beaucoup d'études ont trouvé qu'il est plutôt judicieux de le détruire dans le réseau pour réduire la charge et éviter leur surcoût de transfert en terme de consommation d'énergie.

Étant donné que le délai de propagation sur le lien $l_{i,j}$ est suffisamment petit (de l'ordre de 1 $\mu s$) face au délai de traitement de chaque saut, limiter le délai de bout en bout, revient à réduire le délai de transmission sur chaque saut.

$$d_{e2e} = \sum_{i=1}^{m-1} d_i \tag{5.2}$$

Dans un souci de limiter le délai sur chaque saut à une valeur $d_{max}$ qui est fonction de la charge du nœud ($d_{max} = f(charge)$), nous pensons qu'il est beaucoup plus équitable de ne pas pénaliser les flots traversant plus de sauts, et de faire plutôt un traitement par flot dans cette classe, en intégrant la valeur de délai de bout en bout dans l'entête de chaque paquet. De cette façon, le délai maximal de chaque saut sera $d_{max} = d_{e2e}/m$, où $m$ est la longueur du chemin inclus dans l'entête de chaque paquet, par les protocoles de routage réactif qui sont généralement basés sur le routage par la source. Le délai de traitement par saut, s'étend généralement de la couche réseau à la couche liaison. Chaque nœud doit maintenir le délai de ce type de flots à $d_{e2e}/m$, pour cela nous utilisons le gestionnaire EDF (Earliest Deadline First) dans l'architecture *DiffServ* pour la classe *EF*. Ce gestionnaire est largement utilisé dans ce but, et permet de transmettre les paquets avec le plus petit deadline en premier, et de supprimer ceux qui ont dépassé leur échéance.

Mais, comme le délai maximal par saut $d_{e2e}/m = d_{réseau} + d_{MAC}$, la couche MAC doit exploiter le médium pour garder la somme inférieure ou égale à $d_{max}$. La fonction *EDCF* telle qu'elle est proposée aujourd'hui, ne permet pas de garantir ni une valeur spécifique de délai, ni une borne sur le délai.

Mais, d'après l'équation 5.1, changer la valeur initiale de la fenêtre de contention, revient à changer la priorité et en fait changer le délai de transmission, comme le montre le résultat dans (5.3) :

$$\begin{gathered}
\frac{d_{i+1}}{d_i} = \frac{CW_{i+1,min}}{CW_{i,min}} \Rightarrow \frac{d_{i+1}}{d_i} - 1 = \frac{CW_{i+1,min}}{CW_{i,min}} - 1 \Rightarrow \\
\frac{d_{i+1}}{d_i} - \frac{d_i}{d_i} = \frac{CW_{i+1,min}}{CW_{i,min}} - \frac{CW_{i,min}}{CW_{i,min}} \Rightarrow \frac{d_{i+1} - d_i}{d_i} = \frac{CW_{i+1,min} - CW_{i,min}}{CW_{i,min}} \Rightarrow \\
CW_{i+1,min}^{p+1} = CW_{i,min}^{p}(1 + \frac{d_{i+1}^{p+1} - d_i^p}{d_i^p})
\end{gathered} \tag{5.3}$$

Donc limiter le délai de la couche MAC à une certaine valeur $d_{MAC}$ lors de la transmission de la trame $p+1$, revient à initialiser la fenêtre de contention selon l'équation suivante :

$$CW_{AC_0,min}^{p+1} = CW_{AC_0,min}^{p}(1 + \frac{d_{MAC}^{p+1} - d^p}{d^p}) \tag{5.4}$$

Cette méthode d'initialisation réduit la fenêtre de contention si le délai $d_{MAC}^{p+1}$ attendu pour la trame sous jacente du paquet $p+1$, est plus petit que celui de la trame précédente, et vice versa. Tous les paramètres dans (5.4) sont connus sauf le délai $d_{MAC}^{p+1}$.

Dans la mesure où l'objectif de différenciation au moment de la transmission est réalisé par la fonction *EDCF*, nous pouvons supposer l'absence de toute file d'attente au niveau réseau, afin de négliger le délai de traitement dans cette couche, et on aura $d_{MAC}^{p+1} = d_{e2e}/m$. A ce moment, la fonctionnalité de *DiffServ* dans la couche réseau serait réduite tout simplement au marquage du trafic d'entrée. Mais nous avons préféré adopter l'autre approche, en supposant l'existence de mécanismes de différenciation dans la couche réseau, dans un souci de compatibilité avec la version précédente du standard *802.11b*. Pour cela, nous avons utilisé le mécanisme du chien de garde, qui exploite l'instant d'arrivée du paquet, ajouté par le gestionnaire EDF à l'entête du paquet, pour calculer le délai passé $d_{réseau}$ ainsi que le délai restant $d_{MAC}$. En supposant que la couche MAC est incapable d'accéder à cette valeur intégrée dans l'entête du paquet, ce mécanisme se charge de communiquer séparément cette information à l'algorithme d'initialisation de la fenêtre de contention pour prendre les mesures nécessaires.

**La classe AF (Assured Forwarding) :** cette classe est destinée aux applications exigeantes vis-à-vis de la variation de la bande passante (ou de la variation de débit). L'absence de cette variation nécessite un

taux d'arrivée égal à celui de départ. Selon la théorie des files d'attente, pour une file d'attente de capacité finie, maintenir une longueur constante garantira l'absence de cette variation. Pour cela, la fenêtre de contention de ce type de trafic doit être mise à jour selon la fonction suivante :

$$CW^{p+1}_{AC_i,min}(t_k) = CW^{p}_{AC_i,min}(t_{k-1}) + (L_{seuil,i} - L_i(t_{k-1})) \tag{5.5}$$

Où $L_{seuil,i}$ est la valeur maximale de la longueur de la file d'attente (comme celle utilisée dans l'ordonnanceur RED [FJ93]). $L_i(t_{k-1})$ est la longueur de la file (en nombre de messages) durant la transmission précédente. Cette formule montre clairement que lorsque la longueur dépasse la valeur du seuil, la fenêtre de contention diminue pour augmenter le débit, et augmente dans l'autre cas pour libérer les ressources aux autres hôtes et autres classes.

**La classe BE (best effort) :** cette classe est destinée aux flots tolérants aux variations de délai et de bande passante. Ces flots seront stockés dans la même file d'attente, et une seule fenêtre de contention est suffisante pour ces flots. L'ajustement de cette fenêtre est nécessaire pour empêcher cette classe de surcharger le réseau et dégrader la qualité des autres classes :

$$\begin{aligned} &\frac{(CW^{p+1}_{AC_3,min} - CW^{p}_{AC_3,min})}{CW^{p}_{AC_3,min}} = \frac{I_{seuil} - I^{(p)}}{I^{(p)}} \Rightarrow \\ &CW^{p+1}_{AC_3,min}(t_k) = CW^{p}_{AC_3,min}(t_{k-1}) \times (1 + \frac{I_{seuil} - I^{(p)}}{I^{(p)}}) \end{aligned} \tag{5.6}$$

Où $I_{seuil}$ est la valeur seuil de l'indicateur de congestion, mesurée en fonction du nombre d'unités de temps ($aSlotTime = 9\mu s$) libres entre deux transmissions consécutives sur le médium. $I^{(p)}$ est le nombre moyen d'unités de temps disponibles après la transmission de la trame $p$, dont la valeur moyenne est calculée selon la formule de $RTT$. Par conséquent, si le flot BE sent que le medium est loin du seuil de congestion ($I^{(p)} > I_{seuil}$), il tire profit en réduisant sa fenêtre de contention, et dans le cas contraire, il se contente d'augmenter sa fenêtre de contention en n'occupant que la partie qui reste pour éviter un état de famine. Reste à noter que seul le mécanisme de l'initialisation de la fenêtre de contention est modifié, mais tous les autres mécanismes de *IEEE 802.11e*, comme le ralentissement binaire qui consiste à doubler la fenêtre de contention en cas de collision, sont retenus sans aucune modification durant la simulation.

Le dynamisme dans l'initialisation de la fenêtre de contention dans notre algorithme garantit l'adaptation des flots temps réel à la variation des ressources, s'il en reste assez. Par contre, si les ressources existantes ne sont pas suffisantes, c'est-à-dire lorsque $I^{(p)}$ est trop petit ($I^{(p)} < I_{seuil}$), les flots temps réel ne peuvent plus satisfaire leurs contraintes en *QdS* avec ces mécanismes d'adaptation, et commencent à voir leurs fenêtres de contention et leurs débits tendre vers zéro, à cause du taux élevé de collisions. Pour cela, un mécanisme de contrôle d'admission est nécessaire pour limiter les flots acceptés et empêcher une surcharge du réseau.

### 5.2.2 Mécanisme de contrôle de congestion

Le mécanisme de contrôle d'admission se base sur la bande passante disponible pour accepter ou refuser de nouveaux flots. Dans les réseaux ad hoc, il est difficile d'avoir une estimation précise de la bande passante disponible, qui est partagée entre les nœuds voisins, et d'évaluer l'impact des nouveaux flots sur la *QdS* de ceux qui existent. Mais, même si un mécanisme de contrôle d'admission est utilisé, il n'y a pas la garantie que les ressources disponibles à l'instant $t$ le restent à l'instant $t+\tau$, avec la mobilité et la variation de la bande passante dûe à la moindre source d'interférence. Pour cela, le mécanisme de contrôle d'admission est utilisé seulement avant l'ouverture d'une session, sans prévoir l'influence des nouveaux flots sur la QdS de ceux qui existent.

Mais, si la charge dépasse la capacité du réseau après la transmission des nouveaux flots, un mécanisme de contrôle de congestion doit être activé pour rejeter les flots affectant la *QdS* des autres, et causant la congestion du réseau. En se basant sur ce principe, la transmission d'un nouveau flot commence immédiatement après la vérification de l'existence de ressources nécessaires par le mécanisme de

contrôle d'admission. Ensuite, le mécanisme d'estimation de la bande passante se charge d'estimer l'influence des nouveaux arrivés sur la bande passante, et déclenche l'alarme en cas de congestion de façon similaire à la méthode utilisée dans le modèle du chapitre précédent. Le principe de fonctionnement des mécanismes de contrôle d'admission, de congestion et d'estimation de la bande passante sont les mêmes que ceux utilisés dans la section 4.3.7 du chapitre 4.

## 5.3 Simulations et Résultats

### 5.3.1 Modèle de simulation

Pour réaliser une étude quantitative, nous avons modifié le modèle des réseaux des files d'attente utilisé pour la simulation dans le chapitre 4, et nous avons réalisé une série de tests. La topologie présentée dans la figure 5.2, s'étend sur une surface de 800x200 $m^2$, avec un nombre de nœuds égal à 8. La distance verticale entre deux nœuds est de 200 m, et la zone de couverture d'un nœud est de 250 m. La mobilité n'est pas prise en compte durant la simulation. La bande passante du médium est de 11 Mbps et le temps de la simulation est de 100 s. Les résultats des simulations réalisées ont montré la capacité de notre modèle à maintenir le délai et le débit des flots en deçà des valeurs spécifiées par les applications.

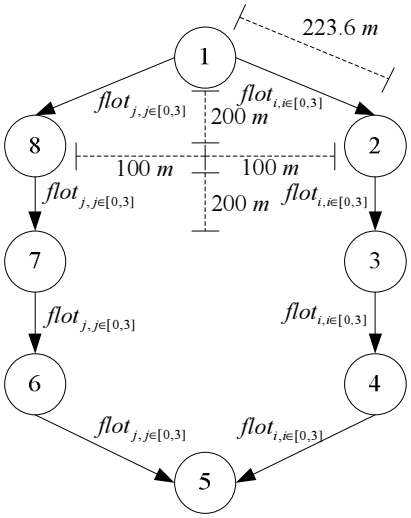


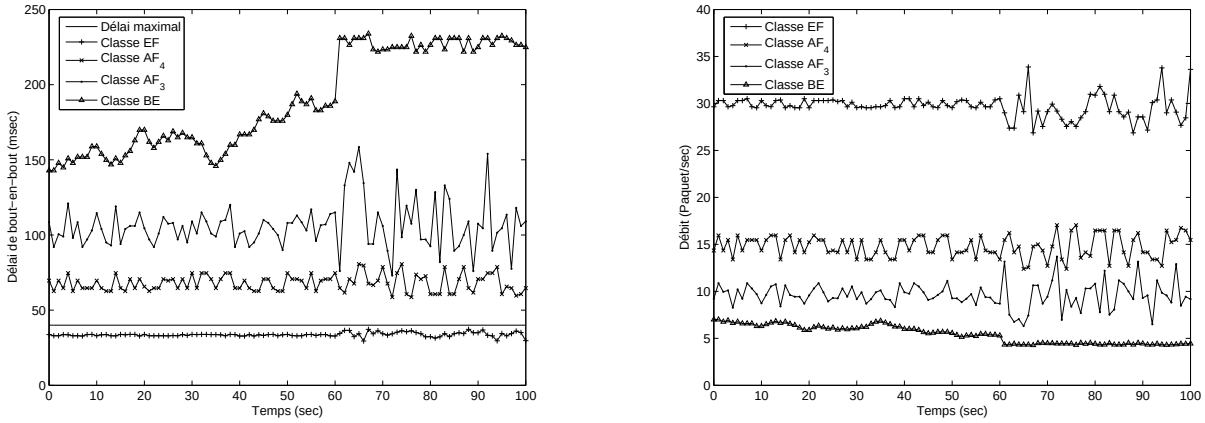

Figure 5.2 – La topologie de la simulation.

### 5.3.2 Résultat des simulations

Dans la première série de simulations, le nœud 1 est la source de 8 flots, et le nœud 5 en est la destination. 4 flots sont routés statiquement à travers le chemin $1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5$ et les 4 autres à travers le chemin $1 \rightarrow 8 \rightarrow 7 \rightarrow 6 \rightarrow 5$, afin de prendre en compte l'influence de la contention sur les autres flots. Les paramètres des flots utilisé durant la simulation, sont ceux utilisés dans le chapitre 4, et ils sont donnés dans le tableau 4.8. Les paramètres correspondant au mode de fonctionnement PHY *IEEE 802.11a* et au niveau MAC sont donnés dans le tableau 4.7. Les paramètres de notre modèle utilisé durant la simulation sont donnés dans le tableau 5.1. Les 4 premiers flots traversant le chemin $1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5$ commencent à l'instant $t$=0, avec une contrainte de délai de bout-en-bout pour le flot audio ($EF$) fixée à 40 ms, et une contrainte de débit minimal de 15.36 koctet/s pour la vidéo ($AF_4$), et 6.6 koctet/s pour le flot TCP interactif ($AF_3$). Le deuxième groupe de flots passant à travers les nœuds $1 \rightarrow 8 \rightarrow 7 \rightarrow 6 \rightarrow 5$ commence à l'instant $t$=60 sec. Les figures 5.3(a) et 5.3(b) montrent les variations du délai et du débit pour les 4 premiers flots traversant le premier chemin, ainsi que les valeurs moyennes dans les figures 5.4(a) et 5.4(b).

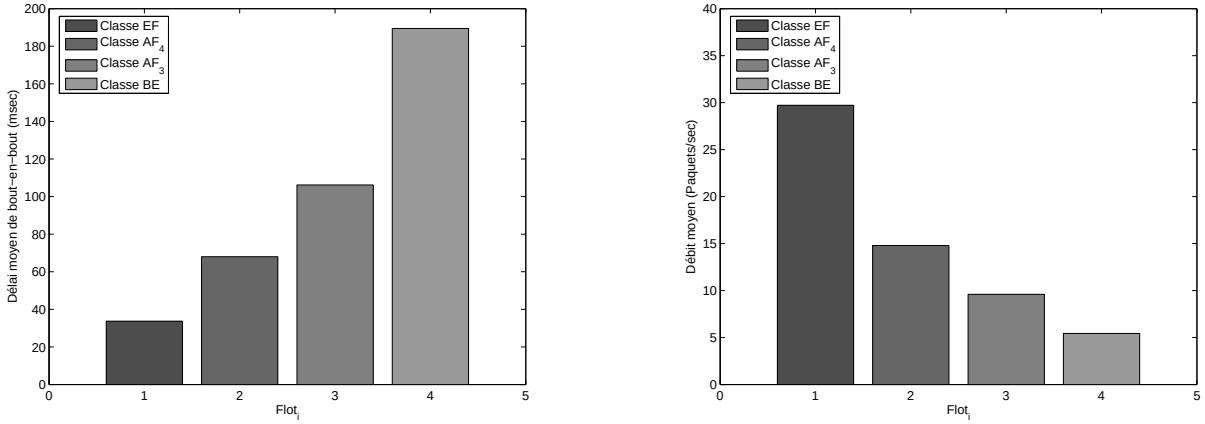
| Paramètre | Valeur |
|---|---|
| $\theta$ | 2 ms |
| $L_{seuil,1}$ | 10 paquets |
| $L_{seuil,2}$ | 50 paquets |
| $I_{seuil}$ | 10 |
| $T_{idle,seuil}$ | 3 |

Tableau 5.1 – Paramètres du modèle.




(a) Délai. (b) Débit.

Figure 5.3 – Variation du délai et de débit.

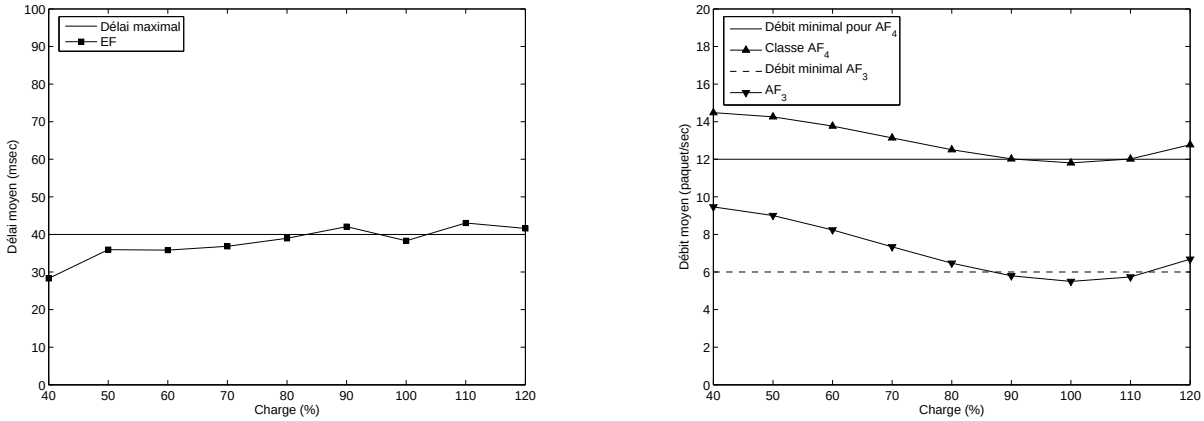



(a) Délai moyen. (b) Débit moyen.

Figure 5.4 – Valeurs moyennes.

Dans la deuxième série de tests, nous étudions l'influence de la charge du réseau sur l'efficacité de notre approche. Nous avons injecté de 1 à 7 flots vidéo (MPEG) avec un débit de 137.5 koctet/sec, de façon à augmenter la charge du réseau de 40% à 120% en augmentant le nombre de flots. Les figures 5.5(a) et 5.5(b) présentent la valeur moyenne du délai de la classe $EF$, ainsi que les débits moyens de deux classes $AF_4$ et $AF_3$. Ces figures montrent l'efficacité de notre approche à maintenir le délai et le débit dans les limites des contraintes des applications. Lorsque la charge dépasse la capacité du réseau, le mécanisme de contrôle d'admission se charge de rejeter les nouveaux flots pour adapter la charge à la

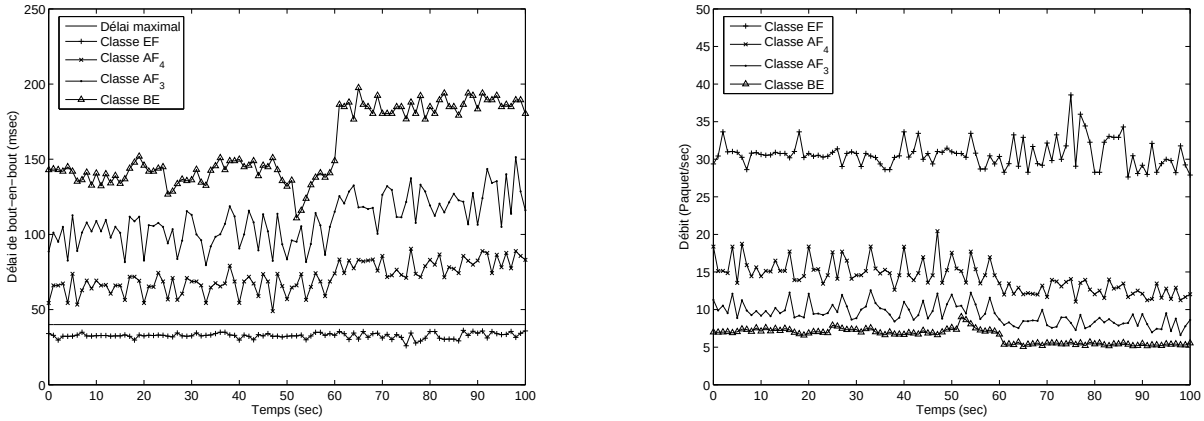
capacité. Pour cela, une amélioration dans le délai et le débit aura lieu aux alentours d'une charge de 90% comme le montre la figure 5.5.




(a) Délai. (b) Débit.

Figure 5.5 – Variation du délai et du débit avec la variation du charge de réseau.

Dans la troisième série de tests, nous avons étudié l'influence d'une loi entre deux arrivées $BP$ sur les performances du modèle. Les résultats de nos simulations sont présentés dans les figures 5.6 et 5.7, où on peut constater que même avec différentes distributions de trafic, le délai de la classe $EF$ est peu perturbé par les autres trafics, et les débits des classes $AF_4$ et $AF_3$ sont toujours supérieurs aux valeurs minimales requises par les applications.

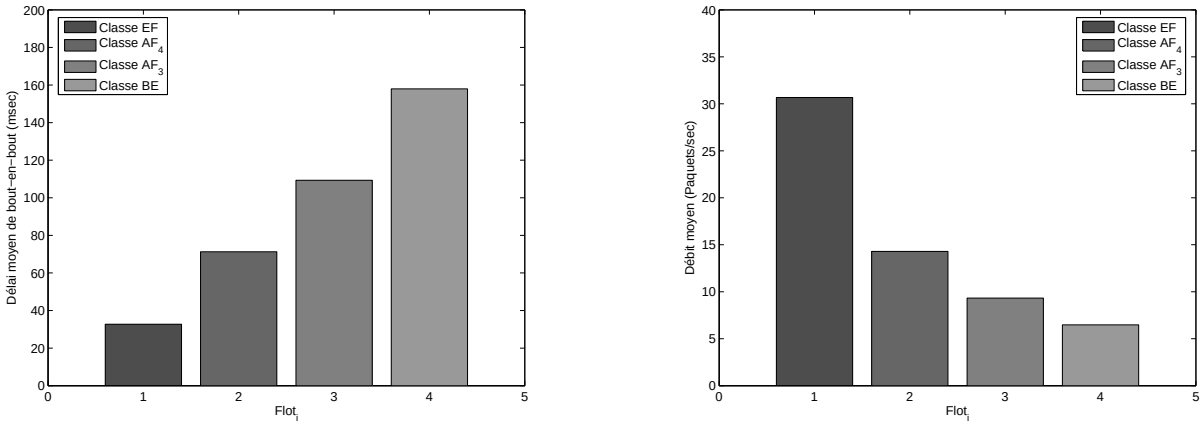



(a) Délai. (b) Débit.

Figure 5.6 – Variation du délai et de débit avec une distribution entre arrivées $BP$.

## 5.4 Conclusion.

Dans ce chapitre, nous avons présenté un nouveau mécanisme de gestion de la qualité de service de bout en bout, basé sur l'extension du modèle proposé dans le chapitre précèdent, mais cette fois pour fournir une différenciation absolue basée sur les contraintes des applications. Ce modèle garantit le respect du délai et du débit de bout-en-bout sans faire de réservation de ressources, et tout en se basant sur un traitement distribué. Par ailleurs, ce traitement ne nécessite pas le stockage d'information d'état liée aux flots sur les nœuds intermédiaires. Les résultats des simulations effectuées montrent l'efficacité de notre modèle sous différentes charges du réseau et différentes lois d'arrivées des paquets.




(a) Délai moyen

(b) Débit moyen

Figure 5.7 – Valeurs moyennes avec une distribution entre arrivées $BP$.

La suite logique de ces travaux est de continuer l'implémentation de ce modèle dans le simulateur NS-2, pour étudier plus précisément l'influence de la mobilité sur les performances. Dans le but d'améliorer encore les performances, nous voulons étudier l'influence de l'addition du mécanisme de contrôle d'admission à la source, utilisée dans SWAN, pour vérifier l'existence des ressources avant l'ouverture d'une session.

# Chapitre 6

# Conclusions et perspectives

Nous dressons dans une première section un bilan des travaux effectués au cours de cette thèse, en rappelant les principaux résultats et en soulignant les bénéfices de chacun de ces axes. Ensuite, nous évoquons les nombreuses perspectives ouvertes par ces travaux pour donner suite à cette thèse.

## 6.1 Conclusions

La spécification d'un système et la vérification des ses propriétés durant la phase de conception, est une étape fondamentale dans le cycle de développement. Les algèbres des processus stochastiques sont des techniques de description formelle de haut niveau, qui permettent de spécifier un modèle d'un système, et de disposer d'une axiomatique et d'un fondement mathématique qui induit les preuves, et qui permet de vérifier la satisfaction des propriétés exigées de l'utilisateur, par le diagramme des états sous-jacent de la spécification réalisée.

Nous nous sommes intéressés à la spécification et à la vérification des systèmes avec des délais sur des activités réelles, et spécialement à l'application de la spécification formelle pour la conception de nouveaux mécanismes de gestion du réseau. Mais la spécification temporelle des délais des actions dans les algèbres de processus stochastiques est limitée aux distributions sans mémoire. Ceci peut sembler avantageux, car la majorité des études (analytiques, numériques et simulations) réalisées sur les réseaux, ont supposé des caractéristiques Markoviennes de trafic.

Mais, cette supposition semble être fausse aujourd'hui, où les mesures réalisées sur le réseau ont montré que les caractéristiques du trafic du réseau (taille des paquets, temps entre arrivées, etc.), suivent plutôt une distribution à décroissance lente (« heavy-tailed ») dont les fonctions de distribution ne sont pas obligatoirement sans mémoire.

Afin de remédier à ce problème au niveau de la spécification temporelle dans le formalisme algébrique, nous avons proposé une méthode d'intégration des distributions générales dans les algèbres de processus stochastiques, à travers la représentation par les distributions phase type, tout en essayant de résoudre les problèmes sous-jacents de la spécification de haut niveau. L'utilisation des distributions phase type engendre une croissance non linéaire de l'espace d'états, et pour éviter l'explosion de ce dernier en gardant une bonne précision d'approximation, nous avons cherché la représentation $PH$ minimale.

Mais les représentations par les formes $PH$ qui sont largement utilisées, ne permettent pas de bien représenter les distributions à décroissance lente avec un nombre de phases petit. Pour cela, nous avons cherché la forme convenable pour la représentation des telles distributions. Nous avons utilisé la distribution hyper-erlang qui permet de bien couvrir toutes les distributions avec un $cv^2 > 0$, pour bien représenter ces distributions avec un nombre de paramètres et de phase minimaux, afin de réduire la complexité de l'algorithme de recherche de ces paramètres.

Un algorithme basé sur le principe du maximum de vraissemblance et l'algorithme EM (Expectation-Maximisation), ont été utilisés pour dériver ces paramètres. Grâce aux résultats numériques obtenus, nous avons pu comparer les nombres de phases requises par les algorithmes classiques et notre algorithme, et nous avons pu montrer l'efficacité dans l'approximation par un petit nombre de phases. Cet algorithme

présente un intérêt pratique considérable pour la modélisation des distributions à décroissance lente, ce qui permet d'augmenter l'expressivité des formalismes algébriques pour la conception des modèles plus concis et plus proches du domaine du problème, tout en évitant l'explosion de l'espace d'états.

Récemment, Bobbio et Telek ont proposé une méthode analytique pour la recherche de la forme $PH$ minimale [BHT05], tout en cherchant à égaliser seulement les trois premiers moments des distributions $G$ et de $PH$. Ce qui permet d'éviter l'utilisation des algorithmes d'approximation avec une complexité non linéaire pour l'approximation des distributions générales. Nous avons programmé cette méthode d'approximation sous MatLab suite à l'absence d'un outil, et de nombreux résultats numériques ont été obtenus par ce moyen.

Mais, l'utilisation des distributions phase type provoque une croissance du nombre d'états dans le diagramme de transition sous jacent. Pour cela, nous ne nous sommes pas contentés de travailler sur la phase d'abstraction pour éviter l'explosion de l'espace d'états. Nous avons cherché à réduire cet espace pour pouvoir traiter des systèmes de plus en plus proches de la réalité. Pour ce faire, nous avons pu exploité les algorithmes d'approximation de distributions de phase pour agréger l'espace d'états, tout en calculant les 3 premiers moments d'une sous chaîne agrégée, et en la remplaçant par la distribution $PH$ correspondante.

Nous avons proposé dans cette thèse des méthodes et des algorithmes basés sur les algèbres de processus stochastiques. Mais ces travaux peuvent être utilisés dans les formalismes voisins, et spécialement avec les automates stochastiques, où des techniques de résolution d'espace d'états grands sont développées.

Ensuite, nos travaux à l'intersection de la modélisation et de l'évaluation des performances se sont tournés vers la qualité de service et la conception de nouveaux mécanismes de gestion de la qualité de service dans les réseaux ad hoc. L'objectif scientifique était de montrer l'utilité d'une approche formelle fondée sur les algèbres de processus stochastiques, pour la conception de tels systèmes.

Nous avons proposé une extension du modèle de gestion de la qualité de service basée sur la différenciation proportionnelle (PDS). Des algorithmes de gestion et de contrôle spécifiques, complètement distribués, ont été élaborés pour faire face à la dynamicité des ressources du réseau, et pour fournir une qualité de service proportionnelle de bout en bout entre les différentes classes de trafic. L'étude du modèle proposé par simulation, a montré que même avec des variations de la charge du réseau ou de la bande passante, le rapport des délais aux différentes classes reste toujours proportionnel.

Mais, en se basant sur le fait que les applications ont différents besoins en termes de paramètres de qualité de service, et généralement des contraintes déterministes sur des valeurs optimales de ces paramètres, nous avons proposé une extension d'un premier modèle pour fournir une QdS absolue de bout en bout, basée sur les contraintes des applications, et tout en répondant aux différentes contraintes des applications en termes de QoS. L'étude des performances de ce modèle montre qu'il s'adapte bien à la dynamicité du réseau pour essayer de satisfaire les contraintes des applications. Les résultats des simulations sur le scénario considéré, montrent la capacité du modèle proposé à écouler les trafics et à satisfaire les contraintes des applications, même sous différentes conditions de surcharge du réseau.

## 6.2 Perspectives et Futurs travaux

Nos travaux futurs vont dans la continuité de notre thème de recherche, où l'ensemble des travaux proposés dans cette thèse donne naissance à de nombreuses perspectives.

Comme nous avons été amenés à considérer en plus des problèmes propres à la gestion de la qualité de service dans les réseaux ad hoc, des problèmes de modélisation et d'analyse dans les formalismes algébriques (réduction de l'espace d'états, expressivité temporelle, etc.), nous présentons nos perspectives autour des deux domaines.

D'une part, des nombreuses recherches restent à effectuer sur les formalismes des algèbres de processus stochastiques, où d'un point de vue modélisation algébrique et analyse de performances, il nous semble important de pousser plus loin notre travail sur les techniques d'agrégation pour réduire significativement l'espace d'états engendré.

La majorité des techniques traitent le problème de l'explosion de l'espace d'états a posteriori (après la dérivation du diagramme). De ce fait, ces techniques se trouvent confrontés à un traitement d'un espace d'état grand, où la réduction de la complexité temporelle, qui sera gagnée lors de la résolution linéaire, est au moins celle engendrée pour l'identification des états agrégables, et la génération du nouveau espace d'état après l'agrégation de ces états. Il nous semble assez logique qu'une stratégie de réduction capable de traiter les modèles de haut niveau avant de générer le diagramme de transitions de bas niveau, ou plutôt un traitement a priori suivi d'un autre a posteriori, doit être mise en œuvre afin d'éviter le traitement d'un espace d'états grand pour l'identification des états agrégables. Ensuite, un gros travail reste à faire sur les méthodes de résolution utilisées dans ce formalisme, et il nous semble intéressant de regarder les travaux effectués dans les formalismes voisins, et surtout autour des automates stochastiques.

Les algèbres de processus stochastiques à temps discret semble avoir été oubliés, et nous avons vu que les distributions $PH$ à temps discret ont la capacité de bien représenter tout type de distribution, et avec un nombre de phases beaucoup plus petit que celles à temps continu. Ensuite, des travaux sur la proposition de nouvelles sémantiques basées sur les distributions phases type est envisageable, où seulement l'opérateur de composition parallèle constitue un obstacle avec les entrelacements des phases. Un formalisme algébrique avec des distributions $PH$ nous semble assez intéressant, où le concepteur s'occupe seulement de la spécification des distributions à utiliser, et la machine prend en charge l'approximation et la dérivation de la chaîne de Markov sous-jacente du modèle.

La modélisation algébrique offre une aide indéniable pour conforter la mise en place d'une simulation, dont on est sûr du fonctionnement. Mais, même avec les nombreux avantages, beaucoup des difficultés persistent au niveau la spécification temporelle, et de la détermination des délais des activités. La précision des résultats dépend des valeurs choisies pour les délais des ces activités. Celles-ci sont généralement choisies au hasard par le concepteur, car même à partir de valeurs obtenues en sondant un réseau réel, ces valeurs résultent rarement de l'exécution d'une seule action, mais plutôt de plusieurs. Pour cela, il nous semble assez intéressant de mettre en place un algorithme permettant la recherche des délais des activités d'un processus, étant donné les valeurs obtenues à partir des mesures prélevées sur un prototype, ou un système existant. Nous envisageons de développer un outil logiciel capable de dériver ces paramètres, afin de systématiser l'estimation de ces paramètres lorsqu'on possède certaines informations sur les composants.

D'autre part, dans nos travaux autour de la qualité de service dans les réseaux ad hoc, il est intéressant dans un premier temps, de finaliser l'implémentation du dernier modèle proposé sous le simulateur NS-2, afin d'étudier l'influence du modèle de mobilité largement utilisé dans la littérature (Random Way-Point) sur les performances des ces mécanismes.

Dans la suite, nous souhaitons approfondir nos contributions par les déploiement réelle. il faut étudier la mobilité de l'utilisateur et de proposer une extension du modèle de gestion de ressources en se basant sur des informations génétique de la mobilité, comme les trajectoires habituelles des utilisateurs, et l'endroit de formation de tels réseaux, où le temps de connexion dans un café ou sur une place, ne sera pas le même que celui passé par une personne connectée au réseau dans un bus. En effet, la tendance actuelle à aller vers la conception et la mise en place d'un modèle algébrique tenant compte de la mobilité. Une telle proposition a donc davantage d'intérêt si elle est associée à la prise en compte des informations partielles qui sont observées.

Toutes les simulations réalisées sur les modèles de gestion de la qualité de service ne prennent pas en compte les interférences et la puissance des émissions. Une étude de l'impact des interférences sur les performances sera plus réaliste, et encore beaucoup plus souhaitable. Une extension pourra être envisagée pour prendre en compte et réagir dynamiquement afin d'anéantir l'influence des interférences sur les performances du modèle, s'il en existe.

Enfin, nous espérons que la lecture de ce document aura apporté des perspectives intéressant et ouvert la porte à de futures recherches en ce sens.

# Liste d'acronymes

**2LQoS :** Two-Layered QoS Model for Manets.

**AC :** Access Category.

**ACP :** Algebra of Communicating Processes.

**ADPH :** Acyclic Discret Phase type distributions.

**AF :** Assured Forwarding.

**AIFS :** Arbitration Inter Frame Space.

**AODV :** Ad hoc On-demand Distance Vector.

**APH :** Acyclic Phase type distributions.

**APHM :** Minimal Acyclic Phase type distributions.

**AQOR :** Ad hoc QoS On-demand Routing.

**ASAP :** Adaptive reServation and pre-Allocation Protocol.

**BA :** Behavior Aggregates.

**BDD :** Binary Decision Diagram.

**BE :** Best Effort.

**BP :** Bounded Pareto.

**BT :** Backoff Timer.

**CAC :** Call Admission Control.

**CCS :** Calculus of Communicating Systems.

**CDMA :** Code Division Multiple Access.

**CEDAR :** Core Extraction Distributed Ad hoc Routing algorithm.

**CF :** Canonical Form.

**CSMA/CA :** Carrier Sense Multiple Access/Collision Avoidance.

**CSP :** Communicating Sequential Process.

**CTL :** Computation Tree Logic.

**CTS :** Clear To Send.

**CW :** Congestion Window.

**DCF :** Distributed Coordination Function.

**DDPN :** Discret Determinstic Petri Net.

**DES :** Discret Event Simulation.

**DiffServ :** Differentiated Services.

**DIFS :** Distributed Inter Frame Space.

**DSCP :** Differentiated Service Code Point.

**DSDV :** Destination-Sequenced Distance-Vector.

**DSPN :** Determinstic Stochastic Petri Net.

**DSR :** Dynamic Source Routing.

**ECN :** Explicit Congestion Notification.

**EDCF :** Enhanced Distributed Coordination Function.

**EF :** Expedited Forwarding.

**EM :** Expectation-Maximisation.

**EMPA :** Extended Markovian Process Algebra.

**ESPN :** Extended Stochastic Petri Net.

**FQMM :** Flexible QoS Model for Manets.

**FTP :** File Transfer Protocol.

**G :** General distribution.

**GSMP :** Generalized Semi Markov Process.

**GSPN :** Generalised Stochastic Petri Net.

**HCF :** Hybrid Coordination Function.

**HPD :** Hybrid Proportional Delay.

**HTTP :** Hypertext Transfer Protocol.

**IGSMP :** Interactive Generalized Semi Markov Process.

**INSIGNA :** IN-band SIGNAling for QoS in Manets.

**IntServ :** Integrated Services.

**IP :** Internet Protocol.

**LOTOS :** Language of Temporal Ordering Specifications.

**LTL :** Linear Temporal logic.

**ML :** Maximum Likelihood.

**MP-DSR :** Multi-Path Dynamic Source Routing Protocol.

**OLSR :** Optimized Link State Routing Protocol.

**PAD :** Proporitional Average Delay.

**PDD :** Proportional Delay Differentiation.

**PEPA :** Performance Evaluation Process Algebra.

**PH :** Phase type distributions.

**PHB :** Per Hop Behavior.

**RED :** Random Early Detection.

**RFT :** Remaining Firing Time.

**RREP :** Route Reply.

**RREQ :** Route Request.

**RRER :** Route Error.

**RSVP :** Resource Reservation Protocol.

**RTS :** Request To Send.

**SIFS :** Short Inter Frame Space.

**SLA :** Service Level Agreement.

**SMP :** Semi Markov Process.

**SOS :** Structural Operational Semantic.

**SPA :** Stochastic Process Algebra.

**SWAN :** Stateless Wireless Ad-hoc Networks.

**TBP :** Ticket Based Probing.

**TBRPF :** Topology dissemination Based on Reverse-Path Forwarding.

**TCA :** Traffic Control Agreement.

**TCP :** Transmission Control Protocol.

**TDMA :** Time Division Multiple Access.

**TIPP :** TImed Process and Performability analysis.

**TORA :** Temporally Ordered Routing Algorithm.

**VoIP :** Voice over IP.

**WTP :** Waiting Time Priority.

**ZRP :** Zone Routing Protocol.

# Bibliographie


[AC01] Imad Aad and Claude Castelluccia. Differentiation Mechanisms for IEEE 802.11. In *Proceedings of IEEE INFOCOM*, pages 209–218, April 2001.

[ACD93] Rajeev Alur, Costas Courcoubetis, and David Dill. Model-Checking in Dense Real-Time. *Information and Computation*, 104 :2–34, 1993.

[ACVS02] Gahng-Seop Ahn, Andrew T. Campbell, Andras Veres, and Li-Hsiang Sun. Supporting Service Differentiation for Real-Time and Best-Effort Traffic in Stateless Wireless Ad Hoc Networks (SWAN). *IEEE Transactions on Mobile Computing*, 1(3) :192–207, September 2002.

[AGRN03] Harpreet Arora, Lloyd Greenwald, Usha Rao, and John Novatnack. Performance Comparison and Analysis of Two QoS schemes : SWAN and DiffServ. Drexel Universitiy Research Fifth Annual Research Day 2003, April 2003.

[Alt85] Tayfur Altiok. On the Phase-Type Approximations of General Distributions. *IEEE Trans.*, 17 :110 –116, 1985.

[ANO96] Soren Asmussen, Olle Nerman, and Marita Olsson. Fitting Phase-Type Distributions via the EM Algorithm. *Scandinavian Journal of Statistics*, 23 :419–441, 1996.

[AS87] David Aldous and Lawrence Shepp. The Least Variable Phase Type Distribution is Erlang. *Communications in Statistics – Stochastic Models*, 3(3) :467–473, 1987.

[AS02] Harpreet Arora and Harish Sethu. A Simulation Study of the Impact of Mobility on Performance in Mobile Ad Hoc Networks. In *Proceedings of the Applied Telecommunication Symposium*, San Diego, California, April 2002.

[ASSB00] Adnan Aziz, Kumud Sanwal, Vigyan Singhal, and Robert Brayton. Model Checking Continuous Time Markov Chains. *ACM Transactions on Computational Logic*, 1(1) :162–170, 2000.

[BB87] Tommaso Bolognesi and Ed Brinksma. Introduction to The ISO Specification Language LOTOS. *Computer Networks and ISDN Systems, Elsevier Science Publishers*, 14(1) :25–59, 1987.

[BB98] Marco Bernardo and Mario Bravetti. Functional and Performance Modeling and Analysis of Token Ring using EMPA. In *Proceedings of The* $6^{th}$ *Italian Conference on Theoretical Computer Science (ICTCS'98)*, pages 204–215, Prato (Italy), November 1998.

[BBC+98] Steven Blake, David Black, Mark Carlson, Elwyn Davies, Zheng Wang, and Walter Weiss. An Architecture for Differentiated Services. RFC 2475, December 1998.

[BBG98] Mario Bravetti, Marco Bernardo, and Roberto Gorrieri. Towards Performance Evaluation with General Distributions in Process Algebras. In D. Sangiorgi and R. de Simone, editors, *Proceedings CONCUR'98*, volume 1466 of *Lecture Notes in Computer*, pages 405–422, Nice, France, 1998. Springer-Verlag.

[BC92] Andrea Bobbio and Aldo Cumani. ML Estimation of the Parameters of a PH Distribution in Triangular Canonical Form. In G. Balbo and G. Serazzi, editors, *Computer Performance Evaluation*, pages 33–46. Elsevier Science Publishers, 1992.

[BCD02] Marco Bernardo, Paolo Ciancarini, and Lorenzo Donatiello. Architecting Families of Software Systems with Process Algebras. *ACM Trans. Softw. Eng. Methodol.*, 11(4) :386–426, 2002.

[BCS94] Robert Braden, David Clark, and Scott Shenker. Integrated Services in the Internet Architecture : an Overview. RFC 1633, June 1994.

[BCV01] Michael Barry, Andrew Campbell, and Andras Veres. Distributed Control Algorithms for Service Differentiation in Wireless Packet Networks. In *IEEE INFOCOM*, April 2001.

[BDG98] Marco Bernardo, Lorenzo Donatiello, and Roberto Gorrieri. A Formal Approach to the Integration of Performance Aspects in the Modeling and Analysis of Concurrent Systems. *International journal of Information and Computation*, 144(2) :83–154, August 1998.

[BDPS04] P.P. Bocharov, C. D'Apice, A.V. Pechinkin, and S. Salerno. *Queuing Theory (Modern Probability and Statistics)*. Brill Academic, Netherlands, May 2004.

[Ben03] Anne Benoit. *Méthodes et algorithmes pour l'évaluation des performances des systèmes informatiques à grand espace d'états*. PhD thesis, Institut National Polytechnique de Grenoble, June 2003.

[Ber97a] Marco Bernardo. An Algebra Based Method to Associate Rewards with EMPA Terms. In *Proceedings of The* $24^{th}$ *International Colloquium on Automata, Languages and Programming (ICALP'97)*, pages 358–368, June 1997.

[Ber97b] Marco Bernardo. Enriching EMPA with Value Passing : A Symbolic Approach Based on Lookahead. In Ed Brinksma and A. Nymeyer, editors, *Proceedings of the* $5^{th}$ *International Workshop on Process Algebras and Performance Modelling*, pages 35–49, Enschede, Netherlands, June 1997.

[Ber99] Marco Bernardo. *Theory and Application of Extended Markovian Process Algebra*. PhD thesis, University of Bologna, Italy, Februray 1999.

[BG98] Marco Bernardo and Roberto Gorrieri. A Tutorial on EMPA : A Theory of Concurrent Processes with Nondeterminism, Priorities, Probabilities and Time. *Theoretical Computer Science*, 201 :1–54, July 1998.

[BG99] Mario Bravetti and Roberto Gorrieri. Interactive Generalized Semi-Markov Processes. In Jane Hillston and M. Silva, editors, *In the Proceedings of the* $7^{th}$ *Internation Workshop on Process Algebras and Performance Modeling*, pages 83–98, Zaragoza (Spain), September 1999.

[BG02] Mario Bravetti and Roberto Gorrieri. The Theory of Interactive Generalized Semi-Markov Processes. *Theor. Computer Science*, 282(1) :5–32, June 2002.

[BH01] Ed Brinksma and Holger Hermanns. Process Algebra and Markov Chains. *Lectures on Formal Methods and Performance Analysis*, 2090 :183–231, 2001.

[BHT05] Andrea Bobbio, András Horváth, and Miklós Telek. Matching Three Moments With Minimal Acyclic Phase Type Distributions. *Stochastic Models*, 21 :303–326, 2005.

[Bia00] Giuseppe Bianchi. Performance Analysis of the IEEE 802.11 Distributed Coordination Function. *IEEE Journal on Selected Areas in Communications*, 18(3) :535–547, March 2000.

[BK85] Jan Bergstra and Jan Willem Klop. Algebra of Communicating Processes with Abstraction. *Theoretical Computer Science*, 37 :77–121, 1985.

[BMJ+98] Josh Broch, David A. Maltz, David B. Johnson, Yih-Chun Hu, and Jorjeta Jetcheva. A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols. In *Proceedings of the Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking*. ACM, October 1998.

[BN02] Christian Bonnet and Navid Nikaein. A Glance at Quality of Service Models for Mobile Ad Hoc Networks. In *16$^e$ Congrès De Nouvelles Architectures pour les Communications (DNAC)*, Paris, France, December 2002.

[BPS01] Jan Bergstra, Alban Ponse, and Scott Smolka, editors. *Handbook of Process Algebra*. Elsevier Science Publishers, 2001.

[BPST98] Andrea Bobbio, Antonio Puliafito, Marco Scarpa, and Miklòs Telek. WebSPN : Non-Markovian Stochastic Petri Net Tool. In *Proceedings of the International Conference on WEB-based Modeling and Simulation*, San Diego, USA, January 1998.

[Bra02] Mario Bravetti. *Specification and Analysis of Stochastic Real-Time Systems*. Dottorato di ricerca in informatica, Università di Bologna, Padova, Venezia, February 2002.

[Bra04] Mario Bravetti. Real Time and Stochastic Time. In *Proceedinds of the International School on Formal Methods for the Design of Computer, Communication and Software Systems*, volume 3185, pages 132–180. Springer, September 2004.

[Bry86] Randal E. Bryant. Graph-Based Algorithms for Boolean Function Manipulation. *IEEE Transactions on Computers*, 35(8) :677–691, 1986.

[Bry92] Randal E. Bryant. Symbolic Boolean Manipulation with Ordered Binary-Decision Diagrams. *ACM Computing Surveys*, 24(3) :293–318, September 1992.

[BZB+97] Robert Braden, Lixia Zhang, Steven Berson, Shai Herzog, and Sigih Jamin. Resource ReSerVation Protocol(RSVP) – Version 1 Functional Specification. RFC 2205, September 1997.

[CCL03] Imrich Chlamtac, Marco Conti, and Jennifer J.-N. Liu. Mobile Ad Hoc Networking : Imperatives and Challenges. *Elsevier Ad Hoc Networks Journal*, 1(1) :13–64, July 2003.

[CES86] Edmund M. Clarke, E. Allen Emerson, and A. Prasad Sistla. Automatic Verification of Finite-State Concurrent Systems using Temporal Logic Specifications. *ACM Trans. Program. Lang. Syst.*, 8(2) :244–263, 1986.

[CGHR00] Graham Clark, Stephen Gilmore, Jane Hillston, and Marina Ribaudo. Exploiting Modal Logic to Express Performance Measures. In *TOOLS '00 : Proceedings of the* $11^{th}$ *International Conference on Computer Performance Evaluation : Modelling Techniques and Tools*, pages 247–261, London, UK, 2000. Springer-Verlag.

[CGL94] Gianfranco Ciardo, Reinhard German, and Christoph Lindemann. A Characterization Of The Stochastic Process Underlying A Stochastic Petri Net. *IEEEE Transactions on software engineering*, 20(7), July 1994.

[CGP99] Edmund M. Clarke, Orna Grumberg, and Doron A. Peled. *Model Checking*. MIT Press, 1999.

[CH96] Graham Clark and Jane Hillston. Towards Automatic Derivation of Performance Measures from PEPA Models. In Rob Pooley and Jane Hillston, editors, *Proceedings of the Twelfth UK Performance Engineering Workshop*, pages 65–81, September 1996.

[CH02] Graham Clark and Jane Hillston. Product Form Solution for an Insensitive Stochastic Process Algebra Structure. *Performance Evaluation*, 50(2–3) :129–151, 2002.

[Che99] Shigang Chen. *Routing Support for Providing Guaranteed End-to-End Quality-of-Service*. PhD thesis, University Illinois at Urbana-Champaign, 1999.

[Chi91] Giovanni Chiola. GreatSPN 1.5 Software Architecture. In *Proceedings of the Fifth Internation Conference on Modeling Techniques and Tools for Computer Performance Evaluation*, Torino, Italy, February 1991.

[Cia95] Gianfranco Ciardo. Discrete-Time Markovian Stochastic Petri Nets. In W. J. Stewart, editor, *Computation with Markov Chains*, pages 339–358, Raleigh, NC, USA, Jan 1995.

[CJ03] Thomas Clausen and Phillipe Jacquet. Optimized Link State Routing Protocol (OLSR). RFC 3626, October 2003.

[CKT94] Hoon Choi, Vidyadhar G. Kulkarni, and Kishor S. Trivedi. Markov Regenerative Stochastic Petri Nets. *Perf. Eval.*, 20 :337–357, 1994.

[Cla99] Graham Clark. Stochastic Process Algebra Structure For Insensitivity. In Jane Hillston, editor, *Proceedings of the $7^{th}$ Workshop on Process Algebra and Performance Modelling*, pages 63–82. prensas universitarias de zaragoza, September 1999.

[Cla00] Graham Clark. *Techniques for the Construction and Analysis of Algebraic Performance Models*. PhD thesis, University of Edinburgh, 2000.

[CM99] Scott Corson and Joseph Macker. Mobile Ad Hoc Networking (MANET) : Routing Protocol Performance Issues and Evaluation Considerations. RFC 2501, January 1999.

[CMBC93] Giovanni Chiola, Marco Ajmone Marsan, Gianfranco Balbo, and Gianni Conte. Generalized Stochastic Petri Nets : A Definition at the Net Level and Its Implications. *IEEE Trans. Software Eng.*, 19(89–107), February 1993.

[CN99] Shigang Chen and Klara Nahrstedt. Distributed Quality of Service Routing in Ad-Hoc Networks. *IEEE Journal on Selected Areas in Communications*, 17(8) :1488–1505, August 1999.

[Cum82] Aldo Cumani. On the Canonical Representation of Homogeneous Markov Processes Modelling Failure-Time Distributions. *Microelectronics and Reliability*, 22 :583–602, 1982.

[CVB02] Periklis Chatzimisios, Vassilis Vitsas, and Anthony C. Boucouvalas. Throughput and Delay analysis of IEEE 802.11 Protocol. In *Proceedings of the 5th IEEE International Workshop on Networked Appliances*, pages 168–174, Liverpool John Moores University, UK, October 2002.

[D'A99] Pedro R. D'Argenio. *Algebras and Automata for Timed and Stochastic Systems*. PhD thesis, Department of Computer Science, University of Twente, 1999.

[DC99] Dr-Jiunn Deng and Ruay-Shiung Chang. A Priority Scheme for IEEE 802.11 DCF Access Method. *IEICE Trans. Commun.*, E82-B(1), January 1999.

[DCC+02] Daniel Deavours, Graham Clark, Tod Courtney, David Daly, Salem Derisavi, Jay Doyle, William Sanders, and Patrick Webster. The Möbius Framework and its Implementation. *IEEE TRANSACTIONS ON SOFTWARE ENGINEERING*, 28(10) :1–15, October 2002.

[DHKK01] Pedro R. D'Argenio, Holger Hermanns, Joost-Pieter Katoen, and Ric Klaren. MoDeST – A Modelling and Description Language for Stochastic Timed Systems. In Luca de Alfaro and Stephen Gilmore, editors, *Process Algebra and Probabilistic Methods (PAPM-ProbmiV 2001)*, volume 2165 of *Lecture Notes in Computer Science*, pages 87–104. Springer Verlag, September 2001.

[Dij59] Edsger Wybe Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1 :269–271, 1959.

[DKB97] Pedro R. D'Argenio, Joost-Pieter Katoen, and Ed Brinksma. A Stochastic Automata Model and its Algebraic Approach. In *Proceedings of the $5^{th}$ Workshop on Process Algebras and Performance Modelling*, pages 1–16, Enschede, The Netherlands, 1997.

[DKS89] Alan Demers, Srinivasan Keshav, and Scott Shenker. Analysis and Simulation of a Fair Queuing Algorithm. In *Proceedings of ACM SIGCOMM'89*, pages 1–12, September 1989.

[DLR77] Arthur Dempster, Nan Laird, and Donald Rubin. Maximum Likelihood from Incomplete Data via the $EM$ Algorithm. *Journal of the Royal Statistical Society : Series B (Statistical Methodology)*, 39(1) :1–38, 1977.

[Don93] Susanna Donatelli. Superposed Stochastic Automata : A Class of Stochastic Petri Nets with Parallel Solution and Distributed State Space. *Performance Evaluation*, 18(1) :21–36, 1993.

[DQT98] Corentin Durbach, Franck Quessette, and Alexis Troubnikoff. Solving Large Markov Model Based on Stochastic Automata Networks. In *Proceedings of the Advances in Computer and Information Sciences*, Belek-Antalya, Turkey, 1998. IOS Press.

[DR00] Constantinos Dovrolis and Parameswaran Ramanathan. Proportional differentiated services, Part II : Loss Rate Differentiation and Packet Dropping. In *Proceedings of the Eight International Workshop on Quality of Service (IWQoS)*, pages 52–61, 2000.

[DR01] Constantinos Dovrolis and Parameswaran Ramanathan. Dynamic Class Selection : From Relative Differentiation to Absolute QoS. In *Proceeding of the* $9^th$ *International Conference on Network Protocols (ICNP 2001)*, pages 120–128, 2001.

[DSR99] Constantinos Dovrolis, Dimitrios Stiliadis, and Parameswaran Ramanathan. Proportional Differentiated Services : Delay Differentiation and Packet Scheduling. In *Proc. ACM SIGCOMM*, 1999.

[DSR02] Constantinos Dovrolis, Dimitrios Stiliadis, and Parameswaran Ramanathan. Proportional Differentiated Services : Delay Differentiation and Packet Scheduling. *IEEE/ACM Trans. NETWORKING*, 10(1) :12–26, 2002.

[DTGN84] Joanne Bechta Dugan, Kishor S. Trivedi, Robert Geist, and Victor Nicola. Extended Stochastic Petri Nets : Applications and Analysis. Technical report, Department of Computer Science, Duke University, 1984.

[ER00] Moncef Elaoud and Parameswaran Ramanathan. Adaptive Allocation of CDMA Resources for Network-level QoS Assurances. In *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking (MOBICOM-00)*, pages 191–199, Aug 2000.

[Es05] Paal Engelstad and Olav Østerbø. Analysis of QoS in WLAN. *Telektronikk*, 1, 2005.

[Fer98] Paulo Fernandes. *Méthodes numériques pour la solution de systèmes markoviens à grand espace d'états*. PhD thesis, Institut National Polytechnique de Grenoble, France, 1998.

[FJ93] Sally Floyd and Van Jacobson. Random Early Detection Gateways for Congestion Avoidance. *IEEE/ACM Transactions on Networking*, 1(4) :397–413, Aug 1993.

[Fou97] Jean-Michel Fourneau. Réseaux d'automates stochastiques : Exemples et applications aux télécommunications. In *Proceedings of the Ecole d'informatique des systèmes parallèles et répartis*, Toulouse, France, 1997.

[FP02] Jean-Michel Fourneau and Nihal Pekergin. An Algorithmic Approach to Stochastic Bounds. *Lectures Note in computer science, Springer Verlag*, 2459 :64–88, 2002.

[Gei06] Nil Geisweiller. *Étude sur la Modélisation et la Vérification Probabiliste d'Architectures de Simulations Distribuées pour l'Évaluation de Performances*. PhD thesis, École nationale supérieure de l'aéronautique et de l'espace, 2006.

[Ger00] Reinhard German. *Performance Analysis of Communication Systems : Modeling with Non-Markovian Stochastic Petri Nets*. Number ISBN in 0-471-49258-2. John Wiley and Sons, 2000.

[GH03] Stephen Gilmore and Jane Hillston. A survey of the PEPA Tools. In *Proceedings of the Second Workshop on Process Algebra and Stochastically Timed Activities*, pages 40–49, June 2003.

[GHR93] Norbert Götz, Ulrich Herzog, and Michael Rettelbach. TIPP – A Stochastic Process Algebra. In Jane Hillston and F. Molle, editors, *Proceedings* $1^{st}$ *Workshop on Process Algebras and Performance Modelling*, pages 31–36, University of Edinburgh, UK, July 1993.

[GHR01] Stephen Gilmore, Jane Hillston, and Marina Ribaudo. An Efficient Algorithm for Aggregating PEPA Models. *IEEE Transactions on Software Engineering*, 27(5) :449–464, May 2001.

[GL94] Reinhard German and Christoph Lindemann. Analysis of Stochastic Petri Nets by the Method of Supplementary Variables. *Performance Evaluaiton*, 20(1–3) :317–335, 1994.

[GLT95] Reinhard German, Dimitris Logothetis, and Kishor S. Trivedi. Transient Analysis of Markov Regenerative Stochastic Petri Nets : A Comparison of Approaches. In *Proceedings of the* $6^{th}$ *International Workshop on Petri Nets and Performance Models*, pages 103–112, Durham, NC, USA, 1995. IEEE Computer Society Press.

[GMMP89] Carlo Ghezzi, Dino Mandrioli, Sandro Morasca, and Mauro Pezze. A General Way to Put Time in Petri Nets. In *Proc. 5th Int. Workshop on Software Specification*, pages 60–67, Pittsburgh, Pennsylvania, USA, May 1989.

[GW05] Rajarshi Gupta and Jean Walrand. *Advances in Control, Communication Networks, and Transportation Systems*, chapter Achieving Fairness in a Distributed Ad-Hoc MAC. Springer-Birkhauser, July 2005.

[Hay00] Simon Haykin. *Communication systems*. Wiley, 2000.

[HBWW99] Juha Heinanen, Fred Baker, Walter Weiss, and John Wroclawski. Assured Forwarding PHB Group. RFC 2597, June 1999.

[Her02] Holger Hermanns. Interactive Markov Chains. *Lecture Notes in Computer Science*, 2428, 2002.

[HH95] Peter Harrison and Jane Hillston. Exploiting Quasi-Reversible Structures in Markovian Process Algebra Models. In *Proceedings of* $3^{rd}$ *International Workshop on Process Algebras and Performance Modelling*, pages 510–520, Dec 1995.

[HHK02] Holger Hermanns, Ulrich Herzog, and Joost-Pieter Katoen. Process Algebra For Performance Evaluation. *Theoretical Computer Science*, 24(1–2) :43–87, 2002.

[HHM98] Holger Hermanns, Ulrich Herzog, and Vassilis Mertsiotakis. Stochastic Process Algebras – Between LOTOS and Markov chains. *Computer Networks and ISDN Systems*, 30(9–10) :901–924, 1998.

[Hil96] Jane Hillston. *A compositional Approach to Performance Modelling*. PhD thesis, Cambridge University Press, 1996.

[Hil00] Jane Hillston. Exploiting Structure in Solution : Decomposing Composed Models. In *the Proceedings of the Eighth Annual Workshop on Process Algebra and Performance Modelling*, Geneva, Switzerland, sep 2000.

[HJ94] Hans Hansson and Bengt Jonsson. A Logic for Reasoning about Time and Reliability. *Formal Aspects of Computing*, 6(5) :512–535, 1994.

[HK01] Jane Hillston and Leilla Kloul. An Efficient Kronecker Representation for PEPA Models. In L. de Alfaro and S. Gilmore, editors, *Proceedings of the First joint PAPM-PROBMIV Workshop*, volume 2165, Aachen, Germany, September 2001. Lecture Notes in Computer Science, Springer-Verlag.

[HM85] Matthew Hennessy and Robin Milner. Algebraic Laws for Non-Determinism and Concurrency. *Journal of the ACM*, 32(1) :137–161, January 1985.

[HM96] Holger Hermanns and Vassilis Mertsiotakis. A Stochastic Process Algebra Based Modelling Tool. In M. Merabti, M. Carew, and F. Ball, editors, *Performance Engineering of Computer and Telecommunications Systems*, pages 187–201. Springer, 1996.

[HMKS00] Holger Hermanns, Joachim Meyer-Kayser, and Markus Siegle. Multi Terminal Binary Decision Diagram to Represent and Analyse Continuous Time Markov Chains. In *Proceedings of the* $3^{rd}$ *International Workshop on the Numerical Solution of Markov Chains*, Prensas Universitarias de Zaragossa, Zaragossa, Spain, 2000.

[HOA85] Charles Antony Richard HOARE. *Communicating Sequential Processes*. Prentice-Hall, 1985.

[Hor03] András Horváth. *Approximating non-Markovian Behaviour by Markovian Models*. PhD thesis, Budapest University of Technology and Economics, 2003.

[How71] Ronald A. Howard. *Dynamic Probabilistic Systems*, volume 2 of *Semi Markov and Decision Processes*. John Wiley & Sons, 1971.

[HPS02] Zygmunt Haas, Marc Pearlman, and Prince Samar. Zone Routing Protocol (ZRP) for Ad Hoc Networks. Internet Draft-draft-ietf-manet-zone-zrp-04.txt, July 2002.

[HS97] Jan Ellsberger Dieter Hogrefe and Amardeo Sarma. *SDL – Formal Object-Oriented Language for Communication Systems*. Prentice Hall, 1997.

[HT98] Jane Hillston and Nigel Thomas. Product Form Solution for a Class of PEPA Models. In *Proceedings of IEEE International Computer Performance and Dependability Symposium*, Durham, NC, September 1998. IEEE Computer Society Press.

[HT00] András Horváth and Miklós Telek. Approximating Heavy Tailed Behaviour with Phase Type Distributions. In *Proceedings of the 3rd International Conference on Matrix-Analitic Methods in Stochastic models*, pages 191–214, Leuven, Belgium, June 2000.

[HTT00] Christophe Hirel, Bruno Tuffin, and Kishor S. Trivedi. SPNP : Stochastic Petri Nets Version 6.0. In *Proceedings of the* $11^{th}$ *International Conference on Computer Performance Evaluation : Modelling Techniques and Tools*, pages 354–357, London, UK, 2000. Springer-Verlag.

[Inf05] Information Sciences Institute. NS-2 network simulator. Software Package, 2005. http ://www.isi.edu/nsnam/ns/.

[JMH03] David B. Johnson, David A. Maltz, and Yih-Chun Hu. The Dynamic Source Routing Protcol for Mobile Ad Hoc Networks. Internet Draft, draft-ietf-manet-dsr- 09.txt, April 2003.

[JNP99] Vann Jacobson, Kathleen Nichols, and Kedarnath Poduri. An Expedited Forwarding PHB. RFC 2598, June 1999.

[JT91] Mary A. Johnson and Michael R. Taaffe. A Graphical Investigation of Error Bounds for Moment Based Queueing Approximations. *Queueing Systems*, 8 :295–312, 1991.

[Kir94] Matthias Kirschnick. The Performance Evaluation and Prediction System for Queueing Networks (PEPSY-QNS). Technical Report TR-I4-18-94, Department of Computer Science, University of Erlangen, Germany, 1994.

[Kle64] Leonard Kleinrock. A Delay Dependent Queue Discipline. *Naval Research Logistics Quarterly*, 11(4), December 1964.

[Kle75] Leonard Kleinrock. *Queueing Systems*, volume I. John Wiley, New York, 1975.

[KLS+01] Vikram Kanodia, Chengzhi Li, Ashutosh Sabharwal, Bahareh Sadeghi, and Edward Knightly. Distributed Multi-Hop Scheduling and Medium Access with Delay and Throughput Constraints. *Mobile Computing and Networking*, August 2001.

[Kno00] William J. Knottenbelt. *Parallel Performance Analysis of Large Markov Models*. PhD thesis, Imperial College London, United Kingdom, February 2000.

[KNP02] Marta Kwiatkowska, Gethin Norman, and David Parker. Probabilistic Symbolic Model Checking with PRISM : A Hybrid Approach. In *Proceedings of the 8th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'02)*, pages 52–66, London, UK, 2002. Springer-Verlag.

[Koz83] Dexter Kozen. Results on the propositional $\mu$-calculus. *Theoretical computer Science*, 27 :333–345, 1983.

[KS60] John G. Kemeny and J.Laurie Snell. *Finite Markov Chains*. Van Nostrand, Princeton, NJ, 1960.

[KSC91] Manolis Katevenis, Stefanos Sidiropoulos, and Costas Courcoubetis. Weighted Round-Robin Cell Multiplexing in a General-Purpose ATM Switch Chip. *IEEE Journal on Selected Areas in Communications*, 9(8) :1265–1279, Oct 1991.

[KSH03] Rachid El Abdouni Khayari, Ramin Sadre, and Boudewijn Haverkort. Fitting World-Wide Web Request Traces with the EM-algorithm. *Perform. Eval.*, 52(2-3) :175–191, 2003.

[LAS03] Anders Lindgren, Andreas Almquist, and Olov Schelén. Quality of Service Schemes for IEEE 802.11 Wireless LANs – An Evaluation. *Mobile Networks and Applications*, 8(3) :223–235, June 2003.

[LAZC00] Seoung-Bum Lee, Gahng Seop Ahn, Xiaowei Zhang, and Andrew Campbell. INSIGNIA : An IP-Based Quality of Service Framework for Mobile ad Hoc Networks. *Journal on Parallel and Distributed Computing*, 60(4) :374–406, April 2000.

[Lin98] Christoph Lindemann. *Performance Modelling with Deterministic and Stochastic Petri Nets*. John Wiley and Sons, 1998.

[Lin01] Chunhung Richard Lin. On-Demand QoS Routing in Multihop Mobile Networks. In *Proceedings INFOCOM*, 2001.

[LLP+01] Roy Leung, Jilei Liu, Edmond Poon, Ah-Lot Charles Chan, and Baochun Li. MP-DSR : A QoS-Aware Multi-Path Dynamic Source Routing Protocol for Wireless Ad-Hoc Networks. In *26th Annual IEEE International Conference on Local Computer Networks (LCN'01)*, 2001.

[LLY01] Matthew Leung, John Lui, and David Yau. Adaptive Proportional Delay Differentiated Services Characterization and Performance Evaluation. *IEEE/ACM Transactions on Networking*, 9(6) :801–817, December 2001.

[LT94] Patrick Lascaux and Raymond Théodor. *Analyse numérique matricielle appliquée à l'art de l'ingénieur*. Masson, 1994.

[LTP95] Dimitris Logothetis, Kishor S. Trivedi, and Antonio Puliafito. Markov Regenerative Models. In *IEEE Internation Computer Parallel Distributed Systems*, pages 134–142, Erlangen, Germany, April 1995.

[Mar80] Raymond Marie. Calculating Equilibrium Probabilities for $\lambda(n)/C_k/1/N$ Queues. In *Proceedings of the 1980 International Symposium on Computer Performance Modelling, Measurement and Evaluation*, pages 117–125, New York, NY, USA, 1980. ACM Press.

[Mat62] Klaus Matthes. Zur theorie der bedienungsprozesse. *Transaction $3^{rd}$ Prague Conference on Information Theory, Stat. Dec. Fns. and Random Processes*, pages 513–528, 1962.

[MBC84] Marco Ajmone Marsan, Gianfranco Balbo, and Gianni Conte. A Class of Generalized Stochastic Petri Nets for the Performance Analysis of Multiprocessor Systems. *ACM Transaction Computer Systems*, 2(1), May 1984.

[MBC+95] Marco Ajmone Marsan, Gianfranco Balbo, G. Conte, Susanna Donatelli, and G. Franceschinis. *Modelling with Generalized Stochastic Petri Nets*. John Wiley & Sons, 1995.

[MBD98] Marco Ajmone Marsan, Andrea Bobbio, and Susanna Donnatelli. Petri Nets in Performance Analysis : An Introduction. *Lectures Notes in Computer Science*, 1998.

[MC87] Marco Ajmone Marsan and Giovanni Chiola. On Petri Nets with Deterministic and Exponentially Distributed Firing Times. *Lecture Notes in Computer Science, Springer-Verlag*, 266 :132–145, 1987.

[MCM+02] Stefan Mangold, Sunghyun Choi, Peter May, Ole Klein, Guido Hiertz, and Lothar Stibor. IEEE 802.11e Wireless LAN for Quality of Service. In *Proceedings of European Wireless 2002 (EW2002)*, pages 32–39, Florence, Italie, February 2002.

[Med02] Jyotiprasad Medhi. *Stochastic Models in Queueing Theory*. Academic Press, New York, NY, USA, 2nd edition, November 2002.

[Mil89] Robin Milner. *Communication and Concurrency*. Prentice Hall, 1989.

[Mit82] Isi Mitrani. *Simulation Techniques for Discrete Event Systems*. Cambridge University Press, 1982.

[MN02] Jim Martin and Arne Nilsson. On Service Level Agreements for IP Networks. In *proceedings of the IEEE INFOCOM'2002*, volume 2, pages 855–863, June 2002.

[MNTB04] Mohammad Malli, Qiang Ni, Thierry Turletti, and Chadi Barakat. Adaptive Fair Channel Allocation for QoS Enhancement in IEEE 802.11 Wireless LANs. In *IEEE International Conference on Communications (ICC'04)*, pages 3470–3475, Paris, France, June 2004.

[MST01] Mohammad Mirhakkak, Nancy Shult, and Duncan Thomsom. Dynamic Bandwidth Management and Adaptive Applications for a Variable Bandwidth Wireless Environment. *IEEE JSAC*, 19(10), October 2001.

[NABT03] Qiang Ni, Imad Aad, Chadi Barakat, and Thierry Turletti. Modeling and Analysis of Slow CW Decrease for IEEE 802.11 WLAN. In *Proceedings of the fourteenth IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC 2003)*, pages 1717–1721, Beijing, Chine, Septembre 2003.

[NAS04] Ibrahima NIANG, Hossam AFIFI, and Dominique SERET. Couplage de services différenciés pour une qos de bout en bout. In *Proceedings of the Colloque Africain sur la recherche en Informatique*, Hamamet, Tunisie, Novembre 2004.

[NAT04] Qiang Ni, Imad Aad, and Thierry Turletti. A Survey of QoS Enhancements for IEEE 802.11 Wireless LAN. *Journal of Wireless Communications and Mobile Computing*, 4(5) :1–20, 2004.

[NBBB98] Kathleen Nichols, Steven Blake, Fred Baker, and David Black. Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers. RFC 2474, December 1998.

[NBMR02] Navid Nikaein, Christian Bonnet, Yan Moret, and Idris A. Rai. Two–Layered Quality of Service Model for Reactive Routing Protocols for Mobile Ad Hoc Networks. In *Proceedings of the Sixth World Multiconference on Systemics, Cybernetics and Informatics*, Orlando, Floride, USA, July 2002.

[Neu75] Marcel F. Neuts. Probability Distributions of Phase Type. In *Liber Amicorum Professor Emeritus H. Florin*, pages 173–206, University of Louvain, Belgium, 1975.

[Neu81] Marcel F. Neuts. *Matrix Geometric Solutions in Stochastic Models*. The Johns Hopkins University Press, 1981.

[Nik00] Maria Nikolskaia. *Binary Decision Diagram and Applications to Reliability Analysis*. PhD thesis, LaBRI, Université Bordeaux I, Janvier 2000.

[OHB03a] Takayuki Osogami and Mor Harchol-Balter. A Closed Form Solution for Mapping General Distributions to Minimal PH Distributions. In *International Conference on Performance Tools – TOOLS 2003*, pages 200–217, Urbana, IL, USA, February 2003. Springer : LNCS 2794.

[OHB03b] Takayuki Osogami and Mor Harchol-Balter. Necessary and Sufficient Conditions for Representing General Distributions by Coxians. In *International Conference on Performance Tools – TOOLS 2003*, pages 182–199, Urbana, IL, USA, Sept 2003. Springer : LNCS 2794.

[OOS03] Philippe Olivier, Philippe Owezarski, and Kavé Salamatian. Quelques éléments caractéristiques du trafic internet. Colloque International Mesures de l'Internet, mai 2003.

[Oso05] Takayuki Osogami. *Analysis of Multi-Server Systems via Dimensionality Reduction of Markov Chains*. PhD thesis, Carnegie Mellon University, Pittsburgh, June 2005.

[OTL04] Richard G. Ogier, Fred L. Templin, and Mark G. Lewis. Topology Dissemination Based on Reverse-Path Forwarding (TBRPF). RFC 3684, February 2004.

[PA91] Brigitte Plateau and Karim Atif. Stochastic Automata Networks for Modeling Parallel Systems. *IEEE Transactions on Software Engineering*, 17(10), October 1991.

[PB94] Charles E. Perkins and Pravin Bhagwat. Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers. In *ACM SIGCOMM '94 : Proceedings of the conference on Communications architectures, protocols and applications*, pages 234–244, New York, NY, USA, 1994. ACM Press.

[PC97] Vincent D. Park and M. Scott Corson. A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks. In *the Proceedings of the Sixteenth Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'97)*, volume 3, pages 1405–1413, April 1997.

[Pla84] Brigitte Plateau. *De l'evaluation du parallelisme et de la synchronisation*. PhD thesis, Université de Paris-Sud, Centre d'Orsay, France, November 1984.

[Plo81] Gordon D. Plotkin. A Structural Approach to Operational Semantics. Technical report, Computer Science Department, Aarhus University, September 1981.

[Pnu77] Amir Pnueli. The Temporal Logic of Programs. In *Proceedings of the 18th Annual Symposium on Foundations of Computer Science (FOCS'77)*, pages 46–57. IEEE Comp. Soc. Press, Octobre 1977.

[PRD01] Charles E. Perkins, Elizabeth M. Royer, and Samir R. Das. Quality of Service for Ad hoc On-Demand Distance Vector Routing. Internet Draft draft-ietf-manetaodvqos- 00.txt, July 2001.

[PRD03] Charles E. Perkins, Elizabeth M. Royer, and Samir Das. Ad Hoc On-Demand Distance Vector (AODV) Routing. RFC 3561, July 2003.

[PST98] Antonio Puliafito, Marco Scarpa, and Kishor S. Trivedi. Petri Nets with $k$ Simultaneously Enabled Generally Distributed Timed Transitions. *Perfromance Evaluation*, 32(1) :1–34, 1998.

[QS02] Daji Qiao and Kang Shin. Achieving Efficient Channel Utilization and Weighted Fairness for Data Communications in IEEE 802.11 WLAN under the DCF. In *Proceedings of the Tenth International Workshop on Quality of Service (IWQoS)*, pages 227–36, 2002.

[RDS04] Alma Riska, Vesselin Diev, and Evgenia Smirni. An EM-based Technique for Approximating Long-Tailed data sets with PH Distributions. *Performance Evaluation*, 55 :147–164, 2004.

[RFB01] K. K. Ramakrishnan, Sally Floyd, and David Black. The Addition of Explicit Congestion Notification (ECN) to IP. RFC 3168, Sept 2001.

[Rib95] Marina Ribaudo. On the Aggregation Techniques in Stochastic Petri Nets and Stochastic Process Algebras. In S. Gilmore and J. Hillston, editors, *Proceedings of the Third International Workshop on Process Algebras and Performance Modelling*, pages 600–611, December 1995.

[RNT03] Lamia Romdhani, Qiang Ni, and Thierry Turletti. Adaptive EDCF : Enhanced Service Differentiation for IEEE 802.11 Wireless Ad-Hoc Networks. In *Proceedings of the IEEE Wireless Communications and Networking Conference*, New Orleans, USA, March 2003.

[RR04] Jeffrey Robinson and Tejinder Randhawa. Saturation Throughput Analysis of IEEE 802.11e Enhanced Distributed Coordination Function. *IEEE Journal on Selected Areas in Communications*, 22(5) :917–928, June 2004.

[Saa96] Yousef Saad. *Iterative Methods for Sparse Linear Systems*. PWS Publishing Company, Boston, 1996.

[SB03a] Osman Salem and Abdelmalek Benzekri. A Software Performance Evaluation Approach Using Stochastic Process Algebra Tools. In *Proceedings of the International Conference on Software Engineering Research and Practice*, volume 1, pages 140 – 145, Las Vegas, Nevada, USA, June 2003.

[SB03b] Osman Salem and Abdelmalek Benzekri. Functional Modelling and Performance Evaluation for Two Class DiffServ Router using Stochastic Process Algebra. In *Proceedings of the 17th European Simulation Multiconference*, pages 257–262, Nottingham – UK, June 2003.

[SB03c] Osman Salem and Abdelmalek Benzekri. Modélisation et évaluation de performances en exploitant l'algèbre des processus stochastiques. Actes du $8^{ime}$ Atelier d'évaluation de performances (AEP8), Mai 2003.

[SB04a] Osman Salem and Abdelmalek Benzekri. An Algebraic Model of an Adaptive Extension of DiffServ for MANET. In *Network Control and Engineering for QoS, Security and Mobility (IFIP'04)*, pages 221–234, Palma de Mallorca, Nov 2004.

[SB04b] Osman Salem and Abdelmalek Benzekri. Modelling and Analyzing Dynamic Source Routing Protocol with General Distributions. In *proc. of the 11th International Confernence on Analytical and Stochastic Modelling Techniques and Applications*, pages 173–179, Magdeburg, Germany, June 2004.

[SB06a] Osman Salem and Abdelmalek Benzekri. Towards Absolute End-to-End QoS in Ad Hoc Networks. In *Proceedings of IEEE Advanced International Conference on Telecommunications (AICT'06)*, Gouadeloupe, French Caribbean, February 2006.

[SB06b] Osman Salem and Abdelmalek Benzekri1. Towards End-to-End QoS in Ad Hoc Networks. In *Third Annual Conference on Wireless On-demand Network Systems and Services (IFIP WONS 2006)*, Les Ménuires, France, Jan 2006.

[SC75] Charles Sauer and K. Mani Chandy. Approximate Analysis of Central Server Models. *IBM Journal of Research and Development*, 19 :301–313, 1975.

[SCFJ03] Henning Schulzrinne, Stephen Casner, Ron Frederick, and Van Jacobson. RTP : A Transport Protocol for Real-Time Applications. RFC 3550, July 2003.

[SH00] Ben Strulo and Peter Harrison. Spades – A Process Algebra for Discrete Event Simulation. *Logic Computation*, 10(1) :3–42, 2000.

[Sha04] Srikant Sharma. Analysis of 802.11b MAC : A QoS, Fairness, and Performance Perspective. *ArXiv Computer Science e-prints*, November 2004.

[SK96] João L. Sobrinho and A. S. Krishnakumar. Real-Time Traffic over the IEEE 802.11 Medium Access Control Layer. *Bell Labs Technical Journal*, 1(2) :172–187, 1996.

[SM91] Ryan Stansifer and Dan Marinescu. Petri Net Models of Concurrent Ada Programs. *Microelectronics and Reliability*, 31(4) :577–594, 1991.

[SOQW95] William Sanders, W.Douglas Oball, Muhammad Akber Qureshi, and F. K. Widjanarko. The UltraSAN Modeling Environment. *Perform. Eval.*, 24(1) :89–115, 1995.

[SPG97] Scott Shenker, Craig Partridge, and Roch Guerin. Specification of Guaranteed Quality of Service. RFC 2212, September 1997.

[SSB99] Prasun Sinha, Raghupathy Sivakumar, and Vaduvur Bharghavan. CEDAR : A Core Extraction Distributed Ad hoc Routing Algorithm. *IEEE Journal on Selected Areas in Communications*, 17(8) :1454–1465, August 1999.

[ST05] Theodoros Salonidis and Leandros Tassiulas. Distributed Dynamic Scheduling for End-to-End Rate Guarantees in Wireless Ad Hoc Networks. In *Proceedings of ACM MOBIHOC 2005*, 2005.

[Ste94] William J. Stewart. *Introduction to the Numerical Solution of the Markov Chains*. Princeton University Press, 1994.

[Ste95] William J. Stewart. *Computation with Markov Chains*. Kluwer, Boston, 1995.

[Str93] Ben Strulo. *Process Algebra for Discrete Event Simulation*. PhD thesis, Department of Computing, Imperial College, University of London, 1993.

[TBT05] Axel Thümmler, Peter Buchholz, and Miklós Telek. A Novel Approach for Fitting Probability Distributions to Real Trace Data with the EM Algorithm. In *Proceedings of the International Conference on Dependable Systems and Networks (DSN'05)*, pages 712–721, Washington, DC, USA, 2005. IEEE Computer Society.

[TG98] Nigel Thomas and Stephen Gilmore. Applying Quasi-Separability to Markovian Process Algebra. In Corrado Priami, editor, *Proceedings of the Sixth Annual Workshop on Process Algebras and Performance Modelling*, Nice, France, sep 1998. Universitá Degli Studi di Verona.

[TH02a] Miklós Telek and Armin Heindl. Matching Moments for Acyclic Discrete and Continuous Phase-Type Distributions of Second Order. *International Journal of Simulation Systems, Sciences & Technology*, 3(3–4) :47–57, 2002.

[TH02b] Miklòs Telek and Armin Heindl. Moment Bounds for Acyclic Discrete and Continuous Phase Type Distributions of Second Order. In *Proceedings of UK Performance Evaluation Workshop, UKPEW*, 2002.

[VBG00] Nitin Vaidya, Paramvir Bahl, and Seema Gupta. Distributed Fair Scheduling in a Wireless LAN. In *Proceedings of the sixth annual international conference on Mobile Computing and Networking (MobiCom 2000)*, pages 167–178, Aug 2000.

[VBO98] Véronique Vèque and Jalel Ben-Othman. MRAP : A Multiservices Resource Allocation Policy for Wireless ATM Network. *Computer Networks and ISDN Systems*, 29(17–18) :2187–2200, 1998.

[VL02] Vladimir Vishnevsky and Andrey Lyakhov. IEEE 802.11 Wireless LAN : Saturation Throughput Analysis with Seizing Effect Consideration. *Cluster Computing*, 5(2) :133–144, April 2002.

[VP85] Michel VERAN and Dominique POTIER. QNAP 2 : A Portable Environment for Queueing System Modelling. In Dominique POTIER, editor, *Proceedings of the Interantion Conference on Modelling Techniques and Tools for Computer Performance Evaluation*, pages 25–63, North-Holland, 1985.

[WG99] IEEE 802.11 WG. Wireless LAN Media Access Control (MAC) and Physical Layer (PHY) Specifications. IEEE Standard 802.11, 1999.

[WG05] IEEE 802.11 WG. Draft Supplement to Part 11 : Wireless Medium Access Control (MAC) and Physical Layer (PHY) specifications : Medium Access Control (MAC) Enhancements for Quality of Service (QoS). IEEE Std 802.11e/D13.0, Jan 2005.

[WH01] Kui Wu and Janelle Harms. QoS Support in Mobile Ad Hoc Networks. *Crossing Boundaries – the GSA Journal of University of Alberta*, 1(1) :92–106, November 2001.

[Whi82] Ward Whitt. Approximating a Point Process by a Renewal Process : Two Basic Methods. *Operations Research*, 30 :125–147, 1982.

[Wol89] Ronald Wolff. *Stochastic Modeling and the Theory of Queues*. Prentice Hall, 1989.

[WPL+02] Haitao Wu, Yong Peng, Keping Long, Shiduan Cheng, and Jian Ma. Performance of Reliable Transport Protocol over IEEE 802.11 Wireless LAN : Analysis and Enhancement. In *Proc. IEEE INFOCOM*, New York, USA, June 2002.

[Wro97] John Wroclawski. Specification of the Controlled–Load Network Element Service. RFC 2211, September 1997.

[WZLX05] Junfeng Wang, Hongxia Zhou, Lei Li, and Fanjiang Xu. Accurate Long-tailed Network Traffic Approximation and its Queueing Analysis by Hyper-Erlang Distributions. In *Proceedings of the IEEE Conference on Local Computer Networks*, pages 148–155, Nov 2005.

[XG03] Qi Xue and Aura Ganz. Ad Hoc QoS On-demand Routing (AQOR) in Mobile Ad Hoc Networks. *Journal of Parallel and Distributed Computing*, 63(2) :154–165, February 2003.

[Xia04] Yang Xiao. Performance Analysis of IEEE 802.11e EDCF under Saturation Conditions. In *Proceedings of ICC04*, Paris, France, June 2004.

[XSA03] Jianbo Xue, Patrick Stuedi, and Gustavo Alonso. ASAP : An adaptive QoS protocol for Mobile Ad Hoc Networks. In *IEEE International Symposium on Personal Indoor and Mobile Radio Communications (PIMRC 2003)*, pages 2616–2620, Sept 2003.

[XSLC00] Hannan Xiao, Winston Seah, Anthony Lo, and Kee Chaing Chua. A Flexible Quality of Service Model for Mobile Ad Hoc Networks. In *Proceedings of the fifty-first IEEE Vehicular Technology Conference*, pages 445–449, Tokyo, Japon, May 2000.

[YK04] Yaling Yang and Robin Kravets. Distributed QoS guarantees for realtime traffic in ad hoc networks. In *Conference on Sensor and Ad Hoc Communications and Networks*, pages 118–127, Oct 2004.

[ZC02] Chenxi Zhu and Scott Corson. QoS Routing for Mobile Ad Hoc Networks. In *Proceedings of the Twenty-first Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE Infocom)*, New York, USA, June 2002.

[ZFGH00] Armin Zimmermann, Jörn Freiheit, Reinhard German, and Günter Hommel. Petri Net Modelling and Performability Evaluation with TimeNET 3.0. In *Proceedings of the* $11^{th}$ *International Conference on Computer Performance Evaluation : Modelling Techniques and Tools (TOOLS '2000)*, pages 188–202, London, UK, 2000. Springer-Verlag.

[ZFH01] Armin Zimmermann, Jörn Freiheit, and Günter Hommel. Discrete Time Stochastic Petri Nets for Modeling and Evaluation of Real-Time Systems. In *Int. Workshop on Parallel and Distributed Real-Time Systems*, San Francisco, April 2001.