N° d'ordre : 06/2007 - D/IN

REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITÉ DES SCIENCES ET DE LA TECHNOLOGIE
HOUARI BOUMEDIENE
FACULTÉ D'ELECTRONIQUE ET D'INFORMATIQUE

THÈSE

Présentée pour l'obtention du diplôme de

DOCTORAT

En : Informatique
Spécialité : Informatique

Par : **Nawel GHARBI BOUHAL**

Sujet

Évaluation des Performances et de la Fiabilité des Systèmes Multi-classes avec Rappel à l'aide des Réseaux de Petri Stochastiques Colorés

Soutenue publiquement le 08/07/2007, devant le jury composé de :

Mr	Amar AISSANI	Professeur à l'USTHB	Président
Mme	Malika IOUALALEN	Professeur à l'USTHB	Dtrice de Thèse
Mr	Djamil AISSANI	Professeur à l'Université de Bejaia	Examinateur
Mr	Kamel BARKAOUI	Professeur au CNAM - Paris	Examinateur
Mr	Kamel BOUKHETALA	Professeur à l'USTHB	Examinateur
Mr	Jean-Michel ILIÉ	M.C à l'Université P&M Curie	Examinateur

Man Do What He Can,
But God What He Will

À mes parents et ma grand-mère
À mon époux
À Nazim et à Liza

Remerciements

Je tiens à remercier Monsieur le Professeur Amar AISSANI pour l'honneur qu'il me fait en présidant le jury de ma thèse, et aussi pour le rôle qu'il a joué dans mon orientation vers le domaine des files d'attente avec rappel.

J'adresse mes sincères remerciements à Messieurs les Professeurs Djamil AISSANI, Kamel BARKAOUI et Kamel BOUKHATALA, d'avoir en dépit d'une intense activité, bien accepté de participer au jury de soutenance.

Merci à Monsieur Jean-Michel ILIÉ d'avoir bien voulu accepté de lire ce travail, pour ses discutions pertinentes et sa participation au jury. Profitons de la circonstance pour lui témoigner toute la gratitude qui lui revient, pour sa fructueuse collaboration.

Je voudrais également exprimer à Madame le Professeur Malika IOUALALEN BOUKALA tous mes remerciements et ma reconnaissance d'avoir dirigé cette thèse et encadré mes travaux pendant toutes ces années. Ses suggestions pertinentes et ses idées fructueuses ont permis de compléter ma recherche de manière significative et efficace.

Que Madame Claude DUTHEILLET LAMONTHEZIE soit aussi remerciée pour le chaleureux accueil qu'elle m'a prodigué au laboratoire LIP6, et surtout pour la contribution active qu'elle a apporté dans l'accomplissement de mes travaux.

Enfin, je remercie tous les membres de ma famille pour leur soutien moral et leur aide précieuse. Plus particulièrement ma grand-mère, mes parents et mon époux qui ont en plus fait preuve de beaucoup de patience.

Table des matières

Table des figures

Liste des tableaux

Introduction Générale

Les systèmes avec rappel (ou systèmes avec appels répétés) apparaissent dans beaucoup de domaines tels que les réseaux téléphoniques, les réseaux informatiques et les télécommunications. Ces systèmes sont caractérisés par le fait que les clients qui trouvent tous les serveurs occupés ou non disponibles à leur arrivée, doivent quitter immédiatement l'espace de service et rappeler ultérieurement, dans un ordre aléatoire, indépendamment les uns des autres et à des intervalles de temps aléatoires.

La conception de ces systèmes nécessite par sa complexité et ses implications économiques, des outils d'aide qui permettent la modélisation et l'évaluation des performances du système, en vue de vérifier sa correction, d'optimiser l'utilisation de ses ressources et d'augmenter sa fiabilité. À cet effet, un modèle appelé *modèle des files d'attente avec rappel : FAR* (Retrial Queues) a été introduit pour étudier les situations d'appels répétés.

La première contribution sérieuse sur le sujet a été publiée en 1957 par W.J. Cohen [1] quand les chercheurs ont découvert que le modèle de file d'attente traditionnel et le modèle d'Erlang avec perte, étaient inadéquats pour expliquer les comportements stochastiques des systèmes téléphoniques dans lesquels les abonnés répètent leurs appels dès la réception du signal occupé. C'est ainsi, que le modèle des files d'attente avec rappel, qui occupe une situation intermédiaire entre le modèle d'Erlang et le modèle de file d'attente classique, a vu le jour.

Durant ces deux dernières décennies, un effort considérable a été consacré par beaucoup de praticiens et de chercheurs théoriciens, à l'évaluation des performances des systèmes avec rappel, et plus précisément aux files d'attente avec rappel, qui est le modèle conventionnel habituellement appliqué pour l'analyse des performances de ces systèmes. L'intérêt porté à ce thème est principalement expliqué par les développements et les avancées technologiques notamment dans les domaines de l'informatique et des télécommunications, conduisant à l'utilisation de nouvelles facilités telle que : répéter-dernier-numéro (repeat-last-number), auto-recomposition (auto-repeat) et sonner-quand-libre (ring-back-when-free) [2]. Ces systèmes continuent à attirer l'attention de beaucoup de chercheurs du fait de la possibilité de leurs applications dans des domaines importants tels que les réseaux LAN (Local Area Network) avec le protocole de communication CSMA (Carrier-Sense Multiple Access), les réseaux informatiques où plusieurs terminaux font des rappels pour recevoir un service d'un processeur central, les réseaux de télécommunications où les messages sont retransmis après une tentative sans succès pour accéder au canal de communication, ainsi que les réseaux mobiles cellulaires et les systèmes aéronautiques. D'ailleurs, l'intérêt croissant du thème des FAR est reflété par la publication, durant ces dernières dé-

cennies, d'un nombre important de papiers. Pour une synthèse des principaux résultats et des progrès réalisés dans ce domaine, le lecteur peut consulter [3, 4, 5, 6, 7, 8, 9, 10, 11, 12] et les références qu'ils contiennent.

Cependant, la prise en compte du phénomène d'appels répétés a engendré beaucoup de problèmes théoriques. En fait, la considération d'un second flux d'appels répétés de l'orbite qui est superposé au flux des arrivées primaires, rend les méthodes classiques et les résultats de la théorie des files d'attente standards inadéquats, et introduit ainsi de grandes difficultés pour l'obtention de résultats analytiques dans le domaine des FAR. Ainsi, les chercheurs se sont concentrés principalement sur le développement d'algorithmes numériques et des méthodes d'approximation et de simulation. En fait, les résultats analytiques détaillés existent pour certaines files d'attente avec rappel particulières, avec des hypothèses contraignantes sur certains paramètres, tel que le nombre de serveurs, la taille de la population, la fiabilité des serveurs, l'homogénéité des clients et des serveurs, etc., alors que pour beaucoup d'autres systèmes, les résultats obtenus restent très limités.

Les files d'attente multi-serveurs avec rappel, peuvent être vues comme des processus de quasi-naissance-et-mort dépendants du niveau (level dependent quasi-birth-and-death) [13]. La principale caractéristique de leur générateur infinitésimal est l'hétérogénéité spatiale causée par les transitions associées aux appels répétés [5]. Cette absence d'homogénéité explique la difficulté analytique des FAR multi-serveurs. C'est la raison pour laquelle il est très difficile, si ce n'est impossible, de déduire des formules analytiques exactes pour les probabilités stationnaires et les autres caractéristiques de performance. Par conséquent, les études analytiques des modèles multi-serveurs sont limitées et dans beaucoup de cas restreintes à des systèmes à *un ou deux serveurs au plus*. Pour remédier au problème, des solutions sont obtenues par des méthodes d'approximation et des modèles tronqués [14].

D'autre part, à cause de la structure compliquée des processus stochastiques utilisés habituellement pour la modélisation des systèmes avec rappel, la plupart des travaux considèrent des modèles avec une source (population) infinie de clients et un flux poissonnien des arrivées primaires. En réalité, cette hypothèse a principalement l'avantage de permettre une description mathématique plus simple des problèmes de rappel [1]. Cependant, dans beaucoup de situations pratiques, il est important de prendre en compte le fait que le taux de génération des nouveaux appels primaires décroît quand le nombre de clients dans le système croît. Ceci peut se faire en considérant des modèles ayant une source finie de clients, ou bien des modèles à entrée quasi-aléatoire (quasi-random input models) [5, 11, 12, 15, 16]. En effet, dans un système téléphonique, par exemple, la taille de la population qui correspond au nombre d'abonnés, peut être importante mais elle est réellement limitée.

Les modèles avec rappel et source finie apparaissent dans la modélisation et l'analyse des performances des systèmes à disque-mémoire magnétique (magnetic disk-memory systems) [17], des réseaux mobiles cellulaires [18, 19], des réseaux à commutation de circuit avec une architecture hybride fibre-coaxial (hybrid fiber-coax systems) [20] et des réseaux LAN (Local Area Networks) avec le protocole CSMA/CD [21], avec une topologie en étoile [22, 23, 15], avec des protocoles à accès aléatoire [24] et avec des protocoles à accès multiple [25].

Par ailleurs, en pratique, certains composants des systèmes sont sujet à des pannes aléatoires, suite auxquelles le service est interrompu pour une durée de temps aléatoire. Il est donc d'une importance basique d'étudier les performances et la fiabilité des systèmes avec rappel, où les serveurs sont sujets à des pannes et des réparations aléatoires, et ce à cause de la forte influence des pannes, qui peuvent avoir un impact non négligeable sur les mesures de performance du système. On parle alors de systèmes avec rappel et serveurs non-fiables. En fait, l'analyse de la fiabilité de ces systèmes s'avère indispensable du fait que de plus en plus d'applications importantes dans la téléphonie, les banques et les compagnies aériennes par exemple, nécessitent une qualité de service particulière même en présence des pannes.

Cependant, malgré que l'étude de la fiabilité est d'une grande importance, les travaux qui prennent en considération le phénomène de rappel des clients avec la non-fiabilité des serveurs restent assez limités et dans la plupart des cas supposent que la source est infinie et la station de service comprend un serveur unique [26, 27, 28, 29]. En ce qui concerne les modèles avec rappel, source finie et serveur non-fiable, les travaux sont plus rares [30, 31, 32], particulièrement pour les modèles multi-serveurs [33, 34], et les résultats sont obtenus par des méthodes numériques.

Vu l'importance pratique des modèles avec rappel, à source finie de clients et serveurs non-fiables, nous nous intéressons dans cette thèse à l'étude de la combinaison du phénomène de rappel, source finie, multiplicité et non-fiabilité des serveurs, ce qui rend le système plutôt compliqué. La présence de ces différentes caractéristiques dans un même système, introduit de multiple problèmes de synchronisation, en plus des phénomènes de blocage liés aux appels répétés.

Par opposition aux modèles de files d'attente, les réseaux de Petri stochastiques généralisés (RdPSG) [35] constituent un important modèle graphique et mathématique, qui est très adapté à la description des systèmes parallèles en présence des contraintes de synchronisation et des mécanismes de blocage. En effet, ces systèmes avec rappel sont des systèmes stochastiques à événements discrets, caractérisés par des phénomènes de blocage et de synchronisation, et dans lesquels il n'y a aucun mécanisme d'ordre entre les appels primaires et celui des appels répétés. Par conséquent, la puissance d'expression du modèle des RdPSG, qui permet une modélisation très détaillée et sémantiquement précise, constitue une des principales raisons de son application pour les systèmes avec rappel. Ainsi, nous proposons dans cette thèse, une approche de modélisation et d'évaluation des performances et de la fiabilité des systèmes multi-serveurs avec rappel et source finie en se basant sur les réseaux de Petri stochastiques généralisés. Nous considérons dans un premier lieu le cas de serveurs fiables ensuite les systèmes avec pannes. Ceci nous permet d'une part une description simple, détaillée et efficace de ces systèmes, et aussi l'analyse des propriétés qualitatives et quantitatives, en appliquant les outils et les techniques très avancés développés dans le domaine des RdPSG.

D'autre part, la plupart des files d'attente avec rappel étudiées supposent que le flux d'entrée est homogène du point de vue des caractéristiques des clients, telles que les distributions des temps d'inter-arrivée, des temps de service et des temps de rappel. Cependant,

en pratique, ces caractéristiques peuvent varier pour les différents types de clients. Ceci nous conduit aux *systèmes multi-classes avec rappel*. Ces systèmes apparaissent dans divers domaines d'applications, tels que les réseaux de télécommunications, les systèmes à commutation téléphonique et les réseaux mobiles cellulaires.

Les modèles multi-classes sont beaucoup plus difficiles à analyser mathématiquement que les modèles à classe unique. Ainsi, les résultats explicites sont disponibles seulement dans quelques cas particuliers [36, 37, 38, 4, 39] avec une population infinie et une station mono-serveur. Par ailleurs, certains travaux récents ont porté sur des FAR multi-classes à source finie, où chaque classe est limitée à un seul client et la station de service comprend un serveur unique [40, 41, 42, 32] ou plusieurs serveurs identiques [43].

D'autre part, les modèles de FAR avec serveurs hétérogènes sont très peu étudiés, malgré leur importance dans les systèmes réels. D'ailleurs, on ne trouve dans la littérature que quelques références [44, 33, 34], dans lesquelles les clients sont supposés être homogènes et les serveurs sont hétérogènes. Pour ce qui est des modèles avec rappel, clients et serveurs hétérogènes à la fois, aucune étude n'a été faite à ce jour, et ce ni pour le cas de serveurs fiables ni serveurs non-fiables. Ceci est dû essentiellement à la complexité de ces modèles, dont l'analyse s'est toujours basée sur la théorie des files d'attente. En effet, les formalismes des réseaux de Petri n'ont jamais été appliqués dans ce domaine.

Ainsi, nous proposons dans la deuxième partie de cette thèse, une approche de modélisation et d'évaluation des performances et de la fiabilité des systèmes multi-classes avec rappel et source finie, à l'aide du modèle des réseaux de Petri stochastiques généralisés colorés (RdPSGC) [45, 46]. Nous considérons les systèmes avec plusieurs classes de clients et aussi plusieurs classes de serveurs, où chaque classe a ses propres caractéristiques stochastiques et comprend un nombre arbitraire de clients (ou de serveurs).

L'avantage de l'utilisation des RdPSG colorés est le fait qu'ils constituent un modèle graphique et mathématique de haut-niveau, approprié pour la description et l'analyse des performances des systèmes avec des composants hétérogènes. D'autre part, ce modèle coloré permet, en plus de la représentation explicite des choix conditionnels, de la synchronisation et du blocage, l'intégration des contraintes provenant de l'identification et de la particularisation éventuelle des différents clients et serveurs. Ainsi, nous montrons à travers cette thèse la simplicité avec laquelle ces systèmes multi-classes assez compliqués peuvent être décrits et analysés grâce à ce modèle stochastique coloré, dans lequel chaque jeton correspondant à un client ou à un serveur doit porter une information qui dénote son type.

Les RdPSG colorés ont une très forte puissance d'expression. En effet, la possibilité d'associer une information aux jetons et de paramétrer le franchissement de transitions, permet la représentation des systèmes complexes d'une manière très concise et détaillée, ce qui aurait nécessité des réseaux non-colorés beaucoup plus volumineux et probablement illisibles. En effet, les RdPSG colorés fournissent une abréviation des RdPSG en regroupant certaines caractéristiques des systèmes à l'intérieur d'objets de plus haut niveau.

L'approche proposée dans cette thèse, qui consiste en l'utilisation des modèles de haut

niveaux pour l'analyse des systèmes avec rappel, présente plusieurs avantages. D'un point de vue description, elle permet essentiellement une facilité de modélisation et d'expression des propriétés, en plus de la représentation des contraintes de synchronisation et de blocage. D'un point de vue performance, contrairement aux modèles de files d'attente avec rappel, les RdPSG et les RdPSGC permettent une analyse des propriétés qualitatives du système modélisé et offrent aussi un riche moyen d'expression des indices de performance et de fiabilité exacts en régime stationnaire. Ces indices s'expriment en fonction des éléments de base du réseau (places, transitions, marquages) et des probabilités stationnaires. Par conséquent, ces modèles nous permettent de faire une **analyse numérique exacte** des systèmes avec rappel et source finie, en se basant sur les techniques et les outils définis dans ce domaine. En fait, ces techniques sont très attrayantes car elles offrent une approche d'évaluation des performances basée sur une description formelle. Ceci permet l'utilisation du même modèle pour la spécification, la validation et l'évaluation des performances.

Nous nous intéressons particulièrement à l'étude du comportement stationnaire, car ce qui intéresse le modélisateur, dans de nombreuses applications, est le comportement du système une fois qu'il se stabilise. Ceci suppose bien entendu qu'un tel comportement (état d'équilibre) existe, ce qui peut être facilement vérifié pour les modèles de RdPSG et RdPSGC, en se basant simplement sur certaines propriétés qualitatives.

Motivation pratique de l'étude

Depuis ces deux dernières décennies, les besoins en téléphonie et en télécommunications ont augmenté et les exigences des usagers de ce secteur en terme de qualité, coût et délai ne cessent de s'accroître, surtout que le phénomène d'appels répétés a un effet négatif non-négligeable sur les indices de performance. Pour cela, des efforts considérables sont réalisés afin d'améliorer la qualité des systèmes avec rappel, ce qui nécessite par conséquent, une évolution permanente.

Cette évolution est le fait de différentes motivations : on cherchera à titre d'exemple à augmenter la rapidité d'un système en faisant coopérer des machines indépendantes. On cherchera également à optimiser l'utilisation des diverses ressources. Sur un autre plan, on pourra augmenter la fiabilité ou la tolérance aux pannes d'un système en faisant fonctionner en parallèle plusieurs sous-systèmes identiques. Un autre point essentiel, est le fait que dans les réseaux modernes à grande rapidité d'établissement des appels, la durée moyenne des intervalles de répétition est courte, et par conséquent, les abonnés peuvent rappeler rapidement. Ceci entraîne la saturation des équipements communs de commande, ainsi les abonnés auront l'impression que le système offre une très mauvaise qualité de service.

Toutes ces évolutions rendent la gestion de ces systèmes de plus en plus complexe, et face aux besoins des usagers exprimés en terme de délai ou de temps de réponse, la nécessité de maîtriser ces systèmes devient évidente.

Par ailleurs, l'évaluation des performances des systèmes a de nos jours, de plus en plus d'importance. Il devient en effet inconcevable de construire un système quelconque (que ce

soit un système informatique, un réseau de communication ou autre), sans avoir auparavant fait l'analyse des performances. La pression des enjeux économiques est actuellement telle que l'on ne peut aboutir à un système sous-dimensionné et que l'on doit éviter au maximum le surdimensionnement. Construire le système adapté, en respectant le plus possible les objectifs du cahier des charges, est une démarche qui passera obligatoirement par une étape de modélisation et d'analyse des performances.

En fait, l'évaluation des performances des systèmes avec rappel peut intervenir à deux niveaux : en conception ou en exploitation.

En conception : cela signifie que le système n'existe pas encore et qu'il s'agit de le créer. Par conséquent, il est clair qu'on ne peut pas effectuer de mesures dessus.
Dans la pratique, les ingénieurs devront concevoir un système avec rappel en respectant un cahier des charges. Lorsqu'un système est simple, l'experience du concepteur peut être suffisante. Cependant, la plupart des systèmes réels actuels sont de plus en plus complexes. L'expérience de l'expert, si compétent soit-il n'est alors plus suffisante. Il y a en effet beaucoup trop de paramètres à prendre en compte. Il faut pourtant être capable, pour une configuration donnée du système, de calculer les paramètres de performance, afin de vérifier qu'ils sont bien en accord avec le cahier des charges. Concevoir un système sans avoir mené au préalable d'analyse des performances, peut aboutir à la création d'un système inutilisable, car sous-dimensionné et ne respectant pas les objectifs initiaux. À l'inverse, on peut aboutir à un système surdimensionné et pour lequel on aura gaspillé inutilement de l'argent.

En exploitation : cette fois-ci, le système avec rappel existe, mais on souhaite le modifier ou le tester en dehors de son point de fonctionnement normal. Ainsi, il s'agit de concevoir un système différent répondant à de nouveaux objectifs. L'ingénieur par exemple, propose de changer telle machine pour la remplacer par une machine deux fois plus rapide. Il faudra alors analyser les performances de ce nouveau système, pour vérifier que ce changement permet d'atteindre l'augmentation désirée. Si ce n'est pas le cas, cela signifie peut-être que ce n'était pas cette machine qui ralentissait le fonctionnement globale du système. Son changement aurait donc été inutile, ou en tout cas insuffisant, vis-à-vis de l'objectif fixé. Il peut être également intéressant de tester le système avec rappel dans des conditions anormales de fonctionnement telles que des pannes de machines ou des surcharges de rappels. C'est encore un cas où la mesure est impossible ou en tout cas non envisageable.

L'évaluation des performances d'un système réel peut être schématisée de la façon présentée dans la Figure 1.

Ce schéma se décompose en une étape de *modélisation* permettant de passer du système au modèle et une étape d'*analyse des performances* du modèle. Le rebouclage n'a lieu que si les performances obtenus ne sont pas celles espérées.

Les files d'attente avec rappel ont été traditionnellement utilisées pour la conception et l'analyse des systèmes avec rappel. Ce modèle est plutôt complexe pour le développement direct de processus stochastiques, lorsqu'il s'agit de systèmes compliqués. Ainsi, leur étude à l'aide d'un formalisme de description plus sophistiqué et puissant serait pratique.

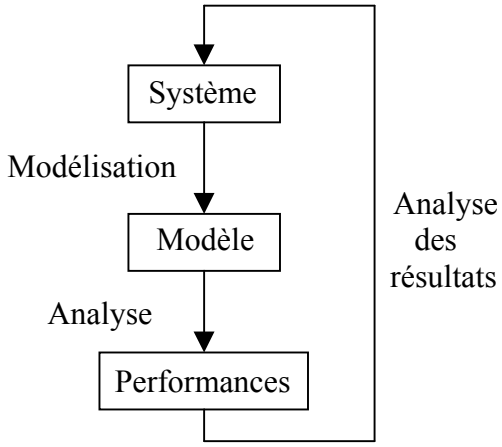

FIG. 1 – Schéma d'évaluation des performances d'un système

Les réseaux de Petri stochastiques généralisés (ordinaires ou colorés), représentent des formalismes de haut-niveau permettant la modélisation et l'analyse des systèmes parallèles dont les comportements sont caractérisés par les phénomènes de synchronisation, de blocage et de concurrence.

Ainsi, l'intérêt de la démarche présentée dans cette thèse, réside essentiellement dans la proposition d'un modèle d'analyse des systèmes avec phénomènes d'appels répétés, autre que le modèle conventionnel des files d'attente avec rappel, et donc d'adapter les méthodes existantes pour ce modèle, en l'occurrence le modèle des RdPSG ordinaires ou colorés, en vue de l'obtention de résultats exacts, pour des systèmes complexes avec appels répétés.

Organisation du document

Dans cette thèse, nous commençons tout d'abord, par la présentation du modèle des files d'attente avec rappel. Nous exposons dans ce cadre un état de l'art des méthodes et des principaux résultats obtenus pour les modèles multi-serveurs. Nous discutons des difficultés liées à la prise en compte du flux des appels répétés et nous présentons les principaux travaux et résultats des modèles multi-serveurs à source finie. Nous étudions dans ce cadre, les modèles avec serveurs fiables, serveur(s) non fiable(s), serveurs hétérogènes ainsi que les FAR avec clients hétérogènes.

Le deuxième chapitre débute par la description du modèle des réseaux de Petri simples qui constituent la base de tous les réseaux de haut-niveau. Nous présentons les principaux concepts du modèle et nous couvrirons principalement l'étude des propriétés qualitatives qui permettent d'analyser le fonctionnement du système modélisé. Dans un deuxième temps, nous développons les points clés des chaînes de Markov. Nous définissons ensuite le modèle des réseaux de Petri stochastiques généralisés et le modèle des réseaux de Petri stochastiques généralisés colorés. Nous nous intéressons particulièrement à l'analyse des performances à l'aide de ces modèles. Avant de conclure ce chapitre, nous abordons une

étude comparative entre les réseaux de Petri stochastiques généralisés et d'autres modèles d'évaluation des performances, en l'occurrence les chaînes de Markov et les files d'attente.

Le but du troisième chapitre est la présentation d'une approche de modélisation et d'analyse des performances et de la fiabilité des systèmes multi-serveurs avec rappel et source finie à l'aide des réseaux de Petri stochastiques généralisés. Nous commençons par l'étude des systèmes avec serveurs fiables. Nous enchaînons ensuite par les systèmes avec serveurs non-fiables. Nous proposons dans ce cadre, une définition d'une discipline de panne plus générale que celles étudiées dans la littérature. Nous donnons le modèle correspondant à chaque discipline de panne et nous déduisons ensuite les formules des indices de performance et de fiabilité correspondants, permettant l'obtention de résultats numériques exacts.

Dans le chapitre suivant, nous présentons une extension de l'approche proposée, aux systèmes multi-classes avec rappel et source finie. Nous considérons dans ce cadre, les systèmes avec plusieurs classes de clients et de serveurs qui sont supposés dans un premier temps fiables. Ensuite, nous étendons l'étude au cas non-fiable en considérant les différentes politiques de panne et disciplines de service. Pour cela, la modélisation est basée sur le formalisme des réseaux de Petri stochastiques généralisés colorés. À partir de là, les principaux indices de performance et de fiabilité stationnaires sont obtenus.

Dans le dernier chapitre, nous exposerons quelques expérimentations relatives aux systèmes avec rappel mono-classes (homogènes), puis celles des systèmes multi-classes (hétérogènes), en utilisant l'outil software GreatSPN 2.0.2. Ceci nous permettra d'une part, de valider les modèles proposés, et d'autre part, d'étudier l'effet du phénomène de rappel, des politiques de panne et des disciplines de service sur les performances des systèmes.

Enfin, une conclusion générale résumera les différents points abordés à travers cette thèse, et donnera quelques perspectives intéressantes de notre travail.

Chapitre 1

État de l'Art

1.1 Introduction

La théorie des files d'attente est une théorie mathématique relevant du domaine des probabilités, qui étudie les solutions optimales de gestion des systèmes ou structures à files d'attente. Celles-ci se rencontrent en permanence dans la vie courante, par exemple, dans un guichet dont le titulaire effectue un service, dans un atelier de fabrication, dans des structures de gestion des feux lumineux d'un réseau routier, dans une centrale téléphonique, etc.

Ce domaine de recherche est né en 1917, des travaux de l'ingénieur Erlang, portant sur l'analyse statistique des réseaux téléphoniques [47, 48]. Ces travaux marquèrent les développements ultérieurs dans le domaine des probabilités modernes et des processus stochastiques, et furent étendus par la suite pour l'analyse des performances et la planification des systèmes de production [49, 50], des systèmes informatiques et des réseaux de télécommunications [51].

Cependant, la théorie classique des files d'attente donne deux principales méthodes pour résoudre le conflit qui se produit lorsqu'un client qui arrive dans le système, trouve le(s) serveur(s) occupé(s) ou non disponible(s) :

- Le client peut quitter définitivement le système sans être servi, ceci correspond au *modèle d'Erlang avec refus* (Erlang loss system) appelé aussi *modèle à appels perdus*, symbolisé par la fameuse formule de perte d'Erlang [52],
- ou bien, le client peut attendre, en file d'attente, pour être servi après la libération du serveur, selon une certaine discipline de service (FIFO, LIFO, ...) [53, 54].

Une situation intermédiaire envisage la possibilité pour un client qui trouve le (tous les) serveur(s) occupé(s) de rappeler ultérieurement pour le service, autant de fois que nécessaire et à des intervalles de temps distribués selon une certaine loi de probabilité, jusqu'à ce qu'il trouve un serveur libre et que son service puisse commencer. Entre ces appels répétés ou rappels, le client est dit : *en orbite*. Ces systèmes sont appelés : *systèmes avec rappel* ou bien *systèmes avec appels répétés*.

Ainsi, les systèmes avec rappel sont caractérisés par le fait que les clients (appels) qui trouvent tous les serveurs occupés ou non disponibles rejoignent l'orbite pour rappeler

ultérieurement dans un ordre aléatoire et à des intervalles de temps aléatoires.

L'orbite ou encore le pool des sources d'appels répétés, peut être vu comme une sorte de file d'attente (ou buffer), où les clients bloqués se comportent indépendamment les uns des autres et font des rappels jusqu'à ce qu'ils soient servis, puis quittent le système.

Les systèmes avec rappel apparaissent dans beaucoup d'applications concrètes, essentiellement dans les domaines de la téléphonie, des télécommunications et des réseaux informatiques. Nous citons comme exemples typiques, les systèmes téléphoniques avec les fonctions *auto-recomposition*, *recomposer-dernier-numéro* et *sonner-quand-libre* décrites dans [2], les réseaux LAN (Local Area Networks) avec le protocole de communication CSMA (Carrier Sense Multiple Access), dans lequel les messages sont retransmis après une tentative sans succès pour accéder au bus de transmission et ce après un temps aléatoire, ou encore les réseaux informatiques où plusieurs terminaux font des rappels pour recevoir un service d'un processeur central.

La conception de ces systèmes comme tous les autres systèmes modernes, nécessite par sa complexité et ses implications économiques, des outils d'aide qui permettent la modélisation et l'évaluation des performances du système, en vue de vérifier sa correction, d'optimiser l'utilisation de ses ressources et d'augmenter sa fiabilité. À cet effet, un modèle appelé *modèle des files d'attente avec rappel* (FAR) ou encore *modèle des files d'attente avec appels répétés* (Retrial Queues) a été introduit pour étudier les situations d'appels répétés.

Ce modèle analytique permet de décrire le fonctionnement d'un système avec rappel en termes mathématiques, et d'évaluer ensuite ses paramètres de performance. Une étude détaillée nous a montré que le modèle des files d'attente avec rappel est le modèle conventionnel habituellement utilisé pour l'analyse des performances des systèmes avec rappel.

L'étude des files d'attente avec rappel remonte à la fin des années quarante, où des chercheurs tel que Kosten [55], Wilkinson [56] et Cohen [1] ont découvert que les modèles d'attente traditionnels et les modèles de perte d'Erlang, ne permettaient pas de modéliser le service des abonnés dans une centrale téléphonique, ni d'expliquer le comportement stochastique des systèmes de télécommunications, où les abonnés répètent leurs appels en recomposant le numéro dès la réception du signal occupé. À vrai dire, un abonné qui reçoit le signal *occupé* ne peut être inséré dans une file d'attente classique pour attendre la terminaison de la communication en cours, mais il va plutôt rappeler pour tenter sa chance après un certain temps aléatoire et indépendamment des autres abonnés.

Depuis les premiers travaux, la théorie des files d'attente avec rappel a été largement utilisée dans diverses applications tels que les ateliers de production [57, 58, 59, 60, 61], les réseaux informatiques [62], les réseaux mobiles cellulaires [63, 18, 64, 19], les systèmes de télécommunications [1, 65, 66, 67, 68, 69, 70], ainsi que les réseaux à commutation téléphonique. Ce modèle peut également modéliser le service des avions à l'atterrissage dans un aéroport, d'où l'origine de l'expression : *entrer en orbite*.

Cependant, la prise en considération d'un second flux d'appels répétés rend les mé-

thodes classiques et les résultats de la théorie des files d'attente standards inadéquats, et introduit ainsi de grandes difficultés pour l'obtention de résultats analytiques dans le domaine des FAR qui est assez complexe. Cette difficulté est due essentiellement à la compétition entre deux flux indépendants : un flux des appels primaires et un second flux d'appels répétés qui a une structure plus complexe.

En fait, des résultats analytiques détaillés existent pour certaines files d'attente avec rappel particulières, avec des hypothèses contraignantes sur certains paramètres, tel que le nombre de serveurs, leur fiabilité, l'homogénéité des clients, etc., alors que pour beaucoup d'autres systèmes, les résultats sont obtenus par des méthodes (algorithmes) numériques, des méthodes d'approximation et de simulation. Par conséquent, beaucoup de travaux et de résultats restent à développer dans le domaine des files d'attente avec rappel.

Pour une synthèse détaillée de la littérature, des applications, des méthodes fondamentales et des principaux résultats obtenus dans ce domaine, le lecteur est référé au livre de Falin et Templeton [5], ainsi qu'aux papiers synthèse de Aissani [71], Artalejo [11, 10], Falin [4, 12], Kulkarni [9] et Yang et Templeton [3]. D'autre part, pour une riche bibliographie classifiée et accessible de ce domaine, voir Artalejo [7, 8] et les références qu'ils contiennent.

Notre objectif majeur dans ce chapitre est d'exposer au lecteur une synthèse des méthodes et des principaux résultats obtenus dans le domaine des files d'attente multi-serveurs avec rappel. Pour cela, nous commençons par la description du modèle. Nous discutons aussi les ressemblances et les différences entre les files d'attente standards et les files d'attente avec rappel. Nous nous intéressons aux modèles de FAR multi-serveurs markoviens. Nous présentons dans ce cadre, les principales méthodes d'analyse par approximation définies pour le modèle à source infinie, ce qui nous permettra de discuter les difficultés liées à la prise en compte du flux des appels répétés. Dans les sections suivantes, nous donnons un état de l'art basé sur une littérature assez récente, des principaux travaux et résultats des modèles de FAR multi-serveurs à source finie. Nous étudions les modèles avec serveur(s) non fiable(s), avec serveurs hétérogènes ainsi que les FAR avec clients hétérogènes. En fait, il ne s'agit pas ici de présenter un éventail exhaustif de toutes les FAR étudiées à ce jour, mais seulement des modèles ayant rapport avec notre travail. Enfin, nous donnons une conclusion dans laquelle nous mentionnons les problèmes ouverts de ces variantes.

1.2 Description du modèle des files d'attente avec rappel

Un système de file d'attente [53, 54, 72] est un système dans lequel les clients (modélisant les activités qui ont besoin d'accéder aux ressources), arrivent suivant une loi probabiliste, pour recevoir un service auprès d'une station de service (modélisant les ressources). Ainsi, une file d'attente standard est composée d'un buffer et d'une station qui peut comprendre un ou plusieurs serveurs parallèles.

À l'arrivée d'un client dans le système, il vérifie si un serveur est disponible. Dans ce cas, il entre en service et conserve la ressource pendant toute la durée du traitement.

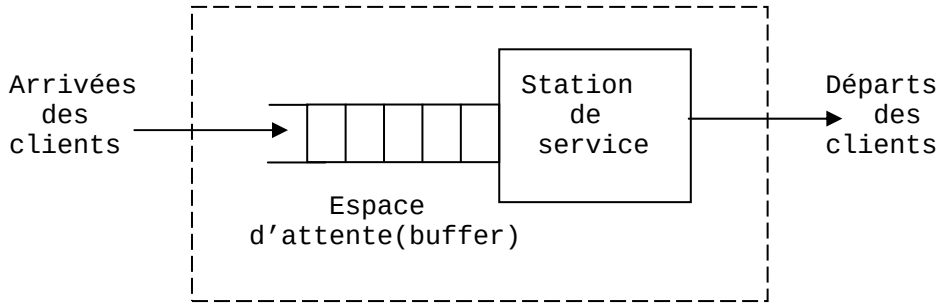

FIG. 1.1 – Représentation graphique d'une file d'attente standard

À la fin de son service, le client libère le serveur qui devient alors disponible, et quitte immédiatement le système. Si par contre, tous les serveurs sont occupés, le client devra attendre dans un espace d'attente dit *buffer*, jusqu'à ce qu'un serveur soit disponible. Dans ce cas, le client à servir sera sélectionné selon une certaine discipline de service (FIFO, LIFO, random ou autre).

Une représentation graphique du modèle de file d'attente classique est donnée dans la figure 1.1.

Cependant, les modèles de files d'attente standards ne prennent pas en compte le phénomène des appels répétés, et par conséquent ne peuvent pas être appliqués pour résoudre un nombre de problèmes importants pratiques. D'ailleurs, Kosten dans [73], a précisé que : *tout résultat théorique qui ne prend pas en considération cet effet de répétition, devrait être suspecté*. Ainsi, les files d'attente avec rappel (FAR) ont été introduites pour résoudre cette imperfection.

1.2.1 La structure générale des files d'attente avec rappel (FAR)

Une file d'attente avec rappel est composée d'une *station de service* qui comprend s ($s \geq 1$) serveurs parallèles et indépendants et d'une *orbite* ou pool de rappel qui représente un espace d'attente imaginaire, pouvant être à capacité finie ou infinie.

À l'arrivée d'un client à la station de service, s'il y a un ou plusieurs serveurs disponibles, le client sera servi immédiatement (par un seul serveur) et quittera le système dès la fin du service. Par contre, si tous les serveurs sont occupés, le *client bloqué* sera obligé de quitter la station de service pour entrer en orbite, et devient ainsi une *source d'appels répétés* ou un *client en orbite*, qui rappellera pour le service à des intervalles de temps suivant une loi de probabilité, jusqu'à ce qu'un serveur soit libre. Chacun de ces clients de l'orbite est traité comme un *client primaire*, c'est à dire un nouveau client qui arrive de l'extérieur du système. S'il trouve un serveur libre, il sera servi immédiatement puis quittera le système. Autrement, si tous les serveurs sont encore occupés, le client revient une autre fois en orbite, et ce sans aucune influence sur le processus de service.

L'intervalle de temps entre deux tentatives consécutives faites par un même client de l'orbite est dit : *temps de rappel*. Ce temps est indépendant de tous les temps de rappel

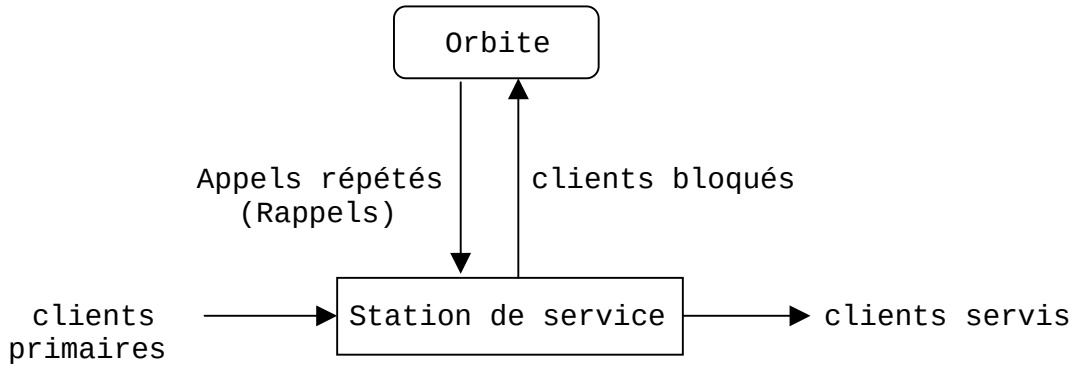

FIG. 1.2 – Structure générale d'une file d'attente avec rappel

précédents.

Ainsi, dans ce modèle, les clients qui ne peuvent pas être immédiatement pris en charge, ne seront pas servis en FIFO ou en LIFO, mais rappelleront plutôt pour le service chacun indépendamment des autres et de manière aléatoire. On peut donc considérer qu'ils passent leur temps d'attente en orbite avant de rappeler. Par conséquent, le modèle des files d'attente avec rappel occupe une situation intermédiaire entre le modèle d'Erlang avec refus [52] et le modèle des files d'attente classiques, qui en constituent les modèles limites dans les cas de faible et forte intensité de rappel respectivement.

La structure générale d'une file d'attente avec rappel est schématisée dans la figure 1.2.

1.2.2 Les caractéristiques des files d'attente avec rappel

Un système de file d'attente avec rappel est caractérisé par :
- Le mécanisme d'arrivée des clients dans le système ;
- Le mécanisme de service ;
- Le mécanisme de rappel ;
- Le nombre de serveurs ayant des caractéristiques statistiques identiques ou pas ;
- La capacité maximale de l'orbite ;
- La capacité de la source (population) de clients.

Remarquons que le processus décrit par ce modèle est un processus stochastique, car on ne peut connaître à l'avance ni le temps d'arrivée d'un client, ni le temps de rappel, ni la durée de service qu'il demandera.

Lorsque la station est formée d'un seul serveur, le modèle est dit *mono-serveur*. Mais lorsque la station est formée de s serveurs parallèles ($s \geq 2$), le modèle est dit *multi-serveurs*.

D'autre part, on distingue principalement deux types de systèmes de files d'attente avec rappel :

- **Les systèmes ouverts (source infinie) :** Ils sont alimentés par une population infinie ou une source disposant d'un nombre infini d'unités. Ainsi, le nombre des arrivées est illimité. Comme exemple, nous citons les programmes soumis à un ordinateur.
- **Les systèmes fermés (source finie) :** Ils sont plutôt alimentés par un nombre maximum d'unités fixé, correspondant par exemple, au nombre d'abonnés dans un réseau téléphonique.

Système de notation : [3, 11]
En se basant sur la notation de Kendall, un modèle de file d'attente avec rappel est noté comme suit : $A/B/s/L/K$ où :

- A décrit la distribution des temps des interarrivées des clients ;
- B décrit la distribution du temps de service de chaque client ;
- s désigne le nombre de serveurs dans la station de service ;
- L est la capacité du système (station de service + taille de l'orbite) ;
- K est la taille de la source (ou la population) de clients.

La loi des interarrivées ou celle du service peut être : exponentielle (markovienne notée M), déterministe (D), générale (G), loi d'Erlang d'ordre k (E_k), géométrique (Geo) ou autre. Par contre, la distribution des temps de rappel est supposée généralement exponentielle de taux ν : autrement dit, la durée moyenne des intervalles de rappel est de $1/\nu$. C'est la raison pour laquelle elle est omise de la notation. Cependant, il est important de préciser que plusieurs travaux récents considèrent des modèles avec une loi de rappel non-exponentielle.

D'autre part, quand la capacité du système L ou la source de clients K sont infinies, elles sont omises de la notation.

Exemple : La notation $M/M/3//10$ représente une file d'attente avec rappel markovienne à 3 serveurs, orbite infinie et une source limitée à 10 clients.

En fait, le modèle de file d'attente avec rappel décrit ci-dessus est un modèle général. Plusieurs variantes ont été définies dans la littérature pour la modélisation et l'analyse de systèmes particuliers. Nous trouvons entre autres : le modèle avec orbite et buffer, le modèle avec clients non-persistants, le modèle à serveurs hétérogènes, etc.

Nous nous intéressons particulièrement dans cette thèse aux systèmes purement **markoviens** avec un processus d'arrivée, une distribution des temps de service et des temps de rappel exponentiels. Ces modèles sont dits : *modèles markoviens de files d'attente avec rappel*. En fait, la loi exponentielle convient à la modélisation d'évènements dont la distribution temporelle n'est pas connue [35]. D'autre part, nous supposons que les clients sont absolument **persistants**. Ceci signifie que tout client bloqué doit rappeler autant de fois que nécessaire jusqu'à ce qu'il soit servi, contrairement aux modèles de FAR avec clients impatients (non-persistants), dans lesquelles un client, après un certain nombre de rappels

sans succès, pourrait abandonner le service et donc quitter le système définitivement sans être servi.

1.3 Les FAR multi-serveurs

Une des FAR les plus étudiées dans la littérature est la FAR multi-serveurs $M/M/s$, dans laquelle les clients arrivent suivant un processus de Poisson à une station de service composée de s serveurs identiques, indépendants et exponentiels, et où les clients bloqués peuvent rappeler après une durée de temps exponentielle.

Les premiers articles ayant traité ce modèle remontent à plusieurs décennies [55, 56, 1, 65, 74]. Pendant ces dernières années, ce modèle continua à attirer l'attention de beaucoup de chercheurs à cause de ses diverses applications dans des domaines importants caractérisés par le phénomène des appels répétés. Cependant, il est important de préciser que les études analytiques des files d'attente multi-serveurs avec rappel et population infinie, sont à ce jour, limitées et dans beaucoup de cas restreintes aux systèmes à deux serveurs [75, 76, 77, 78, 79, 80]. En effet, nous constatons l'absence de formules explicites pour les caractéristiques de performance principales, telle que la distribution stationnaire, la probabilité de blocage et le nombre moyen de clients dans le système, quand le nombre de serveurs est supérieur à deux [14].

En fait, une étude exhaustive des FAR avec un nombre de serveurs arbitraire, constitue un problème compliqué qui implique des difficultés analytiques associées à la non-homogénéité de l'espace d'états, due aux appels répétés. En d'autres termes, le manque de résultats analytiques est dû à la structure compliquée du processus stochastique qui décrit le comportement de ce modèle.

Quelques travaux théoriques intéressants [1, 81] ont été consacrés à la résolution analytique du modèle avec un nombre arbitraire de serveurs. Cependant, la distribution stationnaire de l'état du système est exprimée soit en fonction d'intégrales de contour [1], ou bien comme des limites de fractions continues étendues [81]. D'un point de vue analytique, les deux solutions représentent des tentatives significatives, mais elles ne sont pas destinées à des usages pratiques [14].

Ainsi, à cause de l'absence de méthodes d'analyse analytique des modèles multi-serveurs avec rappel, les chercheurs se sont concentrés principalement sur le développement d'algorithmes numériques et des méthodes d'approximation [82, 83, 84, 5, 14, 85].

1.3.1 Le formalisme mathématique

Pour comprendre où la difficulté se situe, nous considérons dans cette section, un système de FAR multi-serveurs $M/M/s$ avec une source de clients et une orbite de tailles infinies. Dans ce modèle, les clients primaires arrivent suivant un processus de Poisson de taux $\lambda > 0$. La station de service comprend s serveurs identiques et parallèles. Les temps de service des clients sont indépendants et distribués exponentiellement avec le paramètre $\mu > 0$ et les temps de rappel suivent une distribution exponentielle négative de

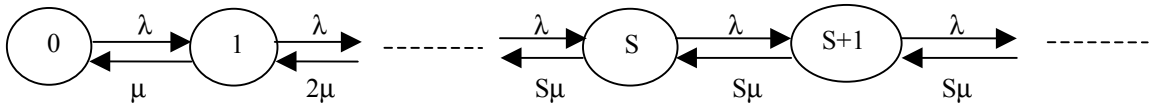

FIG. 1.3 – Évolution d'une file d'attente $M/M/s$ standard

taux $\nu > 0$. Comme d'habitude, les périodes d'inter-arrivées, les temps de service et les temps de rappel sont mutuellement indépendants.

État du système : [4, 14, 10]
L'état du système peut être décrit par un processus à deux variables $X = \{(C(t), N(t)) ; t \geq 0\}$, où $C(t)$ est le nombre de serveurs occupés et $N(t)$ est le nombre de clients en orbite (source d'appels répétés) à l'instant t.
Sous les hypothèses citées ci-dessus, le processus X est une chaîne de Markov à temps continu homogène avec l'espace d'états dénombrable infini $E = \{0, ..., s\} \times \mathbb{N}$.

Contrairement au processus décrivant la file d'attente $M/M/s$ standard, présenté dans la figure 1.3, $X(t)$ n'est pas un processus de naissance et de mort. D'ailleurs, la difficulté de l'obtention de résultats analytiques pour les modèles de files d'attente avec rappel multi-serveurs est liée au fait que les processus sous-jacents à ces modèles ne soient pas des processus de naissance et de mort.

Définition : Un processus de naissance et de mort (birth-and-death process) est un cas particulier des chaînes de Markov homogènes à temps continu, dont les taux de transition sont de la forme :

$$\lambda_{i,j} = \begin{cases} \lambda_i & \text{si } j = i+1, \\ \mu_i & \text{si } j = i-1, \\ 0 & \text{si } |i-j| \geq 2. \end{cases}$$

Le processus décrivant la file d'attente $M/M/s$ standard est un processus de naissance et de mort, qui passe à l'arrivée d'un client, de l'état i à l'état $i+1$ avec un taux $\lambda_i = \lambda$, et à la fin d'une période de service d'un client, de l'état i à l'état $i-1$ avec un taux $\mu_i = Min(i, s).\mu$.

Cependant, en prenant en compte le flux des appels répétés, le processus correspondant devient beaucoup plus compliqué.

Les probabilités de transition d'état dans un intervalle de temps infiniment petit dt sont résumées dans le tableau suivant :

Transition d'état	Probabilité de transition d'état	Description de l'évènement
$(i,j) \rightarrow (i+1,j)$ $0 \leq i \leq s-1$, $j \geq 0$	$\lambda\ dt + o(dt)$	Arrivée d'un appel primaire et son début de service
$(s,j) \rightarrow (s,j+1)$ $j \geq 0$	$\lambda\ dt + o(dt)$	Arrivée d'un appel primaire et son entrée en orbite
$(i,j) \rightarrow (i+1,j-1)$ $0 \leq i \leq s-1$, $j \geq 1$	$j\nu\ dt + o(dt)$	Arrivée d'un appel répété et son début de service
$(i,j) \rightarrow (i-1,j)$ $1 \leq i \leq s$, $j \geq 0$	$i\mu\ dt + o(dt)$	Fin de service d'un client

Les taux de transition $q_{(i,j)(m,n)}$ du processus X sont donnés comme suit : [14]

pour $0 \leq i \leq s-1$:

$$q_{(i,j)(m,n)} = \begin{cases} \lambda, & \text{si } (m,n) = (i+1,j), \\ i\mu, & \text{si } (m,n) = (i-1,j), \\ j\nu, & \text{si } (m,n) = (i+1,j-1), \\ -(\lambda + i\mu + j\nu), & \text{si } (m,n) = (i,j), \\ 0, & \text{sinon.} \end{cases}$$

et pour $i = s$:

$$q_{(s,j)(m,n)} = \begin{cases} \lambda, & \text{si } (m,n) = (s,j+1), \\ s\mu, & \text{si } (m,n) = (s-1,j), \\ -(\lambda + s\mu), & \text{si } (m,n) = (s,j), \\ 0, & \text{sinon.} \end{cases}$$

En triant les états tel que $E = \{(0,0),...,(s,0),(0,1),...,(s,1),...\}$, le générateur infinitésimal Q du processus X peut être exprimé sous la forme d'une matrice à blocs, définie comme suit : [10]

$$Q = \begin{pmatrix} A_0^{(0)} & A_0^{(+1)} & 0 & 0 & ... \\ A_1^{(-1)} & A_1^{(0)} & A_1^{(+1)} & 0 & ... \\ 0 & A_2^{(-1)} & A_2^{(0)} & A_2^{(+1)} & ... \\ 0 & 0 & A_3^{(-1)} & A_3^{(0)} & ... \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix}$$

où $A_j^{(-1)}$, $A_j^{(0)}$ et $A_j^{(+1)}$ sont les matrices carrées $(s+1) \times (s+1)$ suivantes :

$$A_j^{(-1)} = \begin{pmatrix} 0 & j\nu & 0 & \dots & 0 \\ 0 & 0 & j\nu & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & \dots & \dots & j\nu \\ 0 & \dots & \dots & \dots & 0 \end{pmatrix}$$

$$A_j^{(+1)} = \begin{pmatrix} 0 & \dots & \dots & 0 & 0 \\ 0 & \dots & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & \dots & 0 & 0 \\ 0 & \dots & \dots & 0 & \lambda \end{pmatrix}$$

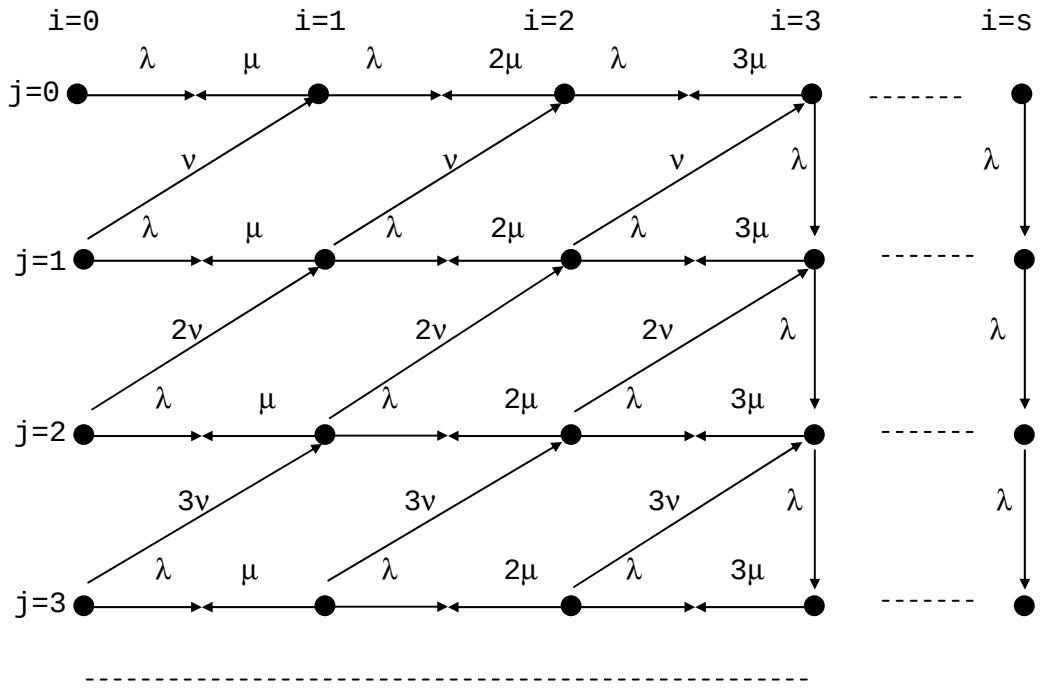

FIG. 1.4 – Espace d'états et transitions du processus X

$$A_j^{(0)} = \begin{pmatrix} -(\lambda + j\nu) & \lambda & 0 & 0 & \dots & 0 \\ \mu & -(\lambda + \mu + j\nu) & \lambda & 0 & \dots & 0 \\ 0 & 2\mu & -(\lambda + 2\mu + j\nu) & \lambda & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & \dots & \dots & s\mu & -(\lambda + s\mu) \end{pmatrix}$$

Le comportement stochastique du processus X peut être représenté géométriquement par le diagramme de transition donné dans la figure 1.4

En examinant ce processus X, nous constatons qu'il est infini, ce qui est logique car l'espace d'états correspond à un produit d'un ensemble fini $\{0, ..., s\}$ avec l'ensemble des entiers positifs $\mathbb{N}$. Par ailleurs, il est important de préciser que deux théories importantes ont été introduites par Neuts [86] et Malyshev [87] concernant les chemins aléatoires définis sur un tel espace. L'hypothèse principale de ces théories est la condition suivante de *l'homogénéité spatiale* [10].

$$A_j^{(k)} \equiv A^{(k)}, \ si \ j \geq j^*$$

pour tout k et un certain entier positif j^*.

Cette hypothèse permet une analyse mathématique étendue du comportement transitoire et stationnaire de divers processus, tel que la FAR $M/M/1$ et la file d'attente classique sans rappel $M/M/s$. Ainsi, la condition d'application de ces théories est essentiellement l'homogénéité spatiale du processus, et ce indépendamment du fait que l'espace

d'états soit fini ou infini.

Contrairement à ces processus, la caractéristique principale du générateur infinitésimal des FAR multi-serveurs $M/M/s$ avec politique de rappel classique, est l'hétérogénéité spatiale due aux appels répétés. En effet, les transitions entre les états (i, j) dépendent de la seconde coordonnée j, qui correspond au nombre de clients en orbite. Les principales difficultés analytiques et la plupart des propriétés intéressantes des FAR sont en rapport avec cette caractéristique de non-homogénéité [10].

Pour montrer la nature des difficultés avec plus de détails, nous présenterons plus loin le problème le plus simple qui correspond au calcul de la distribution stationnaire $P = (p_{ij})_{(i,j)\in E}$ du processus X.

Condition d'érgodicité : Un problème important dans l'étude des files d'attente avec rappel est l'obtention des conditions nécessaires et suffisantes pour l'existence d'un régime stationnaire de leur fonctionnement (ergodicité). Pour les FAR $M/M/s$, ce problème a été examiné par plusieurs chercheurs tel que Cohen [1], Deul [88] et Falin [89, 90]. L'idée est que l'état stationnaire est atteint si et seulement si le nombre moyen de serveurs occupés à l'état d'équilibre, qui est égal à λ/μ, est inférieur au nombre total de serveurs disponibles.

Ainsi, le processus X est ergodique (récurrent positif) si et seulement si l'*intensité du traffic* $\rho = \lambda/s\mu < 1$ [90, 5].

Distribution stationnaire : Dans le cas où l'état stationnaire est atteint ($\lambda < s\mu$), les probabilités limites $p_{ij} = \lim_{t\to\infty} P\{C(t) = i,\ N(t) = j\}$ existent $\forall (i, j) \in E$ et sont positives (et uniques). Par conséquent, les indices de performance du modèle à l'état d'équilibre peuvent être obtenus.

Le calcul de cette distribution stationnaire $P = (p_{i,j})_{(i,j)\in E}$ du processus X (où i est le nombre de serveurs occupés et j le nombre de clients en orbite), peut se faire en appliquant le principe des flux qui stipule : la somme des flux entrants est égale à la somme des flux sortants de chaque état de la chaîne de Markov. Ainsi, nous pouvons obtenir le système d'équations suivant, dont les inconnues correspondent aux probabilités d'état : [76, 91]

$$\begin{cases} (\lambda + i\mu + j\nu)p_{i,j} = \lambda p_{i-1,j} + (j+1)\nu p_{i-1,j+1} + (i+1)\mu p_{i+1,j}, & 0 \le i < s,\ j \ge 0, \\ (\lambda + s\mu)p_{s,j} = \lambda p_{s-1,j} + (j+1)\nu p_{s-1,j+1} + \lambda p_{s,j-1}, & j \ge 0. \end{cases} \tag{1.1}$$

où $p_{-1,j} = p_{i,-1} = 0\ \forall i$ et j.

D'autre part, le calcul des probabilités stationnaires se fait habituellement à l'aide des équations de Kolmogorov $PQ = 0$. En essayant d'exploiter le fait que le générateur infinitésimal Q soit bien structuré, et en divisant le vecteur des probabilités stationnaires P tel que : $P = (p_0, p_1, ...)$ où $p_j = (p_{oj}, ..., p_{sj})$, nous pouvons écrire les équations de Kolmogorov sous la forme matricielle suivante : [10]

$$p_{j-1}A_{j-1}^{(+1)} + p_j A_j^{(0)} + p_{j+1}A_{j+1}^{(-1)} = 0,\ j = 0, 1, ... \tag{1.2}$$

où p_{-1} et $A_{-1}^{(+1)}$ sont égaux à zéro.

Une autre alternative consiste à introduire les fonctions génératrices partielles : [10]

$$p_i(z) = \sum_{j=0}^{\infty} z^j p_{ij},\ 0 \leq i \leq s.$$

et transformer les equations de Kolmogorov en cet ensemble d'équations différentielles :

$$\mu p\prime(z)A(z) = p(z)B(z) \tag{1.3}$$

où $p(z) = (p_0(z), ..., p_s(z))$, $p\prime = (p\prime_0(z), ..., p\prime_s(z))$ et $A(z)$ et $B(z)$ sont les matrices carrées $(s+1) \times (s+1)$ suivantes :

$$A(z) = \begin{pmatrix} z & -1 & 0 & \dots & 0 & 0 \\ 0 & z & -1 & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \\ 0 & 0 & 0 & \dots & z & -1 \\ 0 & 0 & 0 & \dots & 0 & 0 \end{pmatrix}$$

$$B(z) = \begin{pmatrix} -\lambda & \lambda & 0 & \dots & 0 & 0 \\ \mu & -(\lambda+\mu) & \lambda & \dots & 0 & 0 \\ 0 & 2\mu & -(\lambda+2\mu) & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & -(\lambda+(s-1)\mu) & \lambda \\ 0 & 0 & 0 & \dots & s\mu & -(\lambda(1-z)+s\mu) \end{pmatrix}$$

Cependant, dans le cas $s > 2$, le système d'équations infini 1.1, l'ensemble des équations linéaires 1.2 et l'ensemble des équations différentielles 1.3, **n'ont pas de solution en forme close (analytique)** (closed form solution). Par conséquent, **malgré le fait que le générateur infinitésimal Q soit bien structuré, la distribution stationnaire $\{p_{ij}\,;\ (i,j) \in E\}$ ne peut être exprimée sous aucune forme analytique (facile), et ne se prête pas non plus à un calcul récursif direct**. Ceci explique l'absence de formules analytiques explicites pour les caractéristiques de performance du modèle $M/M/s$ avec rappel, quand le nombre de serveurs est supérieur à 2.

Par contre, quand $s \leq 2$, les probabilités stationnaires p_{ij} satisfont un ensemble d'équations de type naissance et de mort. Ainsi, des solutions explicites sont disponibles [5]. Cependant, la considération de plus de deux serveurs ($s > 2$), complique les transitions entre états et, par conséquent, la structure sous-jacente de type naissance-et-mort n'est pas préservée.

Ainsi, des solutions explicites de la distribution stationnaire et des indices de performance, ont été obtenues seulement dans quelques cas particuliers [79, 5, 78, 92, 93].
Pour le modèle $M/M/2$ avec clients persistants, des solutions exactes des probabilités stationnaires ont été dérivées dans [75, 76, 77, 78, 79]. Récemment, la FAR $M/M/2/2$ avec clients impatients a été étudiée dans [80] où les auteurs ont obtenu la distribution jointe du nombre de serveurs occupés et du nombre d'appels en orbite. Par ailleurs, Artalejo a

obtenu récemment dans [94], des résultats pour le modèle $M/M/s$ avec orbite à capacité finie comme hypothèse.

D'autre part, en se basant sur les équations linéaires de Kolmogorov 1.2 et les équations différentielles 1.3, quelques travaux théoriques intéressants ont été consacrés à la solution analytique du modèle avec un nombre arbitraire de serveurs. Essentiellement, nous citons le papier de Cohen [1] où la distribution stationnaire de l'état du système est exprimée en fonction *d'intégrales de contour*, et celui de Pearce [81], où la distribution stationnaire est exprimée comme une *limite de fractions continues étendues*. D'un point de vue analytique, ces deux solutions représentent des tentatives intéressantes (de grande portée) et très significatives dans le domaine des FAR multi-serveurs (c'est l'avis de plusieurs auteurs) et démontre amplement les difficultés associées. Cependant, d'un point de vue pratique, les probabilités stationnaires p_{ij} ne peuvent pas être exprimées sous une forme facile à manier et ne conduisent pas non plus à un calcul récursif direct. Par conséquent, ces solutions analytiques proposées ne conviennent pas à des implémentations pratiques.

Ainsi, pour remédier au problème, les chercheurs travaillant dans le domaine des FAR avec un nombre arbitraire de serveurs et une source infinie, ont orienté leurs recherches vers les méthodes d'approximation et les modèles tronqués, qui sont des méthodes numériques, dont l'implementation est pratique et qui permettent le calcul des probabilités stationnaires.

Cet axe de recherche comprend trois voies :

- Les approximations [95, 96, 82, 97].
- Les modèles tronqués finis (finite truncated models) [56, 83].
- Les modèles tronqués généralisés (generalized truncated models) [98, 84, 5, 14, 85].

Dans la première catégorie, nous incluons les travaux où le modèle original intraitable (difficile) est remplacé par un autre modèle plus simplifié. Souvent, l'approximation est bonne dans le cas où les paramètres du système appartiennent à un certain domaine, ou bien dans des cas particuliers extrêmes où l'on s'intéresse au comportement limite de la FAR, tel que le cas d'un trafic chargé [96] *i.e.* quand $\lambda/\mu s \to 1-$, le cas d'une faible charge [95], le cas d'une faible intensité de rappel ($\nu \to 0$) ou encore une forte intensité de rappel ($\nu \to \infty$) [82]. Dans ces cas, les théorèmes limites nous permettent de comprendre l'influence des appels répétés [5].

D'autres méthodes d'approximation consistent à calculer les bornes supérieures de performance de ce modèle [97].

En effet, dans le cas où le taux de rappel $\nu \to \infty$, i.e. que les intervalles entre deux rappels successifs tendent vers zéro, la FAR $M/M/s$ peut être vue comme une file d'attente standard avec un buffer. Cette observation heuristique générale peut être confirmée par des résultats mathématiques rigoureux. Ceci permet l'obtention de formules asymptotiques pour les caractéristiques de performance stationnaire ([5], section 2.7.1). D'autre part, le comportement limite des FAR $M/M/s$ quand $\nu \to 0$ [5], converge vers le comportement du système d'Erlang avec perte correspondant.

Dans la seconde catégorie, les *modèles tronqués finis* consistent à remplacer l'espace d'états infini initial E, par un autre espace d'états fini tronqué R. Une première possibilité simple consiste à mettre une limite fictive L à la capacité de l'orbite, ce qui a été fait initialement par Wilkinson dans [56]. Ainsi, le système infini d'équations linéaires peut être réduit en un système d'équations fini, qui peut être ensuite résolu par des procédures numériques, ou bien par ordinateur à l'aide des routines standards de résolution des systèmes d'équations linéaires, pour l'obtention de la distribution stationnaire et des principaux paramètres de performance.

En fait, la solution du système tronqué converge vers celle du système original (initial), quand la capacité de l'orbite $L \to \infty$. Ainsi, la limite fictive de la taille de l'orbite doit être finie mais suffisamment grande, pour que la distribution stationnaire du système tronqué soit une bonne approximation de la distribution du système initial.
Cependant, comme aucune base analytique n'est disponible pour le choix du niveau limite adéquat L, Neuts et Rao [84] ont proposé une approche de tests, qui consiste à commencer par une valeur initiale raisonnable pour L, calculer les probabilités stationnaires correspondantes et ensuite l'accroître progressivement jusqu'à l'obtention de la valeur appropriée pour L à partir de laquelle les éléments du vecteur des probabilités stationnaires varient très peu. Ce vecteur peut être évalué par une méthode itérative telle que Gauss-Seidel qui tire profit de la structure de Q [84].

Cette méthode tronquée directe est pratique pour calculer le vecteur des probabilités stationnaires pour des systèmes à faible intensité de trafic ρ avec un taux de rappel assez large. Par contre, pour les systèmes à des hauts niveaux de congestion (taux de rappel faible avec intensité de trafic importante), L croît rapidement et donc la taille du système d'équations tronqué serait très grande. Ainsi, pour de tels cas, l'évaluation du vecteur des probabilités stationnaires par la méthode tronquée directe est très coûteuse du point de vue calcul.

Récemment, de nouvelles méthodes tronquées plus sophistiquées ont été développées. Ces méthodes sont basées sur l'exclusion des états dont les probabilités stationnaires sont négligeables [83]. Là aussi, à des hauts niveaux de congestion, la solution nécessite des ressources de calcul très puissantes.

Pour remédier à cet inconvénient, de nouveaux modèles plus efficaces dits : les *modèles tronqués généralisés* ont été développés [98, 84, 5, 14, 85]. L'idée principale de ces modèles est d'imposer une hypothèse simplifiée qui conduit à un modèle de file d'attente auxiliaire avec un espace d'états infini et un générateur infinitésimal plus approprié. Ceci correspond en fait à une approximation du système infini qui ne peut être résolu directement, par un autre système calculable infini, avec un générateur infinitésimal ayant une forme facile à traiter.

Les méthodes basées sur les modèles tronqués finis et les modèles tronqués généralisés, sont des méthodes d'analyse numérique approximative permettant le calcul (par approximation) de la distribution stationnaire. Cependant, il a été démontré numériquement dans [5, 84], que le fait d'approximer le système infini original par un autre système infini, donne

une meilleure précision que celle des méthodes tronquées finies.

D'autre part, Artalejo et Pozo [14] ont fait une comparaison entre les différents modèles tronqués généralisés existants dans la littérature et qui sont les suivants :

- **Le modèle de Falin [98] :** Ce modèle suppose que le taux de rappel devient égal à l'infini (∞) quand le nombre de clients en orbite dépasse un certain niveau M. Ceci signifie qu'à partir du niveau M, le système se comporte comme une file d'attente ordinaire $M/M/1$ avec un taux d'arrivée λ et un taux de service $s\mu$.

  Ainsi, le taux de rappel ν_j sachant que le nombre de clients en orbite est égal à j est donné par :

$$\nu_j = \begin{cases} j\nu, & \text{si } 0 \leq j \leq M, \\ \infty, & \text{si } j \geq M+1. \end{cases}$$

- **Le modèle de Neuts et Rao [84] :** Neuts et Rao ont proposé une seconde méthode d'approximation simplifiée efficace dite : *approximation géométrique matricielle* (matrix-geometric approximation), qui est une méthode d'analyse algorithmique basée sur la compréhension heuristique du comportement physique du système et qui permet de calculer efficacement le vecteur d'état stationnaire même pour des systèmes de haut niveau de congestion.
  Dans cette approche, ils considèrent que le nombre de clients de l'orbite, autorisés à rappeler est restreint à un nombre approprié N. Ainsi, le taux de rappel est donné par :

$$\nu_j = min(j, N)\nu, j \geq 0$$

  Cette approximation conduit à un générateur infinitésimal qui est homogène à partir du niveau N dont le choix de la valeur est crucial [84].
  Le processus approximatif correspondant est un processus de quasi-naissance et de mort indépendant du niveau, avec un nombre important d'états limite. Ainsi, en appliquant la théorie générale des processus de quasi-naissance et de mort [99], la distribution stationnaire peut être calculée.
  Des exemples numériques illustratifs démontrent l'efficacité et la précision de cette méthode d'approximation, qui permet d'une part d'obtenir les résultats numériques des systèmes qui ne sont pas faciles à traiter analytiquement, et qui a pu surmonter d'autre part les insuffisances de la méthode tronquée directe.

- **Le modèle d'Artalejo [14] :** Artalejo et Pozo ont développé une autre méthode généralisée pour l'approximation numérique de la FAR $M/M/s$. Dans ce modèle, les auteurs supposent que le taux de rappel dépend de l'état du système (i,j). Ainsi, on parle plutôt de taux ν_{ij}, où :

$$\nu_{ij} = \begin{cases} \infty, & \text{si } 0 \leq i \leq c-2 \text{ et } j \geq K+1, \\ j\nu, & \text{sinon.} \end{cases}$$

  En fait, ce processus approximatif n'a pas une interprétation physique particulière. Cependant, il correspond à une généralisation naturelle du modèle de Falin [98].

Ainsi, le diagramme de transitions entre états, est plus proche en termes de graphe du diagramme correspondant à la FAR $M/M/s$ initiale. D'autre part, nous remarquons que le taux de rappel ν_{ij} est non-homogène par rapport à la seconde coordonnée j, pareil que le modèle $M/M/s$ avec rappel principal. Ainsi, cette dernière approche préserve la caractéristique principale des files d'attente avec rappel, i.e. la non-homogèneité spaciale introduite par l'existence d'un flux d'appels répétés, contrairement aux deux autres modèles tronqués généralisés qui ont des taux de rappel homogènes à partir du niveau M (ou N resp.).
En effet, Artalejo et Pozo [14] ont démontré numériquement que cette méthode donne une meilleure approximation par rapport aux deux autres méthodes tronquées généralisées (et évidemment aux modèles tronqués finis).

1.4 Comparaison entre les files d'attente standards et les FAR

Dans cette section, nous établissons une analyse comparative des modèles $M/M/s$ standards par rapport aux modèles $M/M/s$ avec rappel.

L'état du système est décrit par le processus $X = \{(C(t), N(t));\ t \geq 0\}$ dans le cas d'une FAR et par le processus $Y = \{Q(t);\ t \geq 0\}$ qui indique le nombre de clients dans le système à l'instant t, dans le cas d'une file d'attente standard sans appels répétés.
Le processus Y est un simple processus de naissance et de mort avec des taux de naissance (arrivée) $\lambda_i = \lambda,\ i \geq 0$, et des taux de mort (service) $\mu_i = min(i, s)\mu,\ i \geq 1$. Par contre, X n'est pas un processus de naissance et de mort à cause de l'influence des appels répétés.

Comme d'habitude, la première question à étudier est la récurrence positive de X et Y. En effet, il a été démontré que chacun des deux processus est récurrent positif (ergodique), si et seulement si : $\rho = \lambda/s\mu < 1$.
Dans le cas du processus Y, la preuve découle des résultats classiques de la classification des états des processus de naissance et de mort [53, 72]. La preuve pour le processus avec rappel X utilise plutôt le critère de Foster ([5], section 2.2).

Une autre différence essentielle est le fait que le cas particulier $\rho = 1$ nécessite une condition nécessaire et suffisante pour la non-récurrence de Y, tandis que le comportement de X dans le cas $\rho = 1$ dépend du taux de rappel. Par exemple, si $s = 1$ et $\rho = 1$, alors X est non récurrent si et seulement si $\nu \geq \mu$ ([5], section 1.3.1).

Il est intéressant de noter que la condition d'ergodicité du processus X ne dépend pas du taux de rappel ν (dans le cas de rappels linéaires). Ceci implique qu'elle est valable pour ν quelconque et particulièrement pour $\nu = \infty$. En fait, quand $\nu = \infty$, la FAR se comporte comme une file d'attente standard avec une discipline d'ordre aléatoire. Ceci est cohérent, car pour les modèles $M/M/s$ standards (sans rappel), cette condition d'ergodicité est valable.

Si $\rho < 1$, l'état stationnaire existe. Dans le cas de la file d'attente $M/M/s$ standard [53, 72, 54], la distribution stationnaire du nombre de clients dans le système est donnée par :

$$p_i = \begin{cases} p_0(\frac{\lambda}{\mu})^i \frac{1}{i!}, & \text{si } 0 \leq i \leq s \\ p_0 \rho^i \frac{s^s}{s!}, & \text{si } i > s. \end{cases}$$

où :

$$p_0 = (\frac{s^s \rho^{s+1}}{s!(1-\rho)} + \sum_{i=0}^{s} (\frac{\lambda}{\mu})^i \frac{1}{i!})^{-1}$$

En fait, les probabilités stationnaires p_i de ce processus de naissance et de mort peuvent être obtenues en établissant simplement le système des équations d'équilibre de la chaîne de Markov avec la condition de normalisation $\sum_{i\geq 0} p_i = 1$. La résolution de ce système qui contient une infinité d'équations à nombre infini d'inconnues, nous permet (par substitution) d'exprimer chaque inconnue p_i en fonction de p_0. À partir de cette distribution, on peut calculer les différents paramètres de performance du système à l'état stationnaire.

En particulier, dans le cas mono-serveur $M/M/1$ standard, nous avons une distribution géométrique avec le paramètre ρ.

$$p_i = (1-\rho)\rho^i, \ i \geq 0.$$

Pour ce qui est des modèles de FAR, la distribution stationnaire de l'état de la file d'attente $M/M/1$ avec rappel, est donnée par : ([5], section 1.2)

$$p_{0j} = \frac{\rho^j}{j!\nu^j}(1-\rho)^{1+\lambda/\nu} \prod_{k=0}^{j-1} (\lambda + k\nu), \ j \geq 0,$$

$$p_{1j} = \frac{\rho^{j+1}}{j!\nu^j}(1-\rho)^{1+\lambda/\nu} \prod_{k=1}^{j} (\lambda + k\nu), \ j \geq 0,$$

Ainsi, nous avons aussi l'expression suivante pour la distribution stationnaire du nombre total de clients dans le système :

$$p_j = \frac{\rho^j}{j!\nu^j}(1-\rho)^{1+\lambda/\nu} \prod_{k=1}^{j} (\lambda + k\nu), \ j \geq 0,$$

D'autre part, le modèle de FAR avec $s = 2$ peut être réduit à des expressions hypergéométriques ([5], section 2.3). Cependant, la considération de plus de deux serveurs complique les transitions entre états et, par conséquent, la structure sous-jacente de type naissance et de mort n'est pas préservée.

Nous nous intéressons à présent à une autre caractéristique de performance importante, qui est le temps d'attente virtuel $W(t)$ d'un client qui arrive au système, à l'instant t. Suivant cette définition, $W(t)$ signifie le temps que le client passe en attente dans la file (modèle standard) ou dans l'orbite (modèle avec rappel) avant de commencer le service. Comme nous traitons les systèmes à l'état stationnaire, nous notons simplement $W(t)$ par W.

Il est clair que la distribution de W dépend de la discipline de service. En fait, la plupart des systèmes avec rappel fonctionnent suivant une discipline d'accès à ordre aléatoire

(random access discipline). Ceci signifie que tous les appels en attente dans l'orbite ont une chance égale d'être affectés à des serveurs quand ceux-ci deviennent disponibles. Cependant, le service aléatoire est compliqué à cause du phénomène de compétition entre le flux des arrivées primaires et celui des appels secondaires. Ainsi, un client primaire, qui appelle pour la première fois pourrait être servi avant ceux qui sont en orbite, et qui ont effectué plusieurs tentatives.

La fonction de distribution du temps d'attente W dans la file $M/M/s$ standard, peut être exprimée sous la forme d'une intégrale ([100], section 9.1.3) qui peut être développée en utilisant les polynômes orthogonaux de Lagrange ou bien les méthodes des séries-Mclaurin ([101], section 5.15).

Contrairement à ceci, les résultats analytiques de l'analyse du temps d'attente dans la FAR $M/M/s$ restent encore à développer [10].

Par ailleurs, il est important de préciser que dans beaucoup d'applications, il est très difficile d'observer l'orbite (ou le pool de rappel). C'est le cas, par exemple, des réseaux mobiles cellulaires [102, 18]. Ainsi, l'analyse des performances des systèmes basés sur les modèles avec rappel diffère largement de celle basée sur les files d'attente standards car l'orbite est un buffer invisible. Par conséquent, les clients (et les serveurs) n'ont aucune information sur le nombre d'unités en orbite.

1.5 Les FAR à source finie

Dans la théorie des files d'attente avec rappel, il est habituellement supposé que le flux des arrivées primaires est poissonnien (flux de Poisson). Ceci signifie que les arrivées primaires sont générées par une population infinie de clients potentiels. Autrement dit, le nombre de sources est infini, et chacune d'elles génère des arrivées primaires indépendantes. Dans une telle description, la probabilité d'une nouvelle arrivée durant tout intervalle de durée dt est donnée par $\lambda dt + o(dt)$ quand $dt \to 0$, indépendamment de l'état du système à l'instant t, où λ est le taux du processus d'arrivée de Poisson.

L'hypothèse faite sur le nombre de clients supposé infini a l'avantage de permettre une description mathématique plus simple des problèmes de rappel [1]. Cependant, dans certaines applications tels que les réseaux de communication récents, le processus de Poisson n'est pas approprié pour décrire le modèle des arrivées par packets, à cause de la corrélation entre les packets qui arrivent dans les réseaux ATM [103]. Par ailleurs, dans beaucoup de situations pratiques, il est important de prendre en compte le fait que le taux de génération des nouveaux appels primaires décroît quand le nombre de clients dans le système croît. Ceci peut se faire en considérant des modèles avec une source finie de clients, ou bien des modèles à entrée quasi-aléatoire (quasi-random input models), où chaque source individuelle génère son propre flux d'appels primaires. Ainsi, le processus quasi-aléatoire est une généralisation naturelle du processus de Poisson.

Des exemples de ce comportement apparaissent dans la modélisation et l'analyse des performances des systèmes à disque-mémoire magnétique (magnetic disk-memory

systems) [17], des réseaux téléphoniques mobiles cellulaires [18, 19], des systèmes hybrides fibre-coaxial (hybrid fiber-coax systems) [20] et des réseaux LAN (Local Area Networks) avec les protocoles CSMA/CD non-persistants [21], avec une topologie en étoile [22, 23, 15], avec des protocoles à accès aléatoire [24] et avec des protocoles à accès multiple [25] ainsi que des réseaux LAN sans collision.

Les files d'attente standards (sans rappel) avec une population finie ont été étudiées en détail par Takagi dans [104]. Pour ce qui est du modèle des FAR avec un flux d'entrée quasi-aléatoire, il a été traité initialement par Kornyshev [105] qui a étudié le modèle multi-serveurs $M/M/s//K$. Depuis ce premier article, d'autres chercheurs se sont intéressés au domaine vu son importance. Pour une synthèse complète des résultats relatifs au modèle markovien multi-serveurs $M/M/s//K$ avec rappel et source finie, voir les articles de Falin et Artalejo [11, 12, 16], où les auteurs ont dérivé les formules explicites des principales caractéristiques de performance stationnaires, y compris le processus du temps d'attente qui est particulièrement complexe et difficile à étudier pour les FAR, à cause de la concurrence entre les deux flux.

Pour plus de résultats, voir aussi [106, 5, 107, 108, 15, 18, 109] où d'autres variantes du modèle $M/M/s//K$ avec rappel et source finie ont été traitées.

Dans cette section, nous présentons une synthèse des principaux résultats obtenus pour le modèle $M/M/s//K$ avec rappel et source finie.

1.5.1 Description du modèle $M/M/s//K$ avec rappel

Nous considérons une FAR multi-serveurs $M/M/s//K$ avec s serveurs parallèles et identiques (homogènes) et une population de clients (ou source) de taille K finie. Dans ce modèle, chaque client est soit libre, en orbite ou en service à tout instant. Le processus d'arrivée des appels primaires est un processus quasi-aléatoire de taux λ, ce qui signifie que la probabilité qu'un client génère une requête pour le service dans tout intervalle $(t, t+dt)$ est $\lambda(K-i-j)dt+o(dt)$ (où i est le nombre de clients en service et j le nombre de clients en orbite) quand $dt \to 0$, si le client est libre à l'instant t, et zéro si le client est en orbite ou en cours de service à l'instant t, et ce indépendamment du comportement de tous les autres clients.

À l'instant d'arrivée d'un appel primaire, si un serveur au moins est disponible, le client sera immédiatement servi suivant un processus exponentiel de paramètre μ. Par contre, si tous les serveurs sont occupés, le client rejoint l'orbite et produit un flux exponentiel d'appels répétés avec le taux ν.

1.5.2 Description de l'évolution de l'état du modèle $M/M/s//K$ avec rappel

L'état du système $M/M/s//K$ peut être décrit par le processus $X = \{(C(t), N(t)) ; t \geq 0\}$, où $C(t)$ est le nombre de serveurs occupés et $N(t)$ est le nombre de clients en orbite (ou sources d'appels répétés) à l'instant t.

À cause de l'exponentialité des variables aléatoires du modèle, ce processus X est une chaîne de Markov à temps continu homogène avec l'espace d'états fini $E = \{0, ..., s\} \times$

$\{0, ..., K-s\}$.

Nous noterons par $(i,j) = lim_{t\to\infty} X(t)$ où i représente le nombre de clients actifs (en service), et j le nombre de clients en orbite à l'état stationnaire.

Les évènements qui peuvent modifier l'état du système et les probabilités de transition d'état dans un intervalle de temps infiniment petit dt sont résumés dans le tableau suivant :

Transition d'état	Probabilité de transition d'état	Description de l'évènement
$(i,j) \to (i+1,j)$ $0 \le i \le s-1$, $0 \le j \le K-s$	$(K-i-j)\lambda\ dt + o(dt)$	Arrivée d'un appel primaire et son début de service
$(s,j) \to (s,j+1)$ $0 \le j \le K-s-1$	$(K-s-j)\lambda\ dt + o(dt)$	Arrivée d'un appel primaire en son entrée en orbite
$(i,j) \to (i+1,j-1)$ $0 \le i \le s-1$, $1 \le j \le K-s$	$j\nu\ dt + o(dt)$	Arrivée d'un appel répété et son début de service
$(i,j) \to (i-1,j)$ $1 \le i \le s$, $0 \le j \le K-s$	$i\mu\ dt + o(dt)$	Fin de service d'un client

À partir de ce tableau récapitulant l'ensemble des transitions possibles entre les états, on peut générer la chaîne de Markov décrivant l'évolution du modèle multi-serveurs $M/M/s//K$ avec rappel et source finie de taille K (voir Figure 1.5).

Comme le nombre de serveurs et la taille de la population K sont limités, le nombre d'états du processus markovien de la figure 1.5 est fini. Il est égal à : $(s+1)\times(K-s+1)$. De plus, cette chaîne est irréductible (graphe fortement connexe) car tout état peut être atteint à partir de n'importe quel autre état de la chaîne. Ainsi, d'après la théorie classique des processus de Markov, il est bien connu que si une CMTC finie est irréductible, alors, il existe une distribution unique des probabilités stationnaires. Ainsi, ce processus est ergodique et par conséquent, il admet une distribution stationnaire unique, et ce $\forall \lambda > 0, \mu > 0$ et $\nu > 0$.

Les éléments du générateur infinitésimal du processus X sont donnés par : [11]

pour $0 \le i \le s-1$

$$q_{(i,j)(n,m)} = \begin{cases} (K-i-j)\lambda, & \text{si } (n,m) = (i+1,j), \\ i\mu, & \text{si } (n,m) = (i-1,j), \\ j\nu, & \text{si } (n,m) = (i+1,j-1), \\ -((K-i-j)\lambda + i\mu + j\nu), & \text{si } (n,m) = (i,j), \\ 0, & \text{sinon.} \end{cases}$$

pour $i = s$

$$q_{(s,j)(n,m)} = \begin{cases} (K-s-j)\lambda, & \text{si } (n,m) = (s,j+1), \\ s\mu, & \text{si } (n,m) = (s-1,j), \\ -((K-s-j)\lambda + s\mu), & \text{si } (n,m) = (s,j), \\ 0, & \text{sinon.} \end{cases}$$

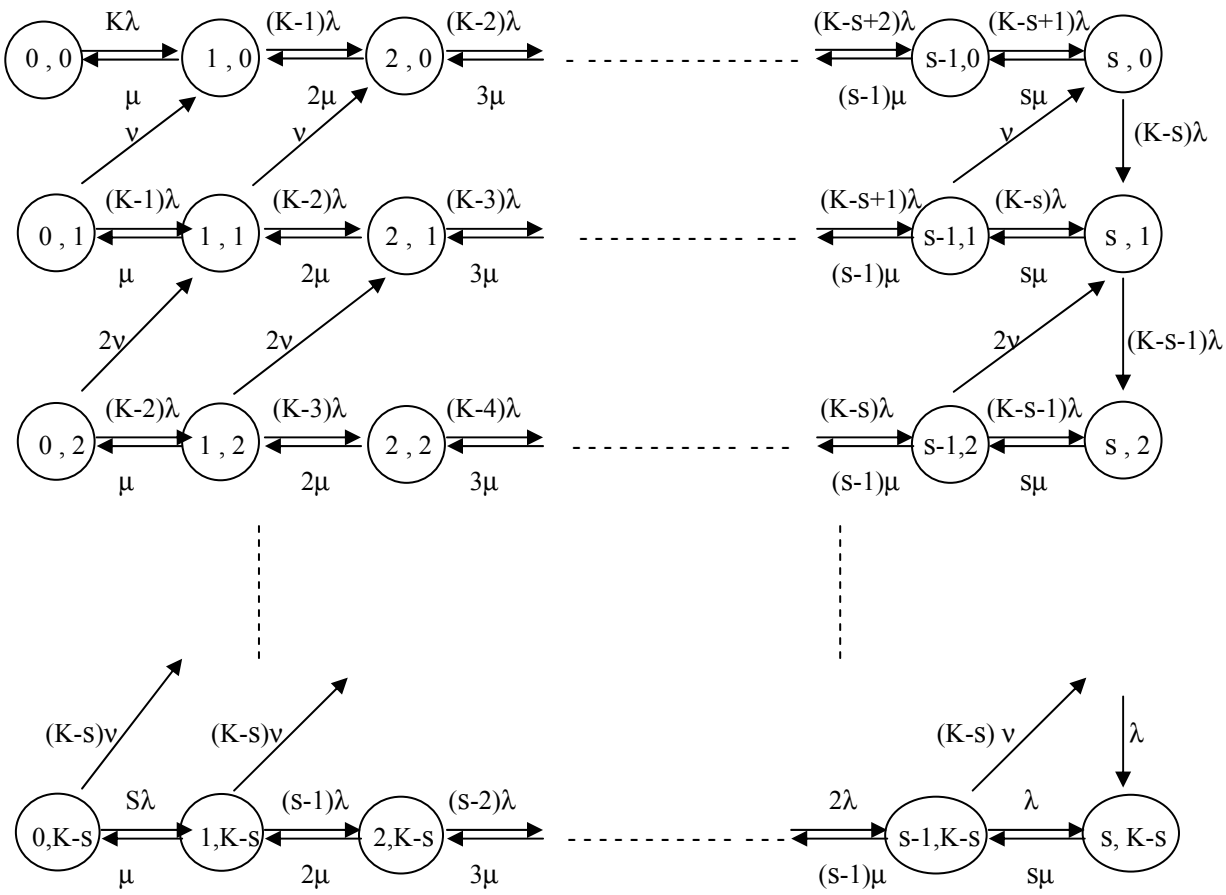

FIG. 1.5 – La CMTC décrivant la FAR M/M/s//K

1.5.3 Analyse du modèle $M/M/s//K$ avec rappel

Les probabilités stationnaires $\{p_{i,j}, (i,j) \in E\}$ satisfont l'ensemble d'équations suivant : [11]

$$\begin{cases} ((K-i-j)\lambda + i\mu + j\nu)\ p_{i,j} \\ = (K-i+1-j)\lambda\ p_{i-1,j} + (j+1)\ \nu\ p_{i-1,j+1} + (i+1)\ \mu\ p_{i+1,j}, \quad 0 \leq i \leq s-1,\ j \geq 0 \\ ((K-s-j)\lambda + s\mu)\ p_{s,j} \\ = (K-s+1-j)\lambda\ p_{s-1,j} + (j+1)\ \nu\ p_{s-1,j+1} + (K-s-j+1)\lambda\ p_{s,j-1}, \quad j \geq 0 \end{cases}$$

où $p_{-1,j} = p_{i,-1} = 0\ \forall i$ et j.

Falin a proposé dans [16], un algorithme récursif pour calculer les probabilités stationnaires $p_{i,j}$. Ainsi, les indices de performance du système peuvent être exprimés en fonction de ces probabilités comme suit : [11, 5]

– Le nombre moyen de sources d'appels répétés

$$N = E[N(t)] = \sum_{i=0}^{s} \sum_{j=0}^{K-s} j.p_{i,j}$$

– Le nombre moyen de serveurs occupés

$$Y = E[C(t)] = \sum_{i=0}^{s} \sum_{j=0}^{K-s} i.p_{i,j}$$

– La probabilité que tous les serveurs sont occupés

$$P_s = P\{C(t) = s\} = \sum_{j=0}^{K-s} p_{s,j}$$

– Le taux moyen de génération des appels primaires

$$\overline{\lambda} = \lambda E[K - C(t) - N(t)] = \lambda(K - Y - N)$$

– La probabilité de blocage des appels primaires

$$B_p = \frac{(K-s)P_s - N_s}{K - Y - N}$$

où $N_s = \sum_{j=0}^{K-s} j.p_{s,j}$

– La probabilité de blocage des appels répétés

$$B_R = \frac{N_s}{N}$$

– La probabilité de blocage global

$$B = \frac{\lambda(K-s)P_s + (\nu - \lambda)N_s}{\lambda(K - Y - N) + \nu N}$$

– Le temps d'attente moyen

$$W = \frac{N}{\overline{\lambda}}$$

1.6 Applications des systèmes avec rappel et source finie

Les systèmes avec appels répétés et source finie apparaissent naturellement dans beaucoup d'applications pratiques dans les réseaux de communication, les réseaux téléphoniques, informatiques, les systèmes à temps réel [3], ainsi que certains problèmes de la vie courante.

Dans cette section, nous donnons quelques exemples de systèmes avec rappel et source finie.

1.6.1 Les systèmes téléphoniques

La première étude de Kornyshev [105] traitant les FAR à source finie, était motivée par l'analyse du comportement des abonnés dans les réseaux téléphoniques réels. Il était évident que les modèles classiques (avec espace d'attente ou avec perte), ne prenaient pas en considération l'existence d'un réel flux d'appels répétés. En effet, les systèmes avec appels répétés offrent une bonne alternative pour comprendre le phénomène de rappel dans les réseaux téléphoniques sachant que tout abonné qui reçoit un signal occupé, recompose le numéro après une certaine période de temps aléatoire. D'autre part, ces systèmes sont caractérisés par une population finie d'abonnés.

1.6.2 Les systèmes à disque-mémoire magnétique (magnetic disk memory systems)

Ohmura et Takahashi [17] ont appliqué le modèle de FAR à population finie pour l'analyse des systèmes à disque-mémoire magnétique. Dans ces systèmes, K unités de mémoire se partagent un contrôleur de disque (serveur) et transmettent une information dès qu'ils trouvent le contrôleur oisif. Les requêtes non satisfaites sont répétées après une rotation du disque ce qui peut être modélisé par un intervalle de rappel constant.

1.6.3 Les réseaux mobiles cellulaires

Dans les systèmes à communication mobile, un espace est divisé en cellules, chacune d'elles est servie par une station de base ayant un nombre limité de canaux. Dans les nouvelles technologies des réseaux mobiles, les micro-cellules sont considérées, ainsi, la taille de la cellule devient plus petite et donc le nombre de mobiles servis dans une cellule est aussi relativement plus petit. Par conséquent, les modèles avec source finie doivent être considérés [18].

1.6.4 Les réseaux LAN avec le protocole CSMA/CD

Le modèle des files d'attente avec rappel et source finie a été souvent appliqué [21] pour l'étude des réseaux LAN (Local Area Network) [25, 110] qui fonctionnent sous le protocole de communication CSMA/CD (Carrier Sense Multiple Access with Collision Detection).

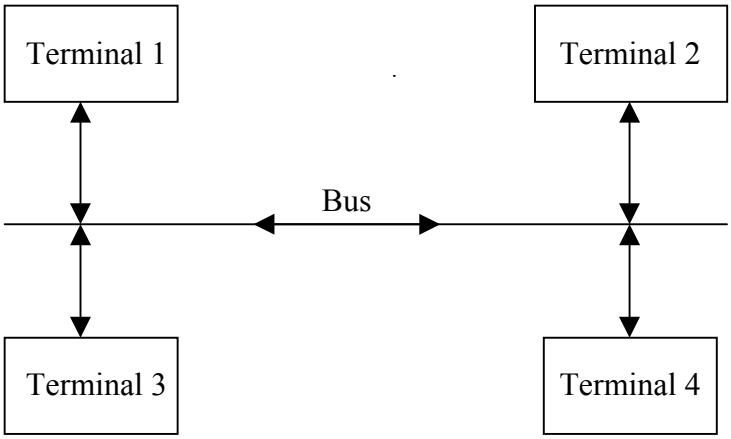

FIG. 1.6 – Réseau LAN simple avec architecture en bus

Dans ces réseaux, un nombre fini K d'utilisateurs (ou de terminaux) sont reliés par un bus (serveur) unique qui est le canal de communication (voir Fig. 1.6). Les terminaux communiquent les uns avec les autres via le bus qui peut être utilisé par un seul terminal à la fois. Une telle architecture de réseau d'ordinateurs local est appelée *architecture en bus* [111].

Sous ce protocole, si un terminal a un message prêt pour la transmission, il vérifie immédiatement l'état du canal. Si celui-ci est occupé, il réessaiera la transmission après une période de temps aléatoire dite : *temps de rappel*. D'autre part, si le canal est libre, la transmission du message commence immédiatement. Après la transmission, le terminal renonce au contrôle du bus qui reste oisif jusqu'à ce qu'un nouveau message arrive de l'extérieur.

À cause des délais de propagation non nuls, durant un certain intervalle de temps, après que le terminal ait commencé la transmission du message, d'autres terminaux pourraient trouver le canal libre et transmettre leur message. Dans un tel cas, une collision pourrait avoir lieu, et par conséquent tous les terminaux impliqués dans la collision abandonnent leur transmission et rejoignent le groupe de rappel (l'orbite) pour rappeler ultérieurement. Àprès une certaine durée de temps, suite à la collision, le canal sera de nouveau disponible.

1.6.5 Les réseaux LAN sans collision (collision avoidance local area networks)

Une caractéristique habituelle dans les réseaux LAN est que plusieurs terminaux utilisent un moyen commun pour la transmission, ainsi des collisions entre les messages ont lieu. Ces collisions impliquent la destruction de l'information venant des différents terminaux, et par conséquent la qualité de performance diminue. Pour éviter ce problème, un nombre de réseaux LAN sans collisions ont été développés (topologie en bus, topologie en étoile, etc). Janssens dans [22] a décrit un réseau LAN composé de K contrôleurs d'accès au réseau et d'un hub. Le réseau est modélisé par une FAR mono-serveur (hub) avec une population finie de taille K (contrôleurs).

1.6.6 Les systèmes informatiques à temps réel

Considérons un système informatique à temps réel dans lequel il y a s ports et m terminaux ($m \geq s$). Pour la connexion d'un terminal à l'ordinateur central, un port exactement doit être utilisé. Les étudiants utilisent l'ordinateur central pour le traitement des données, pendant une durée de temps aléatoire et ce à partir de leurs terminaux. Un terminal est utilisé par un étudiant à la fois. Pour cela, l'étudiant doit envoyer une requête pour sa connexion à l'ordinateur à partir du terminal. S'il y a un port libre, le terminal est connecté immédiatement à l'ordinateur ; autrement l'étudiant reformulera la requête après un temps aléatoire jusqu'à ce qu'il y ait un port libre pour lui.

Dans ce système, les s ports correspondent aux serveurs et les m terminaux représentent la source finie de clients, tandis que l'orbite est à capacité infinie car tous les terminaux (clients) bloqués peuvent rappeler pour se connecter.

1.7 Les FAR avec serveur(s) non-fiable(s)

Les FAR ont été largement utilisées pour modéliser des problèmes qui apparaissent dans les systèmes à commutation téléphonique, les réseaux de télécommunications, les réseaux informatiques, etc. En pratique, certains composants de ces systèmes sont sujets à des pannes aléatoires (voir par exemple [112, 113]) suite auxquelles le service est interrompu pour une durée de temps aléatoire. Il est donc d'une importance basique d'étudier la fiabilité des FAR, où les serveurs sont sujets à des pannes et des réparations en plus du phénomène de rappel, et ce à cause de la forte influence des pannes qui constituent un phénomène naturel et qui peuvent avoir un impact non négligeable sur les performances du système. On parle alors de modèle de *file d'attente avec rappel et serveurs non-fiables*. Dans ce cas, les serveurs ont un taux de panne et un taux de réparation (en plus du taux de service), contrairement aux systèmes avec serveurs fiables où ces deux taux sont nuls.

Cependant, malgré l'importance de l'étude des modèles qui prennent en considération le phénomène de rappel combiné avec la non-fiabilité du serveur, les travaux et les résultats obtenus sont limités. Pour une littérature relative, les lecteurs intéressés peuvent consulter par exemple les articles [114, 115, 26, 27, 116, 28, 29], où les FAR à un serveur unique non-fiable et à source infinie ont été traitées.

En ce qui concerne les modèles avec serveur(s) non fiable(s) et source finie de clients, les travaux sont aussi limités. En fait, des études portant sur des modèles de FAR markoviens ont été faites récemment par une équipe de l'université de Debrecen [30, 31, 32, 33, 34], et les résultats sont obtenus par des méthodes numériques.

Dans ces modèles, les appels primaires sont générés par K ($1 < K < \infty$) **sources dont chacune correspond à un client**, et le service des clients est assuré par s serveurs ($s \geq 1$) sujets à des pannes et des réparations aléatoires. Chacune des sources peut être dans l'un des trois états : libre, en service ou en orbite. Chaque serveur peut être occupé ou oisif. D'autre part, il peut être opérationnel ou en panne.

Les appels primaires arrivent dans le système selon un processus quasi-aléatoire. Si un appel primaire trouve un serveur libre et opérationnel, le client commence le service

immédiatement et quitte le système une fois le service terminé. Si par contre, à l'instant d'arrivée de l'appel, tous les serveurs sont soit occupés soit en panne, alors le client entre en orbite et devient une source d'appels répétés. Toutes les variables aléatoires sont supposées être distribuées exponentiellement et indépendantes les unes des autres.

D'autre part, chaque serveur peut tomber en panne durant l'intervalle $(t, t+dt)$, avec la probabilité $\gamma dt + o(t)$ en étant occupé, et avec la probabilité $\delta d(t) + o(t)$ s'il est oisif. Ainsi, deux politiques de panne ont été considérées dans la littérature [30, 31, 32] :

- Si $\delta = 0$ et $\gamma > 0$: on parle de **pannes actives** (active breakdowns), i.e. que le serveur ne peut tomber en panne qu'en cours de service ;
- Si $\delta = \gamma > 0$: on parle de **pannes indépendantes** (independent breakdowns), i.e. que le serveur peut tomber en panne indépendamment de son état, oisif ou occupé.

Si le serveur tombe en panne en étant occupé, deux alternatives sont possibles une fois qu'il sera réparé :

- Soit il continue le service du client interrompu, on parle alors de **panne à service continu**,
- Ou bien, le client interrompu entre en orbite et rappellera ultérieurement pour le service, avec une remise à zéro du travail réalisé avant que la panne ne survienne, on parle dans ce cas de **panne à service répété**.

Par ailleurs, durant une période où tous les serveurs sont en panne, deux différents cas peuvent être envisagés :

- **Sources bloquées**, où toutes les opérations sont arrêtées y compris la génération des appels primaires et des appels répétés ;
- **Sources intelligentes** (non bloquées), où seulement le service est interrompu et toutes les autres opérations continuent, autrement dit, les nouveaux appels et les appels répétés peuvent être générés indépendamment de l'état des serveurs.

Dans les sections suivantes, nous présentons les résultats obtenus pour différents modèles de FAR à source finie et serveur(s) non-fiable(s).

1.8 Les FAR à source finie homogène et serveur non-fiable

Le modèle de FAR avec une source finie de clients homogènes et **un serveur unique non-fiable** a été étudié récemment par Almasi, Roszik et Sztrik dans [30, 31].

Dans ce modèle, les appels primaires sont générés par K $(1 < K < \infty)$ sources homogènes. Une source libre peut générer des appels primaires suivant un processus quasi-aléatoire de taux λ. Si le serveur est opérationnel et oisif au moment de l'arrivée d'un appel primaire ou répété, cet appel commence immédiatement à être servi suivant une distribution exponentielle de taux μ et le serveur passera à l'état occupé ; autrement le client entre en orbite et devient une source de génération d'appels répétés de taux ν.
À la fin d'un service, le serveur devient oisif et la source devient libre et peut ainsi générer

d'autres appels primaires.

D'autre part, le serveur peut tomber en panne en étant occupé avec un taux γ, et en étant oisif avec un taux δ. Le temps de réparation est distribué exponentiellement avec un taux τ, et ce pour les deux politiques de panne.

L'état du système à l'instant t peut être décrit par le processus $X(t) = (Y(t); C(t); N(t))$, où :

$$Y(t) = \begin{cases} 0 & \text{si le serveur est opérationnel,} \\ 1 & \text{si le serveur est en panne.} \end{cases}$$

$$C(t) = \begin{cases} 0 & \text{si le serveur est oisif,} \\ 1 & \text{si le serveur est occupé.} \end{cases}$$

et $N(t)$ est le nombre de sources d'appels répétés (ou clients en orbite) à l'instant t.

À cause de la propriété d'absence de mémoire de la distribution exponentielle des variables aléatoires de ce modèle, le processus $\{X(t),\ t \geq 0\}$ est une chaîne de Markov avec un espace d'états fini. Comme ce processus est irréductible et fini, on peut conclure qu'il est récurrent positif, et par conséquent ergodique pour toutes les valeurs des taux du modèle [30, 31]. Ainsi, l'état stationnaire existe. D'autre part, les transitions entre les états de cette chaîne de Markov dépendent de la politique de panne du système (pannes actives ou indépendantes) et du type des sources considérées (sources bloquées ou intelligentes). Pour ce qui est des probabilités stationnaires, elles sont définies dans tous les cas, comme suit : [30, 31]

$$P(y, i, j) = lim_{t\to\infty} P(Y(t) = y; C(t) = i; N(t) = j)$$

où : $y = 0, 1 \quad i = 0, 1 \quad j = 0, ..., K^*$.
avec

$$K^* = \begin{cases} K - 1, & \text{pour sources bloquées,} \\ K - i, & \text{pour sources intelligentes.} \end{cases}$$

En ayant les probabilités stationnaires, divers indices de performance et de fiabilité peuvent être dérivés en appliquant les formules suivantes : [30, 31]

– **L'utilisation du serveur**

$$U_S = \sum_{j=0}^{K-1} P(0, 1, j)$$

– **La disponibilité du serveur**

$$A_S = \sum_{i=0}^{1} \sum_{j=0}^{K^*} P(0, i, j)$$

– **L'utilisation du réparateur :** Elle correspond à la non disponibilité du serveur

$$U_R = \sum_{i=0}^{1} \sum_{j=0}^{K^*} P(1,i,j) = 1 - A_S$$

– **Le nombre moyen de clients en orbite**

$$N = E[N(t)] = \sum_{y=0}^{1} \sum_{i=0}^{1} \sum_{j=0}^{K^*} j.P(y,i,j).$$

– **Le nombre moyen de clients dans le système (en orbite ou en service)**

$$M = E[N(t) + C(t)] = N + \sum_{y=0}^{1} \sum_{j=0}^{K-1} P(y,1,j)$$

– **L'utilisation des sources (clients)**

$$U_{SO} = \begin{cases} \frac{E[K-C(t)-N(t);Y(t)=0]}{K}, & \text{pour sources bloquées,} \\ \frac{K-M}{K}, & \text{pour sources intelligentes.} \end{cases}$$

– **L'utilisation totale du système**

$$U_T = U_S + K.U_{SO} + U_R$$

– **Le taux moyen de génération des appels primaires**

$$\overline{\lambda} = \begin{cases} \lambda E[K - C(t) - N(t); Y(t) = 0], & \text{pour sources bloquées,} \\ \lambda E[K - C(t) - N(t)], & \text{pour sources intelligentes.} \end{cases}$$

– **Le temps de réponse moyen**

$$E[T] = \frac{M}{\overline{\lambda}}$$

– **Le temps d'attente moyen**

$$E[W] = \frac{N}{\overline{\lambda}}$$

– **La probabilité de blocage d'un appel primaire**

$$B = \begin{cases} \lambda E[K - C(t) - N(t); Y(t) = 0; C(t) = 1]/\overline{\lambda}, & \text{pour sources bloquées,} \\ \lambda E[K - C(t) - N(t); C(t) = 1]/\overline{\lambda}, & \text{pour sources intelligentes.} \end{cases}$$

Concernant les FAR à source finie et **plusieurs serveurs identiques non-fiables**, nous n'avons trouvé dans la littérature aucun article qui traite ce modèle.

1.9 Les FAR multi-classes

La plupart des modèles de FAR supposent que les clients sont homogènes du point de vue des caractéristiques, telles que les distributions des temps d'inter-arrivées, des temps de service et des temps d'inter-rappel. Cependant, en pratique, ces caractéristiques peuvent varier largement d'un type de clients à un autre. Ceci nous conduit aux modèles de *FAR multi-classes* ([4], section 12).

Dans une description générale, nous pouvons considérer n classes ou types de clients. Les clients primaires de type i arrivent suivant un flux homogène de taux λ_i, l'intensité de rappel associée est ν_i et le taux de service est μ_i. D'un point de vue mathématique, de tels modèles sont beaucoup plus difficiles à analyser que les modèles à une seule classe de clients (ou à clients homogènes), car l'espace d'états correspondant est $\{0, ..., s\} \times \mathbb{N}^n$ plutôt que $\{0, ..., s\} \times \mathbb{N}$ dans le cas d'une classe unique de clients [10]. Ainsi, les résultats explicites sont disponibles seulement pour quelques cas particuliers.

Ces systèmes multi-classes avec rappel apparaissent dans divers domaines tels que :
- Les systèmes Téléphone/Fax qui sont utilisés pour les appels ordinaires et pour la transmission des Fax.
- Les réseaux LAN qui permettent la transmission des données, du son et de l'image.
- Les réseaux mobiles cellulaires [117, 18], dans lesquels la station de base dans chaque cellule traite deux types d'appels : les appels d'origine initiés dans cette même cellule, et les appels hand-off, i.e., les appels en cours de communication, qui occupaient des lignes dans des cellules adjacentes et qui rentrent dans la cellule en question, suite à un déplacement inter-cellulaire des utilisateurs mobiles.

Le modèle des FAR multi-classes a été étudié initialement par Kulkarni [118] qui a considéré la FAR $M/G/1$ avec deux classes (types) de clients, deux orbites, et où les clients de chaque classe ($i = 1, 2$) ont leur propre taux d'arrivée λ_i, taux de rappel ν_i et distribution de temps de service. L'auteur a dérivé des formules explicites pour le nombre moyen de clients dans chaque en orbite, dans le système, le temps d'attente moyen et le nombre moyen de rappels pour chaque type de clients. Plus tard, Falin [119] a généralisé les résultats au modèle $M/G/1$ avec rappel et n types de clients.

Une synthèse détaillée des FAR mono-serveur avec deux types de clients ainsi que leurs applications et les nouveaux résultats des différentes variantes de ce modèle, est donnée par Choi et Chang dans [37]. D'autre part, les modèles multi-serveurs avec rappel et deux types de clients sont étudiés dans [18, 36, 38].

Une étude bibliographique nous a permis de constater que les résultats analytiques sont obtenus pour des cas assez limités, avec des hypothèses sur certains paramètres, tel que le nombre de serveurs, et le nombre de classes de clients limités à deux généralement. En fait, malgré que les caractéristiques de performance du modèle avec deux types d'appels soient disponibles sous forme explicite, certains chercheurs trouvent ces résultats encombrant à cause du fait que les formules obtenues incluent des intégrales de transformées, des solutions d'équations fonctionnelles, etc. Par ailleurs, pour d'autres modèles, l'analyse se fait par approximation ou simulation . Ainsi, beaucoup de travaux restent à développer, tel que le problème de l'obtention de résultats analytiques pour la FAR $M/M/s$ avec deux types

de clients (ou plus) quand $s \geq 2$ [36]. En fait, ce modèle a été étudié par Tran-Gia and Mandjes [18], qui ont donné un algorithme récursif permettant le calcul de la distribution stationnaire pour le cas d'une source infinie et une orbite à capacité finie (modèle tronqué).

D'autre part, l'analyse des modèles multi-classes avec source finie reste aussi un domaine à développer. En effet, des chercheurs se sont intéressés récemment à cet aspect, en considérant des modèles multi-classes particuliers où chaque classe comprend un seul client. On parle alors de modèles de *FAR avec sources (ou clients) hétérogènes.*

1.10 Les FAR à source finie hétérogène

Le modèle des FAR avec source finie hétérogène a été étudié récemment par l'équipe de l'université de Debrecen : Almasi, Bolch, Roszik et Sztrik. Ils ont considéré d'une part dans [41, 42] le cas de serveur unique fiable, et ont proposé dans [40] l'application de ce modèle pour l'analyse des systèmes de communication avec le protocole CSMA/CD. D'autre part, ils ont étudié dans [43] le cas multi-serveurs avec source finie hétérogène. Les résultats obtenus seront présentés dans la suite de cette section.

Ce modèle comprend une station de service de s serveurs identiques et un nombre fini K de sources d'appels hétérogènes où chacune génère une seule requête à la fois. Autrement dit, le modèle a une population finie de K clients potentiels différents. Par conséquent, **la source i correspond à un seul client (client i)**. D'autre part, chaque source est caractérisée par son propre taux d'arrivée des appels primaires, taux de service et taux de rappel.
Quand la source i est libre à l'instant t, elle génére un appel primaire durant l'intervalle $(t, t+dt)$ avec la probabilité $\lambda_i dt + o(dt)$. Si un des serveurs est oisif, alors le service commence et se terminera durant l'intervalle $(t, t+dt)$ avec la probabilité $\mu_i dt + o(dt)$. Durant cette période, la source est dite en service et donc ne peut pas générer de requêtes. Après le service, la source passe à l'état libre et pourra ainsi générer un nouvel appel primaire. Si par contre, tous les serveurs sont occupés à l'instant de l'arrivée d'une requête de la ième source, alors la source commence à générer un flux exponentiel d'appels répétés avec le taux ν_i jusqu'à ce qu'elle trouve un serveur oisif. D'autre part, tous les temps inclus dans ce modèle sont supposés être mutuellement indépendants.

L'état du système à l'instant t peut être décrit par le processus :

$X(t) = \{(\alpha_1, ..., \alpha_s; \beta_1, ..., \beta_{N(t)}), t \geq 0\}$,
où : s est le nombre de serveurs et $N(t)$ est le nombre de sources d'appels répétés à l'instant t.

À cause de l'hétérogénéité des sources, nous avons besoin de les identifier. Nous notons les sources en service par α_i $(i = 1, ..., s)$ et les sources en orbite par β_j $(j = 1, ..., N(t))$.

Comme les variables aléatoires de ce modèle sont exponentielles, le processus $\{X(t), t \geq 0\}$ est une chaîne de Markov. Cette chaîne à espace d'états fini est irréductible, et par conséquent ergodique pour toutes les valeurs des taux du modèle. Ainsi, l'état stationnaire

existe.

Les auteurs ont défini les probabilités stationnaires comme suit : [43]

$P(i_1, ..., i_s; 0) = lim_{t\to\infty} P\{\alpha_1 = i_1, ..., \alpha_s = i_s; N(t) = 0\}$

$P(i_1, ..., i_s; j_1, ..., j_k) = lim_{t\to\infty} P\{\alpha_1 = i_1, ..., \alpha_s = i_s; \beta_1 = j_1, ..., \beta_k = j_k\}$,
$k = 1, ..., K - s$

Une fois que les probabilités stationnaires sont obtenues, les principaux indices de performance du modèle peuvent être dérivés en appliquant les formules explicites suivantes définies dans [43] :

- **La probabilité que le client i est en orbite :** Ceci correspond au nombre moyen de clients de type i en orbite, du fait que l'on a **un seul client de chaque type**.

$$N_i = \sum_{i_1,...,i_s} \sum_{k=1}^{K-s} \sum_{\substack{j_1,...,j_k \notin \{i_1,...,i_s\} \\ i \in \{j_1,...,j_k\}}} P(i_1, ..., i_s; j_1, ..., j_k),\ i = 1, ..., K$$

- **La probabilité que le client i est en service :** Elle correspond au nombre moyen de clients de type i en service.

$$Y_i = \sum_{\substack{i_1,...,i_s \\ i \in \{i_1,...,i_s\}}} P(i_1, ..., i_s; 0) + \sum_{\substack{i_1,...,i_s \\ i \in \{i_1,...,i_s\}}} \sum_{k=1}^{K-s} \sum_{j_1,...,j_k \notin \{i_1,...,i_s\}} P(i_1, ..., i_s; j_1, ..., j_k),\ i = 1, ..., K$$

- **Le taux moyen de génération des appels primaires du client i :**

$$\overline{\lambda}_i = \lambda_i(1 - Y_i - N_i),\ i = 1, ..., K$$

- **Le temps d'attente moyen du client i :**

$$\overline{W}_i = \frac{N_i}{\overline{\lambda}_i},\ i = 1, ..., K$$

- **Le temps de réponse moyen du client i :**

$$\overline{R}_i = \frac{N_i}{\overline{\lambda}_i} + \frac{1}{\mu_i},\ i = 1, ..., K$$

- **L'utilisation du client i :**

$$U_i = 1 - N_i - Y_i,\ i = 1, ..., K$$

1.11 Les FAR à source finie hétérogène et serveur non-fiable

Le modèle des FAR avec un nombre fini de sources hétérogènes et un serveur unique non-fiable sujet à des pannes et des réparations aléatoires a été étudié récemment dans [32].

Dans ce modèle, chaque source (client) est caractérisée par un taux d'arrivée, un taux de service et un taux de rappel particulier. Par contre, le taux de panne et le taux de réparation du serveur est constant pour toutes les sources.

L'état du système à l'instant t peut être décrit par le processus :

$X(t) = \{(Y(t); \alpha_{C(t)}; \beta_1, ..., \beta_{N(t)}), t \geq 0\}$, où :

$$Y(t) = \begin{cases} 0, & \text{si le serveur est opérationnel,} \\ 1, & \text{si le serveur est en panne.} \end{cases}$$

$$C(t) = \begin{cases} 0, & \text{si le serveur est oisif,} \\ 1, & \text{si le serveur est occupé.} \end{cases}$$

$N(t)$ est le nombre de sources d'appels répétés à l'instant t et $\alpha_{C(t)}$ est la source en service à l'instant t si le serveur est occupé.
Pour identifier les sources (hétérogènes) en orbite, nous les notons par β_j $(j = 1, ..., N(t))$. Par contre, si l'orbite est vide, la troisième composante est nulle.

Le processus $\{X(t), t \geq 0\}$ est une chaîne de Markov irréductible et à espace d'états fini. Elle est donc ergodique pour toutes les valeurs des taux du modèle.

Les auteurs ont défini les probabilités stationnaires comme suit : [32]

$P(q; 0; 0) = lim_{t\to\infty} P(Y(t) = q; C(t) = 0; N(t) = 0)$ $q = 0, 1$

$P(q; j; 0) = lim_{t\to\infty} P(Y(t) = q; \alpha_1 = j; N(t) = 0)$,
$q = 0, 1,\ j = 1, ..., K$

$P(q; 0; i_1, ..., i_k) = lim_{t\to\infty} P(Y(t) = q; C(t) = 0; \beta_1 = i_1, ..., \beta_k = i_k)$
$q = 0, 1,\ k = 1, ..., K^*$

$P(q; j; i_1, ..., i_k) = lim_{t\to\infty} P(Y(t) = q; \alpha_1 = j; \beta_1 = i_1, ..., \beta_k = i_k)$,
$q = 0, 1,\ k = 1, ..., K - 1$

où

$$K^* = \begin{cases} K - 1, & \text{pour sources bloquées,} \\ K, & \text{pour sources intelligentes.} \end{cases}$$

Une fois que les probabilités stationnaires sont obtenues, les principaux indices de performance et de fiabilité du modèle peuvent être dérivés en appliquant les formules explicites suivantes définies dans [32] :

- **L'utilisation du serveur par rapport au client j :** Elle correspond à la probabilité que le serveur est opérationnel et occupé avec le client (source) j, ce qui revient à sommer toutes les probabilités où la première composante est 0 et la seconde composante est j. Formellement :

$$U_j = \sum_{k=0}^{K-1} \sum_{i_1,...,i_k \neq j} P(0; j; i_1, ..., i_k)$$

Ainsi, l'utilisation du serveur est :

$$U_s = E[Y(t) = 0; C(t)] = \sum_{j=1}^{K} U_j$$

– **L'utilisation du réparateur :** Ceci correspond à la non disponibilité du serveur

$$U_R = E[Y(t)] = \sum_{j=0}^{K} \sum_{k=0}^{K^*} \sum_{i_1,...,i_k \neq j} P(1; j; i_1, ..., i_k)$$

– **La disponibilité du serveur**

$$A_S = 1 - U_R$$

– **La probabilité que le client i est en orbite**

$$P_O^{(i)} = \sum_{q=0}^{1} \sum_{\substack{j=0 \\ j \neq i}}^{K} \sum_{k=1}^{K^*} \sum_{i \in (i_1,...,i_k)} P(q; j; i_1, ..., i_k)$$

– **La probabilité que le client i est en service**

$$P_S^{(i)} = \sum_{q=0}^{1} \sum_{k=0}^{K^*} \sum_{i \neq (i_1,...,i_k)} P(q; i; i_1, ..., i_k)$$

– **La probabilité que le client i est en service ou en orbite**

$$P^{(i)} = P_S^{(i)} + P_O^{(i)}$$

– **Le temps de réponse moyen du client i**

$$E[T_i] = \frac{P^{(i)}}{\mu_i U_i}, \ i = 1, ..., K$$

– **Le temps d'attente moyen du client** i : correspond au temps passé en orbite par le client i (indépendamment du fait que le serveur est opérationnel ou en panne) et le délai de temps dû à la panne du serveur.

$$E[W_i] = E[T_i] - 1/\mu_i = \frac{P^{(i)} - U_i}{\mu_i U_i}, \ i = 1, ..., K$$

– **Le nombre moyen de clients en orbite :**

$$N = E[N(t)] = \sum_{i=1}^{K} P_O^{(i)}$$

– **Le nombre moyen de clients dans le système (en orbite ou en service) :**

$$M = E[C(t) + N(t)] = \sum_{i=1}^{K} P^{(i)} = \sum_{i=1}^{K} P_S^{(i)} + \sum_{i=1}^{K} P_O^{(i)}$$

– **Le taux moyen de génération des appels primaires :**

$$\overline{\lambda} = \sum_{i=1}^{K} \mu_i U_i$$

– **La probabilité de blocage d'un appel primaire i :**

$$B_i = \begin{cases} \frac{\lambda_i \sum_{j=1, j\neq i}^{K} \sum_{k=0}^{K-1} \sum_{i\neq i_1,\ldots,i_k} P(0;j;i_1,\ldots,i_k)}{\overline{\lambda}}, & \text{pour sources bloquées,} \\ \frac{\lambda_i \sum_{j=1, j\neq i}^{K} \sum_{k=0}^{K-1} \sum_{i\neq i_1,\ldots,i_k} (P(0;j;i_1,\ldots,i_k)+(P(1;j;i_1,\ldots,i_k)}{\overline{\lambda}}, & \text{pour sources intelligentes.} \end{cases}$$

Ainsi, la probabilité de blocage des appels primaires est :

$$B = \sum_{i=1}^{K} B_i$$

Il est important de préciser que le modèle de FAR avec source finie hétérogène et plusieurs serveurs non-fiables n'a pas encore été étudié.

1.12 Les FAR avec serveurs hétérogènes

Dans la plupart des modèles de files d'attente avec rappel étudiés, tous les serveurs sont supposés être identiques. En effet, les travaux sur les modèles de FAR avec plusieurs serveurs hétérogènes sont rares. Cependant, les systèmes avec appels répétés et des serveurs de différents types (hétérogènes), ayant des temps de service et éventuellement des temps de panne et des temps de réparation différents, apparaissent souvent en pratique.

Notre recherche bibliographique nous a permis de constater qu'on ne trouve dans la littérature que les quelques articles de Pourbabai [44, 120, 121] et ceux de Sztrik et Roszik [33, 34], qui considèrent l'hétérogénéité des serveurs.

En fait, Pourbabai s'est intéressé dans [44, 120, 121], à l'analyse des modèles non-markoviens avec une source infinie et des serveurs hétérogènes fiables. Les résultats sont obtenus par approximation (heuristiques basées sur des simplifications). Par contre, Sztrik et Roszik [33, 34] ont étudié par une méthode numérique, des modèles markoviens avec une source finie et des serveurs hétérogènes non-fiables.

Ainsi, très peu d'articles traitent l'hétérogénéité des serveurs et les résultats sont obtenus seulement par méthode numérique, par approximation ou simulation.

1.13 Les FAR avec serveurs hétérogènes non-fiables

Nous présentons dans cette section, les résultats obtenus pour les modèles markoviens à source finie de clients homogènes et plusieurs serveurs hétérogènes sujets à des pannes et des réparations aléatoires [33, 34].

Dans ce modèle, les appels primaires sont générés par K $(1 < K < \infty)$ sources homogènes suivant un processus quasi-aléatoire de taux λ. Si un des serveurs $(s \geq 1)$ est opérationnel et oisif au moment de l'arrivée de l'appel, alors le service commence. Dans le cas de la discipline de **service aléatoire** (random service), le client est servi par l'un des serveurs disponibles, qui est choisi aléatoirement. Par contre, dans le cas de la stratégie du **service le plus rapide** (SPR), le client est servi par le serveur disponible le plus rapide, et si celui-ci est occupé ou en panne, alors le client vérifie le prochain serveur et ainsi de suite. Par conséquent, la disponibilité et l'oisiveté des serveurs sont toujours vérifiées suivant un certain ordre de priorité qui dépend du taux de service μ_i associé au serveur i.

Par ailleurs, chaque serveur peut tomber en panne durant l'intervalle $(t, t + dt)$, avec la probabilité $\gamma_i dt + o(t)$ en étant occupé, et avec la probabilité $\delta_i d(t) + o(t)$ s'il est oisif. Pour la réparation des serveurs en panne, le réparateur suit une discipline FIFO et le temps de réparation du serveur i suit une loi exponentielle de taux τ_i.

D'autre part, si tous les serveurs sont occupés ou en panne au moment de l'arrivée d'un appel, celui-ci entre en orbite pour rappeler suivant un processus markovien de taux ν.

L'état du système à l'instant t peut être décrit par le processus : [33, 34]

$X(t) = \{(\alpha_1(t), ..., \alpha_s(t); N(t)), t \geq 0\}$, où $N(t)$ est le nombre de sources d'appels répétés et $\alpha_i(t)$, $i = 1, ..., s$, représente l'état du ième serveur à l'instant t.

$$\alpha_i(t) = \begin{cases} 1, & \text{si le serveur } i \text{ est en service,} \\ 0, & \text{si le serveur } i \text{ est opérationnel et oisif,} \\ -1, & \text{si le serveur } i \text{ est en panne.} \end{cases}$$

Les auteurs ont défini les probabilités stationnaires comme suit : [33, 34]

$$P(i_1, ..., i_s, j) = lim_{t\to\infty} P\{\alpha_1(t) = i_1, ..., \alpha_s(t) = i_s, N(t) = j\},\ i_1, ..., i_s = -1, 0, 1,\ j = 0, ..., K^*,$$

où

$$K^* = K - \sum_{i_k, i_k = 1} i_k$$

De plus, nous notons par $C(t)$ le nombre de serveurs occupés et par $A(t)$ le nombre de serveurs disponibles à l'instant t.

Une fois que les probabilités stationnaires sont obtenues, les principaux indices de performance et de fiabilité du modèle peuvent être dérivés en appliquant les formules explicites suivantes : [33, 34]

– **Le nombre moyen de sources d'appels répétés**

$$N = E[N(t)] = \sum_{i_1,...,i_s} \sum_{j=1}^{K^*} jP(i_1, ..., i_s, j)$$

– **L'utilisation du serveur** k

$$U_k = \sum_{\substack{i_1,...,i_s \\ i_k=1}} \sum_{j=0}^{K^*} P(i_1, ..., i_s, j),\ k = 1, ..., s$$

– **Le nombre moyen de serveurs occupés**

$$C = E[C(t)] = \sum_{\substack{i_1,...,i_s \\ K^*>0}} \sum_{j=0}^{K^*} K^* P(i_1, ..., i_s, j) = \sum_{k=1}^{s} U_k$$

– **Le nombre moyen de clients dans le système (en orbite ou en service)**

$$M = E[N(t) + C(t)] = N + C$$

– **L'utilisation du réparateur**

$$U_R = \sum_{\substack{i_1,...,i_s \\ -1 \in \{i_1,...,i_s\}}} \sum_{j=0}^{K^*} K^* P(i_1, ..., i_s, j)$$

– **La disponibilité d'un serveur au moins**

$$A_s = P\{\alpha_k > -1\}, k \in \{1, ..., s\} = 1 - \sum_{j=0}^{K} P(-1, ..., -1, j)$$

– **L'utilisation des sources**

$$U_{SO} = \begin{cases} \frac{E[K-C(t)-N(t);A(t)>0]}{K}, & \text{pour sources bloquées,} \\ \frac{E[K-C(t)-N(t)}{K}, & \text{pour sources intelligentes.} \end{cases}$$

– **L'utilisation totale du système**

$$U_O = C + K.U_{SO} + U_R$$

– **Le taux moyen de génération des appels primaires**

$$\overline{\lambda} = \begin{cases} \lambda E[K - C(t) - N(t); A(t) > 0], & \text{pour sources bloquées,} \\ \lambda E[K - C(t) - N(t)], & \text{pour sources intelligentes.} \end{cases}$$

– **Le temps d'attente moyen**

$$E[W] = \frac{N}{\overline{\lambda}}$$

– **Le temps de réponse moyen**

$$E[T] = \frac{M}{\overline{\lambda}}$$

En examinant toutes les formules dérivées pour ces modèles à source finie, nous remarquons que le calcul des paramètres de performance et de fiabilité, est basé sur celui des probabilités d'état stationnaire. La méthode traditionnelle consiste à déduire les équations de Kolmogorov relatives à ces probabilités, et en utilisant la condition de normalisation, nous résolvons l'ensemble d'équations. Cependant, à cause du fait que l'espace d'états des chaînes de Markov décrites dans les sections précédentes, soit très large (particulièrement pour les modèles hétérogènes), ces calculs sont difficiles à réaliser. Ainsi, pour simplifier la procédure et pour rendre l'étude plus pratique, ces chercheurs ont développé un outil software dit : MOSEL (Modeling, Specification and Evaluation Language) [122, 123], qui permet de formuler ces différents modèles et de calculer les probabilités stationnaires ainsi que les indices de performance et de fiabilité correspondants.

1.14 Conclusion

Le modèle des files d'attente avec rappel permet l'analyse des performances des systèmes téléphoniques, des réseaux informatiques et des réseaux de télécommunication. En fait, les systèmes caractérisés par le phénomène d'appels répétés ont toujours été exclusivement modélisés et analysés par le formalisme de FAR.

Dans cette synthèse, nous nous sommes intéressés particulièrement aux FAR markoviennes multi-serveurs. Nous avons présenté brièvement les problèmes majeurs liés à l'introduction des phénomènes de rappel, ainsi que les techniques qui traitent certains de ces problèmes et les principaux résultats obtenus. En effet, la prise en considération des phénomènes d'appels répétés a rendu les résultats obtenus pour les files d'attente standards inadéquats et a introduit de grandes difficultés analytiques. D'ailleurs, les résultats analytiques des caractéristiques de performance des FAR multi-serveurs à source infinie, sont à ce jour, limités aux systèmes à deux serveurs [14]. Pour ce qui est des modèles avec un nombre de serveurs arbitraire, leur analyse se fait à l'aide des méthodes d'approximation et des modèles tronqués qui permettent de calculer numériquement certains indices de performance [14].

Par ailleurs, les modèles de FAR multi-serveurs à source finie apparaissent dans beaucoup d'applications pratiques [18, 19, 20, 21, 22, 23, 15]. Les travaux dans ce domaine restent assez limités, et les résultats sont obtenus généralement par des méthodes numériques [11, 12, 16, 18].

Récemment, une équipe de chercheurs de l'Université de Debrecen a étudié quelques variantes du modèle de FAR à source finie, et a proposé leur analyse par une méthode numérique basé sur l'utilisation de l'outil MOSEL. Les modèles étudiés sont :

– Le modèle avec sources homogènes et un serveur non-fiable [30, 31] ;
– Le modèle avec sources hétérogènes et un serveur fiable [40, 41, 42] ;

- Le modèle avec sources hétérogènes et plusieurs serveurs fiables [43] ;
- Le modèle avec sources hétérogènes et un serveur non-fiable [32] ;
- Le modèle avec sources homogènes et plusieurs serveurs hétérogènes non-fiables [33, 34].

Ainsi, notre recherche bibliographique nous a permis de constater que **les modèles de FAR à source finie, qui prennent en compte l'hétérogénéité des clients, la non-fiabilité et l'hétérogénéité des serveurs ne sont pas étudiés au jour d'aujourd'hui**. En fait, nous n'avons trouvé dans la littérature aucune référence traitant le modèle de FAR avec clients et serveurs hétérogènes à la fois, ni dans le cas de serveurs fiables ni non-fiables. Ceci est dû essentiellement à la complexité du problème. D'autre part, dans tous les travaux qui considèrent les FAR à sources hétérogènes, les auteurs supposent que chaque source est limitée à un seul client, ce qui constitue une hypothèse contraignante, car dans les systèmes réels avec plusieurs types de clients, les clients de même type constituent une classe unique. Par ailleurs, les serveurs pourraient aussi être partitionnés en classes, dont chacune a des caractéristiques particulières. Par conséquent, il serait plus réaliste de parler de systèmes multi-classes i.e. systèmes avec plusieurs classes de clients et/ou plusieurs classes de serveurs que de systèmes avec clients et/ou serveurs hétérogènes.

Ainsi, nous proposons dans le cadre de cette thèse une approche de modélisation et d'analyse des systèmes mono-classe et multi-classes avec rappel et serveurs éventuellement non-fiables à l'aide des modèles de réseaux de Petri stochastiques généralisés ordinaires et colorés, qui feront l'objet du chapitre suivant.

Chapitre 2

Réseaux de Petri Stochastiques Généralisés

2.1 Introduction

Avec l'avènement du parallélisme, surviennent de nombreux problèmes qui lui sont propres. En particulier : Comment exprimer ou modéliser la communication et la synchronisation dans un système ou entre différents systèmes parallèles ?

Un des premiers à tenter de résoudre ces questions fût le mathématicien allemand **Carl Adam Petri**, qui a développé dans les années 60-62 [124], un modèle spécifique du parallélisme dit : **Les réseaux de Petri** (RdP), afin de modéliser les concepts d'actions asynchrones et concurrentes.
Ce modèle est devenu ensuite célèbre et commença à être utilisé de façon plus large, grâce notamment aux travaux des chercheurs américains qui l'ont utilisé dès les années 70, en particulier par les équipes de A. Holt et J. Dennis, dans le cadre du projet MAC [125, 126] au M.I.T (Massachusetts Institute of Technology).

Dès lors, il bénéficie d'études et de recherches à l'échelle mondiale, ce qui l'a doté de résultats théoriques et pratiques abondants. Ainsi, ce modèle a connu un développement important par l'intérêt qu'il a suscité tant dans le domaine de la recherche théorique que dans le monde industriel.

Un réseau de Petri est un modèle graphique formé d'un ensemble de places représentant les ressources et d'un ensemble de transitions symbolisant les évènements ou les opérations. C'est un modèle puissant qui permet d'exprimer les caractéristiques des systèmes parallèles, telle que la synchronisation, la concurrence, le non-déterminisme et l'allocation de ressources. Ce qui le rend très approprié à la représentation de tels systèmes. Ce modèle permet également de faire l'analyse et la vérification de la correction du système modélisé. Cette analyse permet de révéler des caractéristiques importantes du système concernant sa structure et son comportement, et les résultats de cette analyse sont utilisés pour l'évaluer et l'améliorer.

Grâce à leur représentation graphique simple et aux nombreux résultats accumulés, les réseaux de Petri constituent aujourd'hui un des outils formels les plus avancés et les

plus complets, pour la spécification, la modélisation et la validation des systèmes parallèles complexes, ce qui a permis leur utilisation dans de nombreux domaines, telle que l'analyse des protocoles de communication [127, 128], le pilotage des ateliers de production et la conception des logiciels à temps réel.

Malgré tous leurs avantages, l'évaluation possible n'est que qualitative et aucune relation ou contrainte temporelle n'est prise en compte lors de la modélisation d'un système. En effet, ce modèle ne permet pas la description ou l'analyse du fonctionnement des systèmes dans lesquels le temps apparaît comme un paramètre quantifiable et continu. D'autre part, la vérification des propriétés qualitatives seulement n'est pas suffisante pour s'assurer du bon fonctionnement d'un système modélisé. On doit pouvoir évaluer ses paramètres quantitatifs, tels que le temps moyen d'exécution d'une tâche ou le taux de perte de messages sur un réseau, par exemple.

Par conséquent, l'utilisation des RdP pour répondre aux questions relatives au temps et pour l'évaluation des performances quantitatives nécessite l'introduction d'une temporisation dans le modèle qualitatif de base.

Ceci a motivé Ramchandani à développer les réseaux de Petri temporisés (timed Petri nets) [129] dans lesquels, une durée de temps de franchissement, fixe est associée à chaque transition. Cependant, dans la pratique, il existe peu de cas où les durées associées aux évènements sont déterministes (fixes). En fait, la plupart des systèmes réels sont non-déterministes avec des délais de temps aléatoires. C'est pour cette raison que Florin, Molloy et Natkin ont proposé une nouvelle extension du modèle des RdP dénommée **réseaux de Petri stochastiques (RdPS)** [130, 131, 132]. Dans cette classe de réseaux, on associe une distribution de probabilité aux temps de franchissement des transitions. Ainsi, les délais de tir associés aux transitions sont des variables aléatoires (non déterministes).

Cependant, pour la modélisation de situations dans lesquelles certains évènements prennent une durée de temps aléatoire et d'autres arrivent instantanément, telle qu'aucune temporisation ne peut leur être associée, le modèle des RdPS n'est pas suffisant. Pour cela, une nouvelle extension dite : **réseaux de Petri stochastiques généralisés (RdPSG)** [133] a été proposée. Ce modèle a l'avantage de combiner les transitions temporisées ayant une durée de franchissement aléatoire, avec les transitions immédiates qui sont franchies en un temps nul et qui peuvent être utilisées pour mieux modéliser des aspects purement logiques du comportement du système, telles que certaines contraintes de synchronisation par exemple.

Ainsi, les réseaux de Petri stochastiques généralisés [133, 35] représentent un formalisme puissant pour la modélisation des systèmes parallèles et distribués. Ils conviennent à la modélisation des aspects de synchronisation, de blocage et de concurrence. Par ailleurs, ils permettent l'analyse qualitative ainsi que l'évaluation quantitative de ces systèmes.

Durant les deux dernières décennies, les RdPSG ont reçu une attention considérable des chercheurs dans le domaine de l'analyse des performances et de la fiabilité, et ont été largement appliqués à la modélisation des réseaux informatiques [134], des réseaux téléphoniques mobiles cellulaires [135], des systèmes de communication, de production

[136, 137] et aérospatiaux.

Ce chapitre débute par la description du modèle des réseaux de Petri simples qui constituent la base de tous les réseaux de haut-niveau. Nous présentons les principales définitions et concepts fondamentaux du modèle et nous couvrirons principalement l'étude des propriétés qualitatives qui permettent d'analyser le fonctionnement d'un système modélisé. Dans un deuxième temps, nous développons les points clés des processus stochastiques et plus précisément des chaînes de Markov, car leurs méthodes d'analyse servent de point de départ pour l'analyse des réseaux de Petri stochastiques généralisés. Ensuite, nous abordons l'étude et l'analyse des RdPSG. Dans la section suivante, nous introduisons les réseaux de Petri stochastiques généralisés colorés. Ensuite, nous présentons une comparaison entre les RdPSG et d'autres modèles d'évaluation des performances. Enfin, nous terminerons par une conclusion.

2.2 Réseaux de Petri

Un réseau de Petri est un modèle formé d'un ensemble de places représentant les ressources et d'un ensemble de transitions symbolisant les évènements ou les opérations.

2.2.1 Définition formelle

Définition 1 : Réseau de Petri [138, 128]
Un réseau de Petri R se définit par le tuple $(P, T, Pré, Post)$, où :
- $P = \{p_1, p_2, ..., p_n\}$ *est un ensemble fini de places ;*
- $T = \{t_1, t_2, ..., t_m\}$ *est un ensemble de transitions, disjoint de P ;*
- $Pré$, $Post : P \times T \rightarrow \mathbb{N}$ *est une application d'incidence avant et d'incidence arrière respectivement, correspondant aux arcs ; tels que $Pré(p, t)$ contient la valeur entière associée à l'arc allant de la place p à la transition t, et $Post(p, t)$ contient la valeur entière associée à l'arc allant de la transition t à la place p.*

Remarque :
- $Pré(p, t)$ définit le nombre minimal de marques dans p nécessaire au franchissement de la transition t, et retirées de p en cas de franchissement.
- $Post(p, t)$ définit le nombre apporté à p par le franchissement de t.

Définition 2 : Réseau de Petri marqué [128]
Un réseau de Petri marqué se définit par un couple (R, M) dans lequel R est un réseau de Petri et $M : P \rightarrow \mathbb{N}$ une application appelée ***marquage****.*

Le symbole M définit le marquage du réseau de Petri et $M(p)$ indique le marquage de la place p, c'est-à-dire le nombre (entier) de jetons contenus dans p. Graphiquement, les marques sont représentées par des points ou des nombres à l'intérieur des places.

Le *marquage initial*, noté M_0, donne la valeur initiale du nombre de jetons dans toutes les places, et précise donc l'état global initial du système considéré. Par définition, l'état initial correspond à une distribution des jetons à un moment initial, considéré comme le

début de l'étude.

Représentation d'un réseau de Petri :

Un réseau de Petri peut être facilement représenté par un graphe orienté biparti avec deux types de sommets : les *places* notées graphiquement par des cercles, et les *transitions* notées graphiquement par des traits ou des rectangles. Ces différents sommets sont reliés entre eux par des arcs notés par des flèches, qui joignent des places aux transitions et des transitions aux places. Notons que les arcs ne relient jamais deux sommets de même type.

Par défaut, un arc possède un poids avec la valeur entière 1. De façon plus générale, le poids d'un arc est donné par les fonctions $Pré$ et $Post$, ainsi, il peut être un entier supérieur à 1, mais une telle valeur devra être indiquée, de manière explicite, sur l'arc correspondant.

Les places qui sont reliées par des arcs à une transition t, sont appelées, par simplification, *places d'entrée* de t, et notées ${}^{\cdot}t$. Les places, reliées à une transition par un arc qui joint t à ces places, sont appelées *places de sortie* de la transition, et sont notées $t^{\cdot}$.

2.2.2 Évolution d'un réseau de Petri

À partir d'un marquage initial, un réseau de Petri peut évoluer. L'évolution, c'est-à-dire *la transition d'un marquage vers un marquage suivant*, ne peut avoir lieu que si toutes les conditions relatives à cette transition sont satisfaites. Ceci définit la règle de *tir* ou de *franchissement* d'une transition.

Dans les réseaux de Petri, le comportement pourra évoluer lorsque toutes les places relatives à une transition contiendront un nombre suffisant de jetons et, plus précisément, lorsque le nombre de jetons dans chaque place d'entrée d'une transition sera supérieur ou égal au poids de l'arc joignant cette place à la transition : la transition sera alors franchissable (sensibilisée ou tirable). Lorsqu'elle sera franchie (ou tirée), son franchissement (ou tir) définira le marquage suivant du réseau. Le tir dépend donc des jetons.

Ainsi, la règle d'évolution des réseaux marqués leur donne une dynamique, tout en précisant comment les transitions permettent de modifier l'état du système.

Définition 3 : Sensibilisation et franchissement [138, 128]
Une transition t est franchissable (tirable ou sensibilisée) pour un marquage M, si et seulement si : $\forall p \in P,\ M(p) \geq Pré(p,t)$.
Le franchissement (ou le tir) de la transition t conduit à un nouveau marquage M' défini par : $\forall p \in P, M'(p) = M(p) - Pré(p,t) + Post(p,t)$.
On note ceci par $M[t\rangle M'$.

Ainsi, la règle de tir signifie que le nouveau marquage M' sera obtenu, à partir du marquage précédent M, en supprimant d'abord, dans les places d'entrée de t, le nombre de jetons indiqué sur les arcs entrants de t ($Pré(p,t)$), et en ajoutant ensuite, à chaque place de sortie p' de t, le nombre de jetons correspondant au poids indiqué sur l'arc sortant

de t vers p' $(Post(p', t))$.

Remarque : Cette règle d'évolution repose sur l'hypothèse fondamentale d'indivisibilité : le tir d'une transition d'un RdP s'effectue de façon *atomique ou indivisible* [128].

En appliquant la règle de franchissement, il est possible d'obtenir l'ensemble des suites d'évolution du système à partir du marquage initial. Cet ensemble de suites définit l'ensemble des marquages accessibles et peut être représenté sous forme de graphe, le graphe des marquages accessibles.

Définition 4 : Séquence de franchissements [128]
Soit (R, M_0) un réseau de Petri marqué. Une ***séquence de franchissements*** $\sigma \in T^*$ *est une séquence ordonnée de transitions $t_1, t_2, ..., t_n$ tel que : $\exists n$ marquages : $M_1, M_2, ..., M_n$ et $\forall i \in [1, n], M_{i-1}[t_i\rangle M_i$.*

Définition 5 : Marquage accessible [128]
Soit (R, M_0) un réseau de Petri marqué. Un marquage M est accessible si et seulement si $\exists \sigma \in T^*$ une séquence de franchissements telle que $M_0[\sigma\rangle M$.

La manière la plus simple de définir le comportement d'un réseau est de considérer l'ensemble des marquages accessibles depuis le marquage initial. En fait, les propriétés comportementales sont des propriétés dynamiques qui dépendent du marquage initial du modèle et leur vérification est basée sur une construction préalable du graphe des marquages accessibles. Ce graphe constitue la représentation formelle la plus usuelle du comportement du réseau et décrit complètement la sémantique du système modélisé.

Définition 6 : Ensemble d'accessibilité [128]
Soit (R, M_0) un réseau de Petri. L'ensemble des marquages accessibles ou ***ensemble d'accessibilité*** *d'un réseau, noté $A(R, M_0)$ ou A, est l'ensemble des marquages atteints par une séquence de franchissement :*
$A(R, M_0) = \{M \in \mathbb{N}^P | \exists \sigma \in T^*$ tel que $M_0[\sigma\rangle M\}$

Définition 7 : Graphe d'accessibilité [128]
Soit (R, M_0) un réseau de Petri. Le ***graphe d'accessibilité*** *(ou graphe des marquages accessibles) d'un réseau, noté $G(R, M_0)$ est défini comme le graphe dont les noeuds (ou sommets) sont les marquages accessibles de $A(R, M_0)$ et dont les arcs, étiquetés par les noms des transitions, sont définis par la relation de tir entre les marquages. Donc, un arc étiqueté par t joint M à M' si et seulement si $M[t\rangle M'$.*

2.2.3 Concurrence et conflit entre transitions

Le phénomène de concurrence dans un système provient du fait que plusieurs évènements peuvent s'exécuter en parallèle. Le modèle des réseaux de Petri permet de représenter aisément les systèmes en présence d'un tel phénomène.

Définition 8 : Transitions concurrentes [139]

Soit R un réseau de Petri, une transition est dite ***concurrente*** *si son franchissement n'est désactivé par le franchissement d'aucune autre transition du modèle R.*
Deux transitions t_1 et t_2 sont dites ***concurrentes***, ***indépendantes*** *ou* ***asynchrones*** *si elles ne partagent aucune place en entrée.*

Un phénomène dual au précédent est celui du conflit entre transitions.

Définition 9 : Conflit entre transitions [138]
Deux transitions t_1 et t_2 sont en ***conflit structurel****, si et seulement si elles ont au moins une place commune en entrée. Autrement dit, $\exists p \in P : Pré(p, t_1) \times Pré(p, t_2) \neq 0$*
Elles sont en ***conflit effectif*** *pour un marquage M si de plus :*
$M[t_1\rangle$ et $M[t_2\rangle$ et $\exists p \in P : M(p) < Pré(p, t_1) + Pré(p, t_2)$
Un conflit effectif correspond donc à un choix exclusif entre deux franchissements.

2.2.4 Propriétés usuelles des réseaux de Petri

Un des intérêts majeurs des modèles formels est la possibilité de décrire sans ambiguïté le comportement d'un système, de définir des propriétés du système modélisé de manière formelle, et de vérifier ces propriétés à l'aide d'algorithmes ou d'heuristiques [128].

Un réseau de Petri associé à un système est une formalisation de la description de ce système et des services rendus. L'analyse du modèle et la vérification de ses propriétés constituent l'évaluation qualitative du système correspondant, qui est une étape primordiale dans la conception de tout système complexe. En effet, ceci permet d'une part de vérifier si le système réalise l'ensemble des fonctions qu'il doit assurer et d'autre part, s'il n'est pas sujet à des situations indésirables tel que l'interblocage par exemple.

Dans ce qui suit, nous définirons les propriétés générales les plus significatives à partir du graphe d'accessibilité.

Une propriété importante que nous allons traiter en premier lieu, concerne la bornitude du réseau. Nous allons caractériser la possibilité pour une place d'accumuler une quantité bornée ou non de marques au cours de l'évolution d'un réseau.
En effet, cette propriété intervient souvent comme premier élément lors de l'analyse d'un système, notamment parce que le marquage d'une place peut s'interpréter comme la quantité de ressources disponibles ou encore le nombre de ressources nécessaires au fonctionnement de ce système. Il y a aussi des cas où la place considérée représente un objet du système dont la charge ne peut pas dépasser une certaine valeur, telle que la taille de la population d'un système fermé par exemple.

Définition 10 : Bornitude [138, 128]
Un réseau de Petri marqué (R, M_0) est borné ssi :
$\exists k \in \mathbb{N}$, $\forall m \in A(R, m_0), \forall p \in P, M(p) \leq k$.

Ainsi, si le réseau est borné, une borne du réseau est un entier supérieur ou égal à tout marquage accessible d'une place quelconque.

PROPOSITION : [128]
Soit (R, M_0) un réseau de Petri. Le réseau (R, M_0) est borné si et seulement si $A(R, M_0)$ est fini.

D'après la définition et la proposition citées ci-dessus, nous constatons que la vérification de la propriété de bornitude repose sur le graphe d'accessibilité. Or, ce graphe peut être infini (cas des réseaux non bornés) ou même de taille très importante, ce qui rendra la vérification impossible ou impraticable (très coûteuse). Pour remédier à ce problème, la vérification de cette propriété peut se faire à l'aide de l'algèbre linéaire, ou bien à l'aide des séquences de franchissement particulières, et ce sans la construction du graphe d'accessibilité.

Définition 11 : Séquence répétitive croissante [138]
Une séquence de franchissement σ est dite répétitive croissante pour la place p si pour tout couple de marquages (M, M') tel que $M[\sigma\rangle M'$, nous avons $M' \geq M$ avec $M'(p) > M(p)$.

Propriété : [138]
Un réseau marqué (R, M_0) est non borné, si et seulement s'il existe une séquence répétitive croissante σ, pour une place p du réseau et un marquage accessible $M \in A(R, M_0)$ tel que : $M[\sigma\rangle$.

Un autre aspect du fonctionnement d'un réseau dont nous nous préoccupons est de savoir si le système s'arrête dans certaines situations. Ainsi, une propriété importante à vérifier concerne *l'absence de blocage* dans l'activité de tout ou une partie des transitions du réseau. C'est en effet une question essentielle de savoir si le système modélisé est capable de réaliser l'ensemble des fonctions pour lesquelles il a été conçu. Par exemple, un système téléphonique ne doit jamais se bloquer, quel que soit le comportement des utilisateurs.

Définition 12 : Réseau sans blocage [138]
Un réseau de Petri (R, M_0) est dit sans blocage si tout marquage accessible depuis M_0 n'est pas un marquage mort.
Un marquage mort (ou marquage puits) est un marquage sans transition franchissable.

Une autre propriété essentielle en rapport avec l'absence de blocage est la vivacité.

Définition 13 : Vivacité [128]
Un réseau de Petri (R, M_0) est vivant si :
$\forall M \in A(R, M_0), \forall t \in T, \exists M' \in A(R, M)$ *tel que* $M'[t\rangle$.
Autrement dit, $\forall M \in A(R, M_0), \forall t \in T, \exists \sigma \in T^*$ *tel que* $M[\sigma.t\rangle$.

La vivacité est une propriété importante pour traduire le bon fonctionnement d'un système. D'ailleurs, un réseau vivant modélise un système en fonctionnement permanent sans aucun blocage, et dont toutes ses actions peuvent être exécutées. La non satisfaction de cette propriété caractérise des situations de blocage total (état du système à partir duquel aucune évolution n'est plus possible), ou de blocage partiel c'est-à-dire qu'une partie

du système est inacessible.

Remarque : Un réseau sans blocage signifie que le système est en fonctionnement permanent mais pas nécessairement vivant, car la propriété de vivacité est plus forte que le non blocage.

Une autre propriété importante consiste à vérifier la possibilité de toujours revenir à un état donné, qui représente la réinitialisation du système, d'où la définition suivante.

Définition 14 : Existence d'un état d'accueil [128]
Un réseau de Petri (R, M_0) *admet un état d'accueil* M_a *si :*
$\forall M \in A(R, M_0), \exists \sigma \in T^*$ tel que $M[\sigma\rangle M_a$
Autrement dit, un état d'accueil est un état accessible quel que soit l'évolution du réseau.

Une manière très simple de vérifier l'existence d'un état d'accueil, consiste à vérifier si le graphe d'accessibilité est fortement connexe.

Définition 15 : Graphe fortement connexe [128]
Une composante fortement connexe d'un graphe est un sous graphe tel qu'il existe un chemin (orienté) entre tout point A et tout point B de ce sous graphe.
Une composante fortement connexe est dite terminale, si aucun sommet de celle-ci ne possède de successeurs dans une autre composante fortement connexe.
Un graphe est dit fortement connexe s'il possède une seule composante fortement connexe.

Proposition : [128]
Soit (R, M_0) un réseau de Petri borné. Le réseau (R, M_0) a un état d'accueil si et seulement si le graphe d'accessibilité comprend une seule composante fortement connexe terminale.

Remarque : Si le marquage initial est un état d'accueil alors le système modélisé est **réinitialisable**. La réinitialisation implique que le modèle peut se réinitialiser lui-même. Ceci est important pour la reprise automatique en cas d'erreur ou de panne, car elle garantit le fait que le système, après un nombre fini d'étapes, retournera à un état admissible. Par contre, si le modèle de RdP n'est pas réinitialisable, il ne peut y avoir de reprise automatique, ainsi l'intervention manuelle est nécessaire.

L'originalité et l'intérêt des RdP tiennent surtout aux nombreuses techniques de validation autres que le simple examen du graphe des marquages accessibles, souvent coûteux et quelquefois même, irréalisable. Parmi ces méthodes alternatives, il y a d'une part le calcul algébrique qui permet de générer des invariants et d'autre part la réduction de réseaux qui ramène l'étude des propriétés à un réseau de taille plus petite.

2.2.5 Analyse par les invariants linéaires

L'analyse d'un système modélisé par un réseau de Petri, peut se faire selon différentes approches dont chacune a ses avantages et ses inconvénients.

Les méthodes d'analyse, basées sur la construction et le parcours de tout ou une partie du graphe d'accessibilité, sont appelées *méthodes comportementales*. Ces méthodes ont l'avantage d'être conceptuellement simples, et permettent de déterminer les propriétés générales définies dans la section précédente, dites propriétés comportementales. Néanmoins, ces méthodes présentent certains inconvénients : elles ne s'appliquent qu'aux réseaux qui engendrent un nombre fini d'états, leur complexité en temps et en espace dépend de la taille du graphe d'accessibilité (très supérieure à celle du réseau) et elles nécessitent de fixer le marquage initial, ainsi, l'analyse du réseau doit être refaite à chaque fois qu'on change d'état initial.

Ainsi, d'autres méthodes qui tirent parti de la structure du réseau afin de diminuer la complexité de l'analyse ou de la rendre applicable à un réseau dont le marquage initial n'est pas fixé ont été développées. Ces méthodes dites *structurelles* sont basées sur le calcul des invariants linéaires (ou flots) pour déterminer les propriétés structurelles du réseau indépendamment du marquage initial.

Dans ce qui suit, nous présentons quelques notions de base de l'analyse par les invariants linéaires, car c'est la méthode utilisée par l'outil GreatSPN, que nous avons utilisé pour la validation numérique.

Définition 16 : [128]

- *Un P-flot est un vecteur non nul* $v \in \mathbb{Z}^P$ *qui vérifie* $v^t.C = \overrightarrow{0}$.
- *Un P-semiflot est un vecteur non nul* $v \in \mathbb{N}^P$ *qui vérifie* $v^t.C = \overrightarrow{0}$.
- *Un T-flot est un vecteur non nul* $v \in \mathbb{Z}^T$ *qui vérifie* $C.v = \overrightarrow{0}$.
- *Un T-semiflot est un vecteur non nul* $v \in \mathbb{N}^T$ *qui vérifie* $C.v = \overrightarrow{0}$.

où : $C = Post - Pré$ est la matrice d'incidence.

Un P-flot (respectivement, un P-semiflot) est une somme pondérée de places à coefficients entiers (respectivement, naturels). Un P-flot peut donc servir à obtenir une valeur entière, à partir d'un marquage quelconque, en pondérant les marquages de chaque place et en les sommant. Un T-semiflot peut s'interpréter comme le vecteur d'occurrences d'une séquence de transitions (i.e. le vecteur qui contient le nombre de franchissements de chaque transition dans la séquence), tandis qu'un T-flot peut s'interpréter comme la différence de deux vecteurs d'occurrences.

Les invariants (flots et semi-flots) possèdent de nombreuses applications. Par exemple, la recherche des P-semiflots se justifie par le fait que toutes les places p du support d'un P-semiflot v sont bornées, et ceci quel que soit le marquage initial [128]. De même, à partir d'un invariant où $M(p_1) + ... + M(p_n) = 1$, on déduit aisément que p_1,...,p_n, ne peuvent être simultanément marquées.

Plus généralement, les invariants sont à l'origine de nombreuses conditions nécessaires et/ou suffisantes de propriétés comportementales. Afin de mettre en exergue cette caractéristique, nous introduisons à présent deux propriétés structurelles des réseaux de Petri.

Définition 17 : Réseaux conservatifs, Réseaux consistants [128]
Soit R un réseau de Petri :
- *R est conservatif s'il existe un P-semiflot υ tel que $\|\upsilon\| = P$.*
- *R est consistant s'il existe un T-semiflot υ tel que $\|\upsilon\| = T$.*

où $\|\upsilon\|$ désigne le support de υ défini par : $\|\upsilon\| = \{i \mid \upsilon(i) \neq 0\}$.

La proposition suivante met en évidence les liens qui unissent les propriétés comportementales (vivacité et bornitude) aux propriétés structurelles (conservation et consistance).

Proposition : [128]
Soit R un réseau de Petri. On a :
- *$\exists \upsilon \in \mathbb{N}^P, \|\upsilon\| = P$ et $\upsilon^t.C \leq \overrightarrow{0}$ si et seulement si R est structurellement borné. En particulier, R conservatif ($\upsilon^t.C = \overrightarrow{0}$) $\Rightarrow$ R structurellement borné ;*
- *(R, M_0) borné et vivant $\Rightarrow$ R consistant.*

Un réseau structurellement borné est un réseau borné pour tout marquage initial.

Cette proposition est particulièrement intéressante, car pour l'analyse d'un système modélisé par un réseau de Petri, l'analyse structurelle peut être réalisée en premier lieu, permettant ainsi de vérifier certaines propriétés du système avant la génération de l'espace d'états correspondant. Ainsi, on peut déduire si le réseau est borné ou pas. En fait, la vérification de la bornitude constitue une étape préliminaire importante, car il est impossible de calculer tout l'espace d'états du modèle, si celui-ci est non borné.

De la même manière, le calcul des T-semiflot et la vérification de la propriété structurelle de consistance ont un rapport avec la propriété de vivacité. En fait, *une condition nécessaire (mais pas suffisante) pour qu'un réseau borné soit vivant est qu'il soit consistant*. Ceci est un résultat direct de la dernière proposition à partir de laquelle on peut conclure que ***tout réseau borné et non consistant est non vivant***. Cependant, pour vérifier si le modèle est vivant, le graphe d'accessibilité doit être généré. En se basant sur ce graphe, les différentes propriétés qualitatives comportementales peuvent être examinées.

Ainsi, avec les techniques fondées sur l'algèbre linéaire, les résultats paraissent de portée limitée. En effet, cette méthode permet de vérifier la bornitude d'un réseau de Petri, ainsi que certaines propriétés dites de *sûreté*. Cependant, pour les autres propriétés comportementales, on ne dispose que des conditions nécessaires (mais pas suffisantes). Par exemple, *l'existence d'un T-semiflot est une condition nécessaire pour que le réseau soit réinitialisable*. Par conséquent, l'application de la méthode des invariants seule ne permet pas de valider le système étudié. C'est la raison pour laquelle dans divers outils d'analyse des réseaux de Petri tel que le GreatSPN par exemple, les deux approches d'analyse sont combinées.

Il est important de souligner que l'efficacité des méthodes basées sur le graphe d'accessibilité dépend fortement de la taille de celui-ci, et que ces méthodes ne s'appliquent qu'à des réseaux bornés. D'autre part, toute méthode basée sur la construction du graphe d'accessibilité possède une complexité que l'on ne peut prédire. Ceci justifie l'intérêt des

méthodes structurelles. Cependant, aucune méthode ne peut fournir des résultats significativement meilleurs que ceux de la construction du graphe d'accessibilité [128].

Par ailleurs, dans certaines applications, le modèle des réseaux de Petri tel qu'il a été défini pourrait être limité voir même incapable de spécifier et de modéliser certaines contraintes, tel que le test à zéro. Ainsi, plusieurs extensions ont été proposées pour augmenter la puissance du formalisme et permettre d'aborder des applications plus complexes. Une de ces extensions est celle des *réseaux de Petri avec arcs inhibiteurs.*

2.2.6 Réseaux de Petri à arcs inhibiteurs

Le pouvoir d'expression des réseaux de Petri est proche d'un langage de programmation travaillant sur des entiers. Il manque cependant aux réseaux le test d'égalité entre le marquage d'une place et une valeur fixe, pour parvenir en faire un véritable langage de programmation [128]. C'est à cette fin qu'ont été introduits les réseaux à arcs inhibiteurs. Dans ce modèle, les matrices d'incidence sont complétées par une matrice d'inhibition qui impose que le marquage d'une place soit strictement inférieur à une valeur donnée, pour permettre le franchissement d'une transition.

Définition 18 : Réseau de Petri à arcs inhibiteurs [128]
Un réseau de Petri à arcs inhibiteurs est défini par un tuple $R = \langle P, T, Pré, Post, Inh \rangle$ *où :*

- P *est un ensemble fini de places et* T *un ensemble fini de transitions ;*
- $Pré$, $Post : P \times T \rightarrow \mathbb{N}$, *sont les fonctions d'incidence avant et d'incidence arrière respectivement ;*
- $Inh : P \times T \rightarrow (\mathbb{N} \setminus \{0\})$, *est la fonction d'inhibition.*

Définition 19 : Franchissement dans un réseau à arcs inhibiteurs [128]
Soit M *un marquage d'un réseau de Petri à arcs inhibiteurs et* t *une transition :*

- t *est franchissable à partir de* M *si et seulement si :*
  $\forall p \in P,\ M(p) \geq Pré(p,t)$ *et* $M(p) < Inh(p,t)$.
- *le franchissement de* t *à partir de* M *conduit au marquage* M' *défini par :*
  $\forall p \in P,\ M'(p) = M(p) - Pré(p,t) + Post(p,t)$.

Ainsi, seule la condition de franchissabilité est modifiée. La représentation graphique des réseaux à arcs inhibiteurs ne diffère de celle des réseaux standards, que par le fait qu'un arc inhibiteur est représenté par une flèche avec un petit cercle à l'extrémité.

Nous nous intéressons dans cette thèse, à un modèle particulier des réseaux de Petri, dits : *les réseaux de Petri stochastiques généralisés.* Pour cela, nous présentons dans la section suivante, les points clés des processus stochastiques et plus précisément des chaînes de Markov, qui sont la base de l'analyse de ces réseaux.

2.3 Chaînes de Markov

Les processus stochastiques sont des modèles mathématiques utiles pour la description des systèmes comportant des phénomènes de nature probabiliste en fonction d'un paramètre qui est généralement le temps.

Dans cette section, nous présentons une classe particulière de processus stochastiques dits : **les processus markoviens**. La caractérisation fondamentale de ces derniers, appelés aussi les processus sans mémoire, est que tout l'historique du processus est résumé par l'état courant. La connaissance de cet état suffit à prédire l'évolution future du système.

Dans notre travail, on s'intéressera particulièrement aux processus markoviens à espace d'états dénombrable, appelés : **chaînes de Markov**.

Toute la théorie des files d'attente et des réseaux de Petri stochastiques est fondée sur celle des probabilités et des processus stochastiques. Il est donc nécessaire de donner quelques rappels concernant les variables aléatoires, les processus stochastiques et les chaînes de Markov.

2.3.1 Variable aléatoire

Une variable aléatoire est une fonction X réelle définie sur un espace de probabilité. L'ensemble des valeurs possibles de la fonction constitue l'espace d'états E de la variable aléatoire.

On distingue deux grands types de variables aléatoires : les variables aléatoires discrètes et les variables aléatoires continues.

Définition 20 : Variable aléatoire continue [140]
Une variable aléatoire continue est définie par sa fonction densité de probabilité $f_x(x)$ *pour* $x \in]-\infty, +\infty[$, *qui est telle que* $\int_{-\infty}^{+\infty} f_x(x)dx = 1$. *Ainsi, l'espace d'états* $E \subseteq \mathbb{R}$. $f_x(x)dx$ *(dx petit) est donc la probabilité pour que* X *soit comprise entre* x *et* $x + dx$. *Dans le cas où* $E \subseteq \mathbb{R}^+$, *soit* $f_x(x) = 0$, *pour* $x \in]-\infty, 0[$, *on dit que* X *est une variable aléatoire continue à valeurs positives. Dans ces conditions, on remplace souvent la variable* x *par* t *pour faire référence au temps.*

Nous citons comme exemple de variable aléatoire continue, la variable de **loi exponentielle**. Cette loi est sans aucun doute, la plus utilisée dans la théorie des files d'attente et des processus stochastiques. Elle permet généralement de modéliser des temps dont la valeur n'est pas constante.

Définition 21 : [140]
Une variable aléatoire continue T *à valeurs positives, de loi exponentielle de paramètre* λ, *est définie par sa densité de probabilité comme suit :*

$$f_T(t) = \begin{cases} \lambda e^{-\lambda t} & \text{pour } t \geq 0, \\ 0 & \text{pour } t < 0 \end{cases}$$

Propriété : La moyenne d'une variable aléatoire T de loi exponentielle de paramètre λ est : $E[T] = \frac{1}{\lambda}$.

Propriété : sans mémoire [140]
La variable aléatoire exponentielle est la seule variable aléatoire continue à posséder la propriété sans mémoire. Elle est définie comme suit : $P[T \leq t + t_0 \mid T > t_0] = P[T \leq t]$. En effet, cette propriété essentielle fait que cette loi exponentielle soit très largement utilisée.

Interprétation de la propriété sans mémoire : Si T est une variable aléatoire exponentielle, le passé de cette variable ne permet en aucun cas de prédire l'avenir. Considérons l'exemple suivant pour lequel le temps entre deux passages consécutifs d'un bus à un arrêt est distribué de façon exponentielle. Supposons que l'origine des temps est choisie de telle façon qu'un bus vient juste de quitter l'arrêt au temps $t = 0$. Le fait de savoir qu'on a déjà attendu un temps t_0 et que le prochain bus n'est toujours pas arrivé, ne nous permet pas d'affirmer qu'il a plus de chances d'arriver rapidement. En effet, la probabilité pour que le bus arrive avant t unités de temps supplémentaires, sachant qu'il n'est pas arrivé au temps t_0 : $P[T \leq t + t_0 | T > t]$ est égale à la probabilité pour que le bus soit arrivé pendant les t premières minutes : $P[X \leq t]$. Donc, si les bus passent en moyenne toutes les $15mn$ et qu'on en attend un depuis 1 heure, tout ce que l'on peut dire c'est qu'il mettra $15mn$ en moyenne pour arriver.

2.3.2 Processus stochastiques

Les processus stochastiques sont des modèles mathématiques utiles pour la description des phénomènes qui dépendent du hasard.

Définition 22 : Processus stochastique
Un processus stochastique $\{X(t), t \in T\}$ *est une famille de variables aléatoires définies sur le même espace de probabilité, prenant des valeurs dans l'espace d'états* E *et indexée par le paramètre* t.
L'ensemble des temps T *de même que* E*, peuvent être discrets ou continus* [140].

Définition 23 : Processus markoviens [141]
Un processus markovien est un processus stochastique qui satisfait la ***propriété de Markov*** *ou encore la propriété de* ***perte de mémoire*** *définie comme suit :*
$P[X(t) \leq x | X(t_n) = x_n, ..., X(t_0) = x_0] = P[X(t) \leq x | X(t_n) = x_n]$ *pour tout* $t > t_n > t_{n-1} > ... > t_0$.

Interprétation : La propriété de Markov définit un processus stochastique dont le comportement dans le futur (à l'instant t) ne dépend que de l'état courant (à l'instant t_n) et pas de l'historique (les états atteints aux instants : $t_{n-1}, ..., t_0$). Autrement dit, la caractéristique fondamentale de ces processus est que la connaissance de l'état courant suffit à prédire l'évolution future du système. Ainsi, les processus markoviens sont des processus sans mémoire.

Processus de Poisson : Le processus de Poisson est, au même titre que la loi ex-

ponentielle pour les variables aléatoires, le processus stochastique le plus utilisé dans la théorie des files d'attente. Il modélise généralement le processus d'arrivée des clients dans un système. On parlera alors d'*arrivées poissonniennes.*

Définition 24 : [140]
Le processus de Poisson est un processus stochastique à espace d'états discret et à temps continu, tel que les temps d'interarrivées sont des variables aléatoires indépendantes et identiquement distribuées selon une loi exponentielle.

Interprétation : Si les arrivées des clients dans un système suivent un processus de Poisson de taux λ, alors les interarrivées ont une distribution exponentielle de paramètre λ.

Processus de naissance et de mort : [140]
Le processus de naissance et de mort est un cas particulier d'un processus markovien dans lequel, les seules transitions possibles, à partir d'un état e_k, sont vers les états voisins e_{k-1} et e_{k+1}. Ce processus permet de prouver beaucoup de résultats importants et intéressants dans la théorie des files d'attente.

2.3.3 Chaînes de Markov

Dans cette section, nous considérons les processus markoviens à espace d'états discret (dénombrable), connus sous le nom de *chaînes de Markov.* Ce modèle fournit des outils simples de modélisation et d'analyse d'une classe particulière de systèmes à évènements discrets.

Définition 25 : Chaîne de Markov [141, 140]
Une chaîne de Markov est un processus markovien à espace d'états discret ($E \subset \mathbb{N}$).
- *Si $T \subset \mathbb{N}$ (ou $\mathbb{Z}$), le processus est une chaîne de Markov à temps discret.*
- *Si $T \subset \mathbb{R}$ (ou $[0, +\infty[$), le processus est une chaîne de Markov à temps continu.*

Définition 26 : Processus semi-markoviens [128]
Un processus semi-markovien est une extension des chaînes de Markov à temps continu où les temps de séjour dans les états ont une distribution quelconque.

2.3.4 Chaînes de Markov à temps discret (CMTD)

Définition 27 : Chaîne de Markov à temps discret [140]
Un processus stochastique $\{X_n\}_{n \in IN}$ est une chaîne de Markov à temps discret (CMTD) si et seulement si :

$$P[X_n = j | X_{n-1} = i_{n-1}, X_{n-2} = i_{n-2}, ..., X_0 = i_0] = P[X_n = j | X_{n-1} = i_{n-1}]$$

Ainsi, la probabilité pour que la chaîne soit dans un certain état à la nième étape du processus ne dépend donc que de l'état du processus à l'étape précédente (la n-1 ième étape) et pas des états dans lesquels il se trouvait aux étapes antérieurs (les étapes j pour $j = 0, ..., n-2$).

Une classe particulière intéressante des CMTD est celle des **CMTD homogènes**. Celles-ci sont telles que les probabilités $P[X_n = j \mid X_{n-1} = i]$ ne dépendent pas de n. On peut alors définir la *probabilité de transition* d'un état i vers un état j, p_{ij}, qui ne dépend donc pas de l'étape n :

$p_{ij} = P[X_n = j \mid X_{n-1} = i], \quad \forall n \in \mathbb{N}$, où $\sum_{j \in E} p_{ij} = 1$.
Notons que $p_{ii} \geq 0$ car il est possible de rester dans un certain état i entre deux étapes consécutives.

2.3.4.1 Représentation graphique

Il est commode d'utiliser une représentation graphique d'une CMTD sous forme d'un graphe orienté, dans lequel on associe à chaque état de la chaîne un noeud et à chaque transition possible entre deux états, un arc orienté pondéré par la probabilité de transition.

2.3.4.2 Matrice de transition

La matrice de transition $P = [p_{ij}]_{i,j \in E}$ d'une CMTD, est une matrice carrée d'ordre n correspondant à la cardinalité de l'espace d'état E, et dont les éléments correspondent aux probabilités de transition entre états, où l'élément p_{ij} représente la probabilité de se rendre dans l'état j en quittant l'état i.

2.3.4.3 Analyse des CMTD

L'analyse du régime permanent se résume à deux questions : la limite du vecteur des probabilités stationnaire existe-t-elle ? Et si oui, comment la calculer ? Pour répondre à ces questions, certaines propriétés de la chaîne de Markov doivent être vérifiées.

Définition 28 : CMTD irréductible [140]
Une chaîne de Markov à temps discret est dite irréductible ssi de tout état i on peut atteindre tout état j (en un nombre fini d'étapes) :
$\forall i, j \in E$, $\exists m > 1$ *tel que* $p_{i,j}^{(m)} \neq 0$
où : $p_{i,j}^{(m)}$ *désigne la probabilité de transition de l'état i à l'état j en m étapes :*
$p_{i,j}^{(m)} = P[X_{n+m} = j \mid X_n = i]$, $\forall n \in \mathbb{N}$.

Définition 29 : CMTD apériodique [140]
Un état j est périodique, si on ne peut y revenir qu'après un nombre d'étapes multiples de $k > 1$: $\exists k > 1$ *tel que* $p_{i,j}^{(m)} = 0$ *pour m non multiple de k.*
La période de l'état j est alors le plus grand entier k vérifiant cette propriété.
La période d'une CMTD est égale au PGCD de la période de chacun de ses états. Une CMTD est dite périodique si sa période est supérieure à 1 (et ***apériodique*** *si sa période est égale à 1).*

Propriété : [140, 142]
Une chaîne de Markov finie, apériodique et irréductible est ergodique.

Définition 30 : Stationnarité[143]
Une chaîne de Markov admet une distribution stationnaire sur ses états si elle est ergodique.

Une distribution stationnaire est équivalente à une distribution qui, lorsque atteinte, le reste durant toutes les étapes suivantes du processus. Ainsi, le système qu'elle modélise se stabilise après l'écoulement d'un temps infini. Il y a alors possibilité d'obtenir plusieurs paramètres de performance stationnaires du système.

Ainsi, si la CMTD est ergodique, le vecteur π des probabilités stationnaires existe toujours et est solution du système d'équations linéaires suivant : [140]

$$\begin{cases} \pi = \pi.P \\ \sum_{i \in E} \pi_i = 1 \end{cases}$$

2.3.5 Chaînes de Markov à temps continu (CMTC)

On considère un processus stochastique $\{X(t)\}_{t \geq 0}$ à espace d'état discret et à temps continu. L'espace des états peut être de dimension finie ou infinie (mais dénombrable car discret).

Définition 31 : Chaîne de Markov à temps continu [140]
Un processus stochastique $\{X(t)\}_{t \geq 0}$ est une chaîne de Markov à temps continu si et seulement si :

$P[X(t_n) = j \mid X(t_{n-1}) = i_{n-1}, X(t_{n-2}) = i_{n-2}, ..., X(t_0) = i_0] = P[X(t_n) = j \mid X(t_{n-1}) = i_{n-1}]$, $\forall n$ *et* $\forall t_0 < t_1 < ... < t_n$.

Contrairement à ce qui se passe pour les chaînes de Markov à temps discret, on ne dispose jamais, dans le cas d'une CMTC, d'un historique complet du processus. En effet, on observe celui-ci à certains instants dans le temps, choisis aussi nombreux que l'on veut et répartis comme on veut, mais on ignore ce qui se passe entre deux observations. Cependant, la propriété citée ci-dessus permet d'affirmer qu'une connaissance très détaillée du passé ne fournit pas plus d'information quant à l'évolution future du processus que la connaissance de la dernière observation. En d'autres termes, la probabilité pour que la chaîne soit dans un certain état à l'instant t_n ne dépend donc que de l'état du processus à l'instant t_{n-1} et pas des états dans lesquels il se trouvait aux instants antérieurs $(t_0, ..., t_{n-2})$.

Une classe particulièrement intéressante est celle des **CMTC homogènes**.

Définition 32 : CMTC homogène [140]
Une CMTC homogène est caractérisée par le fait que les probabilités $P[X(t_n) = j \mid X(t_{n-1}) = i]$ ne dépendent pas des instants d'observation t_n et t_{n-1}, mais uniquement de la durée $(t_n - t_{n-1})$ qui sépare les deux observations.
On peut alors définir la probabilité $p_{ij}(t)$ de se trouver en j alors qu'on était en i, t instants de temps plus tôt, comme suit :

$p_{ij}(t) = P[X(s+t) = j \mid X(s) = i] \; \forall s \geq 0.$

2.3.5.1 Caractérisation d'une CMTC

L'évolution d'une CMTC peut se voir comme une répétition de deux phases [140] :
- on reste un certain temps distribué selon une loi exponentielle dans un état ;
- lorsque l'on quitte cet état, on va vers un état de destination qui ne dépend ni du temps passé dans l'état ni du chemin par lequel on est arrivé dans l'état. Les transitions sont plutôt probabilistes, d'où la définition suivante :

Définition 33 : [140]
Une chaîne de Markov à temps continu est un processus stochastique à espace d'états discret et à temps continu tel que :
- le temps de séjour dans tout état i a une distribution exponentielle de taux μ_i ;
- l'évolution future dépend seulement de l'état présent et pas de l'historique ;
- les transitions d'un état i vers les autres états sont probabilistes.

Dans les CMTC, en raison de l'absence de mémoire de la loi exponentielle, l'évolution à tout instant est uniquement conditionnée par son état courant.

Remarque : Dans les CMTC, nous considérons les taux de transition au lieu des probabilités de transition d'état. En fait, le temps de transition de l'état i à l'état j est distribué selon une loi exponentielle de taux $\mu_{ij} = \mu_i.p_{ij}$, où p_{ij} est la probabilité de se rendre dans l'état j en quittant l'état i et le temps passé dans l'état i est exponentiel de taux μ_i.

Soit un état i d'une CMTC, avec une transition vers l'état j et une autre vers l'état k. Le temps passé dans l'état i est exponentiel de taux μ_i. Partant de l'état i, la variable aléatoire mesurant le temps de transition vers l'état j (resp. état k) suit une loi exponentielle de taux μ_{ij} (resp. μ_{ik}). On a alors :
$\mu_i = \mu_{ij} + \mu_{ik}$
$\mu_{ij} = \mu_i.p_{ij}$
$\mu_{ik} = \mu_i.p_{ik}$
$p_{ij} = \frac{\mu_{ij}}{\mu_{ij}+\mu_{ik}}$
$p_{ik} = \frac{\mu_{ik}}{\mu_{ij}+\mu_{ik}}$

Par ailleurs, pour savoir lequel des états j ou k sera visité en sortant de i, on *"tire"* une réalisation particulière pour chacune de ces deux variables aléatoires (suivant respectivement des lois exponentielles de taux μ_{ij} et μ_{ik}). La plus petite des deux valeurs générées correspond à la transition qui sera effectivement empruntée.

2.3.5.2 Description des chaînes de Markov à temps continu

En pratique, une chaîne de Markov à temps continu [141, 140] peut être décrite soit par un *diagramme de transition d'état* ou bien par une matrice des taux de transition dite : *générateur infinitésimal*

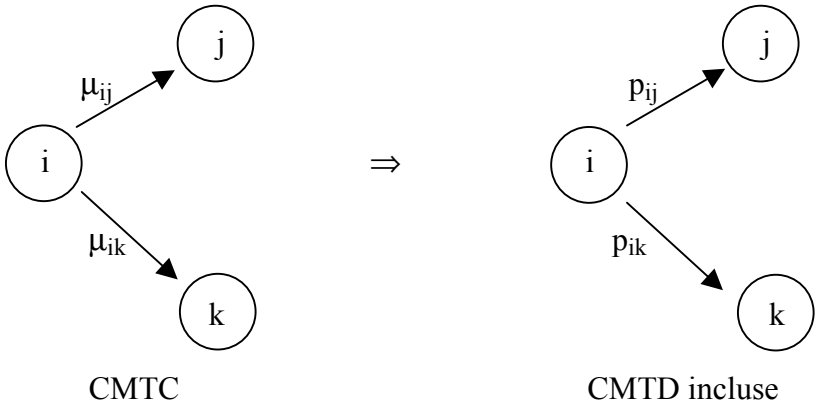

FIG. 2.1 – CMTD incluse dans une CMTC

- Le diagramme de transition est un graphe orienté et libellé dont les sommets correspondent aux états de la chaîne de Markov et les arcs sont étiquetés par les taux de la distribution exponentielle associés à la transition d'un état à un autre.
- Le générateur infinitésimal Q est une matrice carrée, d'ordre égal au nombre d'états de la chaîne. Un élément q_{ij} désigne le taux de transition μ_{ij} de l'état i vers l'état j, $i \neq j$. Les éléments diagonaux q_{ii} sont choisis, par définition, égaux à l'opposé de la somme des autres éléments de la ligne :

$$q_{ij} = \begin{cases} \mu_{ij} & \text{si } i \neq j \\ -\sum_{k=1,k\neq i}^{n} \mu_{ik} & \text{si } i = j \end{cases}$$

où :

- n correspond au nombre d'états de la chaîne de Markov.
- μ_{ij} désigne le taux de la distribution associée à la transition de l'état i à l'état j. S'il n'y a pas de transition, l'élément est nul.

Remarque : [140]
Lorsque la CMTC contient des rebouclages directs, c'est-à-dire qu'à l'issue d'un temps exponentiel de taux μ_i, on peut revenir dans l'état i avec un taux μ_{ii}, ces rebouclages peuvent être supprimés sans affecter d'aucune façon le comportement de la chaîne.

2.3.5.3 Conditions d'existence et calcul d'une distribution stationnaire

Il existe un lien très fort entre les CMTC et les CMTD et ce lien peut être formalisé par la définition de la *CMTD incluse* dans la CMTC.

Définition 34 : Chaîne de Markov incluse [140]
La CMTD incluse dans la CMTC de générateur infinitésimal Q, dont les termes q_{ij} $(i \neq j)$ sont donnés par $q_{ij} = \mu_i.p_{ij}$, est une chaîne de Markov à temps discret dont la matrice de

transition a pour éléments les p_{ij} (voir Fig. 2.1).

Ainsi, si la CMTC est caractérisée par des taux de transition μ_{ij} entre états, les probabilités de transition de la chaîne incluse sont données par :

$$p_{ij} = \frac{\mu_{ij}}{\mu_i} = \frac{\mu_{ij}}{\sum_{k \neq i} \mu_{ik}}$$

Pour confirmer l'existence du régime stationnaire, on doit vérifier si la CMTC est irréductible.

Propriété : [140]
Une CMTC est irréductible si et seulement si la CMTD incluse est irréductible.

En termes de graphe, une chaîne de Markov est irréductible, si et seulement si elle comporte une unique composante fortement connexe.

Propriété : [140]
Une CMTC finie et irréductible est ergodique.

Ainsi, l'existence d'une distribution stationnaire requiert l'existence d'une unique composante fortement connexe. Dans ce cas, l'état stationnaire est atteint après l'écoulement d'un temps infini.

Propriété : [140]
Dans une CMTC erogodique, le vecteur π des probabilités stationnaires $\pi_i = lim_{t \to \infty} \pi_i(t)$ existe toujours et est l'unique solution du système matriciel suivant :

$$\begin{cases} \pi.Q = 0 \\ \sum_{i \in E} \pi_i = 1 \end{cases}$$

où : E est l'espace des états de la chaîne de Markov.

Mathématiquement, le problème de l'analyse des chaînes de Markov paraît simple. Cependant, la résolution de ce système n'est souvent réalisable que pour des chaînes particulières ayant une structure très simple [109]. Par ailleurs, des complications peuvent survenir à partir du calcul dans le cas des chaînes de Markov avec un grand nombre d'états (*problème d'explosion combinatoire*). Pour pallier à ces différents problèmes, on utilise, dans la majorité des cas, des techniques numériques, qui consistent à résoudre le système de façon itérative en utilisant une approche de type "point fixe". Les deux techniques les plus simples et les plus utilisées sont : la *méthode des puissances* et la *méthode de Gauss-Seidel*. Ces deux techniques présentent beaucoup d'avantages, mais ne peuvent s'appliquer qu'à des CMTC dont l'espace d'états est fini [140].

L'avantage de l'utilisation des chaînes de Markov dans le domaine de l'évaluation des performances est qu'elles fournissent des résultats exacts avec des méthodes simples. Cependant, lorsque l'on souhaite utiliser une chaîne de Markov pour représenter le comportement d'un système, on se trouve confronté au problème de la description de cette

chaîne ; car ceci nécessite l'énumération de tous les états et la détermination de toutes les transitions possibles entre chaque couple d'états [144]. Ceci introduit d'une part, un risque d'erreur ou d'omission croissant, et d'autre part, la taille de l'espace d'états, généralement trop important, constitue toujours un obstacle dans la résolution numérique. Pour cela, des outils de modélisation probabilistes plus abstraits ont été proposés. Les outils les plus importants sont basés sur la théorie des files d'attente. Ce modèle se caractérise par la puissance et l'efficacité de la solution lors de l'analyse des performances d'un système. Cependant, il ne permet pas la formalisation des phénomènes de synchronisation [145]. Ainsi, le modèle des réseaux de Petri stochastiques a été proposé.

2.4 Réseaux de Petri stochastiques généralisés

Un réseau de Petri stochastique (RdPS) est un réseau de Petri dans lequel est associée à chaque transition une variable aléatoire représentant le délai de franchissement ou encore le travail qui doit être réalisé pour qu'une opération aboutisse et produise le franchissement de la transition correspondante. De manière précise, le délai de franchissement d'une transition représente l'intervalle de temps qui doit s'écouler entre l'instant de sensibilisation de cette transition et l'instant de son franchissement (la fin de la tâche associée).

Les réseau de Petri stochastiques ont été introduits de manière pragmatique, à la fin des années 70, afin de pouvoir inclure dans les modèles des temporisations aléatoires, et aussi pour bénéficier des méthodes d'évaluation des chaînes de Markov.

Un des intérêts majeurs des RdPS est de pouvoir combiner l'analyse qualitative et l'analyse quantitative.

Cependant, lors de la construction de la topologie d'un RdPS, la modélisation pourrait insérer des transitions qui correspondent à des aspects purement logiques du comportement du système, telle qu'aucune temporisation ne peut leur être associée. Il peut s'agir des opérations d'allocation de ressources, des synchronisations pures ou encore des choix et autres structures de contrôle.
Par ailleurs, dans le même modèle, peuvent apparaître des activités très rapides et d'autres plus lentes, et la négligence de cet aspect peut donner un modèle qui est logiquement incorrect [146, 35]. En fait, la modélisation de ces actions urgentes par une loi exponentielle à taux très élevé n'est pas satisfaisante car, d'une part le choix du taux est arbitraire et d'autre part les calculs numériques souffrent de valeurs d'amplitudes très différentes.

Pour résoudre ces problèmes, une nouvelle classe de RdPS est apparue sous le nom de *RdPS généralisés* (RdPSG) [133], où des *transitions immédiates* (avec une distribution concentrée en 0) ont été introduites. Ce modèle comporte deux types de transitions [133] :

- Les *transitions temporisées* à qui correspondent des variables aléatoires déterminant la durée de franchissement. Ces transitions sont utilisées pour modéliser les délais aléatoires associés à l'exécution des activités ;
- Les *transitions immédiates ou instantanées* qui se caractérisent par une priorité de franchissement supérieure à celle des transitions temporisées et par leur franchisse-

ment immédiat (délai de franchissement nul).
Ces transitions permettent la représentation des actions logiques qui ne consomment pas de temps, comme elles peuvent être utilisées pour modéliser des contraintes de synchronisation, des évènements d'urgence ou des activités prioritaires, dont les délais associés n'ont aucun impact sur les performances du système.

Nous nous intéresserons aux *réseaux de Petri stochastiques généralisés markoviens* qui constituent une classe particulièrement intéressante des RdPSG, dans laquelle les délais de franchissement des transitions temporisées sont des variables aléatoires avec des distributions exponentielles négatives. Ainsi, à toute transition temporisée correspond un taux de franchissement.
Du fait de la propriété d'absence de mémoire de la loi exponentielle, il y a alors à chaque franchissement de transition, une perte de mémoire des travaux déjà réalisés [144, 130].

Remarque :
Dans la représentation graphique d'un RdPSG, les transitions immédiates sont représentées par des traits et les transitions temporisées par des rectangles.

Définition 35 : RdPSG
Un réseau de Petri stochastique généralisé (RdPSG) est un huit-uplet
<P,T,Pré,Post,Inh,pri,W,M_0> où :
- P est l'ensemble des places ;
- T est l'ensemble des transitions temporisées et des transitions immédiates ;
- $Pré$, $Post$ et $Inh : P \times T \rightarrow \mathbb{N}$ sont les fonctions d'incidence avant, d'incidence arrière et d'inhibition respectivement ;
- $pri : T \rightarrow \{0, 1\}$ est la fonction de priorité qui associe à chaque transition temporisée la valeur 0 et à chaque transition immédiate la valeur 1 ;
- $W : T \rightarrow R^+$ est une fonction qui associe à chaque transition temporisée un taux de franchissement et à chaque transition immédiate un poids ;
- $M_0 : P \rightarrow \mathbb{N}$ est le marquage initial du réseau.

Remarque : Les poids sont utilisés pour le calcul des probabilités de franchissement des transitions immédiates et éventuellement pour la résolution probabiliste des conflits entre plusieurs transitions immédiates sensibilisées.

2.4.1 Sémantique stochastique des RdPSG

La sémantique stochastique des RdPSG, appelée aussi *politique d'exécution* du modèle [147, 148] est définie par la politique de mémoire, la politique de service et la politique de choix.

2.4.1.1 Politique de mémoire :

La politique de mémoire spécifie comment le processus de marquage associé au RdP est conditionné par rapport à son passé, c'est-à-dire qu'elle indique ce que devient le travail déjà réalisé ou bien le temps associé à une transition à chaque changement d'état. Il existe trois politiques de mémoire [148] qui correspondent aux situations les plus classiques.

Néanmoins, à cause de la propriété de perte de mémoire de la distribution exponentielle, ces trois politiques sont sémantiquement équivalentes. Ainsi, la politique de mémoire des RdPSG markoviens est une politique Resampling (interruption), où à chaque changement de marquage, le temps déjà passé ou encore le travail réalisé par les activités qui ne sont pas terminées est perdu. Le seul travail qui est pris en compte est celui effectué par l'activité correspondante à la transition qui a été tirée et qui est responsable du changement de l'état du système.

2.4.1.2 Politique de service :

La sémantique de service indique le comportement de la transition lorsque le degré de franchissement (enabling degree) de celle-ci est supérieur à 1 (plusieurs jetons présents). Le **degré de franchissement** d'une transition t en un marquage M noté **ED(t,M)** correspond au nombre de jetons qui peuvent être tirés simultanément par la transition t.

$$ED(t,M) = k \ ssi \begin{cases} \forall p \in^{\cdot} t, M(p) \geq k.Pré(p,t) \\ \text{et } \exists p \in^{\cdot} t, M(p) < (k+1).Pré(p,t). \end{cases}$$

Exemple : Soit le réseau de la figure 2.2.

Les degrés de franchissement correspondant aux transitions t_1, t_2 et t_3 sont : $ED(t_1, M) = 2$; $ED(t_2, M) = 1$; $ED(t_3, M) = 3$.

Remarque : Nous notons par $W(t)$ le taux de franchissement de t et $W(t, M)$ le taux de franchissement de t en M.

Dans la littérature, trois politiques de service sont appliquées : [148]

1. **Serveur unique (single server) :** À chaque instant, un seul client est servi à la fois. Ainsi, pour un marquage donné M, le taux de franchissement d'une transition t de loi exponentielle est : $W(t, M) = W(t)$. C'est la sémantique par défaut des réseaux de Petri stochastiques généralisés.
2. **Serveurs multiples (K) (multiple servers) :** Au plus K clients peuvent être servis simultanément. Le taux de franchissement est dans ce cas : $W(t, M) = Min(K, ED(t, M)).W(t)$.
3. **Serveurs infinis (infinite servers) :** Autant de clients que possible, peuvent être servis en même temps. Ainsi, le taux de franchissement est : $W(t, M) = ED(t, M).W(t)$.

Par conséquent, le taux de franchissement d'une transition exponentielle t en M est en général donné par : $Min(K, ED(t, M)).W(t)$, où $K = 1$ pour la politique de serveur unique, $K = K$ pour K-serveurs multiples et $K = +\infty$ pour le cas de serveurs infinis.

2.4.1.3 Politique de choix

Cette politique consiste à définir la règle du choix de la transition à tirer, parmi toutes les transitions franchissables dans un marquage donné. Il existe deux politiques de choix :

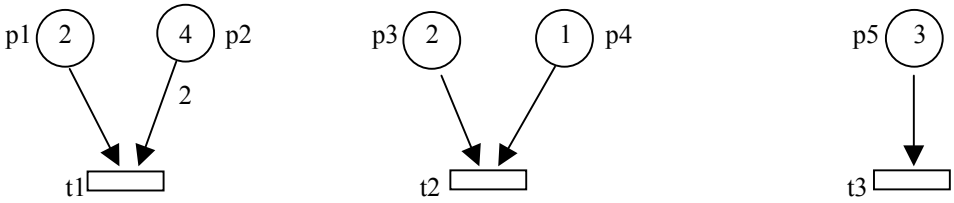

FIG. 2.2 – Degré de franchissement

1. **La politique de présélection** : Dans ce cas, la transition à tirer est choisie suivant une fonction de probabilité indépendante des délais de franchissement des transitions.
2. **La politique de course (modèle concurrentiel)** [149, 147] : cette politique choisit comme transition à tirer parmi les transitions franchissables, celle qui a le plus petit délai de franchissement. Ainsi, dans le cas markovien, le temps de séjour dans un marquage est égal à la valeur du délai minimum.

La politique de choix la plus appliquée dans la pratique pour les transitions temporisées, est la politique de course, tandis que la politique de présélection est habituellement utilisée pour les transitions immédiates.

2.4.2 Processus stochastique associé à un RdPSG

Les RdPSG contiennent des transitions immédiates avec des délais de franchissement nuls et des transitions temporisées avec des temps de franchissement distribués exponentiellement.

En examinant l'évolution d'un RdPSG, nous pouvons constater que la distribution du temps de séjour dans un marquage arbitraire peut être exprimée comme une composition de distributions exponentielles négatives et de distributions déterministes nulles. Nous pouvons donc déduire que *le processus stochastique engendré par un RdPSG est un processus semi-markovien* ([150], chap. 9). Ce processus comprend deux types de marquages :

- Les *marquages tangibles (tangible markings)* : qui sensibilisent seulement les transitions temporisées ainsi, aucune transition immédiate n'est franchissable ;
- Les *marquages évanescents (vanishing markings)* : dans lesquels au moins une transition immédiate est sensibilisée. Le nom évanescent paraît approprié puisque les transitions immédiates sont tirées en un temps nul.

2.4.2.1 Temps de séjour dans un marquage

Le temps de séjour dans un marquage est le temps passé par le RdP dans cet état. Il correspond à la durée de franchissement de la transition qui a provoqué le changement d'état. Cette durée est en fait, la plus petite des durées de franchissement de l'ensemble des transitions franchissables (modèle concurrentiel).

En fait, dans un RdPSG, le temps de séjour dépend du type de marquage atteint. Dans le cas d'un marquage évanescent, ce temps est nul. Par contre, le temps de séjour dans un marquage tangible M_i est distribué exponentiellement avec un paramètre qui est la somme des taux de toutes les transitions temporisées sensibilisées en M_i ([150], chap.9) :

$$q_i = \sum_{t_k \in E(M_i)} \omega_k$$

où : $E(M_i)$ est l'ensemble des transitions franchissables en M_i.

Ainsi, la valeur moyenne du temps de séjour dans un marquage tangible M_i est donnée par la formule suivante ([150], chap. 9) :

$$TS_i = \frac{1}{q_i} = \frac{1}{\sum_{t_k \in E(M_i)} \omega_k} \tag{2.1}$$

Remarque : Comme les temps de séjour dans les marquages évanescents sont nuls, nous pouvons déduire que les probabilités stationnaires de ces états sont aussi nulles.

2.4.2.2 Temps moyen de franchissement d'une transition (TF)

Le temps moyen de franchissement d'une transition t_j est le temps moyen qui doit s'écouler pour que la transition considérée soit tirée. Ce temps est une variable aléatoire de loi exponentielle, dont la moyenne se calcule comme suit ([150], chap. 9) :

$$TF_j = \frac{1}{\omega_j}$$

2.4.3 Politique de franchissement dans les RdPSG

Le franchissement des transitions dans un RdPSG dépend du type de marquage examiné : marquage tangible ou évanescent. Ceci implique un nombre de possibilités quand il s'agit de déterminer laquelle des transitions tirer, quand plusieurs sont franchissables à la fois dans un même marquage.

- Dans le cas d'un marquage tangible, toute transition sensibilisée peut être tirée. Le franchissement d'une transition dépend des taux de franchissement des transitions temporisées sensibilisées ;
- Dans le cas d'un marquage évanescent, seulement les transitions immédiates sensibilisées peuvent être tirées car elles sont plus prioritaires que les transitions temporisées. Dans ce cas, quelques autres règles sont appliquées :

  1. Les transitions immédiates concurrentes sensibilisées sont tirées simultanément ;
  2. Pour les transitions immédiates sensibilisées en conflit, une seule transition peut être tirée. Le choix probabiliste de la transition à tirer se fait par une post-sélection (calcul de probabilités en fonction des poids des transitions immédiates franchissables).

Ainsi, quand plusieurs transitions immédiates sont sensibilisées dans un même marquage (évanescent), choisir laquelle des transitions sera tirée n'a de sens qu'en cas de

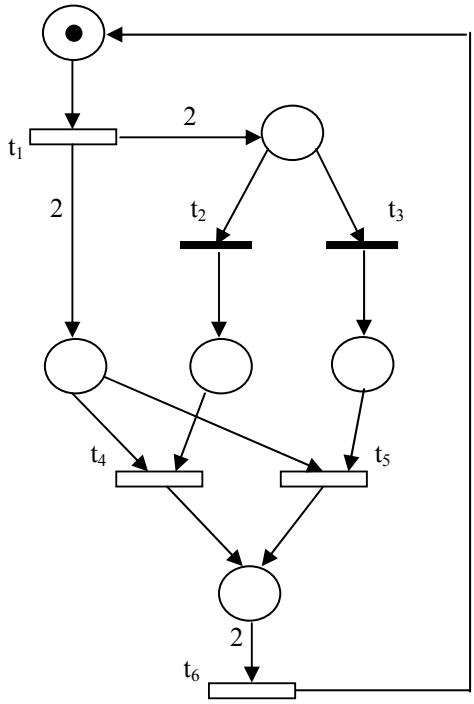

FIG. 2.3 – Un RdPSG avec des transitions immédiates concurrentes

conflit. D'ailleurs, pour des transitions immédiates sensibilisées et en conflit, une seule transition peut être tirée à la fois suivant la règle définie ci-dessous qui attribue des priorités pour résoudre le conflit. Si par contre, ces transitions immédiates sensibilisées sont concurrentes, elles seront tirées simultanément (en un temps nul) et donc la sélection n'est pas nécessaire dans ce cas.

Exemple : Soit le RdPSG de la figure 2.3 avec deux transitions immédiates concurrentes t_2 et t_3 qui sont plus prioritaires que les transitions temporisées. À chaque transition t_i correspond le taux ou le poids ω_i. Le graphe d'accessibilité est donné dans la figure 2.4.

L'expression générale définissant la probabilité pour que dans un marquage M_i (tangible ou évanescent), une transition franchissable $t_k \in E(M_i)$ (temporisée ou immédiate) soit tirée est donnée par ([150], chap. 9) :

$$P\{t_k|M_i\} = \frac{\omega_k}{q_i} = \frac{\omega_k}{\sum_{t_j \in E(M_i)} \omega_j}$$

Quand le marquage est tangible, les paramètres ω_k et ω_j des transitions temporisées franchissables dans ce marquage, sont les taux de leur distribution exponentielle négative. Par contre, quand le marquage est évanescent, ces paramètres sont les poids des transitions immédiates sensibilisées dans ce marquage et définissent la politique de sélection pour faire le choix.

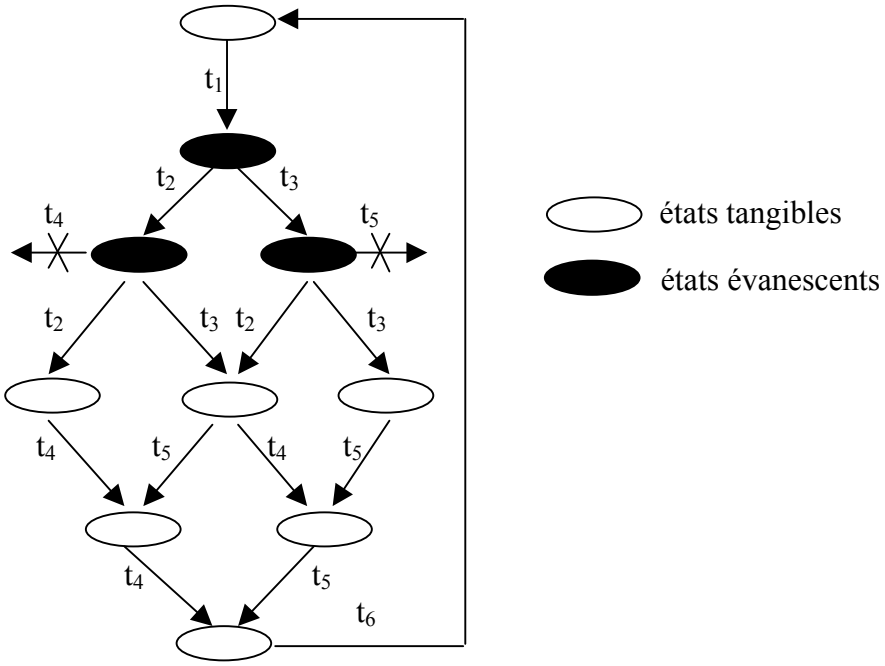

FIG. 2.4 – Graphe d'accessibilité du RdPSG de la figure 2.3

Par ailleurs, le nouveau marquage M' généré à partir de M par le franchissement d'une transition t (temporisée ou immédiate) est calculé selon la règle de franchissement standard des RdP simples :

$$\forall p \in P,\ M'(p) = M(p) - Pre(p,t) + Post(p,t)$$

Remarque : Dans les RdPSG, le franchissement des transitions est une opération atomique, *i.e.* que les jetons sont enlevés des places en entrée et déposés dans les places en sortie en une seule opération indivisible.

Dépendance du marquage : Les taux de franchissement des transitions temporisées et les poids des transitions immédiates, peuvent être dépendants des marquages, c'est-à-dire qu'ils diffèrent d'un marquage à un autre. Citons à titre d'exemple, le cas des modèles avec politique à serveurs infinis. Ainsi, le taux (ou le poids) d'une transition t_k dans un marquage M_i est alors $\omega_k(M_i)$.
Dans ce cas, les équations données pour le calcul de la distribution du temps de séjour dans un marquage aussi bien que la probabilité de la transition à tirer, peuvent être généralisées en supposant que tous les paramètres sont dépendants des marquages [150].

2.4.4 Isomorphisme entre RdPSG et chaîne de Markov incluse

Définition 36 : Isomorphisme
Deux modèles stochastiques sont dits isomorphes si et seulement si :

1. il existe une application entre les espaces des états des deux modèles ;
2. il existe une transition dans un modèle ssi il existe une transition dans l'autre ;

3. la probabilité d'occurrence d'une transition pendant une durée d dans un modèle est égale à la probabilité d'occurrence d'une transition pendant la même durée d dans l'autre modèle.

Le processus stochastique engendré par un RdPSG est un processus semi-markovien qui peut être analysé en identifiant une chaîne de Markov incluse (à temps discret) qui décrit les transitions entre les états du processus ([150], chap. 9).

Cette chaîne de Markov incluse (CMI) peut être obtenue en s'intéressant uniquement à l'ensemble des états du processus semi-markovien et sans prendre en compte le fait que le temps passé dans les états est nul ou pas. Par conséquent, cette CMI comprend les marquages tangibles et les marquages évanescents. Les taux et les poids des transitions du RdPSG sont suffisants pour calculer les probabilités de transition entre les états d'une telle chaîne. Ainsi, il existe un isomorphisme entre le graphe d'accessibilité d'un RdPSG et la CMI décrivant son comportement. Celle-ci est obtenue en appliquant les règles suivantes :

1. L'espace d'états de la CMI correspond à l'ensemble d'accessibilité du RdPSG ;
2. À chaque transition entre deux états du graphe d'accessibilité, correspond une probabilité de changement d'état entre les marquages équivalents de la chaîne.

Ainsi, les RdPSG peuvent être utilisés pour générer automatiquement le processus stochastique sous-jacent qui peut être ensuite analysé et depuis lequel on peut déduire divers paramètres de performance.

2.4.5 Analyse des RdPSG

L'analyse d'un système est basée sur deux aspects essentiels. D'une part, nous avons l'aspect qualitatif qui consiste à vérifier la correction du modèle et les propriétés qualitatives du système modélisé, et d'autre part, l'aspect quantitatif qui a pour but le calcul des paramètres de performance pour l'évaluation de l'efficacité du système étudié.

– **Analyse qualitative :** Pour la vérification des propriétés qualitatives telle que : la vivacité, la bornitude, les états d'accueil, etc, les méthodes d'analyse utilisées pour les RdP simples peuvent être appliquées [151].
Cependant, comme les transitions immédiates ont une priorité sur les transitions temporisées, *le modèle non-temporisé sous-jacent à un RdPSG est un RdP avec priorité* et non pas un RdP simple. En fait, la présence des priorités dans le modèle des RdPSG réduit le nombre de marquages accessibles par rapport au RdP simple. Ainsi, l'espace d'état d'un RdPSG est isomorphe au graphe d'accessibilité du RdP non-temporisé avec priorité ($t_i \rightarrow \omega_i$). D'autre part, l'ensemble d'accessibilité du RdP avec priorité est inclus dans l'ensemble d'accessibilité du RdP sans priorité :
$A(\Sigma_{pri}) \subseteq A(\Sigma)$
Par conséquent, les propriétés du RdP simple sans priorité ne sont pas forcément maintenues en ajoutant une fonction de priorité, par exemple :
– L'accessibilité : $M \in A(\Sigma) \not\Rightarrow M \in A(\Sigma_{pri})$;
– Bornitude : Σ *borné* $\Rightarrow \Sigma_{pri}$ *borné* mais Σ *non borné* $\not\Rightarrow \Sigma_{pri}$ *non borné* ;

– Vivacité et état d'accueil : La priorité peut introduire ou éliminer la vivacité ou un état d'accueil. En effet, chaque fois qu'une transition se trouve en conflit avec une transition de priorité supérieure, c'est cette dernière qui est franchie. Ainsi, une transition vivante dans un réseau sans priorité peut alors devenir simplement quasi-vivante, voire jamais franchissable.

Ainsi, pour l'analyse des propriétés comportementales d'un RdPSG combinant des transitions immédiates avec des transitions exponentielles, nous pouvons examiner le RdP non-temporisé avec priorité sous-jacent. En d'autres termes, nous traitons toutes les transitions comme des transitions simples sans temporisation mais avec une certaine priorité. Cette étape constitue donc l'évaluation qualitative du système correspondant [152].

D'autre part, les techniques d'analyse structurelle des RdPSG permettent la vérification des propriétés structurelles directement sur le réseau et sans prendre en compte le marquage initial. Cette analyse est faite en appliquant les techniques d'algèbre linéaire à la description matricielle du RdPSG (matrices d'incidence). La méthode conduit au calcul des P-semiflots et T-semiflots.
La couverture de toutes les places par des P-semiflots est une condition suffisante pour que le modèle soit borné. Les P-semiflots peuvent quelques fois être utilisés pour prouver certaines propriétés telle que l'exclusion mutuelle.
Un T-semiflot identifie un ensemble de transitions, tel que, partant de tout marquage m, le franchissement de toute séquence de transitions appartenant à l'ensemble, permet de revenir au marquage m. L'existence de T-semiflots qui couvrent toutes les transitions du réseau est une condition nécessaire mais pas suffisante pour la vivacité du modèle.

– **Analyse quantitative :** Cette partie d'analyse consiste à calculer les probabilités stationnaires et les indices de performance. Elle est basée essentiellement sur les techniques d'analyse des chaînes de Markov, et ce grâce à l'isomorphisme entre les RdPSG et les chaînes de Markov. On peut ainsi évaluer les performances du système modélisé, car le problème se transforme en l'étude d'une CMI. Néanmoins, ceci nécessite la vérification de certaines propriétés résumées dans la condition **d'ergodicité** du RdPSG.
La propriété d'ergodicité garantit l'existence d'un régime stationnaire ou permanent du comportement d'un RdPSG, et permet ainsi l'application des techniques d'analyse des chaînes de Markov pour le calcul de la distribution stationnaire et d'autres indices de performance stationnaire du système modélisé.

Théorème : [153]
Un RdPSG borné et tel que son graphe des marquages accessibles est fortement connexe est ergodique.

Théorème : [151]
Un RdPSG borné est ergodique s'il admet le marquage initial comme état d'accueil.

Propriété :[150]
Le processus stochastique engendré par un RdPSG borné avec le marquage initial comme état d'accueil, peut être classé comme un processus semi-markovien à temps continu, à espace d'états fini, stationnaire et irréductible.

Ainsi, si le graphe d'accessibilité d'un RdPSG est fini et de plus possède une seule composante fortement connexe, une chaîne de Markov incluse ergodique équivalente peut être dérivée et ensuite analysée.

2.5 Résolution numérique des RdPSG

Une fois l'ergodicité du modèle est prouvée, on peut alors évaluer les performances du système modélisé. L'évaluation consiste à construire d'abord la matrice des probabilités de transition U de la chaîne de Markov incluse à partir de la spécification du modèle, et ce en utilisant l'expression suivante, pour les transitions temporisées aussi bien que les transitions immédiates :

$$u_{ij} = \frac{\omega_k}{q_i}$$

Par conséquent, la matrice des probabilités de transition U est une matrice carrée d'ordre N où N est la cardinalité de l'espace d'états de la CMI avec marquages tangibles et marquages évanescents.

En ordonnant les marquages tels que les évanescents correspondent aux premières entrées de la matrice et les tangibles aux dernières entrées, la matrice U peut être décomposée de la manière suivante ([150], chap. 9) :

$$U = A + B$$

où

$$A = \begin{pmatrix} U_{VV} & U_{VT} \\ 0 & 0 \end{pmatrix}$$

et

$$B = \begin{pmatrix} 0 & 0 \\ U_{TV} & U_{TT} \end{pmatrix}$$

Une fois cette matrice générée, la distribution des probabilités à l'état stationnaire ψ peut être obtenue comme solution du système d'équations linéaires suivant :

$$\begin{cases} \psi = \psi.U \\ \psi.1^T = 1 \end{cases}$$

Cependant, cette méthode nécessite le calcul de la probabilité d'état stationnaire de tous les marquages y compris les évanescents, sachant que le temps passé dans ces marquages est nul et par conséquent la probabilité de s'y trouver est aussi nulle. De plus, les marquages évanescents ne nécessitent pas seulement des calculs sans intérêt mais augmentent aussi la taille de la matrice de probabilité de transition U, ce qui rend le calcul de la solution plus coûteux en temps et en espace mémoire.

Ainsi, une autre méthode a été proposée dans le but de limiter le calcul des probabilités stationnaires aux marquages tangibles. Pour cela, la CMI doit être réduite en éliminant les marquages évanescents avant la résolution de la chaîne incluse. La chaîne obtenue est dite : **chaîne de Markov incluse réduite** (CMIR). Par conséquent, la matrice des probabilités de transition correspondante U' contiendra seulement les probabilités de transition entre marquages tangibles.

En fait, l'élimination des marquages évanescents de la chaîne de Markov n'influe pas sur le comportement dynamique du système, car le processus passe un temps nul dans ces marquages [133].

Le principe d'élimination des marquages évanescents est le suivant : Soit M_b un marquage évanescent directement accessible à partir du marquage tangible M_a, et soit S l'ensemble des marquages tangibles accessibles à partir de M_b en passant par une séquence de marquages évanescents seulement. Dans ce cas, le marquage évanescent M_b et tous ceux atteignables à partir de M_b par le franchissement de transitions immédiates peuvent être éliminés en introduisant simplement un arc reliant directement M_a à M_c $\forall M_c \in S$ $(M_c \neq M_a)$, et en calculant convenablement la probabilité de transition de M_a à M_c [133].

Pour illustrer la méthode de réduction de la chaîne de Markov incluse en une chaîne de Markov incluse réduite (CMIR), nous considérons les deux exemples suivants :

Exemple 1 : La chaîne de Markov incluse correspondante au RdPSG de la figure 2.3 est représentée dans la figure 2.5, où les probabilités de transition sont données par :
$u_2 = \omega_2/(\omega_2 + \omega_3)$
$u_3 = \omega_3/(\omega_2 + \omega_3)$
$u_4 = \omega_4/(\omega_4 + \omega_5)$
$u_5 = \omega_5/(\omega_4 + \omega_5)$

La chaîne de Markov incluse réduite obtenue après élimination des marquages évanescents est donnée dans la figure 2.6.

Exemple 2 : Soit le RdPSG de la figure 2.7. Le graphe d'accessibilité correspondant est représentée dans la figure 2.8.

À partir du marquage initial $m_0 = p_1$, le système peut passer au marquage $m_1 = p_3$ en suivant deux chemins différents. Le premier correspond au franchissement de la transition T_1 avec la probabilité $\frac{\mu_1}{(\mu_1+\mu_2)}$ et qui conduit au marquage m_1 en une seule étape. Le second correspond à sélectionner la transition T_2 suivie de la transition t_1. Le premier des deux évènements arrive avec une probabilité de $\frac{\mu_2}{(\mu_1+\mu_2)}$ et le second avec une probabilité $\frac{\alpha}{(\alpha+\beta)}$. Ainsi, la probabilité de ce second chemin de m_0 à m_1 est de $\frac{\mu_2}{(\mu_1+\mu_2)} \cdot \frac{\alpha}{(\alpha+\beta)}$.
Notons que le marquage $m_2 = p_2$ est un marquage évanescent qui sera supprimé de la CMI tout en rectifiant la probabilité de transition totale de m_0 à m_1 qui est dans ce cas :
$\mu'_{01} = \frac{\mu_1}{(\mu_1+\mu_2)} + \frac{\mu_2}{(\mu_1+\mu_2)} \cdot \frac{\alpha}{(\alpha+\beta)}$

Nous présentons dans ce qui suit les équations matricielles [133, 35] qui sont pré-

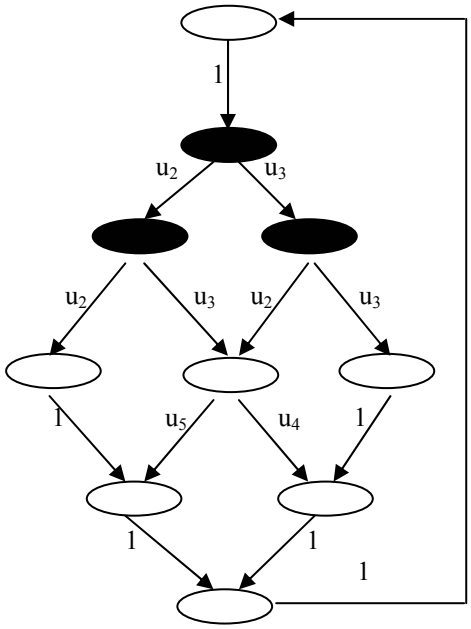

FIG. 2.5 – La chaîne de Markov incluse du RdPSG de la figure 2.3

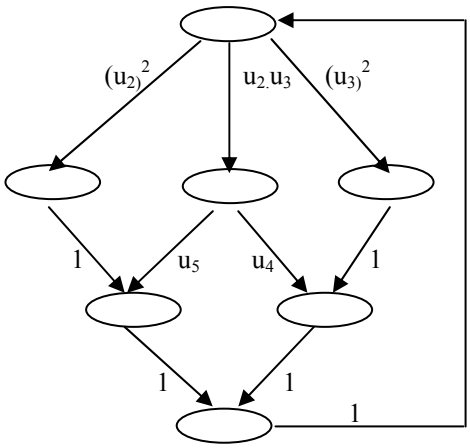

FIG. 2.6 – La chaîne de Markov incluse réduite correspondante

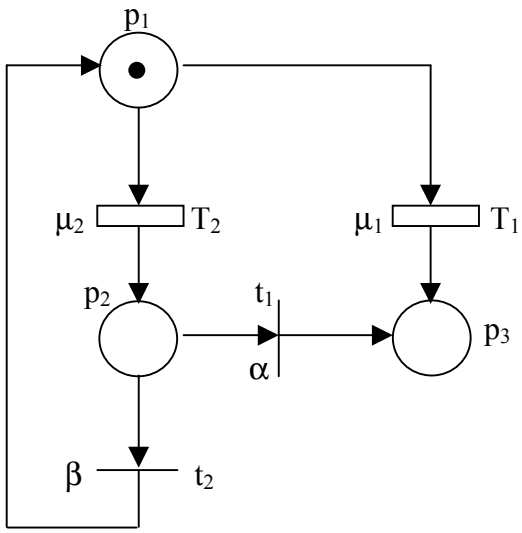

FIG. 2.7 – Un RdPSG avec chemins multiples entre marquages tangibles

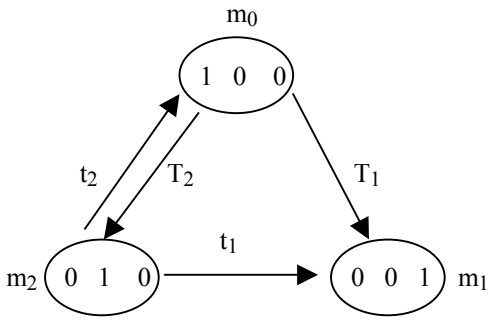

FIG. 2.8 – La chaîne de Markov incluse du RdPSG de la figure 2.7

vues pour calculer les poids sur toutes les séquences des transitions immédiates et pour prendre en compte ces poids automatiquement dans la réduction du graphe d'accessibilité.

En effet, la matrice U' des probabilités de transition de la chaîne de Markov incluse réduite, peut être obtenue **automatiquement** en appliquant la formule suivante [150, 128] :

$$U' = U_{TT} + U_{TV}.H.U_{VT} \tag{2.2}$$

où :

$$H = \begin{cases} \sum_{k=0}^{\infty}(U_{VV})^k, & \text{pas de boucles entre marquages évanescents,} \\ (Id_{VV} - U_{VV})^{-1}, & \text{boucles entre marquages évanescents.} \end{cases}$$

et Id_{VV} est la matrice identité sur les marquages évanescents.

En fait, la relation 2.2 est une simple traduction de la phrase suivante : *pour aller d'un marquage tangible à un marquage tangible, soit on y va directement, soit on passe en premier lieu par un marquage évanescent, puis on passe par une suite de marquages évanescents, et enfin d'un dernier marquage évanescent au marquage tangible final.*

La distribution de probabilité à l'état stationnaire $\psi\prime$ de la CMIR, peut être obtenue comme une solution du système d'équations linéaires suivant :

$$\begin{cases} \psi\prime = \psi\prime.U' \\ \psi\prime.1^T = 1 \end{cases}$$

La construction de la CMIR définie sur l'ensemble des marquages tangibles implique qu'une transformation du processus semi-markovien associé au RdPSG en une CMTC est possible.

La distribution de probabilité à l'état stationnaire sur les marquages tangibles peut alors être aussi obtenue par une résolution directe de cette CMTC. Dans note cas, le générateur infinitésimal Q' de la CMTC associée au RdPSG peut être construit à partir de la matrice des probabilités en divisant chacun de ses éléments par le temps de séjour moyen du marquage tangible correspondant à la ligne (voir formule 2.1).

$$q'_{ij} = \begin{cases} \frac{u'_{ij}}{TS_i}, & i = j, \\ -\sum_{j \neq i} u'_{ij}, & i \neq j. \end{cases}$$

Ce résultat montre que les RdPSG peuvent être analysés en se basant sur les CMTC associées. La distribution de probabilité stationnaire est la solution du système d'équations linéaires suivant :

$$\begin{cases} \pi\prime.Q' = 0 \\ \pi\prime.1^T = 1 \end{cases}$$

où : 0 et 1^T sont des vecteurs de la même taille que π avec tous les composants égaux à 0 et 1 respectivement.

Ainsi, l'analyse d'un RdPSG peut être faite en étudiant une CMTC. Le diagramme de transition d'état de cette chaîne correspond au graphe d'accessibilité tangible du RdPSG.

En conclusion, un RdPSG est équivalent à une chaîne de Markov incluse [133]. En supprimant les marquages évanescents, nous obtenons une chaîne de Markov incluse réduite. Cette chaîne, qui contient seulement les marquages tangibles, est la chaîne qui est utilisée pour le calcul des probabilités d'état stationnaires. Par conséquent, l'introduction des transitions temporisées et des transitions immédiates permet une très grande flexibilité au niveau de la modélisation, et ce sans augmenter la dimension de l'espace d'états tangible final à partir duquel les mesures de performance désirées sont calculées.

Remarque :
Le graphe des marquages accessibles réduit (sans marquages évanescents) n'est pas isomorphe au graphe des marquages du RdP non-temporisé sous-jacent. Par conséquent, la validation **qualitative** du réseau non-temporisé n'est pas valable pour la chaîne de Markov réduite. Ainsi, pour une analyse qualitative d'un RdPSG, il faudrait plutôt examiner le GMA avec marquages tangibles et évanescents. Ensuite, la réduction se fait pour l'évaluation **quantitative**.

2.5.1 Évaluation des indices de performance

Une fois l'ergodicité du modèle vérifiée, on peut alors calculer la distribution de probabilité à l'état stationnaire π. Cette distribution est la base de l'évaluation quantitative du comportement du RdPSG, qui est exprimée en termes d'indices de performance.
Ainsi, à partir des probabilités stationnaires, on déduit par des calculs matriciels les principaux indices (critères ou paramètres) de performance et de fiabilité, tels que la fréquence moyenne de franchissement des transitions, le nombre moyen de marques dans les places, le temps moyen de séjour des marques dans les places, la probabilité d'un évènement donné, etc.

Remarque : Les trois premiers paramètres cités ci-dessus peuvent représenter, dans le cas d'un RdPSG modélisant un protocole de communication par exemple, le débit du canal de transmission, la charge du canal et le temps moyen d'attente pour la transmission. Par contre, dans un système de production, ces paramètres correspondent au débit du système, aux capacités moyennes des stocks et au temps d'attente.

En fait, l'avantage de l'évaluation des indices de performance est de pouvoir choisir les meilleurs paramètres du système pour réaliser une performance désirée.

Nous présentons maintenant les formules de calcul des principaux paramètres quantitatifs. Ces résultats s'expriment en fonction des éléments de base (places, transitions et marquages) du réseau de Petri et des probabilités stationnaires de la chaîne de Markov sous-jacente.

1. **Fréquence moyenne de franchissement d'une transition ($\overline{\lambda}(t_i)$) :**
   On appelle fréquence moyenne (ou encore débit moyen) de franchissement d'une transition t_i, le nombre moyen de tirs de t_i en une unité de temps. Elle est calculée

par :

$$\overline{\lambda}(t_i) = \sum_{M_j \in E(t_i)} \lambda_i(M_j).\pi_j$$

où :
- $E(t_i)$ est l'ensemble des marquages où la transition t_i est franchissable.
- $\lambda_i(M_j)$ est le taux de franchissement de t_i en M_j.

2. **Nombre moyen de marques dans une place ($n(p)$) :**
Le nombre moyen de marques dans une place p est calculé en appliquant la formule :

$$n(p) = \sum_{i:M_i \in E} M_i(p).\pi_i$$

où : $M_i(p)$ est le nombre de jetons dans la place p pour la marquage M_i et E est l'ensemble des marquages accessibles.

3. **Le temps moyen de séjour d'une marque dans un sous-réseau :**
Le délai moyen qu'un jeton passe dans une partie S (dite sous-réseau) d'un RdPSG à l'état stationnaire, peut être calculé en utilisant la formule de Little [53, 154].

$$E[T] = \frac{E[N]}{E[\gamma]}$$

où $E[T]$ est le délai moyen, $E[N]$ est le nombre moyen de jetons dans le sous-réseau S et $E[\gamma]$ est le taux d'arrivée effectif des jetons dans S. Ce taux correspond à la somme des débits moyens des transitions qui tirent dans S (entrantes). En fait, à l'état d'équilibre, ce taux d'entrée moyen est égal au taux de sortie moyen des jetons du sous-réseau qui est égal à la somme des débits moyens des transitions sortantes qui tirent depuis S vers l'extérieur de S.

4. **La probabilité d'un évènement :** La probabilité d'un évènement particulier Υ peut être calculée en sommant les probabilités de tous les marquages dans lesquels la condition correspondante à la définition de l'évènement est vérifiée. Ainsi, cette probabilité est calculée comme suit :

$$Prob\{\Upsilon\} = \sum_{m_i \in X} \pi_i.$$

où $X = \{m_i \in A(R, m_0) : \Upsilon(m_i) = True\}$ est l'ensemble des marquages accessibles où la condition Υ est vérifiée.

2.6 Analyse des RdPSG à espace d'états large

Les RdPSG représentent une méthodologie bien établie pour l'analyse des performances des systèmes concurrents. D'ailleurs, ils sont considérés comme l'un des meilleurs

formalismes de haut-niveau pour la modélisation de ces systèmes.

Pour l'analyse des RdPSG, deux approches de résolution complètement différentes sont généralement appliquées : la simulation et l'analyse basée sur l'espace d'états. La simulation [155] (en utilisant les méthodes de Monte-Carlo par exemple) permet d'associer des distributions de loi générale aux durées des activités (transitions) et permet aussi de réduire la complexité spatiale. Cependant, le principal inconvénient de la simulation est qu'elle nécessite plusieurs exécutions pour l'obtention de résultats significatifs. Ainsi, cette technique pourrait être très coûteuse et la précision des résultats n'est pas toujours garantie.

L'analyse basée sur l'espace d'états est la plus appliquée. Elle consiste à traduire le modèle en une chaîne de Markov finie qui peut être résolue numériquement. Ainsi, des résultats exacts peuvent être obtenus. Cependant, cette résolution pourrait être limitée par le problème d'*explosion combinatoire* de l'espace d'états, dont la taille croît exponentiellement en fonction du nombre de jetons dans le marquage initial et du nombre de places dans le réseau, ce qui constitue le principal inconvénient de la résolution numérique.

Pour pallier à ce problème, un effort considérable a été fourni durant ces dernières décennies, par plusieurs équipes de chercheurs qui ont proposé diverses approches permettant la résolution des RdPSG avec des espaces d'états extrêmement larges. Les méthodes les plus importantes sont les suivantes :

- Les méthodes d'approximation telles que les méthodes tronquées (truncation methods) [156], les méthodes de décomposition [157, 158, 159, 128], le calcul des bornes [160, 161] en utilisant la programmation linéaire, les symétries et les techniques d'agrégation [162, 163, 46], ainsi que l'approche de résolution matricielle-géométrique développée initialement pour les RdPS à espace d'états infini ;
- Les algorithmes de résolution ad-hoc, les techniques PN-driven [164] et les algorithmes pour RdPS en forme-produit [165, 166, 167]. Dans ce cas, les modèles doivent vérifier certaines propriétés particulières ;
- Les approches numériques exactes, telles que les méthodes basées sur l'algèbre tensorielle [150, 128] et autres [168, 169] ;
- Les approches de résolution basées sur la description de Kronecker [170, 171] ;
- Les algorithmes distribués [172, 173, 174], permettant une construction parallèle de l'espace d'états, ainsi qu'une implémentation distribuée du processus de résolution ;
- Les modèles hiérarchiques [164], dans lesquels le modèle peut être composée à partir de sous-modèles, où chacun est résolu séparément, ensuite les résultats passent au sous-modèle de niveau supérieur.

D'autre part, les techniques et les méthodes de résolution (itératives, directes, etc) [109, 175, 176] qui ont été développées pour les chaînes de Markov à espace d'états large, peuvent aussi être utilisées pour les RdPSG.

2.7 Réseaux de Petri stochastiques généralisés colorés

Les RdPSG définissent un cadre théorique simple et puissant pour l'étude et l'analyse des systèmes concurrents avec des aspects de synchronisation, et ce dans de vastes domaines d'application. Cependant, ce modèle pourrait être limité pour les concepteurs d'applications industrielles importantes. Cette limitation est due principalement au fait qu'il est impossible de modéliser des comportements similaires au moyen d'une seule représentation condensée. En effet, lorsque les systèmes étudiés sont complexes, leur représentation en RdP peut conduire à des modèles d'une taille énorme, et par conséquent totalement inexploitables. D'autre part, l'analyse du RdPSG n'est pas paramétrable, c'est à dire qu'il faut redessiner le réseau et recommencer son analyse lorsqu'on change le nombre d'objets (clients ou ressources) du système. En plus, ce formalisme met en avant le contrôle au détriment de la structuration des données, d'ailleurs les jetons du réseau ne sont pas identifiés.

Pour pallier à ces inconvénients, et afin de prendre en compte ces besoins sans modifier la sémantique de base des réseaux de Petri, les réseaux de Petri colorés ont été introduits par K. Jensen dans [177], comme une abréviation des réseaux de Petri.
En identifiant les ressources de même type par une même couleur, les réseaux colorés permettent d'éviter de construire plusieurs fois des parties de réseau similaires. En fait, ces parties sont automatiquement repliées les unes sur les autres, et l'on obtient ainsi des modèles nettement moins volumineux, bien que représentant exactement les mêmes fonctionnalités du système. Afin d'identifier les couleurs choisies lors du franchissement d'une transition, les arcs sont étiquetés par des fonctions de couleur, qui permettent de spécifier le comportement de ces différents objets.

Ainsi, les réseaux de Petri stochastiques généralisés colorés (RdPSGC) se présentent comme une extension naturelle des modèles stochastiques généralisés dans lesquels la notion de couleur a été introduite [178], permettant ainsi une modélisation fine des différents éléments d'un système. Ces réseaux colorés constituent un formalisme abrégé des réseaux ordinaires, tout en permettant une représentation concise de systèmes complexes.

En effet, les RdPSG colorés [178] sont considérés par plusieurs chercheurs du domaine, comme un formalisme de modélisation supérieur ([150], chap. 4), utile pour le développement de modèles très compacts et faciles à comprendre, pour des systèmes symétriques avec des composants concurrents et des comportements similaires qui doivent être distingués.

Les réseaux colorés se caractérisent par une syntaxe plus ou moins naturelle, par les modalités de paramétrisation et par la possibilité d'analyse directe, opérant sur le formalisme de haut niveau lui-même. Du point de vue sémantique, ils sont caractérisés par une sémantique fonctionnelle simple : une information de couleur (ou de type) est associée aux jetons et aux franchissements, et les valuations des arcs sont des fonctions de couleur qui précisent le nombre et la couleur des jetons consommés ou produits lors du franchissement d'une transition qui se produit par une instance particulière de couleur.

Aucune contrainte syntaxique n'est imposée sur ces fonctions de couleur, ce qui simplifie la phase de modélisation, mais rend difficile la définition ou l'extension des techniques d'analyse directe.

2.7.1 Description du modèle

Les réseaux colorés offrent une possibilité d'expression plus compacte que les réseaux ordinaires. En effet, dans un réseau coloré, une place peut contenir des jetons de différentes couleurs et une transition peut être franchie de différentes manières, selon la couleur sélectionnée. Ceci est réalisé en attachant un domaine de couleur à chaque place et à chaque transition. Par conséquent, une place (ou une transition) dans un modèle coloré, représente plusieurs places (ou transitions resp.) dans le modèle RdPSG déplié correspondant. Ainsi, pour un nombre de places et de transitions identique, le nombre de comportements qui peuvent être exprimés par un réseau coloré est nettement plus élevé qu'avec un réseau ordinaire. En fait, les transitions sont paramétrées et le type du paramètre est choisi dans l'ensemble des domaines de couleur. Une instance d'une transition colorée est obtenue en associant un objet au paramètre de la transition. D'autre part, l'ensemble des jetons colorés qui sont consommés à partir des places en entrée et produits dans les places en sortie après le franchissement d'une transition donnée, est défini par des fonctions qui sont les étiquettes des arcs correspondants. Ces fonctions opèrent sur les domaines de couleur en vue de modifier l'état du réseau.

Domaines de couleur : Le type de données associé aux jetons colorés dans une place est dit : *domaine de couleur* de la place.
Un domaine de couleur peut correspondre à un ensemble d'objets élémentaires ayant le même comportement, tant sur le plan qualitatif que quantitatif (par exemple, un ensemble de processeurs) [128]. Cet ensemble peut être réduit à une couleur unique, appelée couleur neutre. Dans ce cas, les jetons de cette couleur sont équivalents aux jetons des réseaux ordinaires.

D'une manière générale, un domaine de couleur peut être une composition tel qu'un produit cartésien d'ensembles finis ou infinis. Cette composition permet de définir des couleurs (tuples) qui sont des associations entre objets, différents ou de même type (si un même ensemble d'objets apparaît plusieurs fois dans le domaine). Par exemple, une couleur peut définir une association de type <serveur,client utilisant ce serveur>.

Exemple : On considère un réseau coloré modélisant une synchronisation d'une occurrence d'objet du domaine de couleur C et d'une occurrence d'objet du domaine D. La synchronisation est représentée par un produit cartésien.
$C(p_1) = C$, $C(p_2) = D$, $C(t) = C \otimes D$.

On peut noter que le domaine de couleur d'une transition fait intervenir les objets ayant un rôle discriminant dans le franchissement. Ainsi, le domaine de couleur d'une transition est obtenu en effectuant le produit cartésien des domaines des variables différentes qui apparaissent autour de la transition [151].

Fonctions de couleur : Les fonctions de couleur définissent le comportement des objets lors d'un franchissement de transition. Ces fonctions sont des applications linéaires

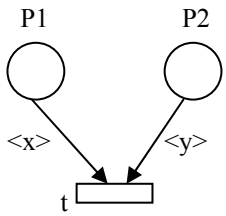

FIG. 2.9 – Exemple de réseau coloré

quelconques qui étiquettent les arcs reliant une place et une transition. Elles déterminent, pour chaque instance de franchissement, le nombre de jetons par couleur qui doivent être consommés ou produits.

La forme générale de la fonction étiquetant un arc reliant une place p de domaine $C(p)$ et une transition t de domaine $C(t)$ est : [151]

$F : Bag(C(t)) \rightarrow Bag(C(p))$.
où : $Bag(E)$ est l'ensemble des multi-ensembles sur E (voir Annexe).

Exemple : La fonction Identité Id est définie par :
$\forall (p,t) \in P \times T, \forall c \in C, Id(p,t)(c) = c$.

Définition 37 : RdPSG coloré [128, 179]
Un réseau de Petri stochastique généralisé coloré (RdPSGC) est défini par :
$< P, T, C, Pré, Post, Inh, pri, M_0, W >$ où :

- P est un ensemble fini de places ;
- T est un ensemble fini (disjoint de P) de transitions temporisées et transitions immédiates ;
- $C : P \cup T \rightarrow \omega$ est la fonction de couleur qui associe à chaque élément $s \in P \cup T$ un domaine de couleur $C(s) \in \omega$, où ω est un ensemble qui contient des ensembles finis et non vides de couleurs. Un élément de $C(s)$ est appelé couleur de s ;
- $Pré$, $Post$ et Inh sont les fonctions d'incidence et d'inhibition, qui associent à chaque place $p \in P$ et à chaque transition $t \in T$, une application de $C(t)$ vers $Bag(C(p))$;
- $pri : T \rightarrow \{0, 1\}$ est la fonction de priorité qui associe à chaque transition temporisée la valeur 0 et à chaque transition immédiate la valeur 1 ;
- M_0 le marquage initial, est une fonction définie sur P, telle que $\forall p \in P, M_0(p) \in Bag(C(p))$;
- $W : \forall t \in T, W(t) : C(t) \rightarrow R^+$ est une fonction qui associe à chaque transition temporisée (ou immédiate), pour une couleur donnée $c \in C(t)$ un taux de franchissement (ou un poids resp.).

Définition 38 : Marquage [128]
Un marquage M d'un RdPSG coloré est un vecteur indexé par P où, pour chaque place $p \in P$, $M(p) \in Bag(C(p))$ désigne le nombre de marques colorées dans cette place p.

Définition 39 : Règle de franchissement [128]
Une transition t est franchissable pour une couleur $c \in C(t)$ dans un marquage M ssi :

$\forall p \in P, M(p)(c) \geq Pré(p,t)(c)$ et $M(p)(c) < Inh(p,t)(c)$.
Ce qui signifie que le marquage M contient les jetons de couleur c nécessaires au franchissement de la transition t.
Le nouveau marquage engendré M' est défini par :

$\forall p \in P, M'(p) = M(p) - Pré(p,t)(c) + Post(p,t)(c)$.

On note alors $M[t(c)\rangle M'$, ce qui signifie que le franchissement de la transition t par rapport à la couleur c pour le marquage M donne le marquage M'.

Il faut noter que toutes les règles de franchissement définies pour les RdPSG restent valables dans le cas coloré. Ainsi, la politique de choix sera naturellement celle du plus court délai, si aucune transition immédiate n'est franchissable, et une sélection probabiliste conditionnée par le poids des transitions immédiates franchissables, sinon. D'autre part, toutes les instances de couleur d'une transition ont la même priorité.
La probabilité de tir d'une instance $t(c)$ franchissable en M vaut [128] :

$$\frac{W(t)(c)}{\sum_{(t',c')} W(t')(c')} \text{ avec } pri(t') = pri(t) \text{ et } M[t'(c')\rangle.$$

Pour ce qui est du choix de la politique de mémoire, il est sans importance, puisque nous utilisons uniquement des lois exponentielles. Enfin, les RdPSG colorés autorisent des dépendances des paramètres stochastiques vis-à-vis du marquage courant. Il est alors facile de décrire les différentes politiques de service.

Remarque : Le choix de la couleur de franchissement d'une transition se fait de façon non-déterministe : si une transition t est franchissable pour une couleur c_1 et pour une couleur c_2, rien ne précise, dans le formalisme, le choix de l'instance de franchissement de t.

Exemple : On considère le réseau coloré de la figure 2.10, où la seule fonction de couleur apparaissant est l'identité (Id).
$\forall p \in P, C(p) = \{a, b\}$; $\forall t \in T, C(t) = \{a, b\}$
$\forall (p,t), Id(p,t)(a) = \{a\}$ et $Id(p,t)(b) = \{b\}$

Ici, lorsque t_1 a été franchie pour a et t_2 franchie pour b, la transition t_3 n'est pas franchissable, alors que dans un réseau non coloré, dès que t_1 et t_2 ont été franchies chacune une fois, la transition t_3 devient franchissable.
Par conséquent, le fonctionnement d'un réseau coloré diffère de celui du réseau ordinaire correspondant.

2.7.2 Analyse des RdPSG colorés

Les RdPSG colorés constituent un outil d'évaluation de haut niveau, qui permet aussi bien l'analyse qualitative que quantitative du système modélisé. Au même titre que les RdPSG ordinaires, l'analyse des RdPSG colorés est basée sur le développement du graphe des marquages accessibles qui comprend des marquages tangibles et des marquages évanescents. Ce graphe est isomorphe à une chaîne de Markov, dont la résolution nous permet l'obtention des probabilités stationnaires et par conséquent de divers indices de performance. L'existence et l'unicité de ce vecteur de probabilités sont assurées, si et seulement si le graphe des marquages accessibles possède une seule composante fortement connexe

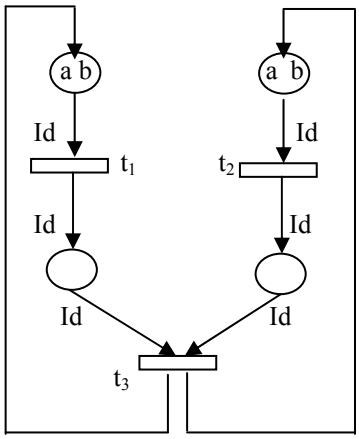

FIG. 2.10 – Un RdPS coloré

terminale [151]. Ainsi, les étapes de l'analyse des RdPSG colorés sont identiques à celles des RdPSG ordinaires. D'ailleurs, l'algorithme de résolution ne tire aucun avantage de la structure colorée du réseau, puisque cette structure n'implique pas de propriété particulière de la chaîne de Markov [151].

Cependant, la taille du graphe d'accessibilité des réseaux colorés peut croître exponentiellement en fonction du nombre de places et de la cardinalité des domaines de couleur. Ainsi, dans le but de réduire la taille de ce graphe et par conséquent d'optimiser la résolution de ces modèles, Jensen a proposé de regrouper certains marquages en classes d'équivalence. La construction des classes est basée sur la définition (non automatique) de symétries, d'où l'on déduit une relation d'équivalence qui est utilisée comme critère de regroupement. En d'autres termes, un *graphe des marquages accessibles agrégé* (réduit), où chaque élément correspond à un ensemble de marquages symétriques, peut être obtenu pour les RdPSG colorés, en exploitant un ensemble de symétries définies par le modélisateur, pour construire des classes de marquages équivalents. Ainsi, en ne développant un sous-graphe d'accessibilité que pour un seul marquage de chaque classe, la taille diminue considérablement.

Le principe de la démarche de résolution des RdPSG colorés est schématisé dans la figure 2.11.

Définition 40 : Symétrie [151]
Deux marquages m_1 et m_2 sont équivalents si et seulement s'il existe une symétrie φ telle que : $m_1 = \varphi(m_2)$.

Le terme de symétrie est utilisé pour nommer des fonctions qui, appliquées à un marquage M, donnent un autre marquage à partir duquel le système a une évolution similaire à son évolution à partir de M. Ces fonctions sont l'identité, les rotations ou les permutations sur les ensembles de couleurs du réseau. Le choix des symétries qui assurent des évolutions similaires est laissé à l'utilisateur.

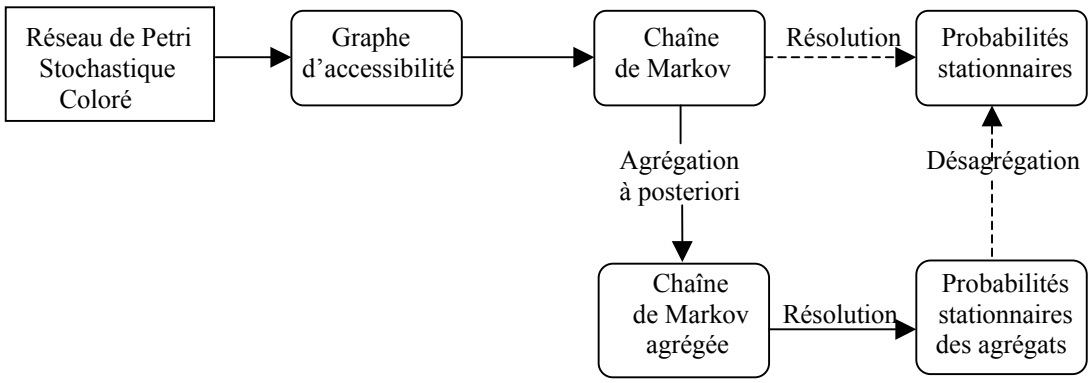

FIG. 2.11 – Résolution des réseaux de Petri stochastiques colorés

Ainsi, le principe de cette seconde approche d'analyse des RdPSG colorés se base sur l'utilisation des symétries du modèle pour la construction des classes d'états, réduisant ainsi le nombre d'états analysés au nombre de classes d'équivalence. Cependant, ces classes ne présentent d'intérêt que si elles simplifient effectivement la résolution du processus, et donc en particulier si elles préservent son caractère markovien. C'est ainsi que l'agrégation markovienne est devenue la technique de base pour l'analyse des réseaux stochastiques colorés, avec l'avantage de fournir des résultats exacts.

L'agrégation markovienne [180] est une technique d'analyse des processus markoviens, qui consiste à regrouper les états d'un processus en classes et à étudier le nouveau processus formé des classes d'états. Cependant, pour que le processus agrégé puisse être analysé de manière efficace, ce regroupement doit préserver la propriété markovienne du processus. D'autre part, la définition des classes d'états dans ces modèles est basée sur l'identification des symétries du système. Par conséquent, l'utilisation de l'équivalence de marquages comme base de l'agrégation est la démarche naturelle de simplification pour analyser les RdPSG colorés.

Plusieurs méthodes ont été présentées dans cette direction. Certaines se basent sur une construction complète du graphe d'accessibilité, puis une agrégation des états en classes [178, 153, 181], et d'autres proposent de construire directement un graphe formé de classes d'états [182]. Le seul inconvénient de ces méthodes est qu'elles ne permettent pas une construction entièrement automatique du graphe condensé. Ceci est dû en particulier à la description des fonctions de couleur, trop générale, qui ne permet pas de conserver la structuration inhérente au modèle étudié d'une part, et ne permet pas d'autre part de détecter de manière automatique les symétries d'un modèle.

Cependant, la construction du graphe d'accessibilité réduit ne peut être effectuée pour un réseau coloré quelconque de manière automatique, s'il n'existe aucune contrainte dans la manière de décrire un modèle. En fait, le problème est la détermination des symétries

admissibles qui peuvent être exploitées pour réduire la taille de l'espace d'états. Pour des classes particulières de réseaux colorés tels que les réseaux stochastiques bien-formés [151], un graphe réduit dit graphe symbolique, peut être construit de manière automatique. Cette construction repose sur la connaissance à priori d'un ensemble de symétries de comportement pour les différents objets composant le système étudié et contenues dans la description du RdP lui-même. Grâce à la syntaxe particulièrement structurée de cette sous-classe, les symétries sont déterminées de manière systématique, ce qui permet une construction entièrement automatique du graphe réduit (graphe symbolique) [46].

L'intérêt des réseaux colorés est fortement lié à l'existence d'outils de validation opérant directement sur le modèle coloré, et ce sans être obligé de se ramener au réseau non coloré correspondant (dépliage). En effet, l'espace des classes d'états qui est isomorphe à une chaîne de Markov permet la validation et l'analyse du modèle en construisant la matrice de transition entre classes d'états et ensuite la résolution des probabilités d'états pour les différentes classes en régime permanent. Cependant, il est important de préciser que l'extension de techniques classiques et la définition de techniques spécifiques, sont rendues particulièrement difficiles dans le cas général, du fait de la non-structuration des fonctions et des domaines de couleur.

Par ailleurs, l'introduction de la coloration a engendré des difficultés d'extension de certaines techniques existantes. Prenons l'exemple des méthodes basées sur les invariants linéaires. En effet, ces méthodes n'ont toujours pas été étendues au cas stochastique coloré ni même stochastique bien-formé. C'est la raison pour laquelle l'analyse du modèle des RdPSG coloré est basée essentiellement sur le développement du graphe des marquages accessibles (ordinaire ou agrégé).

D'autres développements théoriques visent à élargir le champ d'application des RdPSG colorés, en proposant d'autres méthodes pour éviter le problème de l'explosion combinatoire des modèles colorés à espace d'états large. Parmi ces méthodes, nous avons essentiellement celles qui combinent l'agrégation markovienne avec la décomposition tensorielle [163, 183, 184, 185]. Par ailleurs, il existe aussi des méthodes de calcul des bornes ([150], chap. 14) pour la classe des RDPS bien-formés.

2.8 Comparaison entre les RdPSG et les chaînes de Markov

Plusieurs techniques de modélisation analytique sont aujourd'hui disponibles. Chacune d'elles étant plus ou moins adaptée aux aspects spécifiques d'analyse des performances. On distingue principalement : les chaînes de Markov, les files d'attente et les réseaux de Petri stochastiques.

Les RdPSG constituent un mécanisme simple et élégant pour la description et l'évaluation des systèmes parallèles. Ils sont équivalents aux chaînes de Markov à temps continu [131] du point de vue de la solution. En effet, leur analyse peut être exprimée comme la solution d'un système d'équations linéaires et les résultats obtenus sont exacts.

Les chaînes de Markov sont le formalisme classique utilisé pour représenter les systèmes sans mémoire. Ils peuvent être considérés comme un modèle graphique représenté par le diagramme de transition d'états de la chaîne. Cependant, lorsqu'on souhaite utiliser une chaîne de Markov pour représenter le comportement d'un système, on se trouve confronté au problème de la description de cette chaîne ; car ceci nécessite l'énumération de tous les états et la détermination de toutes les transitions possibles entre chaque couple d'états. Ceci introduit d'une part, un risque d'erreur ou d'omission croissant, et d'autre part, la taille de cette chaîne généralement importante constitue un obstacle dans la représentation du système. Par conséquent, il faut avoir recours à des modèles dont la sémantique de fonctionnement est une chaîne de Markov, mais dont la description s'opère de manière relativement concise. Parmi ces outils de modélisation, les modèles de files d'attente et les RdPSG sont les plus couramment utilisés.

Par ailleurs, pour le modèle des chaînes de Markov, une variable d'état est identifiée initialement et le comportement est décrit en spécifiant les transitions de chaque état possible du modèle. Ainsi, tout changement dans le système pourrait nécessiter le changement de la définition de la variable d'état ce qui constitue un inconvénient majeur. D'autre part, la structure des modules composant un système complexe est perdue dans la représentation graphique d'une chaîne de Markov, puisque la variable d'état global du système complet est considérée ([150], chap. 4).

Les RdP combinent le pouvoir de modélisation comportementale au niveau de la spécification de transition d'état des machines à états finis et celui de la relation entrée/sortie des modèles de réseau. Le concept de variable d'état local présent dans les réseaux de Petri, permet l'extension d'un modèle abstrait et par conséquent le développement d'une description plus détaillée. Cette caractéristique n'est pas prévue par les modèles de files d'attente et de chaînes de Markov d'une manière aussi naturelle ([150], chap. 4).

Un autre inconvénient de l'utilisation des chaînes de Markov, est le fait que le lien entre le modèle et les objets du système à modéliser ne soit pas explicite. En effet, il est généralement difficile de générer directement un processus markovien d'un système complexe. Par contre, les RdPSG correspondent plus étroitement aux systèmes à modéliser. Par conséquent, le comportement de ces systèmes peut être facilement et efficacement représenté en utilisant les RdPSG plutôt que les chaînes de Markov directement.
D'autre part, le changement de l'état initial d'un système modélisé par un RdP revient à modifier uniquement le marquage initial. Par contre, ce changement pourrait produire une chaîne de Markov complètement différente malgré que la structure logique du système n'a pas changé. Ainsi,le RdP peut représenter plusieurs états différents du même système avec la même structure, alors qu'une chaîne de Markov représente un état fixé et doit être complètement modifiée à chaque fois que l'état initial change. Aussi, ajouter de nouvelles ressources ou de nouvelles opérations à un système revient à ajouter au RdP le modélisant de nouvelles places et de nouvelles transitions, ce qui se fait facilement. Par contre, un tel changement dans une CM nécessite la modification de la chaîne entière. De plus, le modèle des RdPSG offre un moyen efficace de génération automatisée des CM correspondantes. Il est donc plus simple de décrire des systèmes complexes à l'aide de RdPSG et de les convertir ensuite en des chaînes de Markov. Ceci a contribué à la popularité des RdPSG.

Ainsi, un des avantages du modèle des RdPSG est qu'il simplifie la génération de la chaîne de Markov. Le problème de dimension de celle-ci est encore là, mais le RdPSG permet une représentation plus compacte et logique du système. Cette caractéristique permet la modélisation de systèmes volumineux et complexes et l'équivalence avec les chaînes de Markov permet le calcul des paramètres de performance en bénéficiant des différentes approches développées pour les chaînes de Markov. D'autant plus, des logiciels informatiques sont utilisés pour traduire automatiquement un RdPSG donné en une chaîne de Markov équivalente d'une manière transparente à l'utilisateur. Par conséquent, les RdPSG sont un outil de modélisation simple et plus puissant que les chaînes de Markov.

2.9 Comparaison entre les RdPSG et les files d'attente

Les files d'attente et les RdPSG sont deux approches analytiques qui permettent la description et l'analyse par calcul des paramètres de performance de divers systèmes parallèles, tels que les systèmes de production, les systèmes informatiques et les réseaux de télécommunications.

Le développement des notations et des méthodes de résolution des deux formalismes a été conduit par deux écoles parallèles, qui se sont appuyées sur des approches différentes. En même temps, il y a au moins un certain recouvrement des deux puissances d'expression, ce qui fait que plusieurs systèmes peuvent être analysés au moyen de l'un ou l'autre des deux modèles. D'ailleurs, on pourrait même dire que ces approches sont fondamentalement identiques, du fait que chacune nécessite une modélisation préalable, et une fois que le modèle formel est décrit, les étapes restantes du processus sont indépendantes de la technique qui a été adoptée initialement. Cependant, dans des applications particulières, certaines caractéristiques telles que la finesse d'expression et l'efficacité de la solution font qu'une technique de modélisation soit plus appropriée que l'autre [186]. Ainsi, ces deux modèles peuvent être identiques dans certains cas et dans d'autres cas, il y a des différences significatives qui font que l'une des deux approches soit plus puissante et par conséquent plus adaptée que l'autre [187].

En fait, la ressemblance fondamentale et la différence intrinsèque entre les deux approches nécessitent au moins trois niveaux de comparaison basés sur les notations de description, la puissance et la finesse de modélisation ainsi que les méthodologies d'évaluation des performances. Il est clair que la faiblesse de l'une des deux approches dans un environnement particulier, peut être compensée au moyen des caractéristiques adéquates de l'autre, plus adaptée à l'environnement en question.

Ainsi, le choix de l'approche la plus appropriée est basé sur le pouvoir de description d'une part, et d'autre part sur l'efficacité des méthodes de résolution qu'on peut appliquer et des indices de performance qui peuvent être calculés à partir du modèle et aussi sur l'effort nécessaire pour cela.

Le modèle des files d'attente et plus généralement le modèle des réseaux de files d'attente (RFA) [53], constitue un outil pratique d'analyse des performances. La raison prin-

cipale de son succès est la combinaison de la puissance d'expression et de l'efficacité de la solution qu'il offre. Par ailleurs, il est essentiellement utilisé pour la modélisation des systèmes dans lesquels il y a un partage de ressources. Toutefois, s'il est bien adapté pour évaluer l'utilisation de ressources partagées entre des processus indépendants, son champ d'application ne s'étend malheureusement pas aux systèmes possédant des schémas de synchronisation entre processus asynchrones. En d'autres termes, ce modèle ne permet pas de prendre en compte les comportements de dépendance entre plusieurs processus d'un même système. Ces processus n'évoluent généralement pas de manière indépendante, il existe entre eux des relations de coopération, de concurrence, et de communication que l'on peut résumer sous le terme de **synchronisation**. Ces relations sont présentes dans la quasi-totalité des systèmes et jouent un rôle déterminant pour leurs performances. Cependant, leur représentation directe sous forme de modèle de file d'attente est très difficile et de plus, viole les hypothèses d'application des méthodes standards. Ainsi, une des limitations du modèle des files d'attente en général, et particulièrement du modèle des files d'attente avec rappel, réside dans la représentation de la synchronisation. De toute façon, l'étude des modèles de files d'attente avec des caractéristiques de synchronisation est l'un des principaux problèmes dans le domaine de l'évaluation des performances.

C'est pour pallier à cette carence que les RdPSG ont été introduits. Ils constituent un important modèle graphique et mathématique utilisé pour étudier le comportement de divers systèmes. C'est un outil prometteur qui offre une grande puissance descriptive, permettant ainsi la modélisation et l'évaluation des systèmes parallèles incluant la concurrence, le non-déterminisme et la synchronisation. En effet, par opposition aux modèles de files d'attente, les RdPSG incluent naturellement et de façon très élégante, les différentes structures de synchronisation, afin d'obtenir une modélisation plus fine des systèmes.
En fait, même si certains mécanismes de synchronisation (d'ailleurs dérivés des RdP) ont été introduits dans les RFA, tels que : les sémaphores, les drapeaux et les mécanismes fork/join [151], les RdP offrent actuellement le formalisme le plus naturel pour représenter ces phénomènes et l'adjonction des modèles de files d'attente par des moyens ad-hoc n'atteint toujours pas la généralité et la simplicité de la modélisation de la concurrence par les réseaux de Petri ([128], chap.9).

D'autre part, les files d'attente à capacité finie sont utilisées pour représenter des systèmes réels avec certaines contraintes sur les ressources. Cependant, cette limitation de capacité engendre un problème de blocage des différentes files d'attente du réseau. Pour représenter les différents comportements des systèmes réels avec ressources limitées, différents mécanismes de blocage ont été définis dans la littérature [188]. Cependant, un autre inconvénient des réseaux de files d'attente est le fait de ne pas pouvoir représenter ces phénomènes de blocage. Pour cela, Gribaudo et Sereno [189], ont proposé une technique qui permet de représenter les réseaux de files d'attente à capacité finie à l'aide des RdPSG. Cette modélisation permet d'une part de représenter des RFA ayant des noeuds avec des types de blocage hétérogènes. D'autre part, elle permet de bénéficier des résultats développés dans le domaine des RdPSG.

Ainsi, la possibilité d'inclure facilement les aspects de synchronisation, et de représenter très aisément les différents mécanismes de blocage dans le modèle des RdPSG, en plus de sa puissance d'expression, constituent les principales caractéristiques qui ont rendu

le modèle très populaire et même supérieur aux réseaux de files d'attente dans certains cas. Cet avantage du point de vue représentation, est essentiellement dû au fait que la modélisation à l'aide des FA est abstraite, par contre les RdP offre la possibilité d'une modélisation très détaillée.

D'autre part, l'un des intérêts majeurs des réseaux de Petri stochastiques généralisés et de pouvoir combiner l'analyse qualitative et l'analyse quantitative [190, 191], tout en utilisant des algorithmes efficaces. D'ailleurs, les résultats de l'analyse des propriétés qualitatives et ceux des algorithmes d'analyse structurelle sont souvent exploités pour la validation du système et aussi pour l'évaluation des performances. Contrairement aux modèles de FA qui ne peuvent pas être utilisés pour mener une étude des propriétés qualitatives de systèmes.

Dans la littérature, beaucoup d'efforts ont été consacrés à l'étude des relations entre les RFA et les RdPS. Une comparaison détaillée est donnée dans [187, 150].
Dans plusieurs cas, les études développées dans le domaine des RFA ont été adaptées aux RdPS, telle que la solution en forme-produit, les méthodes d'approximation, les techniques de borne, etc. Dans tous les cas, les caractéristiques du formalisme RdP ont été utilisées pour l'exploitation des potentialités de ces méthodes. D'autres travaux [192] ont proposé le processus inverse, qui consiste à utiliser les résultats développés dans le domaine des RdP pour l'étude des propriétés des RFA. Cette méthode permet d'obtenir plusieurs avantages pour l'analyse qualitative et quantitative des systèmes. En particulier, elle offre la possibilité d'utiliser les résultats et les outils développés dans le domaine des RdP.

Par ailleurs, les différents formalismes basés sur les RdP ont été introduits en réponse à un besoin réel dans le domaine de l'évaluation des performances des systèmes distribués et parallèles. Aucun des formalismes proposés n'a été spécifiquement désigné à substituer les RFA. En fait, certains étaient désignés à compléter les descriptions RFA pour des applications larges et complexes [193].

En résumé, les principales caractéristiques des RdPSG, qui différencient ce formalisme des RFA sont ([150], chap. 4) :

- La structure du réseau et l'état initial du système peuvent être facilement décrits en termes graphiques ("une image vaut mieux que mille mots") ;
- Le système peut être facilement paramétré grâce au marquage initial (pour les réseaux colorés la structure peut aussi être paramétrée) ;
- La séquence, le choix et la concurrence réelle sont bien représentées ;
- Une puissante représentation des mécanismes de synchronisation, de blocage et autres, ce qui facilite l'introduction des contraintes d'exclusion mutuelle, fork-join, etc ;
- La construction modulaire, hiérarchique et distribuée de larges modèles est naturelle, ceci permet la conception progressive lors des différentes phases d'un projet ;
- Une analyse qualitative aussi bien que quantitative du système étudié ;
- Une analyse systématique de systèmes complexes grâce à l'utilisation d'approches théoriques et des outils pratiques très performants.

Pour ce qui est des RdPSG colorés, certains chercheurs, tel que Zénie [153] les voient

comme une convergence des RFA et des RDPSG. Cet outil généralise les deux approches précédentes. En fait, dans les RFA par exemple, les clients peuvent être partitionnés en classes tels que tous les clients d'une même classe sont statistiquement identiques du point de vue des probabilités de routage et des demandes de service. Ces classes de clients peuvent être modélisées simplement par des couleurs dans un RDPSG coloré.

Ainsi, les principaux avantages des RdPSG sont la grande flexibilité du formalisme lui-même, ce qui permet différentes interprétations et extensions, l'utilisation du formalisme dans différents domaines d'applications ainsi que le choix entre différentes modélisations et techniques d'analyse.

2.10 Conclusion

L'objet de ce chapitre a été la présentation des principaux concepts liés aux réseaux de Petri stochastiques généralisés ordinaires et colorés. Ces modèles constituent des outils puissants de modélisation et d'évaluation des performances des systèmes parallèles. Ils sont suffisamment connus, aussi bien par l'étendue de leurs résultats théoriques que par la diversité de leurs domaines d'application.

Les RdPSG ont une valeur pratique, car en plus des transitions avec des temps de franchissement distribués exponentiellement, ce formalisme comprend aussi des transitions immédiates qui sont utilisées pour modéliser des actions logiques (synchronisation par ex.) ainsi que des activités extrêmement rapides, telle qu'aucune temporisation ne peut leur être associée. Ceci permet d'une part d'augmenter la puissance d'expression du modèle. D'autre part, cette classification des transitions réduit l'espace d'accessibilité du modèle RdPSG par rapport au RdP non-temporisé sous-jacent.

Par ailleurs, les files d'attente avec rappel ont été largement utilisées pour l'évaluation des performances des systèmes avec phénomènes d'appels répétés. Elles sont en effet, très adaptées à la représentation de divers phénomènes tel que le partage de ressources. Toutefois, le pouvoir de modélisation limité des files d'attente en général et plus particulièrement des FAR, les rend inadéquates pour représenter beaucoup de caractéristiques des systèmes parallèles complexes telle que la synchronisation et le blocage. Contrairement à ces modèles, les RdPSG ont l'avantage de permettre d'une part, une représentation très aisée des contraintes de synchronisation et des mécanismes de blocage qui caractérisent le fonctionnement de beaucoup de systèmes parallèles, et ils permettent d'autre part, de faire une évaluation qualitative et quantitative du système modélisé.

Un autre avantage des RdPSG markoviens est que leur graphe d'accessibilité est isomorphe à une chaîne de Markov incluse, ce qui permet l'application des techniques et des méthodes développées pour les chaînes de Markov dans le but de fournir des résultats exacts. En fait, la vérification de l'ergodicité de cette chaîne, nécessaire pour l'évaluation des performances stationnaires, revient à faire une validation qualitative du modèle. Ainsi, les RdPSG permettent en une approche la preuve et l'évaluation du système modélisé.

Toutes ces raisons, nous encouragent à utiliser le modèle des RdPSG et celui des

RdPSG colorés pour l'analyse qualitative aussi bien que pour l'évaluation des performances et de la fiabilité des systèmes avec rappel. En effet, d'un point de vue description, ces modèles nous permettent d'incorporer des caractéristiques et des détails du comportement du système étudié, qui peuvent être assez difficiles à modéliser et à analyser par les méthodes conventionnelles. Un autre avantage essentiel, est le fait que ces modèles de haut-niveau nous permettent de dériver les formules des indices de performance et de fiabilité exacts.

Vu ces avantages et vu le manque de résultats exacts dans le domaine des files d'attente avec rappel, nous proposons dans le chapitre suivant une approche de modélisation et d'analyse des systèmes avec rappel à l'aide des RdPSG. Cette approche offre la possibilité d'utiliser les résultats et les outils très avancés développés dans ce domaine, et permet par conséquent, la résolution de divers problèmes qui n'ont pas encore été résolus en utilisant la théorie des FAR.

Chapitre 3

Analyse des Systèmes Mono-Classe avec Rappel à l'aide des RdPSG

3.1 Introduction

Durant ces deux dernières décennies, un effort considérable a été consacré par beaucoup de praticiens et de chercheurs théoriciens, à l'évaluation des performances des systèmes avec rappel, et plus précisément aux files d'attente avec rappel, qui est le modèle conventionnel habituellement appliqué pour l'analyse des performances de ces systèmes. L'intérêt porté à ce thème est principalement expliqué par les développements et les avancées technologiques, notamment dans les domaines de l'informatique et des télécommunications.

Cependant, la prise en considération des phénomènes d'appels répétés a rendu les résultats des files d'attente standards inadéquats et a introduit de grandes difficultés analytiques particulièrement pour les modèles multi-serveurs. En fait, des résultats explicites existent pour quelques modèles avec des hypothèses contraignantes sur certains paramètres tels que la capacité de la population, le nombre de serveurs, la fiabilité des serveurs, l'homogénéité des clients, etc, alors que pour beaucoup d'autres les résultats sont obtenus par approximation ou par simulation.

Concernant les FAR multi-serveurs à source finie, les travaux dans ce domaine restent assez limités, et les résultats sont obtenus généralement par des méthodes numériques [11, 12, 16, 18, 43]. Pour ce qui est des modèles avec rappel, source finie et plusieurs serveurs non-fiables, les références sont encore plus rares. D'ailleurs, les seuls travaux portant sur ce modèle sont de Roszik et Sztrik [33, 34] où les serveurs sont supposés être hétérogènes.

En fait, il est d'une importance basique d'étudier la fiabilité des systèmes avec rappel où les serveurs sont sujets à des pannes et des réparations aléatoires, et ce à cause de la forte influence des pannes sur les performances du système et aussi du fait que les utilisateurs de ces systèmes avec rappel (systèmes téléphoniques, par exemple) deviennent de plus en plus exigeants. Par conséquent, ces systèmes nécessitent une qualité de service d'un certain niveau même en cas de panne. Ainsi, une importance particulière devrait être accordée à l'étude des systèmes combinant la présence simultanée des phénomènes de rappel et la non fiabilité des serveurs.

Cependant, l'étude de ces systèmes par le modèle conventionnel des files d'attente avec rappel pourrait être assez compliqué. En effet, la présence de ces différentes caractéristiques (source finie, multiple serveurs, rappels, pannes et réparations) dans un même système, introduit de multiples problèmes de synchronisation, en plus des problèmes de blocage liés aux appels répétés.

D'autre part, les réseaux de Petri stochastiques généralisés constituent un important modèle graphique et mathématique, qui offre une grande puissance descriptive, permettant la modélisation et l'analyse des systèmes qui sont caractérisés par le fait d'être parallèles, distribués et stochastiques. La possibilité d'inclure facilement les aspects de synchronisation, de concurrence et de représenter très aisément le blocage ainsi que différents phénomènes stochastiques dans un même modèle, est la principale caractéristique qui les a rendus très populaires.

En effet, depuis ces dernières décennies, les RdPSG représentent un centre d'intérêt pour de nombreuses recherches dans les domaines de la fiabilité et des performances. Ce formalisme de modélisation de haut-niveau a été largement appliqué pour décrire le comportement dynamique des systèmes concurrents et asynchrones et aussi pour étudier le comportement qualitatif et quantitatif de ces systèmes parallèles.

Ainsi, nous proposons dans ce chapitre, la modélisation et l'évaluation des performances des systèmes multi-serveurs avec rappel et source finie à l'aide des RdPSG. Cette approche nous permet de décrire d'une manière simple et précise différents systèmes avec serveurs fiables et serveurs non fiables. En fait, deux types de panne sont définies dans la littérature des FAR : les *pannes actives* qui se produisent lorsque le serveur est en cours de service, et les *pannes indépendantes* qui se produisent indépendamment de l'état du serveur, oisif ou occupé, mais avec la même probabilité, ce qui constitue une contrainte assez forte. La puissance d'expression du modèle des RdPSG nous a permis de définir dans [194] une nouvelle discipline de panne plus générale dite : *discipline de pannes dépendantes*. Dans ce cas, les taux de panne des serveurs actifs (pannes actives) et les taux de panne des serveurs oisifs (pannes oisives), peuvent être égaux ou différents, ce qui ne complique en aucun cas ni la modélisation ni l'analyse.

D'autre part, cette approche nous offre la possibilité d'effectuer une analyse qualitative aussi bien que quantitative de ces systèmes, en se basant sur les méthodes et les outils très avancés développés pour l'évaluation des performances et de la fiabilité des RdPSG, pour apporter des solutions numériques exactes au domaine de l'analyse des systèmes avec rappel.

Dans la première section de ce chapitre, nous présentons le modèle de RdPSG décrivant les systèmes avec rappel, source finie et serveurs fiables. Pour la validation du modèle proposé, nous procédons à une comparaison du processus stochastique sous-jacent au processus de la FAR correspondante. Nous donnons ensuite les formules permettant l'obtention des paramètres de performance exacts. Dans la section qui suit, nous nous intéressons particulièrement aux systèmes avec serveurs non fiables. Nous proposons dans ce cadre, un modèle pour chaque discipline de panne et nous déduirons ensuite les for-

mules des indices de performance et de fiabilité correspondants. Nous terminons par une conclusion.

3.2 Les systèmes avec rappel et serveurs fiables

Dans cette section, nous considérons les systèmes avec rappel, complètement markoviens à source finie de taille K et à s serveurs identiques, parallèles et fiables i.e. que le risque de panne est nul. Dans ces systèmes, les clients (ou appels) primaires arrivent suivant un processus quasi-aléatoire de taux λ, ce qui signifie que la probabilité qu'un client génére une requête pour le service dans tout intervalle $(t, t+dt)$ est $\lambda(K-i-j)dt+o(dt)$ (où i est le nombre de clients en service et j est le nombre de clients en orbite) quand $dt \to 0$, si le client est libre à l'instant t, et zéro si le client est en orbite ou en cours de service à l'instant t, et ce indépendamment du comportement de tous les autres clients. Ainsi, chaque client est soit libre, en service ou en orbite à tout instant. D'autre part, les serveurs peuvent être dans l'un des deux cas : oisif ou occupé. Si un client trouve un serveur oisif, il est immédiatement pris en charge puis quittera le système dès la fin du service dont la durée suit une loi exponentielle de paramètre μ. Si par contre, le client ne trouve aucun serveur disponible (i.e. que tous les serveurs sont occupés), il rejoint l'orbite et génère un flux exponentiel d'appels répétés avec le taux ν.

3.2.1 Modélisation des systèmes multi-serveurs avec rappel

La figure 3.1 présente une modélisation possible de ces systèmes multi-serveurs avec appels répétés et source finie par un RdPSG. Ce modèle représente une généralisation du cas mono-serveur que nous avons présenté dans [195], ainsi qu'une extension des travaux que nous avons présentés dans [196, 197]. D'autre part, cette modélisation est générique, en ce sens que la capacité de la population est représentée par le paramètre entier positif K qui apparaît comme marquage initial de la place Cus_Free.

Dans ce modèle :
- La place Cus_Free contient les clients (ou sources) libres ;
- La place Ser_Idle contient les serveurs disponibles oisifs ;
- La place Cus_Serv contient les clients en cours de service ou encore les serveurs occupés ;
- La place $Orbit$ représente l'orbite ;
- La place $Choice$ représente la condition qu'un client (libre ou de l'orbite) demande à être servi. Ainsi, cette place $Choice$ assure la synchronisation entre l'arrivée des appels primaires et celle des appels répétés.

Le marquage initial du réseau est :
$M_0 = \{M(Cus\_Free), M(Choice), M(Cus\_Serv), M(Ser\_Idle), M(Orbit)\} = \{K, 0, 0, s, 0\}$, ce qui signifie que tous les clients sont initialement libres, que tous les serveurs sont disponibles et que l'orbite est vide.

Remarque : Dans tous les modèles proposés, les transitions immédiates sont représentées par des traits et les transitions temporisées par des rectangles. D'autre part, nous

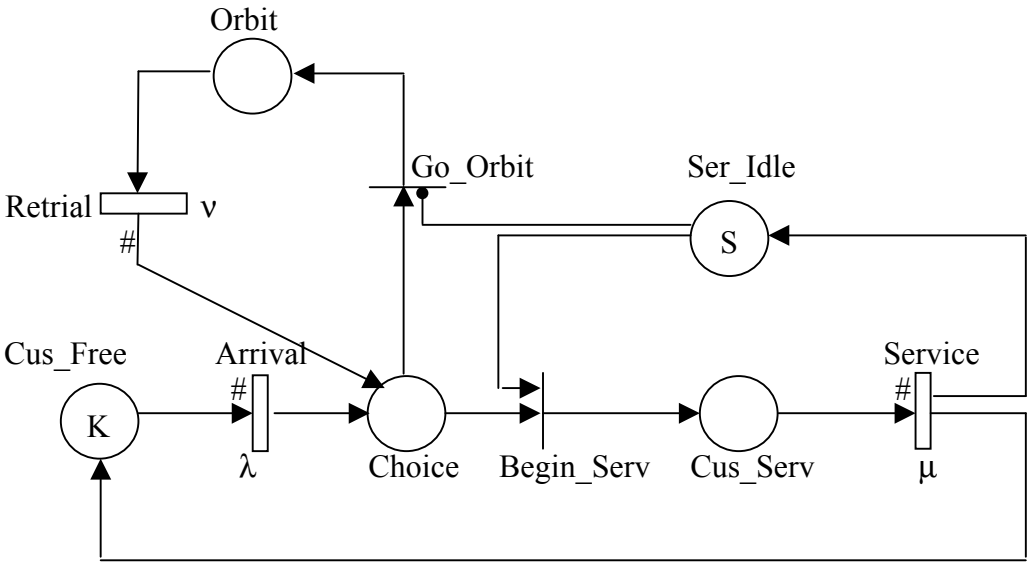

FIG. 3.1 – Le RdPSG modélisant les systèmes avec rappel, serveurs fiables et source finie

supposons que les jetons restent comptabilisés dans les places en entrée d'une transition, tant que son franchissement n'est pas terminé.

Le franchissement de la transition *Arrival* représente l'arrivée d'un appel primaire généré par une source libre. Ainsi, la place *Choice* reçoit un jeton, ce qui représente la condition qu'un client est prêt pour le service. La sémantique de service de cette transition est une *sémantique à serveurs infinis*, ceci signifie que son taux de tir est égal à $ED(Arrival, m).\lambda$, où $ED(\text{Arrival}, m)$ est le degré de franchissement de la transition *Arrival* dans un marquage m. Par conséquent, tous les clients libres peuvent générer des appels primaires. Ceci est représenté par le symbole # placé à côté de la transition *Arrival*.

À l'arrivée d'un client à la place *Choice*, si la place *Ser_Idle* contient au moins un jeton qui correspond à un serveur oisif, la transition immédiate *Begin_Serv* sera tirée instantanément, et le marquage de la place *Cus_Serv* sera incrémenté de 1. Ceci représente le fait que le client a commencé son service et qu'un serveur est passé de l'état oisif à l'état occupé. Par contre, si la place *Ser_Idle* est vide (aucun serveur n'est oisif), c'est plutôt la transition immédiate *Go_Orbit* qui sera tirée et par conséquent le client bloqué rejoint l'orbite. Une fois en orbite, les clients se comportent indépendamment les uns des autres, et chacun d'eux rappelle pour le service après un délai de temps distribué exponentiellement avec un taux ν. Ainsi, la sémantique de la transition *Retrial* doit être à serveurs infinis, i.e. que le taux de tir est dépendant du marquage de la place *Orbit*, ce qui est représenté par le symbole # à côté de la transition *Retrial*. Dès le franchisement de cette transition temporisée, un jeton est déposé dans la place *Choice*. Ceci matérialise l'évènement de rappel d'un client de l'orbite. Ainsi, plusieurs clients de l'orbite peuvent rappeler pour le service simultanément. En fait, un taux de rappel constant, dans ce cas, signifie qu'un seul client à la fois peut rappeler pour le service. Cette politique de rappel constant peut être facilement modélisé avec une transition *Retrial* à sémantique à serveur

unique.

D'autre part, à la fin d'une période de service, un jeton sera déposé dans la place Ser_Idle pour représenter le serveur qui passe de l'état occupé à l'état oisif, et un autre jeton représentant le client qui devient libre est déposé dans la place Cus_Free. Le franchissement de la transition $Service$ est aussi à sémantique à serveurs infinis, car tous les serveurs sont parallèles et donc peuvent travailler simultanément.

3.2.2 Validation du modèle

En étudiant le modèle de RdPSG proposé, avec différentes valeurs pour la source de clients K et le nombre de serveurs s, nous avons constaté que les chaînes de Markov réduites correspondantes, ont la même forme et évoluent de la même manière en fonction de ces deux valeurs. Ainsi, la CMTC réduite illustrant la sémantique de ce RdPSG et décrivant l'évolution des systèmes multi-serveurs avec rappel et source finie, est donnée dans la Figure 3.2.

Pour la validation du modèle de RdPSG proposé dans la figure 3.1, nous avons comparé la CMTC (CM_2) décrivant sa sémantique et donnée dans le figure 3.2, à celle du modèle de FAR $M/M/s//K$ (CM_1) donnée dans la Figure 1.5 du chapitre 1. Ainsi, l'étude comparative est basée sur l'évolution des processus sous-jacents aux deux modèles.

La comparaison de ces deux processus (CM_1 et CM_2), nous a permis de constater que :

1. Le nombre d'états des deux chaînes de Markov est identique et égal à $(s+1) \times (K - s + 1)$;
2. Il existe une bijection (isomorphisme) F entre les espaces d'états des deux modèles, car à tout état $(i, j) \in CM_1$ du processus décrivant la FAR, correspond un seul état de la forme $(K - i - j, 0, i, s - i, j) \in CM_2$ du processus correspondant au RdPSG et inversement, où i est le nombre de serveurs occupés et j est le nombre de clients on orbite. Formellement, la bijection F est donnée comme suit :

   $$\forall (i,j) \in E_1 : \ F(i,j) = (K-i-j, 0, i, s-i, j) \ et \ F^{-1}(K-i-j, 0, i, s-i, j) = (i,j).$$

   où : $0 \leq i \leq s$ et $0 \leq j \leq K - s$ et E_1 est l'espace d'état de la CM_1.

   En effet, dans le modèle de RdPSG, le nombre de jetons dans la place $Choice$ est toujours nul, car c'est une place à partir de laquelle on peut tirer uniquement deux transitions immédiates. D'autre part, le marquage de la place Ser_Idle représente le nombre de serveurs oisifs qui est égal à $s - i$, et celui de la place Cus_Free correspond au nombre de clients libres qui est égal à $K - i - j$.
3. Les taux de transition entre états équivalents (isomorphes) sont identiques. Ainsi, une transition existe dans CM_1 ssi la même transition (avec le même taux) existe dans l'autre processus CM_2. En d'autres termes, si l'état x_1 est isomorphe à y_1 et x_2 est isomorphe à y_2, et il existe une transition entre x_1 et x_2 alors la même transition

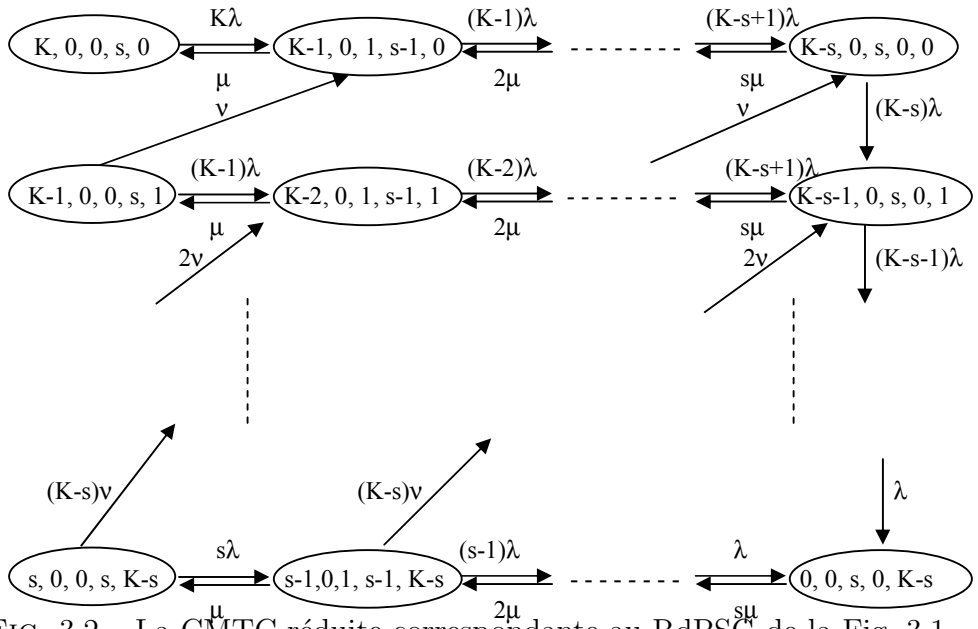

FIG. 3.2 – La CMTC réduite correspondante au RdPSG de la Fig. 3.1

relie y_1 à y_2 et inversement.

Ainsi, nous pouvons déduire qu'il y a une bijection entre les espaces d'états des deux processus. Par conséquent, le modèle de FAR $M/M/s//K$ et le modèle de RdPSG proposé sont sémantiquement équivalents du fait que les processus sous-jacents sont isomorphes, ce qui nous permet de valider le modèle proposé.

3.2.3 Analyse du système

Une fois que le modèle de RdPSG est construit, son analyse consiste d'une part à définir ses propriétés qualitatives (comportementales et structurelles) telles que la bornitude, la vivacité, l'absence de blocage, etc, et d'autre part, à calculer ses paramètres de performance quantitatifs. En fait, l'analyse quantitative n'a de sens que si une analyse qualitative a été préalablement menée. Il est en effet inutile de vouloir obtenir les performances d'un système qui se trouve dans un état de blocage par exemple, car les techniques d'évaluation de performances pourront très bien calculer des paramètres sans se rendre compte de l'éventualité d'un tel blocage. Dans ce cas, ces techniques fourniront, donc, des résultats erronés.

D'autre part, l'évaluation des performances à l'état stationnaire nécessite l'ergodicité du modèle, ce qui revient à vérifier certaines propriétés qualitatives.

Le modèle de RdPSG proposé (Fig. 3.1) est borné, vivant, sans blocage et le marquage

initial est un état d'accueil. Par conséquent, ce modèle admet un état stationnaire. Nous notons par $\pi = (\pi_1, \pi_2, ..., \pi_n)$ la distribution des probabilités de marquage à l'état stationnaire où π_i est la probabilité que le processus est à l'état M_i et $M_i(p)$ est le nombre de jetons dans la place p dans le marquage M_i. Nous notons par A l'ensemble des marquages tangibles accessibles et par $A(t)$ l'ensemble des marquages tangibles accessibles dans lesquels la transition t est franchissable.

L'évaluation des performances s'intéresse au calcul des paramètres de performance d'un système. Parmi les paramètres les plus importants, que l'on souhaite évaluer pour de nombreux systèmes, nous citons : *le débit, le nombre moyen d'une entité donnée, le temps d'attente, le temps de réponse et le taux d'utilisation des ressources.*

En ayant les probabilités d'état stationnaire π, divers indices de performance intéressants peuvent être calculés. Nous donnons dans ce qui suit, les formules explicites de calcul des paramètres des systèmes multi-serveurs avec rappel et source finie, en se basant sur le modèle de RdPSG. Nous nous intéressons particulièrement, au calcul des valeurs moyennes de ces paramètres de performance, car ces valeurs ont un intérêt statistique.

- **Le nombre moyen de serveurs occupés (n_s) :**
  Il correspond au nombre moyen de jetons dans la place Cus_Serv qui est aussi le nombre moyen de clients en service.
$$n_s = \sum_{i:M_i \in A} M_i(Cus\_Serv).\pi_i$$
- **Le nombre moyen de serveurs oisifs (n_d) :**
  Il correspond au nombre moyen de jetons dans la place Ser_Idle.
$$n_d = \sum_{i:M_i \in A} M_i(Ser\_Idle).\pi_i$$
- **Le nombre moyen de clients en orbite (n_o) :**
  Il correspond au nombre moyen de jetons dans la place $Orbit$.
$$n_o = \sum_{i:M_i \in A} M_i(Orbit).\pi_i$$
- **Le nombre moyen de clients libres (n_a) :**
  Il correspond au nombre moyen de jetons dans la place Cus_Free.
$$n_a = \sum_{i:M_i \in A} M_i(Cus\_Free).\pi_i$$
- **Le nombre moyen de clients dans le système (n) :**
  Il correspond au nombre moyen de clients en service ou en orbite.
$$n = n_s + n_o$$
- **Le taux moyen de génération des appels primaires ($\overline{\lambda}$) :**
  Il correspond à la fréquence (débit) de la transition $Arrival$.
$$\overline{\lambda} = \sum_{i:M_i \in A(Arrival)} \lambda.M_i(Cus\_Free).\pi_i$$

– **Le taux moyen de génération des appels répétés ($\overline{\nu}$) :**
Il correspond à la fréquence de rappel des clients en orbite.

$$\overline{\nu} = \sum_{i:M_i \in A(Retrial)} \nu.M_i(Orbit).\pi_i$$

– **Le taux moyen de service ($\overline{\mu}$) :**
Il correspond à la fréquence de la transition *Service*.

$$\overline{\mu} = \sum_{i:M_i \in A(Service)} \mu.M_i(Cus\_Serv).\pi_i$$

– **La probabilité de blocage d'un client primaire (B_p) :**

$$B_p = \frac{\sum_{j:M_j \in A} \sum_{i=1}^{K-s} i.\lambda.Prob[M_j(Cus\_Free) = i \& M_j(Ser\_Idle) = 0]}{\overline{\lambda}}$$

– **La probabilité de blocage d'un client de l'orbite (B_r) :**

$$B_r = \frac{\sum_{j:M_j \in A} \sum_{i=1}^{K-s} i.\nu.Prob[M_j(Orbit) = i \& M_j(Ser\_Idle) = 0]}{\overline{\nu}}$$

– **La probabilité de blocage des clients (B) :**

$$B = B_p + B_r$$

– **L'utilisation de c serveurs (U_c) : ($1 \leq c \leq s$)**
Ceci correspond à la probabilité que c serveurs sont occupés :

$$U_c = \sum_{i:M_i(Cus\_Serv) \geq c} \pi_i$$

– **La disponibilité de c serveurs (A_c) : ($1 \leq c \leq s$)**
Ceci correspond à la probabilité que c serveurs sont oisifs.

$$A_c = \sum_{i:M_i(Ser\_Idle) \geq c} \pi_i$$

– **Le temps d'attente moyen ($\overline{W}$) :**
Le temps d'attente (virtuel) d'un client correspond à la durée de temps entre l'arrivée du client et son début de service. Le temps d'attente moyen $\overline{W}$ à l'état stationnaire, peut être facilement obtenu à l'aide de la formule de Little :

$$\overline{W} = n_o/\overline{\lambda}$$

– **Le temps de réponse moyen ($\overline{R}$) :**

$$\overline{R} = n/\overline{\lambda}$$

3.3 Les systèmes avec rappel et serveurs non-fiables

En pratique, certains composants des systèmes sont sujets à des pannes aléatoires (voir par exemple [112, 113]), il est donc d'une importance basique d'étudier la fiabilité des systèmes avec appels répétés où les serveurs sont sujets à des pannes et des réparations aléatoires, et ce à cause de la forte influence des pannes qui peuvent avoir un impact négatif non négligeable sur les paramètres de performance du système. On parle alors de systèmes avec rappel et serveurs non-fiables. Ainsi, pour l'analyse de ces systèmes, il est important de prendre ceci en compte lors de la construction du modèle. Dans ce cas, les serveurs auront un taux de panne et un taux de réparation aléatoires, contrairement aux systèmes avec serveurs fiables, où ces taux sont nuls.

Dans cette section, nous allons présenter la modélisation ainsi que l'analyse des performances et de la fiabilité des systèmes avec rappel et différentes disciplines de panne, à l'aide du modèle des RdPSG.

3.3.1 Description mathématique

Nous considérons les systèmes avec rappel, une population (source) finie de K ($1 < K < \infty$) clients homogènes et une station de service qui comprend s ($s \geq 1$) serveurs identiques et non fiables.

Chaque client peut être dans l'un des trois états : libre, en service ou en orbite. Les clients libres permettent la génération des appels primaires suivant un processus quasi-aléatoire de taux λ. Chaque serveur peut être occupé ou oisif. D'autre part, il peut être opérationnel ou en panne. Ainsi, à l'arrivée d'un appel primaire ou répété, si un des serveurs est opérationnel et oisif, alors cet appel peut être servi immédiatement suivant une loi exponentielle de taux μ et le serveur passera à l'état occupé ; autrement, si tous les serveurs sont occupés ou en panne, le client entre en orbite et devient une source de génération d'un flux d'appels répétés distribué exponentiellement de taux ν, jusqu'à ce qu'il trouve un serveur opérationnel et oisif pour être servi. À la fin du service, le serveur devient oisif et le client devient libre et peut ainsi générer un autre appel primaire.
En fait, les sources d'appels répétés se comportent indépendamment les unes des autres et sont persistantes. D'autre part, les appels des sources (clients) sont adressés aux serveurs opérationnels oisifs d'une manière aléatoire et sans aucun ordre de priorité.

Chaque serveur peut tomber en panne durant l'intervalle $(t, t+dt)$ avec la probabilité $\gamma dt + o(t)$ s'il est occupé, et avec la probabilité $\delta d(t) + o(t)$ s'il est oisif. En fait, deux politiques de panne ont été considérées dans la littérature [31, 32, 33, 34] :

- La politique de pannes actives, où $\delta = 0$ et $\gamma > 0$;
- La politique de pannes indépendantes, où $\delta = \gamma > 0$.

Dans la première politique, le serveur ne peut tomber en panne qu'en étant actif. Par contre, dans la politique de pannes indépendantes, le serveur peut tomber en panne indépendamment de son état, et la probabilité de panne est la même que le serveur soit oisif ou occupé ($\delta = \gamma > 0$). En fait, cette hypothèse vise principalement à simplifier l'analyse. Cependant, dans les systèmes réels, ces taux peuvent être identiques ou différents. Ainsi, nous avons défini dans [194] une nouvelle politique de panne plus générale que nous avons

appelée **politique de panne dépendante**. Dans ce cas, la probabilité de panne d'un serveur dépend de son état. Ainsi, les taux $\delta > 0$ and $\gamma > 0$ peuvent être égaux ou différents. Par conséquent, cette discipline est une généralisation de la discipline de panne indépendante.

D'autre part, si un serveur tombe en panne en étant occupé, le client interrompu entre en orbite et rappellera ultérieurement pour le service, avec une remise à zéro du travail réalisé avant que la panne ne survienne, on parle dans ce cas de *panne à service répété*. Le temps de réparation d'un serveur quelconque est distribué exponentiellement avec un taux τ. Nous supposons que le réparateur suit la discipline FIFO pour réparer les serveurs en panne et après la réparation, le serveur est aussi bon qu'avant de tomber en panne.

En fait, il est important de préciser que quand les temps de franchissement sont distribués exponentiellement et les mesures de performance auxquelles on s'intéresse sont les indices moyens, les différentes disciplines de service des files d'attente (FIFO, LIFO ou aléatoire) donnent les même résultats. Par conséquent, l'ordre aléatoire qui est la politique par défaut des RdP peut être appliqué et l'analyse donnera les résultats espérés.

Par ailleurs, si tous les serveurs sont en panne, deux différents cas peuvent être envisagés :

- **Cas de sources bloquées**, où toutes les arrivées primaires, les appels répétés et les opérations de traitement sont arrêtés ;
- **Cas de sources intelligentes** (non bloquées), où seulement le service est interrompu et toutes les autres opérations continuent, autrement dit, les nouveaux appels et les appels répétés peuvent être générés indépendamment de l'état des serveurs.

Dans ces systèmes, toutes les variables aléatoires sont supposées être distribuées exponentiellement et indépendantes les unes des autres.

Comme il peut être constaté, ces systèmes multi-serveurs à source finie sont plutôt compliqués, puisqu'ils incluent différentes politiques de panne, et modes de fonctionnement, en plus du phénomène d'appels répétés. Ainsi, notre objectif est de proposer la modélisation et l'analyse de ces systèmes, à l'aide des RdPSG, ainsi que la définition des formules permettant l'obtention des principaux indices de performance et de fiabilité exacts.

3.3.2 Modélisation des systèmes avec pannes actives et sources intelligentes

La Figure 3.3 décrit le modèle RdPSG correspondant aux systèmes multi-serveurs avec rappel, pannes actives et sources intelligentes.

- La place Ser_Idle représente les serveurs oisifs et opérationnels ;
- La place Ser_Down contient les serveurs en panne.

Le marquage initial du réseau est :
$M_0 = \{M(Cus\_Free), M(Choice), M(Cus\_Serv), M(Ser\_Idle), M(Orbit), M(Ser\_Down)\} = \{K, 0, 0, s, 0, 0\}$, ce qui signifie que toutes les sources sont initialement libres, les s serveurs

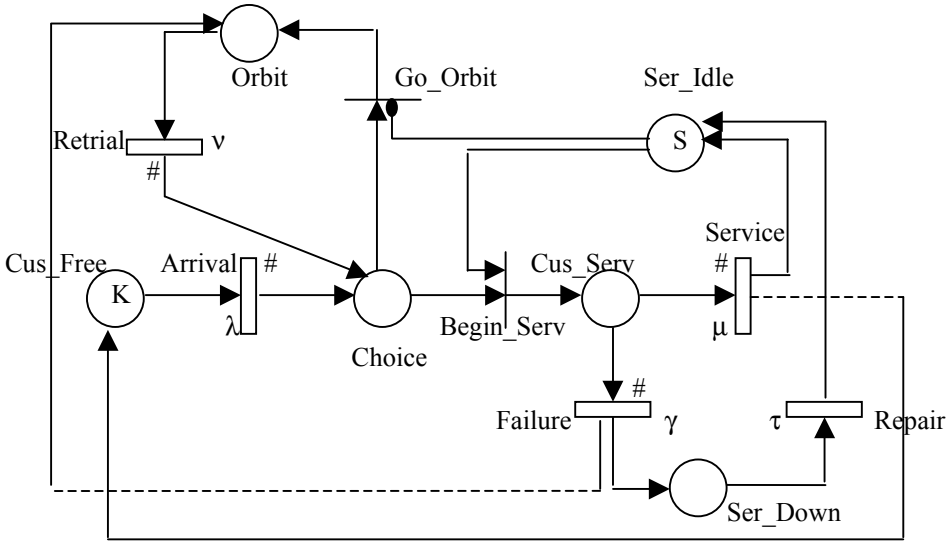

FIG. 3.3 – Le RdPSG modélisant les systèmes multi-serveurs avec rappel, pannes actives et sources intelligentes

sont opérationnels et oisifs et l'orbite est vide.

Dans ce modèle, à l'arrivée d'un client à la place Cus_Serv, ce qui représente son début de service, il y restera jusqu'à ce que l'une des deux transitions *Service* ou $Failure$ soit tirée. Comme le tir est déterminé par deux distributions exponentielles indépendantes, nous ne pouvons pas dire à un instant particulier, laquelle des deux transitions sera tirée en premier (application de la politique de course). Ceci coincide avec la réalité de la situation, car quand le serveur commence à servir un client, on ne sait pas s'il tombera en panne avant la fin du service ou pas.
Si le serveur tombe en panne avant qu'il ne finisse sa tâche, ce qui correspond au franchissement de la transition $Failure$ avant *Service*, le jeton qui se trouvait dans la place Cus_Serv et représentant le client en cours de service, sera consommé et deux jetons seront produits, dont l'un représente le client interrompu qui entre en orbite *Orbit*, après une remise à zéro du travail déjà réalisé par la transition *Service* avant la panne (service répété), et le second jeton représente le serveur en panne qui rejoint la place Ser_Down, où il sera réparé. D'autre part, la transition $Failure$ modélisant la panne des serveurs est reliée uniquement à la place Cus_Serv des serveurs en cours de traitement, ce qui représente convenablement la politique des pannes actives.

Par ailleurs, le franchissement des transitions *Arrival* et *Retrial* qui correspond à l'arrivée d'un appel primaire et répété resp., est indépendant de l'état des serveurs (opérationnels ou en panne), ce qui modélise bien la discipline des sources intelligentes.

Le franchissement de la transition *Repair* représente la fin du temps de réparation et le fait qu'un serveur réparé retourne à l'état opérationnel oisif i.e. à la place Ser_Idle.
Comme le réparateur répare un seul serveur à la fois, la sémantique de service de cette transition *Repair* est à serveur unique. Ceci signifie que le taux de franchissement est constant.

3.3.3 Modélisation des systèmes avec pannes dépendantes et sources intelligentes

Dans le modèle précédent, les serveurs peuvent tomber en panne en étant occupés seulement, ce qui est représenté par la transition $Failure$. À présent, nous considérons la politique des *pannes dépendantes* que nous avons introduite dans [194], où un serveur peut tomber en panne durant l'intervalle $(t, t + dt)$ avec une probabilité $\delta dt + o(dt)$ s'il est oisif, et une probabilité $\gamma dt + o(dt)$ s'il est occupé. Les taux de panne $\delta > 0$ et $\gamma > 0$ peuvent être égaux ou différents. Ceci signifie que la politique des pannes indépendantes où $\delta = \gamma > 0$ est considérée comme un cas particulier.

En fait, les modèles étudiés dans la littérature se limitent au cas de pannes actives ou pannes indépendantes ($\delta = \gamma > 0$), sans doute à cause de la difficulté que pourrait engendrer l'hypothèse $\delta \neq \gamma$, pour l'analyse du modèle de file d'attente avec rappel. Par contre, en utilisant le formalisme des RdPSG, ceci peut être modélisé et analysé facilement et ce pour des taux égaux ou différents. Pour cela, il suffit d'introduire deux transitions séparées $Fail\_Act$ pour les pannes actives et $Fail\_Idle$ pour les pannes oisives. Ainsi, le modèle obtenu serait général et valable pour tous les systèmes. Ceci grâce à la puissance d'expression du formalisme RdPSG.

Le modèle RdPSG correspondant à ces systèmes est donné dans la figure 3.4.

Dans ce modèle, la place Cus_Serv contient les clients en service (ou bien les serveurs occupés). Durant une période de service, un serveur peut tomber en panne. Ceci est représenté par le franchissement de la transition $Fail\_Act$. Dans ce cas, le client interrompu entre en orbite (place $Orbit$) et le serveur en panne rejoint la place Ser_Down où il sera réparé.

Sous la nouvelle discipline de panne proposée, un serveur peut aussi tomber en panne en étant oisif (dans la place Ser_Idle), ce qui correspond au franchissement de la transition $Fail\_Idle$. La sémantique du franchissement de cette transition est aussi à serveurs infinis, car plusieurs serveurs oisifs peuvent tomber en panne en même temps.

Les transitions $Fail\_Act$ et $Fail\_Idle$ représentant les pannes actives et les pannes oisives respectivement, peuvent avoir les même taux ou des taux différents (politiques de pannes indépendantes et dépendantes respectivement).

3.3.4 Modélisation des systèmes avec sources bloquées

Dans les figures 3.3 et 3.4, nous avons considéré des systèmes avec des sources intelligentes (ou non-bloquées). Ceci signifie que si tous les serveurs sont en panne, seul le service est interrompu et toutes les autres opérations continuent. À présent, nous considérons des systèmes avec sources bloquées, où toutes les opérations sont arrêtées (aucun appel n'est généré) sauf la réparation des serveurs.

Pour cela, il suffit d'ajouter aux modèles 3.3 et 3.4, un arc inhibiteur de multiplicité s, allant de la place Ser_Down à la transition $Arrival$ et un autre à la transition

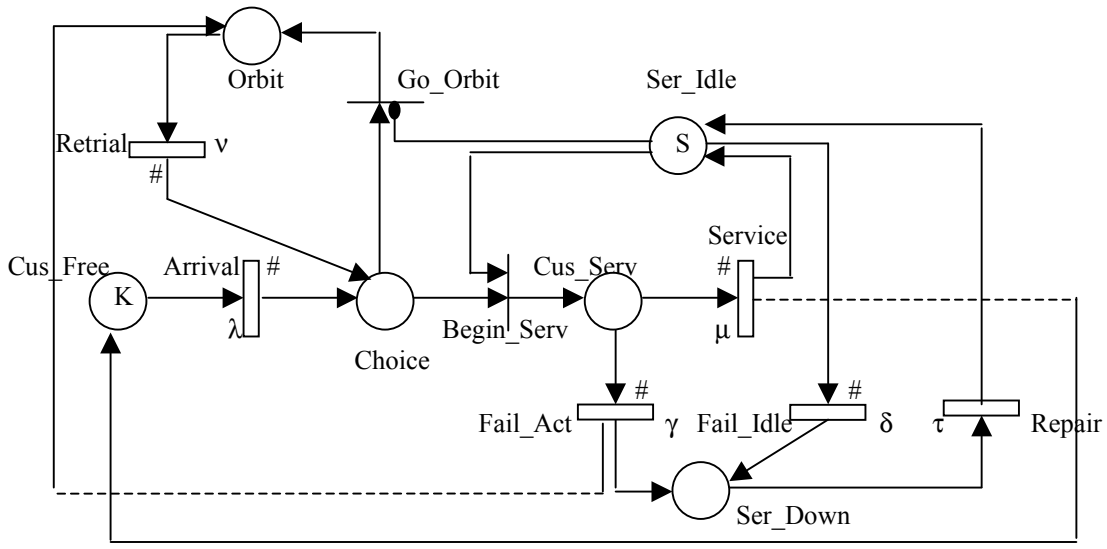

FIG. 3.4 – Le RdPSG modélisant les systèmes multi-serveurs avec rappel, pannes dépendantes et sources intelligentes

Retrial. Ainsi, les appels primaires et les appels répétés peuvent arriver au système seulement quand le marquage de la place Ser_Down est inférieur à s, i.e. quand le nombre de serveurs non opérationnels est inférieur à s. Autrement, si tous les serveurs sont en panne, aucune des deux transitions *Arrival* pour les appels primaires et *Retrial* pour les appels répétés n'est franchissable, et par conséquent, toutes les arrivées sont interrompues.

Remarque 1 : Dans tous les modèles avec pannes proposés, le service d'un client interrompu suite à une panne, est repris à zéro. Ainsi, on ne considère que le service répété. *En fait, la propriété sans mémoire rend inutile la considération des pannes à service continu avec des activités à loi exponentielle.* Ainsi, la considération des pannes à service continu n'a de sens qu'en cas de lois non-markoviennes avec mémoire, telle qu'une loi Cox ou Phase-Type.

Remarque 2 : Tous ces modèles de RdPSG peuvent être adaptés à des systèmes à population infinie. Pour cela, il suffit d'enlever la place Cus_Free et tous les arcs la reliant à d'autres places, et supposer que la sémantique de la transition *Arrival* est à serveur unique, et donc supprimer le symbole #.

3.3.5 Analyse des performances et de la fiabilité

Les RdPSG proposés pour la modélisation des systèmes avec les différentes politiques de panne sont bornés et le marquage initial est un état d'accueil. Par conséquent, ces modèles admettent une distribution stationnaire unique π.

En ayant ces probabilités d'état stationnaire, plusieurs paramètres de performance et indices de fiabilité intéressants peuvent être calculés en appliquant les formules explicites définies ci-dessous.

Pour éviter toute répétition, nous donnons dans ce qui suit uniquement les indices de performance et de fiabilité ayant rapport avec les phénomènes de panne et de réparation. Les paramètres définis dans la section précédente restent valables pour ces systèmes non-fiables.

- **Le nombre moyen de serveurs occupés (n_s) :**
  Il correspond au nombre moyen de jetons dans la place Cus_Serv qui est aussi le nombre moyen de clients en service.

$$n_s = \sum_{i:M_i \in A} M_i(Cus\_Serv).\pi_i$$

- **Le nombre moyen de serveurs opérationnels oisifs (n_d) :**
  Ceci représente le nombre moyen de jetons dans la place Ser_Idle.

$$n_d = \sum_{i:M_i \in A} M_i(Ser\_Idle).\pi_i$$

- **Le nombre moyen de serveurs en panne (n_f) :**
  Il correspond au nombre moyen de jetons dans la place Ser_Down.

$$n_f = \sum_{i:M_i \in A} M_i(Ser\_Down).\pi_i = s - (n_s + n_d)$$

- **La fréquence de panne des serveurs occupés ($\overline{\gamma}$)) :**
  Elle correspond à la fréquence (débit) de la transition $Failure$ (ou $Fail\_Act$) pour le cas de pannes actives et pannes dépendantes respectivement.

$$\overline{\gamma} = \begin{cases} \sum_{i:M_i \in A(Failure)} \gamma.M_i(Cus\_Serv).\pi_i, & \text{si pannes actives,} \\ \sum_{i:M_i \in A(Fail\_Act)} \gamma.M_i(Cus\_Serv).\pi_i, & \text{si pannes dépendantes.} \end{cases}$$

- **La fréquence de panne des serveurs oisifs ($\overline{\delta}$)) :**
  Elle correspond à la fréquence de la transition $Fail\_Idle$. Notons que ce paramètre n'a de sens qu'en cas de politique de pannes dépendantes (ou indépendantes).

$$\overline{\delta} = \sum_{i:M_i \in A(Fail\_Idle)} \delta.M_i(Ser\_Idle).\pi_i$$

- **Le taux moyen de réparation ($\overline{\tau}$) :**
  Il correspond à la fréquence de la transition $Repair$.

$$\overline{\tau} = \sum_{i:M_i \in A(Repair)} \tau.M_i(Ser\_Down).\pi_i$$

- **L'utilisation de c serveurs (U_c) : ($1 \leq c \leq s$)**
  Ceci correspond à la probabilité que c serveurs sont occupés :

$$U_c = \sum_{i:M_i(Cus\_Serv) \geq c} \pi_i$$

– **La disponibilité de c serveurs (A_c) : ($1 \leq c \leq s$)**
Elle correspond à la probabilité que c serveurs sont opérationnels et oisifs.

$$A_c = \sum_{i:M_i(Ser\_Idle)\geq c} \pi_i$$

– **La probabilité de panne de c serveurs (F_c) : ($1 \leq c \leq s$)**

$$F_c = \sum_{i:M_i(Ser\_Down)\geq c} \pi_i$$

– **L'utilisation du réparateur (U_r) :**
Ceci correspond à la probabilité de panne d'un serveur au moins.

$$U_r = F_1 = \sum_{i:M_i(Ser\_Down)\geq 1} \pi_i$$

– **Le temps d'attente moyen ($\overline{W}$) :**
Le temps d'attente moyen $\overline{W}$ des clients à l'état stationnaire, peut être calculé comme précédemment à l'aide de la formule de Little :

$$\overline{W} = n_o/\overline{\lambda}$$

– **Le temps de réponse moyen ($\overline{R}$) :**

$$\overline{R} = (n_o + n_s)/\overline{\lambda}$$

3.4 Conclusion

L'objet de ce chapitre a été la présentation d'une approche de modélisation et d'évaluation des performances et de la fiabilité des systèmes avec rappel, source finie et serveurs fiables ou non fiables, à l'aide du modèle de RdPSG.

Les RdPSG constituent un important modèle graphique et mathématique, qui offre une grande puissance descriptive, permettant ainsi la modélisation et l'analyse des systèmes avec rappel incluant diverses caractéristiques, telle que la source finie, la multiplicité des serveurs, les pannes et les réparations. La présence de ces différentes caractéristiques dans un système introduit de multiples problèmes de synchronisation en plus des phénomènes de blocage qui engendrent les rappels. Par opposition aux modèles de files d'attente, les RdPSG offrent la possibilité d'inclure facilement et de façon très élégante, les différentes structures de synchronisation, et de représenter très aisément les différents mécanismes de blocage. Par conséquent, la puissance d'expression du modèle des RdPSG, constitue une des principales raisons de son application pour les systèmes avec rappel.

D'autre part, la flexibilité de ce formalisme nous a permis une modification facile des modèles pour prendre en compte d'autres caractéristiques supplémentaires, et ce dans le but d'analyser des systèmes qui ont un sens et un intérêt pratique. C'est ainsi, que nous avons pu modéliser et analyser des systèmes avec différentes politiques de panne et introduire même une nouvelle discipline de panne plus générale que celles définies dans la

littérature.

D'un point de vue performance, contrairement aux modèles de files d'attente avec rappel, les RdPSG permettent une analyse des propriétés qualitatives du système modélisé et offrent aussi un riche moyen d'expression des indices de performance et de fiabilité en régime stationnaire. Ces indices s'expriment en fonction des éléments de base du RdP (places, transitions, marquages), ainsi que des probabilités stationnaires. Par conséquent, les RdPSG constituent un modèle intéressant permettant de faire une **analyse numérique exacte** des systèmes avec rappel et source finie.

Enfin, un autre avantage essentiel de cette approche est le fait qu'elle soit applicable pour beaucoup d'autres variantes de systèmes avec rappel. Prenons à titre d'exemple, les systèmes avec politique de rappel constant [198] dans lesquels le taux de rappel est indépendant du nombre de clients en orbite. Dans ce cas, les même modèles restent valables, avec une seule modification au niveau de la sémantique de service de la transition *Retrial* qui devient à serveur unique, et donc le taux de franchissement devient constant. Un autre exemple, est celui des systèmes avec arrivées par lots [5, 199], dans lesquels les arrivées primaires sont des lots de n unités chacun. Ceci peut se faire très simplement par RdPSG, avec un arc modélisant l'arrivée des clients primaires de multiplicité n.

Dans le chapitre suivant, nous nous intéressons à l'analyse des systèmes multi-classes avec rappel à l'aide du modèle des RdPSG colorés qui constituent une extension naturelle du modèle de RdPSG.

Chapitre 4

Analyse des Systèmes Multi-Classes avec Rappel à l'aide des RdPSG colorés

4.1 Introduction

La plupart des modèles de files d'attente avec rappel étudiés supposent que le flux d'entrée est homogène du point de vue des caractéristiques des clients, telles que les distributions des temps d'inter-arrivées, des temps de service et des temps de rappel. Cependant, en pratique, ces caractéristiques peuvent varier largement pour les différents types de clients. Ceci nous conduit aux **systèmes multi-classes avec rappel**. Ces systèmes apparaissent dans divers domaines pratiques, tels que les réseaux de télécommunications, les systèmes à commutation téléphonique et les réseaux mobiles cellulaires.

Ces systèmes multi-classes avec rappel sont habituellement analysés par la théorie des files d'attente. Cependant, les modèles multi-classes sont beaucoup plus difficiles à analyser que les modèles à classe unique. Ainsi, les résultats explicites sont disponibles seulement pour quelques cas particuliers avec des hypothèses contraignantes sur certains paramètres tels que le nombre de serveurs, le nombre de classes de clients, la taille de la population, etc.

Pour une synthèse complète sur les modèles de FAR avec deux classes de clients ($n = 2$), voir [36, 37, 38, 18]. D'autre part, les FAR avec n classes de clients ($n \geq 2$), ont été étudiées par quelques chercheurs tel que Falin [200, 4], Grishechkin [201] et Langaris [39]. Cependant, dans toutes ces références, les seuls modèles considérés sont des modèles mono-serveur à population infinie.

Par ailleurs, certains travaux récents ont porté sur des FAR avec un nombre fini de sources (ou clients) hétérogènes [40, 41, 42, 32, 43], dont chacune a des caractéristiques propres. Cependant, dans ces modèles multi-classes, chaque classe est limitée à un seul client (ou source) et la station de service comprend un serveur unique [40, 41, 42, 32] ou plusieurs serveurs identiques [43].

Les modèles de FAR avec serveurs hétérogènes sont très peu étudiés, malgré leur

importance dans les systèmes réels. D'ailleurs, on ne trouve dans la littérature que les quelques travaux à Pourbabai [44, 120] et à Sztrik [33, 34], dans lesquels les clients sont supposés être homogènes et les serveurs sont hétérogènes. Pour ce qui est des modèles avec rappel, clients et serveurs hétérogènes à la fois, aucune étude n'a été faite à ce jour, et ce ni pour le cas de serveurs fiables ni serveurs non-fiables. Ceci est dû essentiellement à la complexité du problème.

Dans ce chapitre, nous proposons une approche de modélisation et d'évaluation des performances et de la fiabilité des systèmes multi-classes avec rappel et source finie, à l'aide du modèle des réseaux de Petri stochastiques généralisés colorés (RdPSGC) [45, 46]. Nous considérons les systèmes avec plusieurs classes de clients et aussi plusieurs classes de serveurs. En effet, certains systèmes pratiques peuvent comprendre plusieurs types de serveurs de différentes générations avec des vitesses de traitement qui varient d'un type à un autre. Ainsi, ces serveurs au même titre que les clients, peuvent être regroupés en classes, de manière à ce que les serveurs ayant les mêmes paramètres feront parti de la même classe.

Les RdPSG permettent la description de la structure et du fonctionnement des systèmes avec rappel, ainsi que la vérification des propriétés qualitatives et quantitatives. Cependant, ces modèles sont trop abstraits, car ils ne distinguent ni le type de client ni le type de serveur dans un état donné. Une solution possible serait d'introduire un sous-réseau pour chaque type de clients, et d'interconnecter tous ces sous-réseaux à l'aide d'arcs et de transitions. De la même manière, on pourrait introduire des sous-réseaux pour chaque type de serveurs. Cependant, le réseau global deviendrait illisible et impraticable dès que le nombre de classes de clients ou le nombre de classes de serveurs augmente. De plus, il faudrait une nouvelle version du modèle, chaque fois que l'on modifie un de ces nombres. Par conséquent, le problème qui pourrait apparaître en appliquant le formalisme des RdPSG pour les systèmes multi-classes avec rappel, est celui de la complexité des modèles et aussi la complexité du processus de modélisation lui-même. Ainsi, pour remédier à ce problème, nous proposons l'utilisation du formalisme des RdPSG colorés.

L'avantage de l'utilisation des RdPSGC est le fait qu'ils constituent un modèle mathématique de haut-niveau, approprié pour la description et l'analyse des performances des systèmes avec des composants hétérogènes, et qui incluent des mécanismes de synchronisation et de concurrence. D'autre part, ce modèle coloré permet essentiellement l'intégration des contraintes provenant de l'identification et de la particularisation éventuelle (priorité) des différents clients ou serveurs. Ainsi, nous montrons la simplicité avec laquelle ces systèmes multi-classes assez compliqués peuvent être décrits et analysés grâce à ces modèles stochastiques colorés, dans lesquels chaque jeton correspondant à un client ou à un serveur doit porter une information qui dénote son type.

En plus du fait que le formalisme des RdPSG colorés, permet de construire des représentations concises et détaillées de systèmes complexes, il permet aussi l'analyse des propriétés qualitatives ainsi que l'obtention de résultats de performance et de fiabilité exacts, pour des systèmes multi-classes avec rappel, différentes disciplines de service et éventuellement différentes politiques de panne dans le cas où les serveurs ne sont pas fiables.

Dans ce chapitre, nous présentons notre approche de modélisation et d'analyse des systèmes multi-classes avec rappel. En premier lieu, nous définissons les modèles de RdPSG colorés décrivant ces systèmes, et ce pour les deux politiques de service : service aléatoire et service le plus rapide. Nous donnons ensuite, les formules explicites permettant l'obtention des paramètres de performance exacts. Dans la section qui suit, cette étude sera étendue au cas de serveurs non-fiables. Enfin, nous terminons le chapitre par une conclusion.

4.2 Les systèmes multi-classes avec rappel et serveurs fiables

Un système avec rappel peut être parcouru par différentes classes de clients qui se distinguent par : des processus d'arrivée différents, des temps de service différents et/ou des processus de rappel différents. D'autre part, les serveurs peuvent avoir des vitesses de traitement différentes. Ainsi, ces serveurs au même titre que les clients, peuvent être regroupés en classes, de manière à ce que les serveurs ayant les même paramètres feront parti de la même classe. Ainsi, pour définir un système multi-classes avec rappel, il faut définir pour chaque classe de clients et pour chaque classe de serveurs, les distributions associées.

Nous nous intéressons dans cette section, particulièrement aux systèmes multi-classes avec rappel et serveurs fiables.

4.2.1 Description mathématique

Nous considérons des systèmes avec rappel et une source (population) finie composée de n classes ($n \geq 1$) de clients potentiels. Chaque classe i ($1 \leq i \leq n$) contient k_i clients homogènes, ayant les mêmes distributions des temps d'inter-arrivée, des temps de service et des temps de rappel. Chaque client est soit libre, en service ou en orbite à tout instant.

Les appels primaires des clients libres de la classe i arrivent au système suivant un flux quasi-aléatoire de taux λ_i. Ainsi, tout client de la classe i libre à l'instant t, peut générer un appel primaire pour le service dans tout intervalle $(t, t+dt)$ avec la probabilité $\lambda_i(k_i - x_i)dt + o(dt)$ où k_i est la taille de la population de la classe i et x_i est le nombre de clients de la classe i dans le système.

La station de service comprend m classes de serveurs ($m \geq 1$). Chaque classe j ($1 \leq j \leq m$) contient s_j serveurs identiques. Chaque serveur peut être à l'état oisif ou occupé. Si un des serveurs est oisif au moment de l'arrivée d'un appel de client, le service commence immédiatement.

Nous considérons deux disciplines de service différentes :
- Le service aléatoire, où les requêtes sont adressées aux serveurs oisifs de toutes les classes d'une manière aléatoire ;
- Le service le plus rapide, où les requêtes sont adressées à la classe des serveurs oisifs les plus rapides, i.e. que la disponibilité des serveurs est toujours vérifiée suivant

un ordre croissant des indices des classes des serveurs. Ainsi, les serveurs les plus rapides (les plus prioritaires) sont ceux des premières classes.

Les temps de service des clients de la classe i sont indépendants et distribués exponentiellement avec un taux $\mu_{i,j}$ si un serveur de la classe j est utilisé. Par conséquent, le taux de service dépend à la fois du type de client en service et aussi du type de serveur utilisé.

Si tous les serveurs sont occupés au moment de l'arrivée d'un appel de la classe i, le client rejoint l'orbite et commence à générer un flux d'appels répétés distribué exponentiellement avec un taux ν_i jusqu'à ce qu'il trouve un serveur disponible.

Comme d'habitude, nous supposons que les temps d'inter-arrivée, les temps de service et les temps de rappel sont mutuellement indépendants.

Nous présentons dans ce qui suit, une méthode de modélisation et d'analyse des systèmes multi-classes avec rappel, à l'aide des réseaux de Petri stochastiques généralisés colorés (RdPSGC), et ce pour les deux disciplines de service.

4.2.2 Discipline de service aléatoire

Le modèle de RdPSG coloré décrivant les systèmes multi-classes avec rappel étudiés est donné dans la figure 4.1.

Les réseaux colorés ont pour but de distinguer les marques qui modélisent différents objets ayant les mêmes types d'états et sujets aux mêmes types d'actions. Ainsi, le réseau coloré correspondant distingue les différentes classes de clients et classes de serveurs en les identifiant chacune par une couleur. Un second avantage de l'introduction de la couleur est qu'elle nous permet facilement l'obtention d'un modèle paramétré général, dans lequel le nombre des classes de client et de serveur sont des variables.

Une étape importante dans la définition d'un RdPSG coloré est l'identification des domaines de couleur des places et des transitions. Dans les systèmes étudiés, les clients sont divisés en n classes ($n \geq 1$) et les serveurs en m classes ($m \geq 1$). Ceci peut être facilement modélisé par un domaine de couleur des clients C et un domaine de couleur des serveurs R, où :
$C = C(Cus\_Free) = C(Orbit) = C(Choice) = \{c_1, c_2, ..., c_n\}$;
$R = C(Ser\_Idle) = \{r_1, r_2, ..., r_m\}$;
$C(Cus\_Serv) = C \otimes R$.

Le domaine de couleur de chaque place ou transition peut comprendre des couleurs de base, comme il peut être un produit cartésien d'autres domaines de couleur. Par exemple, les jetons de la place Cus_Serv sont des tuples de type $< c, r >$, ce qui représente l'affectation d'un serveur de couleur r à un client de couleur c. Ainsi, le domaine de couleur de cette place est $C \otimes R$.

Le marquage initial du réseau est donné par :

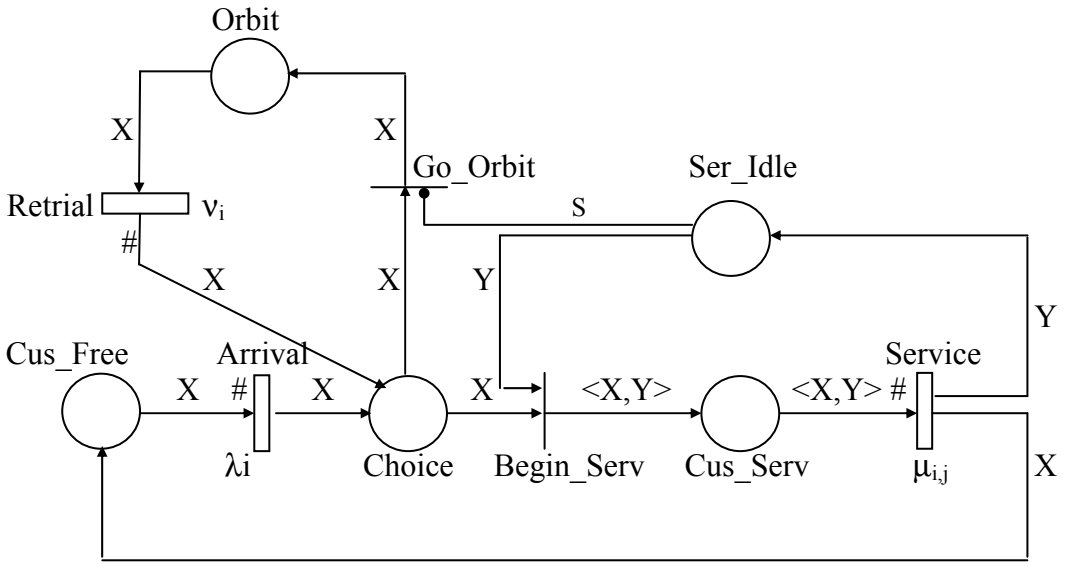

FIG. 4.1 – Le RdPSGC modélisant les systèmes multi-classes avec rappel et serveurs fiables

- $M_0(Cus\_Free) = k_1.c_1 + k_2.c_2 + ... + k_n.c_n$
- $M_0(Ser\_Idle) = s_1.r_1 + s_2.r_2 + ... + s_m.r_m$
- $M_0(Choice) = M_0(Orbit) = M_0(Cus\_Serv) = 0$

Ce marquage initial représente le fait que tous les clients (de toute classe) sont initialement libres, tous les serveurs (de tout type) sont oisifs et l'orbite est vide.

Les fonctions de couleur étiquetant les arcs qui relient les places aux transitions (et les transitions aux places), sont définies comme suit :

- La fonction projection X permet de sélectionner une couleur donnée (un client) du domaine C : $\forall c \in C$, $\forall r \in R$, $X(c) = c$ et $X(c, r) = c$;
  Par exemple on a : $\forall c \in C, Pré(Cus\_Free, Arrival)(c) = c$; et $\forall c \in C, \forall r \in R, Post(Cus\_Free, Service)(c, r) = c$.
- La fonction projection Y permet de sélectionner une couleur donnée (un serveur) du domaine R : $\forall c \in C$, $\forall r \in R$, $Y(r) = r$ et $Y(c, r) = r$;
- La fonction synchronisation S permet de synchroniser toutes les couleurs de la classe R. Autrement dit, elle prend un élément de chaque couleur du domaine R à la fois. Dans ce cas, on a :
  $\forall r \in R, Pré(Ser\_Idle, Go\_Orbit)(r) = \{r_1, ..., r_m\} = r_1 + ... + r_m$.

Les taux des transitions sont paramétrés et dépendant de la couleur de la marque qui franchit la transition. Ce mécanisme nous permet de modéliser des systèmes avec des taux d'arrivée et des taux de rappel dépendant du type de client et aussi des temps de service différents qui dépendent aussi bien du type de client que du type de serveur.

Le franchissement de la transition *Arrival* indique l'arrivée d'un appel primaire. Sa sémantique de service est une *sémantique à serveurs infinis*, car tous les clients libres

peuvent générer des appels primaires. Le taux de franchissement λ_i de cette transition dépend de la couleur du client qui génère l'appel. Dès l'arrivée d'une requête, si la place Ser_Idle contient au moins un serveur oisif d'une classe quelconque, la transition immédiate $Begin\_Serv$ sera tirée. Ainsi, le client commence son service et le serveur passe à l'état occupé. Cependant, si un client de couleur c_i ne trouve aucun serveur oisif dans toutes les classes, la transition immédiate Go_Orbit sera franchie et le client rejoint la place $Orbit$, à partir de laquelle, il va générer un flux d'appels répétés distribués exponentiellement avec un taux ν_i.
Le franchissement de la transition $Retrial$ représente l'arrivée d'un appel répété de l'orbite. Comme les clients de l'orbite peuvent générer des appels répétés pour le service, indépendamment les uns des autres, cette transition doit avoir une sémantique à serveurs infinis.

Quand la transition temporisée $Service$ est franchie, le client en service retourne à l'état libre (à la place Cus_Free) et le serveur devient oisif et prêt à servir un autre client (place Ser_Idle). La sémantique de service de la transition $Service$ est une sémantique à serveurs infinis, car plusieurs serveurs peuvent servir simultanément. De plus, le taux $\mu_{i,j}$ de cette transition dépend de la couleur du client servi et aussi de celle du serveur utilisé.

4.2.3 Discipline du service le plus rapide

Le modèle de la figure 4.1 correspond à la discipline de service aléatoire, dans laquelle les requêtes sont adressées d'une manière aléatoire, aux serveurs oisifs de toutes les couleurs.

Une caractéristique importante des modèles de RdPS colorés est que la politique de service peut être facilement modifiée. En effet, pour modéliser la politique du service le plus rapide, il suffit de définir des priorités entre les couleurs des serveurs. Notons qu'une telle variation dans le modèle des files d'attente avec rappel nécessite des manipulations et des modifications beaucoup plus complexes, particulièrement au niveau de l'analyse.

Dans le cas de la discipline du service le plus rapide, un ordre de priorité doit être défini sur les couleurs du domaine des serveurs R, de façon à ce que l'oisiveté des serveurs soit toujours vérifiée suivant un ordre croissant des indices des couleurs des serveurs. Ainsi, à l'arrivée d'un appel primaire ou répété, s'il trouve un serveur oisif de couleur r_1, le service peut commencer. Autrement, si tous les serveurs de couleur r_1 sont occupés, le client vérifie les serveurs de couleur r_2, et ainsi de suite. Cependant, si les serveurs de toutes les couleurs r_j $(\forall j = \overline{1..m})$ sont occupés, le client rejoint la place $Orbit$.

Pour cela, nous supposons que dans le domaine de couleur des serveurs $R = \{r_1, r_2, ..., r_m\}$, les serveurs de couleur r_j sont plus rapides que ceux de couleur r_q, $\forall j < q$. Ainsi, nous devons compléter le modèle de la figure 4.1 par une fonction de priorité que nous définissons comme suit :

$$\forall (c_i, r_j) \in C(Begin\_Serv), \pi(c_i, r_j) > \pi(c_i, r_q), \forall j < q.$$

4.2.4 Analyse qualitative

Un des principaux avantages du formalisme des RdPSG colorés, est qu'il permet aussi bien l'analyse qualitative que quantitative. En effet, une fois le modèle obtenu, nous pouvons vérifier ses propriétés qualitatives, puis déduire son ergodicité avant d'entamer l'évaluation de ses performances.

L'étude des modèles de RdPSGC proposés pour la description des systèmes multi-classes avec rappel et source finie, nous a permis de déduire qu'ils sont bornés et que le marquage initial est un état d'accueil. Ils sont par conséquent ergodiques, et donc la solution stationnaire existe et est unique, et ce pour la discipline de service aléatoire ainsi que la discipline du service le plus rapide.

4.2.5 Analyse stochastique

Le graphe d'accessibilité correspondant à ces RdPSG colorés est isomorphe à un processus semi-markovien. Ainsi, la solution de ce processus à l'état stationnaire est le vecteur de la distribution de probabilité $\pi = (\pi_1, \pi_2, ..., \pi_n)$, qui peut être calculé en appliquant l'algorithme de résolution des chaînes de Markov incluses.

En ayant la distribution des probabilités d'état stationnaire π, plusieurs paramètres de performance exacts des systèmes multi-classes avec rappel peuvent être obtenus en appliquant les formules données ci-dessous. Nous notons par π_j la probabilité que le processus se trouve dans l'état M_j à l'équilibre, $M_j(p)(c)$ le nombre de jetons de couleur c dans la place p dans le marquage M_j, A l'ensemble des marquages tangibles accessibles et par $A(t, c)$ l'ensemble des marquages tangibles accessibles dans lesquels la transition t est franchissable par rapport à la couleur c.

- **Le nombre moyen de clients libres de type i ($n_F^{(i)}$) :**
  Ceci correspond au nombre moyen de jetons de couleur c_i dans la place Cus_Free.

$$n_F^{(i)} = \sum_{j:M_j \in A} M_j(Cus\_Free)(c_i).\pi_j$$

  Ainsi, le nombre moyen de clients libres n_F est donné par :

$$n_F = \sum_{i=1}^{n} n_F^{(i)}$$

- **Le nombre moyen de clients de type i en orbite ($n_O^{(i)}$) :**

$$n_O^{(i)} = \sum_{j:M_j \in A} M_j(Orbit)(c_i).\pi_j$$

  Ainsi, le nombre moyen de clients en orbite n_O est donné par :

$$n_O = \sum_{i=1}^{n} n_O^{(i)}$$

– **Le nombre moyen de clients de type i en service ($n_S^{(i)}$) :**
Ceci correspond au nombre moyen de tuples $< c_i, r_q >$ dans la place Cus_Serv, où r_q est une couleur de serveur.

$$\begin{aligned} n_S^{(i)} &= \sum_{j:M_j \in A} \sum_{q=1}^{m} M_j(Cus\_Serv)(< c_i, r_q >).\pi_j \\ &= k_i - (n_F^{(i)} + n_O^{(i)}) \end{aligned}$$

où : $M_j(Cus\_Serv)(< c_i, r_q >)$ est le nombre de tuples correspondant aux clients de couleur c_i servis par des serveurs de couleur r_q dans la place Cus_Serv dans le marquage M_j.
Ainsi, le nombre moyen de clients en service n_S est donné par :

$$n_S = \sum_{i=1}^{n} n_S^{(i)} = K - (n_F + n_O)$$

où : $K = \sum_{i=1}^{n} k_i$ est la taille totale de la population.

– **Le nombre moyen de clients de type i dans le système ($n^{(i)}$) :**
Il correspond au nombre moyen de clients de couleur c_i en service ou en orbite.

$$n_{sys}^{(i)} = n_S^{(i)} + n_O^{(i)} = k_i - n_F^{(i)}$$

Ainsi, le nombre moyen de clients dans le système n_{sys} est donné par :

$$n_{sys} = \sum_{i=1}^{n} n_{sys}^{(i)} = K - n_F$$

– **Le nombre moyen de serveurs oisifs de type q ($n_I^{(q)}$) :**
Il correspond au nombre moyen de jetons de couleur r_q dans la place Ser_Idle.

$$n_I^{(q)} = \sum_{j:M_j \in A} M_j(Ser\_Idle)(r_q).\pi_j$$

Ainsi, le nombre moyen de serveurs oisifs n_I est donné par :

$$n_I = \sum_{q=1}^{m} n_I^{(q)}$$

– **Le nombre moyen de serveurs occupés de type q ($n_B^{(q)}$) :**
Ceci correspond au nombre moyen de tuples de couleur $< c_i, r_q >$ dans la place Cus_Serv.

$$\begin{aligned} n_B^{(q)} &= \sum_{j:M_j \in A} \sum_{i=1}^{n} M_j(Cus\_Serv)(< c_i, r_q >).\pi_j \\ &= s_q - n_I^{(q)} \end{aligned}$$

où : s_q est le nombre de serveurs de type q.

Ainsi, le nombre moyen de serveurs occupés n_B est donné par :

$$n_B = \sum_{q=1}^{m} n_B^{(q)} = V - n_I$$

où : $V = \sum_{i=1}^{m} s_i$ est le nombre total de serveurs.

– **L'utilisation des serveurs de type q par les clients de type i ($U^{(i,q)}$) :**
Ceci correspond au nombre moyen de jetons de couleur $< c_i, r_q >$ dans la place Cus_Serv.

$$U^{(i,q)} = \sum_{j:M_j \in A} M_j(Cus\_Serv)(< c_i, r_q >).\pi_j$$

– **Le taux moyen de génération des appels primaires des clients de type i ($\overline{\lambda}_i$) :** Ceci représente la fréquence de la transition $Arrival$ par rapport à la couleur c_i.

$$\overline{\lambda}_i = \sum_{j:M_j \in A(Arrival,c_i)} \lambda_i.M_j(Cus\_Free)(c_i).\pi_j$$

Ainsi, le taux moyen de génération des appels primaires $\overline{\lambda}$ est donné par :

$$\overline{\lambda} = \sum_{i=1}^{n} \overline{\lambda}_i$$

– **Le taux moyen de génération des appels répétés des clients de type i ($\overline{\nu}_i$) :**
Il correspond à la fréquence de la transition $Retrial$ par rapport à la couleur c_i.

$$\overline{\nu}_i = \sum_{j:M_j \in A(Retrial,c_i)} \nu_i.M_j(Orbit)(c_i).\pi_j$$

Ainsi, le taux moyen de génération des appels répétés $\overline{\nu}$ est donné par :

$$\overline{\nu} = \sum_{i=1}^{n} \overline{\nu}_i$$

– **Le taux moyen de service des clients de type i par les serveurs de type q ($\overline{\mu}_{i,q}$) :** Ceci représente la fréquence de la transition $Service$ par rapport aux clients de couleur c_i et aux serveurs de couleur r_q.

$$\overline{\mu}_{i,q} = \sum_{j:M_j \in A(Service,<c_i,r_q>)} \mu_{i,q}.M_j(Cus\_Serv)(< c_i, s_q >).\pi_j$$

– **Le taux moyen de service des clients de type i ($\overline{\mu}_i$) :**
Il représente la fréquence de la transition $Service$ par rapport aux clients de couleur c_i, pour tout type de serveurs.

$$\overline{\mu}_i = \sum_{q=1}^{m} \overline{\mu}_{i,q}$$

– **Le taux moyen de service des serveurs de type q ($\overline{\mu}_q$) :**
Il représente la fréquence de la transition *Service* par rapport aux serveurs de couleur r_q, pour tout type de client.

$$\overline{\mu}_q = \sum_{i=1}^{n} \overline{\mu}_{i,q}$$

– **La disponibilité de s serveurs de type q ($A_s^{(q)}$) : ($1 \leq s \leq s_q$)**
Ceci correspond à la probabilité que s serveurs de couleur r_q sont oisifs.

$$A_s^{(q)} = \sum_{j:M_j(Ser\_Idle)(r_q)\geq s} \pi_j$$

– **La probabilité qu'un serveur au moins de couleur r_q soit utilisé par un client de couleur c_i ($U^{(q,i)}$) :**

$$U^{(q,i)} = \sum_{j:M_j(Cus\_Serv)(<c_i,r_q>)\geq 1} \pi_j$$

Ainsi, la probabilité que des serveurs de couleur r_q soient utilisés est donnée par :

$$U^{(q)} = \sum_{i=1}^{n} U^{(q,i)} = 1 - A_{s_q}^{(q)}$$

où : $A_{s_q}^{(q)}$ représente la disponibilité de tous les serveurs (s_q) de couleur r_q.

– **La probabilité que s serveurs au moins de couleur r_q sont en service ($U_s^{(q)}$) :**
($1 \leq s \leq s_q$)

$$\begin{aligned} U_s^{(q)} &= 1 - A_{s_q - s}^{(q)} \\ &= \sum_{i=1}^{n} \sum_{j:M_j(Cus\_Serv)(<c_i,r_q>)\geq s)} \pi_j \end{aligned}$$

Ainsi, l'utilisation de tous les serveurs de couleur r_q est donnée par :

$$U_{s_q}^{(q)} = 1 - A_0^{(q)}$$

– **Le temps d'attente moyen d'un client de type i ($\overline{W}_i$) :**
Le temps d'attente moyen $\overline{W}_i$ d'un client de couleur c_i à l'état stationnaire, peut être facilement obtenu à l'aide de la formule de Little :

$$\overline{W}_i = \frac{n_O^{(i)}}{\overline{\lambda}_i}$$

– **Le temps moyen de service d'un client de couleur c_i par un serveur de couleur r_q ($\overline{S}_{i,q}$) :**

$$\overline{S}_{i,q} = \frac{1}{\mu_{i,q}}$$

– **Le temps moyen de réponse d'un client de couleur c_i servi par un serveur de couleur r_q ($\overline{R}_{i,q}$) :**

$$\overline{R}_{i,q} = \overline{W}_i + \overline{S}_{i,q} = \frac{n_O^{(i)}}{\overline{\lambda}_i} + \frac{1}{\mu_{i,q}} = \frac{n^{(i)}}{\overline{\lambda}_i}$$

– **La probabilité de blocage d'un client primaire de couleur c_i ($B_p^{(i)}$) :**

$$B_p^{(i)} = \frac{\sum_{j:M_j \in A} \sum_{k=1}^{k_i} k.\lambda_i.Prob[M_j(Cus\_Free)(c_i) = k \& M_j(Ser\_Idle) = 0]}{\overline{\lambda}_i}$$

Ainsi, la probabilité de blocage des clients primaires :

$$B_p = \sum_{i=1}^{n} \frac{\overline{\lambda}_i}{\overline{\lambda}} B_p^{(i)}$$

– **La probabilité de blocage d'un client de couleur c_i en orbite ($B_r^{(i)}$) :**

$$B_r^{(i)} = \frac{\sum_{j:M_j \in A} \sum_{k=1}^{k_i} k.\nu_i.Prob[M_j(Orbit)(c_i) = k \& M_j(Ser\_Idle) = 0]}{\overline{\nu}_i}$$

Ainsi, la probabilité de blocage des clients de l'orbite :

$$B_r = \sum_{i=1}^{n} \frac{\overline{\nu}_i}{\overline{\nu}} B_r^{(i)}$$

– **La probabilité de blocage d'un client de couleur c_i ($B^{(i)}$) :**

$$B^{(i)} = B_p^{(i)} + B_r^{(i)}$$

Ainsi, la probabilité de blocage B des clients est donnée par :

$$B = \sum_{i=1}^{n} B^{(i)}$$

4.3 Les systèmes multi-classes avec rappel et serveurs non-fiables

Notre recherche bibliographique nous a permis de constater que les modèles de FAR à source finie, qui prennent en compte l'hétérogénéité des clients, la non-fiabilité et l'hétérogénéité des serveurs ne sont pas étudiés au jour d'aujourd'hui. Ainsi, nous proposons dans cette section la modélisation et l'analyse des performances et de la fiabilité des systèmes multi-classes avec rappel, source finie et serveurs non-fiables, à l'aide du modèle des RdPSG colorés.

En fait, nous considérons une extension des systèmes étudiés dans la section précédente, dans le sens où la source de clients est composée de $n \geq 1$ classes de clients et la station de service comprend $m \geq 1$ classes de serveurs. Cependant, dans ces systèmes

non-fiables, tout serveur de la classe j $(1 \leq j \leq m)$ peut tomber en panne durant l'intervalle $(t, t+dt)$ avec la probabilité $\delta_j dt + o(t)$ s'il est oisif, et avec la probabilité $\gamma_j d(t) + o(t)$ s'il est occupé. Le temps de réparation des serveurs de cette classe est distribué exponentiellement avec le taux τ_j.

Les appels primaires des clients libres de la classe i $(1 \leq i \leq n)$ arrivent au système suivant un flux quasi-aléatoire de taux λ_i. Si un des serveurs est oisif et opérationnel au moment de l'arrivée d'un appel, le service commence immédiatement. Le taux de service $\mu_{i,j}$ dépend à la fois du type de client i en service, et aussi du type de serveur j utilisé. Pour ce qui est du choix du serveur, les deux disciplines de service : service aléatoire et service le plus rapide peuvent être appliquées.

Par contre, si tous les serveurs de toutes les classes sont occupés ou en panne, au moment de l'arrivée d'un appel d'un client de type i, celui-ci rejoint l'orbite pour rappeler suivant un processus exponentiel de taux ν_i jusqu'à ce qu'il trouve un serveur oisif et opérationnel.

Toutes les variables aléatoires sont supposées être distribuées exponentiellement et indépendantes les unes des autres.

Nous présentons dans ce qui suit, une méthode de modélisation et d'analyse des systèmes multi-classes avec rappel et serveurs non-fiables, à l'aide des réseaux de Petri stochastiques généralisés colorés (RdPSGC), et ce pour les différentes politiques de panne et disciplines de service.

4.3.1 Modélisation des systèmes multi-classes avec pannes actives et sources intelligentes

Nous considérons d'abord les systèmes avec discipline de pannes actives où $\delta_j = 0$ et $\gamma_j > 0$ $\forall j$ $(1 \leq j \leq m)$. D'autre part, les sources sont supposées intelligentes. Ainsi, les clients libres et les clients de l'orbite continuent à générer les appels primaires et les appels répétés en cas de panne général, où tous les serveurs de toutes les classes sont non opérationnels.

Le modèle de RdPSG coloré décrivant les systèmes multi-classes avec rappel, pannes actives et sources intelligentes, est donné dans la figure 4.2.

Une étape importante dans la définition d'un RdPSG coloré est l'identification des domaines de couleur des places et des transitions, ainsi que les fonctions de couleur.

Dans ce modèle, nous avons deux domaines de couleur de base : le domaine de couleur des clients C et le domaine de couleur des serveurs R, où :
$C = C(Cus\_Free) = C(Orbit) = C(Choice) = \{c_1, c_2, ..., c_n\}$;
$R = C(Ser\_Idle) = C(Ser\_Down) = \{r_1, r_2, ..., r_m\}$;
$C(Cus\_Serv) = C \otimes R$.

Le marquage initial du réseau est donné par :

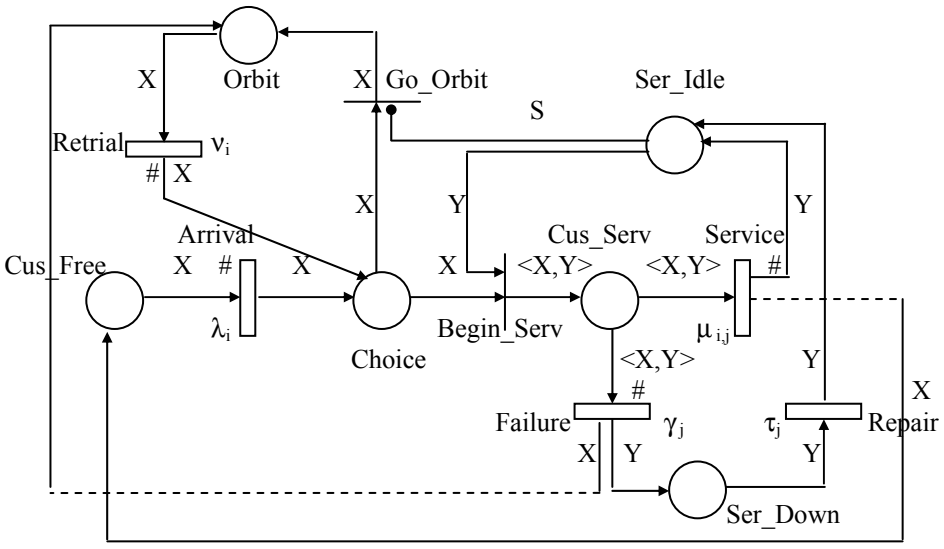

FIG. 4.2 – Le RdPSGC modélisant les systèmes multi-classes avec rappel et pannes actives

- $M_0(Cus\_Free) = k_1.c_1 + k_2.c_2 + ... + k_n.c_n$
- $M_0(Ser\_Idle) = s_1.r_1 + s_2.r_2 + ... + s_m.r_m$
- $M_0(Choice) = M_0(Orbit) = M_0(Cus\_Serv) = M_0(Ser\_Down) = 0$

Ce marquage initial représente le fait que tous les clients (de toute classe) sont initialement libres, tous les serveurs (de tout type) sont oisifs et opérationnels et l'orbite est vide.

Les fonctions de couleur qui étiquettent les arcs, sont les fonctions de projection X et Y et la fonction de synchronisation S définies dans la section précédente.

Dans le modèle de la figure 4.2, un serveur ne peut tomber en panne qu'en étant occupé. Ceci est représenté par le franchissement de la transition $Failure$ dont le taux (de panne) γ_j dépend du type de serveur utilisé. Suite à la panne, le client en cours de service rejoint la place $Orbit$ avec une remise à zéro du travail réalisé par la transition $Service$ avant la panne, et le serveur en panne rejoint la place Ser_Down, où il sera réparé avec un taux τ_j.

4.3.2 Modélisation des systèmes avec pannes dépendantes et sources intelligentes

Dans cette section, nous modélisons les systèmes multi-classes avec rappel et politique des *pannes dépendantes*, que nous avons introduite dans [194], pour les systèmes mono-classe avec rappel. Cette nouvelle discipline de panne stipule que tout serveur peut tomber en panne en étant occupé ou oisif, et avec des taux de panne quelconques.

Grâce à la puissance d'expression du modèle des RdPSG colorés, les systèmes multi-classes avec pannes dépendantes peuvent être facilement modélisés. Il suffit, pour cela, de remplacer la transition $Failure$ des pannes actives par deux transitions séparées $Fail\_Act$

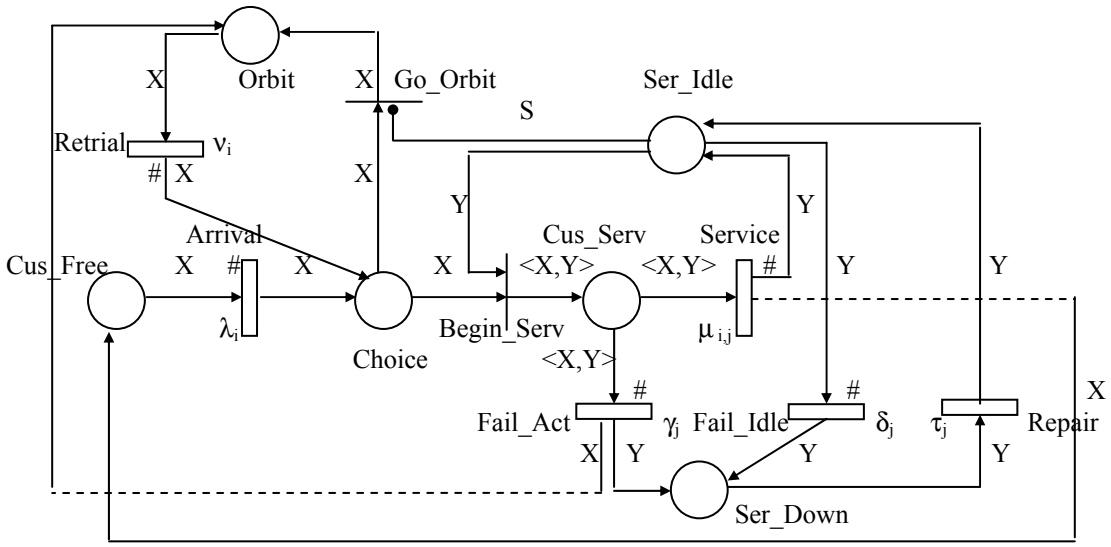

FIG. 4.3 – Le RdPSGC modélisant les systèmes multi-classes avec rappel et pannes dépendantes

pour les pannes actives et $Fail\_Idle$ pour les pannes oisives. Ainsi, le modèle obtenu serait général et valable pour tous les systèmes. Le modèle RdPSG coloré correspondant à ces systèmes est donné dans la figure 4.3.

4.3.3 Modélisation des systèmes multi-classes avec pannes et sources bloquées

Dans les figures 4.2 et 4.3, les modèles correspondent aux systèmes avec sources intelligentes, dans lesquels en cas de panne général de toute la station de service, les clients continuent à générer des appels.

La modélisation des systèmes avec sources bloquées peut se faire d'une façon très simple, en ajoutant à ces modèles un arc inhibiteur de multiplicité V, allant de la place Ser_Down à la transition $Arrival$ et un autre à la transition $Retrial$, où $V = \sum_{i=1}^{m} s_i$. Ainsi, les transitions $Arrival$ et $Retrial$ peuvent être franchies seulement quand le marquage de la place Ser_Down est inférieur à la capacité totale V de la station de service. Autrement, si tous les serveurs de toutes les classes sont en panne (dans $M(Ser\_Down) = V$), toutes les arrivées sont interrompues.

Remarque :
Les systèmes multi-classes non-fiables avec discipline du service le plus rapide, peuvent être modélisés de la même manière que les systèmes avec service aléatoire donnés précédemment, en ajoutant tout simplement une fonction de priorité, la même que celle du modèle fiable, définie comme suit :

$$\forall (c_i, r_j) \in C(Begin\_Serv), \pi(c_i, r_j) > \pi(c_i, r_q), \forall j < q.$$

4.3.4 Analyse des performances et de la fiabilité

Les différents modèles de réseaux colorés proposés pour la description des systèmes multi-classes avec les différentes politiques de panne sont bornés et le marquage initial est un état d'accueil. Par conséquent, ces modèles sont ergodiques, et donc admettent une solution stationnaire unique π.

En ayant les probabilités d'état stationnaire, plusieurs paramètres de performance et indices de fiabilité exacts peuvent être calculés en appliquant les formules explicites données ci-dessous.

Pour éviter toute répétition, nous donnons dans ce qui suit uniquement les indices de performance et de fiabilité ayant rapport avec les phénomènes de panne et de réparation. En fait, les paramètres définis dans la section précédente restent valables pour ces systèmes non-fiables.

– **Le nombre moyen de serveurs occupés de type q ($n_B^{(q)}$) :**
Ceci correspond au nombre moyen de tuples de couleur $< c_i, r_q >$ dans la place Cus_Serv.

$$n_B^{(q)} = \sum_{j:M_j \in A} \sum_{i=1}^{n} M_j(Cus\_Serv)(< c_i, r_q >).\pi_j$$

Ainsi, le nombre moyen de serveurs occupés n_B est donné par :

$$n_B = \sum_{q=1}^{m} n_B^{(q)}$$

– **Le nombre moyen de serveurs opérationnels oisifs de type q ($n_I^{(q)}$) :**
Il correspond au nombre moyen de jetons de couleur r_q dans la place Ser_Idle.

$$n_I^{(q)} = \sum_{j:M_j \in A} M_j(Ser\_Idle)(r_q).\pi_j$$

Ainsi, le nombre moyen de serveurs opérationnels oisifs n_I est donné par :

$$n_I = \sum_{q=1}^{m} n_I^{(q)}$$

– **Le nombre moyen de serveurs en panne de type q ($n_F^{(q)}$) :**
Il correspond au nombre moyen de jetons de couleur r_q dans la place Ser_Down.

$$n_F^{(q)} = \sum_{j:M_j \in A} M_j(Ser\_Down)(r_q).\pi_j = s_q - (n_B^{(q)} + n_I^{(q)})$$

où : s_q est le nombre de serveurs de type q.

Ainsi, le nombre moyen de serveurs en panne n_F est donné par :

$$n_F = \sum_{q=1}^{m} n_F^{(q)} = V - (n_B + n_I)$$

où : V est le nombre total de serveurs (tous types confus).

– **La fréquence de panne des serveurs occupés de type q ($\overline{\gamma}_q$)) :**
Elle correspond à la fréquence (débit) de la transition $Failure$ (ou $Fail\_Act$) pour le cas de pannes actives et pannes dépendantes respectivement.

$$\overline{\gamma}_q = \begin{cases} \sum_{j:M_j \in A(Failure,r_q)} \sum_{i=1}^{n} \gamma_q.M_j(Cus\_Serv)(< c_i, r_q >).\pi_j, & \text{si pannes actives,} \\ \sum_{j:M_j \in A(Fail\_Act,r_q)} \sum_{i=1}^{n} \gamma_q.M_j(Cus\_Serv)(< c_i, r_q >).\pi_j, & \text{si pannes dépendantes.} \end{cases}$$

– **La fréquence de panne des serveurs oisifs de type q ($\overline{\delta}_q$)) :**
Elle correspond à la fréquence de la transition $Fail\_Idle$. Notons que ce paramètre n'a de sens qu'en cas de politique de pannes dépendantes (ou indépendantes).

$$\overline{\delta}_q = \sum_{j:M_j \in A(Fail\_Idle,r_q)} \delta_q.M_j(Ser\_Idle)(r_q).\pi_j$$

Ainsi, le taux moyen de panne des serveurs oisifs $\overline{\delta}$ est donné par :

$$\overline{\delta} = \sum_{q=1}^{m} \overline{\delta}_q$$

– **Le taux moyen de réparation des serveurs de type q ($\overline{\tau}_q$) :**
Il correspond à la fréquence de la transition $Repair$ par rapport à la couleur r_q.

$$\overline{\tau}_q = \sum_{j:M_j \in A(Repair,r_q)} \tau_q.M_j(Ser\_Down)(r_q).\pi_j$$

Ainsi, le taux moyen de réparation $\overline{\tau}$ est donné par :

$$\overline{\tau} = \sum_{q=1}^{m} \overline{\tau}_q$$

– **La probabilité de panne de s serveurs de couleur q ($F_s^{(q)}$) : ($1 \leq s \leq s_q$)**

$$F_s^{(q)} = \sum_{j:M_j(Ser\_Down)(r_q) \geq s} \pi_j$$

Ainsi, la probabilité de panne générale F_V est donnée par :

$$F_V = \sum_{j:M_j(Ser\_Down)=V} \pi_j$$

où : $M_j(Ser\_Down) = \sum_{q=1}^{m} M_j(Ser\_Down)(r_q)$.

– **L'utilisation du réparateur (U_r) :**

$$U_r = F_1 = \sum_{j:M_j(Ser\_Down) \geq 1} \pi_j$$

4.4 Conclusion

Les RdPSG colorés ont une très forte puissance d'expression. En effet, la possibilité d'associer une information aux jetons et de paramétrer le franchissement de transition, permet la représentation des systèmes d'une manière très concise, ce qui aurait nécessité des réseaux non-colorés beaucoup plus volumineux et probablement illisibles. Par conséquent, ce formalisme de haut-niveau permet de réduire la complexité des modèles par rapport aux RdPSG ordinaires.

D'autre part, les modèles de FAR à source finie, qui prennent en compte l'hétérogénéité des clients et des serveurs à la fois, ne sont pas étudiés au jour d'aujourd'hui, et ce ni dans le cas de serveurs fiables ni non-fiables. Ceci est dû essentiellement à la difficulté que pourrait engendrer l'analyse de ces modèles.

L'approche proposée pour la modélisation et l'analyse de ces systèmes avec rappel compliqués, en utilisant le formalisme des RdPSG colorés, présente plusieurs avantages. Un premier avantage est que ce modèle stochastique de haut-niveau permet en plus de la représentation explicite des choix conditionnels et de la synchronisation, l'intégration des contraintes provenant de l'identification et de la particularisation éventuelle (priorité) des différentes ressources ou clients. Ainsi, il nous permet de modéliser facilement des systèmes avec rappel et plusieurs classes de clients et classes de serveurs. Un autre avantage essentiel est le fait que nous pouvons incorporer des dispositifs qui peuvent être quelque peu difficiles à modéliser par des méthodes plus conventionnelles. Par exemple, si les serveurs sont sujets à des pannes aléatoires, nous pouvons facilement modéliser les différentes disciplines de panne d'une façon hiérarchique par les RdPSG colorés. D'autre part, cette approche nous offre la possibilité d'utiliser les méthodes optimales, les résultats et les outils développés dans le domaine des RdPSG colorés. En effet, elle nous offre essentiellement un riche moyen d'expression des indices de performance et de fiabilité exacts, en fonction des éléments du réseau coloré (places, transitions, marquages et couleurs). Par conséquent, elle est bénéfique tant pour la modélisation que pour l'analyse qualitative et quantitative de ces systèmes.

Chapitre 5

Applications

5.1 Introduction

Comme illustration de l'approche d'analyse proposée dans cette thèse, nous exposons dans le présent chapitre, quelques expérimentations relatives aux systèmes avec rappel mono-classe, puis celles des systèmes multi-classes, ce qui nous permettra au passage d'insister à nouveau sur l'importance du bon choix des paramètres adéquats, pour avoir les meilleures performances possibles.

En effet, avec les téléphones de nos jours par exemple, pour effectuer un appel, il suffit d'appuyer sur un bouton. Ainsi, le temps que l'utilisateur met pour rappeler devient plus court en comparaison avec les systèmes téléphoniques conventionnels. Ces rappels auront évidemment un impact négatif non-négligeable sur le temps de réponse des abonnés, en particulier quand la charge du système devient plus importante [18].

Par ailleurs, les composants des systèmes pratiques sont sujets à des pannes aléatoires, ce qui a une forte influence sur les paramètres de performance au même titre que le phénomène de rappel. Ainsi, pour l'analyse des systèmes avec rappel, il est important de prendre ceci en compte lors de la construction du modèle et par la suite lors de son analyse.

Par conséquent, notre étude va porter essentiellement sur l'influence du phénomène de rappel ainsi que la non-fiabilité des serveurs et les politiques de panne, sur les mesures de performance des systèmes.

Les techniques des RdP stochastiques généralisés et stochastiques colorés sont très attrayantes, car elles offrent une approche d'évaluation des performances basée sur une description formelle. Ceci permet l'utilisation du même langage pour la spécification, la validation, l'évaluation des performances et l'implémentation des systèmes.

Dans le cadre de cette étude expérimentale, les résultats numériques ont été calculés en utilisant l'outil software GreatSPN version 2.0.2, qui est un outil très performant principalement dédié à l'évaluation des performances. Il offre à l'utilisateur un environnement graphique simple, lui permettant de définir tous les éléments du modèle, ainsi que tous les paramètres de performance et de fiabilité désirés.

Ainsi, en se basant sur cet outil performant, deux applications ont été effectuées. La première utilise le module des RdPSG pour l'analyse des systèmes mono-classe avec rappel et serveurs non-fiables. Quant à la seconde, elle porte sur les systèmes multi-classes avec rappel, et utilise le module des RdPSG colorés bien-formés. Les résultats sont présentés sous forme de tables et de graphes, qui sont commentés et à partir desquels des conclusions sont tirées.

5.2 Description du GreatSPN

GreatSPN (GRaphical Editor and Analyzer for Timed and Stochastic Petri Nets) [202], est un outil software qui permet la spécification, la validation et l'évaluation des performances des systèmes distribués, en utilisant le formalisme des RdP stochastiques généralisés et leur extension colorés : les RdPS bien-formés. Il comprend des algorithmes d'analyse efficaces, qui permettent son utilisation dans des applications complexes.

Développé à l'université de Torino en Italie, par un groupe de chercheurs qui travaillaient sur les RdPSG et l'évaluation des performances [**?**], il est distribué gratuitement à d'autres universités pour des buts pédagogiques et des expérimentations dans le cadre de la recherche.

La première version de cet outil software a été développée en 1984 suivie de plusieurs autres versions plus améliorées, telle que la version GreatSPN 1.7 [202], dans laquelle de nouveaux algorithmes ont été ajoutés pour améliorer la rapidité des calculs et aussi pour l'analyse des modèles colorés de haut-niveau, qui permettent à l'utilisateur la conception des modèles de systèmes complexes d'une manière plus compacte.
La recherche dans ce domaine continue à évoluer produisant encore de nouvelles versions plus améliorées dont la dernière est la version 2.0.2.

Le package GreatSPN est composé de plusieurs programmes séparés qui coopèrent pour la construction et l'analyse des modèles de RdP en partageant des fichiers. D'autre part, en utilisant les systèmes de réseaux, différents modules d'analyse peuvent s'exécuter sur différentes machines dans un environnement de calcul distribué. Tous les modules sont écrits en langage de programmation C, pour garantir la portabilité et l'efficacité sur différentes machines Unix (Linux, ou autre variante).

Par ailleurs, l'outil GreatSPN permet l'analyse de systèmes complexes et volumineux ayant un espace d'états important. Ceci est possible du fait que tous les modules de résolution utilisent des techniques de stockage particulières pour le gain de mémoire. À titre d'exemple, la limite pratique de la version 1.7 sur des machines SUN SPARC avec une mémoire de 12-16 Mo s'élève à plusieurs centaines de milliers de marquages [202]. En fait, la plupart des algorithmes actuellement implémentés dans le GreatSPN représentent l'état-de-l'art en terme d'une utilisation efficace de la mémoire et de la CPU.

Les principales fonctions et techniques d'analyse implémentées dans l'outil GreatSPN sont :

– Une interface graphique simple à manipuler qui permet à l'utilisateur la saisie du

modèle correspondant au système à analyser. Ce choix s'explique par la nature graphique des représentations du formalisme RdP ;
- Vérification des propriétés structurelles ;
- Calcul des bornes de performance (bornes supérieures et inférieures) à l'aide de la programmation linéaire (pour les modèles non-colorés) ;
- Génération et analyse du graphe d'accessibilité pour les RdPSG et du graphe d'accessibilité symbolique pour les réseaux colorés bien-formés ;
- Résolution markovienne qui permet l'évaluation des performances stationnaires aussi bien que transitoires, en exploitant des techniques numériques matricielles efficaces ;
- La simulation : dans ce cas, plusieurs distributions de probabilité sont admises pour les transitions temporisées des modèles non-colorés, tel que : Erlang, Cox, loi uniforme, discrète, etc.

Par ailleurs, plusieurs autres outils software ont été développés pour l'analyse des modèles de RdPSG. Nous citons essentiellement : SPNP (Stochastic Petri Net Package) [203], TOMSPIN (TOol for Modelling with Stochastic PetrI Nets) [204], TimeNET, UltraSan [205], DSPNExpress [206], etc. Cependant, les points faibles de certains de ces outils étaient la faible portabilité, le manque de flexibilité des programmes et surtout l'absence d'une interface graphique qui est le mode le plus naturel pour la description des modèles basés sur le formalisme RdP. Ainsi, des efforts ont été fournis pour la conception de l'outil GreatSPN, qui est caractérisé par la facilité d'utilisation, la portabilité, la modularité et l'efficacité des modules d'analyse.

En effet, comparé à d'autres outils, la particularité du GreatSPN est le fait qu'il offre un environnement de modélisation et d'analyse plus complet, dans lequel une variété de techniques et d'algorithmes d'analyse sont disponibles. En fait, Chiola a précisé dans [202] que les autres outils offrent de bonnes implémentations de certains de ces algorithmes, mais aucun d'eux ne les offre tous ensemble dans un environnement uniforme tel que GreatSPN. En plus, cet outil permet l'analyse des modèles stochastiques colorés et non-colorés à la fois, ce qui correspond exactement à nos cas d'étude.

1. **Cas des modèles RdPSG :** Une fois que le modèle est complètement spécifié, nous pouvons commencer son analyse. L'analyse structurelle est habituellement accomplie en premier lieu, car elle permet de vérifier des propriétés intéressantes du système, avant de construire son espace d'états. L'analyse structurelle est basée sur le calcul des P-invariants et T-invariants. En fait, si chaque place du réseau est couverte par un certain P-invariant, nous pouvons conclure que le modèle est borné, et par conséquent, il a un espace d'états fini. Ceci est une étape préliminaire importante. D'ailleurs, il n'y a aucun intérêt à essayer de calculer l'espace d'états du modèle pour plus d'analyse, si nous savons que le modèle est non borné. D'autre part, le calcul nous permet de vérifier si toutes les transitions sont couvertes par des T-invariants. Ceci est une condition nécessaire à la vivacité d'un réseau borné. Cependant, cette condition n'est pas suffisante. En fait, pour vérifier si le modèle est vivant par rapport à un marquage initial donné, nous avons besoin de voir la structure du graphe d'accessibilité. Ainsi, une fois que nous avons déterminé que l'espace d'états du modèle est fini, le graphe d'accessibilité tangible sera généré en se basant sur les résultats théoriques présentés dans [207, 146]. Si l'examen de ce graphe fini permet

de déduire que le marquage initial est un état d'accueil, alors on est sûr que la chaîne de Markov sous-jacente est ergodique. Ainsi, elle sera dérivée et résolue pour permettre l'analyse quantitative.

Le graphe des marquages accessibles peut être généré sans aucune information temporelle. Cependant, si nous nous intéressons au calcul des indices de performance, nous avons besoin de définir les poids des transitions immédiates, les taux des transitions temporisées, ainsi que leur sémantique d'exécution : serveur-unique, multiple ou infini.

L'outil GreatSPN offre aussi la possibilité d'estimer l'ordre de magnitude des fréquences de franchissement de certaines transitions et celui du nombre de jetons dans certaines places, et ce sans la génération de tout l'espace d'états. Ceci peut se faire à l'aide du module de calcul des bornes (supérieures et inférieures). L'avantage de cette technique est principalement le fait que la complexité de l'algorithme de résolution soit indépendante de la taille de l'espace d'états.

2. **Cas des modèles colorés :** Une fois que le modèle coloré est complètement décrit, son analyse consiste à générer directement le graphe des marquages accessibles (ordinaires ou symboliques), ainsi que la chaîne de Markov sous-jacente au modèle. Les propriétés qualitatives du système modélisé, sont ensuite vérifiées en se basant sur ce graphe d'accessibilité. Si l'examen de ce graphe permet de déduire que le réseau est borné et que le marquage initial est un état d'accueil, on peut alors affirmer que la chaîne de Markov sous-jacente est ergodique. Ainsi, la distribution des probabilités sera générée et ensuite différents indices de performance seront calculés pour permettre l'analyse quantitative.

Il est important de préciser que dans l'outil GreatSPN, la partie résolution de la chaîne de Markov est commune aux deux modèles RdPS généralisés et colorés. Le calcul de la distribution de probabilité de marquage à l'état stationnaire, se fait en utilisant une méthode numérique itérative basée sur l'algorithme de Gauss-Seidel, et la solution transitoire à un instant donné par rapport à un temps initial, se fait en utilisant une version optimisée de la méthode dite : randomization.

Une fois que la distribution de probabilité d'état stationnaire est obtenue, l'outil permet de calculer automatiquement et implicitement les fréquences de franchissement des transitions. De plus, il permet le calcul d'autres indices de performance qui peuvent être explicitement définis par l'utilisateur, tel que : le nombre moyen de marques dans une place donnée, la probabilité d'un certain évènement ou encore le temps qu'un jeton passe dans un sous-réseau. Une grammaire formelle est utilisée pour la définition des résultats désirés.
Les résultats obtenus peuvent être visualisés sur l'interface graphique du logiciel, comme on peut les récupérer à partir du fichier $file\_name.sta$.

5.3 Application 1 : Systèmes mono-classe avec rappel et serveurs non-fiables

Le but de cette application est d'étudier l'influence du phénomène de rappel, ainsi que la non-fiabilité des serveurs et les disciplines de panne, sur les mesures de performance des systèmes.

Dans un premier temps, les résultats obtenus dans le cas fiable sont validés par un programme Pascal donné dans le livre de Falin et Templeton [5], puisque si le taux de panne dans un modèle non-fiable est très faible et le taux de réparation est très élevé, les valeurs des indices de performance devraient être proches des valeurs correspondantes au modèle fiable.

Les résultats de cette validation sont donnés dans la table 5.1, à partir de laquelle nous pouvons constater que les mesures de performance obtenus pour les modèles avec pannes actives et modèles avec pannes dépendantes sont pratiquement les mêmes d'une part, et sont d'autre part, très proches de ceux correspondants au modèle fiable.

Nous présentons dans ce qui suit, les résultats numériques de l'étude des performances et de la fiabilité des systèmes avec rappel et serveurs identiques non-fiables. En premier lieu, nous illustrons l'effet des paramètres du système et la discipline de panne, sur le temps de réponse moyen. Comme nous étudions les systèmes multi-serveurs avec rappel, pannes et réparations, nous nous intéressons particulièrement à l'influence du taux de rappel, du taux de panne des serveurs occupés, du taux de réparation et du nombre de serveurs sur le temps de réponse moyen. Pour cela, nous considérons les quatre modèles suivants :

- Modèle 1 : Systèmes avec rappel, pannes actives et sources intelligentes.
- Modèle 2 : Systèmes avec rappel, pannes dépendantes et sources intelligentes.
- Modèle 3 : Systèmes avec rappel, pannes actives et sources bloquées.
- Modèle 4 : Systèmes avec rappel, pannes dépendantes et sources bloquées.

Nous nous concentrons ensuite sur quatre descripteurs de performance et de fiabilité importants qui sont : le nombre moyen de sources en orbite, la probabilité de blocage, le nombre moyen de serveurs en panne et l'utilisation du réparateur.

Les paramètres d'entrée des systèmes étudiés sont résumés dans la table 5.2.

Dans les tables 5.3-5.6, nous comparons les temps de réponse moyens pour les différentes disciplines de panne. Nous vérifions respectivement, l'effet du taux de rappel, du taux de panne des serveurs occupés, du taux de réparation et du nombre de serveurs sur le temps de réponse moyen.

- À partir des tables 5.3, 5.5 et 5.6, nous constatons que le temps de réponse moyen décroît quand le taux de rappel, le taux de réparation ou le nombre de serveurs augmente.

TAB. 5.1 – Validations

	Fiable	Non-Fiable	
		Pannes actives	Pannes dépendantes
Nombre de serveurs	4	4	4
Taille de la source	20	20	20
Taux de génération des appels primaires	0.1	0.1	0.1
Taux de service	1	1	1
Taux de rappel	1.2	1.2	1.2
Taux de panne des serveurs	–	1e-25	1e-25
Taux de réparation des serveurs	–	1e+25	1e+25
Nombre moyen de serveurs occupés	1.800748	1.800764	1.800763
Nombre moyen de sources d'appels répétés	0.191771	0.191786	0.191785
Taux moyen génération des appels primaires	1.800748	1.800745	1.800745
Temps d'attente moyen	0.106495	0.1065036	0.1065031

TAB. 5.2 – Paramètres d'entrée des systèmes

	K	s	λ	μ	ν	γ	δ	τ
Table 5.3	30	6	4	5	axe abscisses	0.02	0.01	0.5
Table 5.4	30	6	4	5	6	axe abscisses	0.01	0.5
Table 5.5	30	6	4	5	6	0.02	0.01	axe abscisses
Table 5.6	30	axe abscisses	4	5	6	0.02	0.01	0.5
Figure 5.1	30	1-20	4	5	axe abscisses	0.02	–	0.5
Figure 5.2	30	1-20	4	5	axe abscisses	0.02	–	0.5
Figure 5.3	30	6	4	5	6	axe abscisses	–	0.01 - 5
Figure 5.4	30	1-20	4	5	6	axe abscisses	–	0.5

TAB. 5.3 – L'effet du taux de rappel sur le temps de réponse

Taux de rappel	Modèle 1	Modèle 2	Modèle 3	Modèle 4
0.01	**1.841 499**	1.887 433	1.842 907	1.888 028
0.05	**1.462 016**	1.483 774	1.462 054	1.483 794
0.1	**1.327 160**	0.343 180	0.327 220	0.343 211
0.5	**1.059 739**	1.067 077	1.059 924	1.067 161
1	**0.973 964**	0.978 834	0.974 221	0.978 945
5	**0.855 525**	0.857 012	0.855 908	0.857 139
10	**0.832 569**	0.833 410	0.832 978	0.833 510

TAB. 5.4 – L'effet du taux de panne des serveurs occupés sur le temps de réponse

Taux de panne	Modèle 1	Modèle 2	Modèle 3	Modèle 4
0.01	**0.820 536**	0.821 632	0.820 643	0.821 727
0.05	**0.962 481**	0.964 374	0.962 700	0.964 553
0.06	**1.011 886**	1.013 991	1.012 104	1.014 163
0.07	**1.067 183**	1.069 498	1.067 372	1.069 633
0.08	1.129 232	1.130 876	**1.128 474**	1.130 933
0.09	1.195 295	1.198 009	**1.195 285**	1.197 933
0.1	1.267 797	1.270 695	**1.267 591**	1.270 418
0.3	3.371 101	3.375 188	**3.342 723**	3.348 648
0.5	5.752 723	5.756 764	**5.659 854**	5.663 768
0.7	8.149 896	8.154 028	**7.976 819**	7.980 748
0.9	10.549 268	10.553 480	**10.288 971**	10.292 967
1	11.749 170	11.753 428	**11.443 867**	11.447 903

TAB. 5.5 – L'effet du taux de réparation sur le temps de réponse

Taux de réparation	Modèle 1	Modèle 2	Modèle 3	Modèle 4
0.001	25.823 081	26.515 162	**25.778 794**	26.384 438
0.002	13.743 889	14.090 351	**13.736 046**	14.038 503
0.003	**9.709 730**	9.941 020	9.713 653	9.915 096
0.004	**7.687 022**	7.860 747	7.696 542	7.847 516
0.005	**6.469 041**	6.608 242	6.481 699	6.602 423
0.01	**4.002 684**	4.072 934	4.020 165	4.080 565
0.05	**1.783 052**	1.798 386	1.794 525	1.807 512
0.1	**1.346 321**	1.354 509	1.352 615	1.359 692
0.3	**0.923 416**	0.925 984	0.924 350	0.926 778
0.5	**0.848 240**	0.849 520	0.848 381	0.849 642
0.7	**0.827 751**	0.828 562	0.827 771	0.828 579
0.9	**0.819 159**	0.819 751	0.819 162	0.819 753
1	**0.816 453**	0.816 975	0.816 454	0.816 974

TAB. 5.6 – L'effet du nombre de serveurs sur le temps de réponse

s	Modèle 1	Modèle 2	Modèle 3	Modèle 4
1	6.161 454	6.164 884	**6.151 971**	6.155 147
2	2.966 975	2.968 944	**2.966 584**	2.968 541
3	1.902 934	1.904 456	**1.902 907**	1.904 429
4	**1.371 718**	1.373 051	**1.371 718**	1.373 050
5	**1.054 730**	1.055 986	1.054 757	1.056 014
8	**0.612 652**	0.614 423	0.613 353	0.614 744
10	**0.493 967**	0.496 916	0.495 292	0.496 773
15	**0.351 967**	0.360 979	0.354 056	0.355 527
20	**0.285 129**	0.298 578	0.287 452	0.288 515

- Dans la table 5.3, on voit bien que le taux de rappel a une influence significative sur le temps de réponse moyen quand ce taux est faible. Cependant, quand de plus en plus de requêtes répétées arrivent, la décroissance n'est pas considérable.

- La table 5.4 montre que le temps de réponse moyen croît quand le taux de panne des serveurs occupés croît. Nous pouvons aussi constater que parmi les quatre disciplines, le modèle avec pannes actives et sources intelligentes (modèle 1) donne les meilleures performances, quand le taux de panne des serveurs occupés est inférieur à une certaine valeur qui approche 0.08. Autrement, le modèle avec pannes actives et sources bloquées (modèle 3) donne les meilleurs temps de réponse moyens quand le taux de panne est plus grand que cette valeur. Ainsi, pour les systèmes anciens par exemple, dont les serveurs tombent fréquemment en panne, il est plus intéressant d'appliquer la discipline des pannes actives avec sources bloquées.

- Dans la table 5.5, on montre combien l'augmentation du taux de réparation τ affecte le temps de réponse moyen, principalement dans le cas où ce taux est faible. Par exemple, la différence entre les temps de réponse pour $\tau = 0.001$ et $\tau = 0.005$ est très significative ($\approx -75\%$). Par contre, quand la durée de réparation est minime, la différence n'est pas importante.

- À partir des tables 5.3 et 5.5, nous voyons aussi que les temps de réponse moyens sont meilleurs (plus faibles) dans les modèles avec pannes actives (Modèles 1 et 3) par rapport aux modèles avec pannes dépendantes (Modèles 2 et 4), et ce quand les taux de rappel et de réparation sont faibles. Cependant, quand ces taux augmentent, la différence entre les quatre modèles n'est plus significative.

- À partir de la table 5.6, nous remarquons que des petites modifications dans le nombre de serveurs, particulièrement de 1 à 3 serveurs, produit une très importante baisse dans le temps de réponse moyen ($\approx -70\%$). Cependant, après une certaine valeur ($s = 3$), la décroissance n'est pas considérable. D'autre part, pour les systèmes avec 1 à 3 serveurs, le modèle avec pannes actives et sources bloquées (modèle 3) donne les meilleurs temps de réponse moyens. Cependant, pour plus de quatre serveurs, le modèle avec pannes actives et sources intelligentes (modèle 1) est le meilleur. Dans le cas particulier où $s = 4$, les deux modèles avec pannes actives donnent des résultats identiques.

- Finalement, les résultats numériques dans les tables 5.3-5.6 coïncident avec la logique qui dit que le temps de réponse moyen est meilleur quand le système est neuf (taux de panne faible des serveurs), la durée de réparation est minime, et quand le taux de rappel et le nombre de serveurs sont importants. Nous pouvons aussi voir que les résultats du modèle 1 (modèle 2) sont relativement proches de ceux du modèle 3 (modèle 4 respectivement). Par conséquent, la discipline de panne (active ou dépendante) a plus d'influence sur les performances du système, que le type des sources (bloquées ou intelligentes).

Nous donnons dans ce qui suit, des graphes représentant d'autres mesures de perfor-

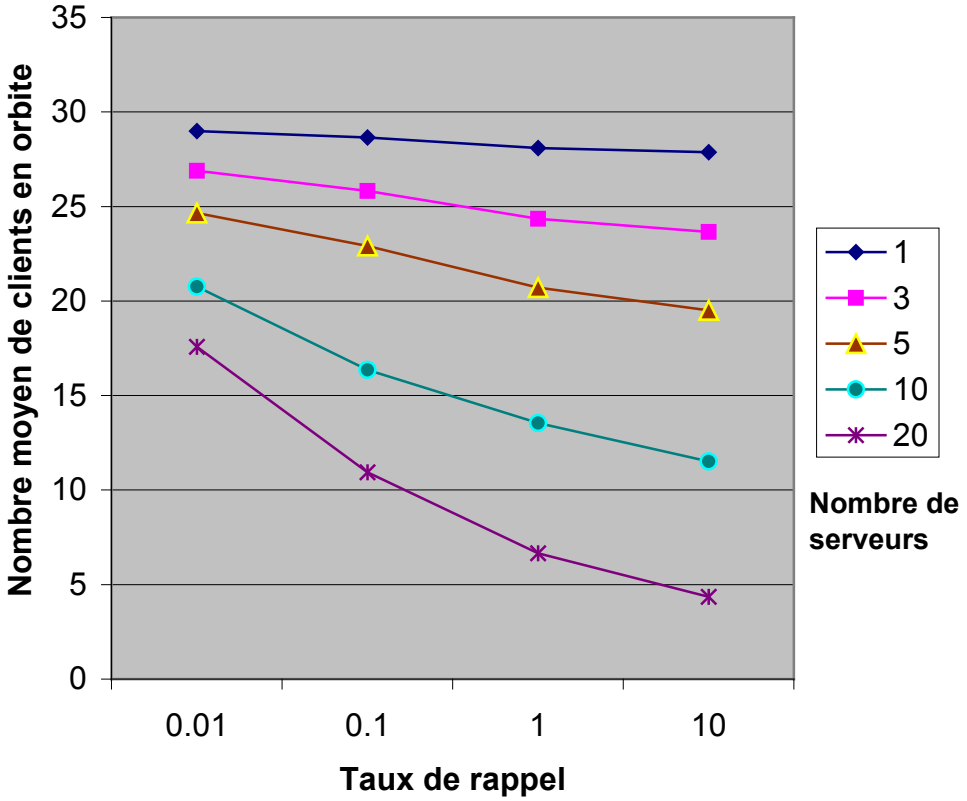

FIG. 5.1 – Nombre moyen de clients bloqués en fonction du taux de rappel

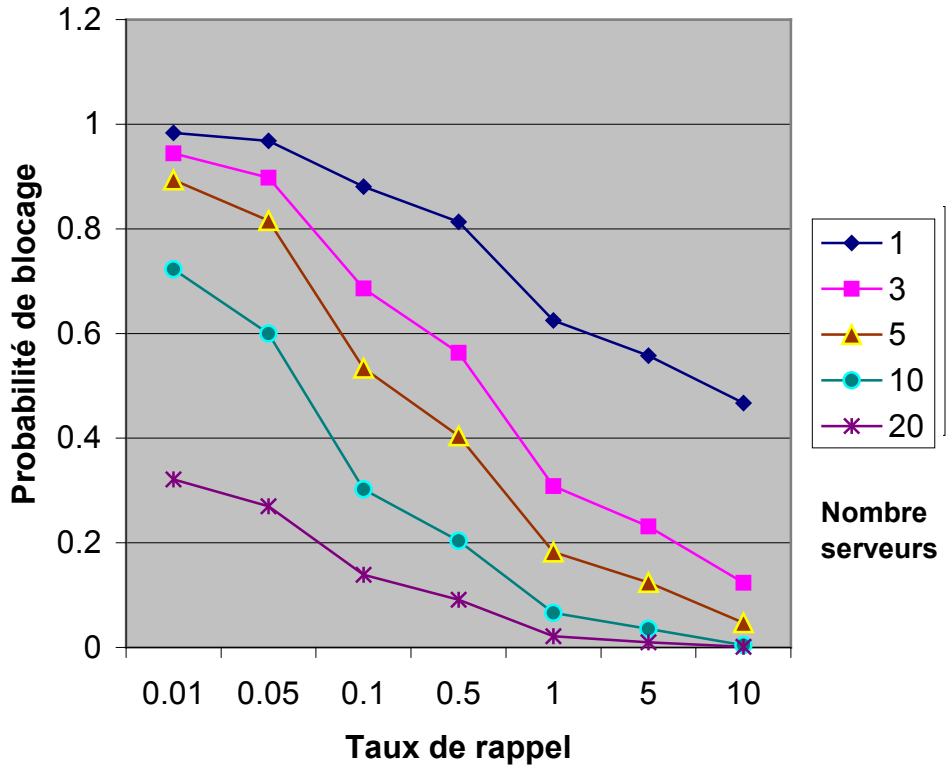

FIG. 5.2 – Probabilité de blocage en fonction du taux de rappel

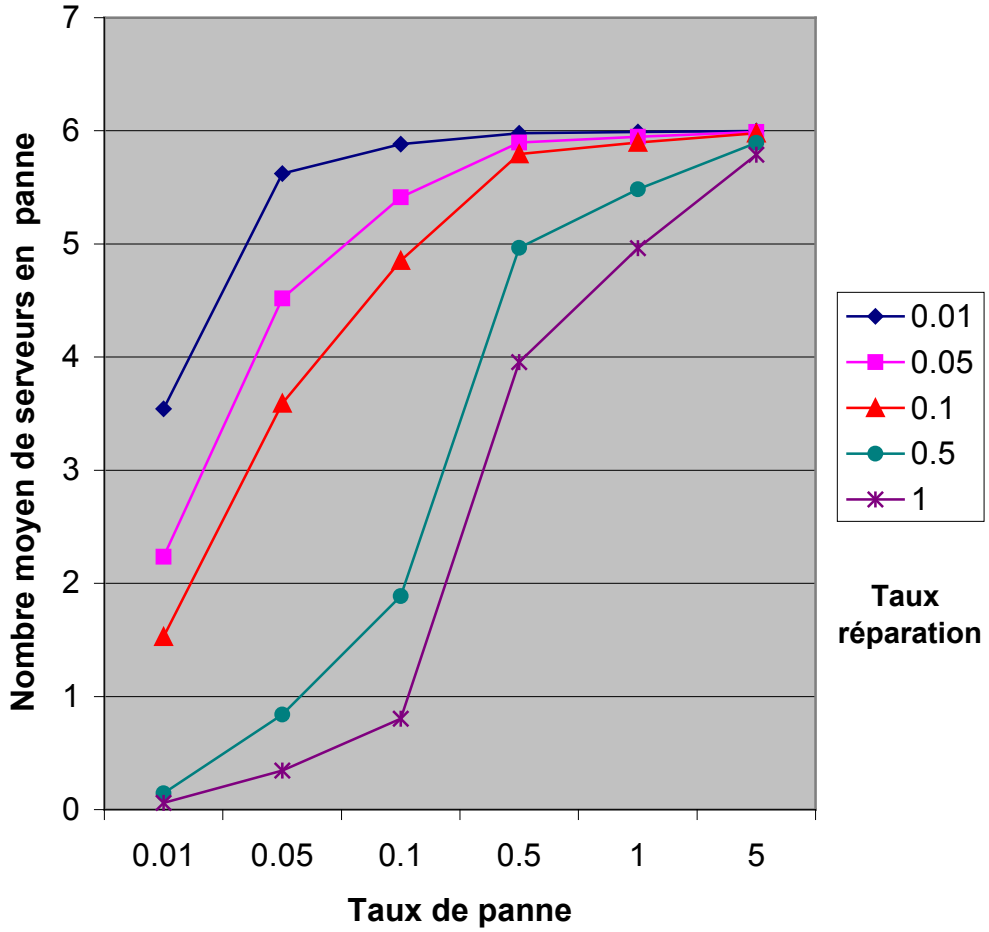

FIG. 5.3 – Nombre moyen de serveurs en panne en fonction du taux de panne

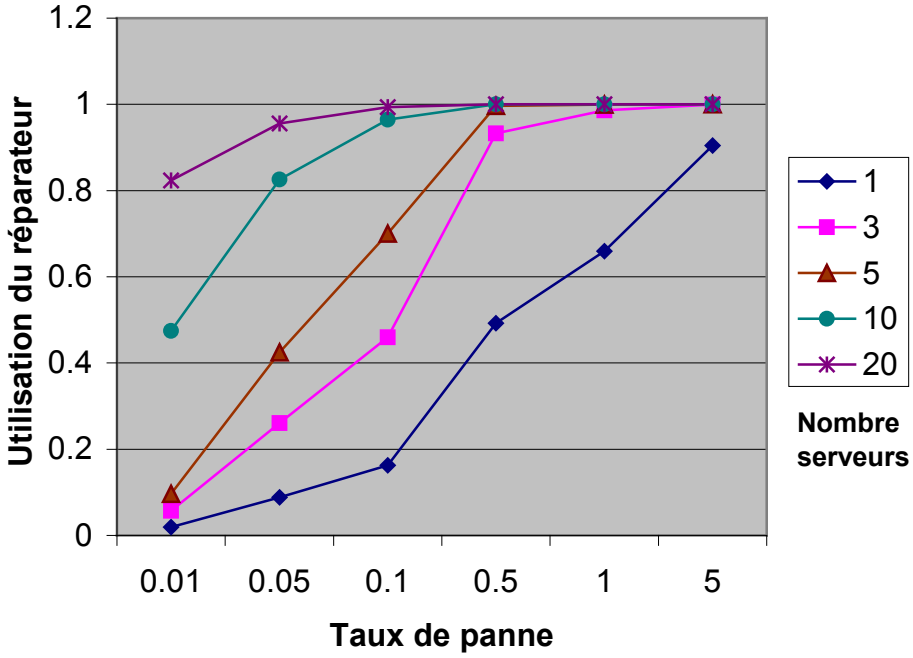

FIG. 5.4 – Utilisation du réparateur en fonction du taux de panne

mance importants, en fonction des deux facteurs : taux de rappel et taux de panne.

- Dans la figure 5.1, le nombre moyen de clients bloqués (en orbite) est représenté en fonction du taux de rappel ν. Nous avons présenté cinq courbes qui correspondent au nombre de serveurs $s = 1, 3, 5, 10$ et 20. À partir de cette figure, on constate que le nombre moyen de clients bloqués décroît en fonction du taux de rappel. Cette décroissance se fait doucement quand les appels répétés s'intensifient. D'autre part, elle est beaucoup plus rapide quand le nombre de serveurs augmente.

- La figure 5.2 montre la probabilité de blocage en fonction du taux de rappel. Là aussi, nous voyons bien qu'une incrémentation du nombre de serveurs dans le système permet de réduire d'une manière significative la probabilité de blocage. Cette figure montre aussi que le choix optimal de la probabilité de blocage correspond au cas où la durée de temps entre deux rappels successifs est courte et le nombre de serveurs est important.

- La figure 5.3 illustre le comportement du nombre moyen de serveurs en panne en fonction du taux de panne des serveurs occupés. Les courbes montrent également l'influence du taux de réparation sur ce nombre moyen de serveurs en panne. Dans cette figure, nous pouvons voir que le nombre de serveurs non opérationnels croît quand le taux de panne croît, et décroît en incrémentant le taux de réparation. Ainsi, les résultats idéaux sont obtenus quand le taux de panne est faible (pannes rares) et le taux de réparation est élevé (réparation rapide), ce qui est intuitivement logique.

- La figure 5.4 montre l'effet du taux de panne des serveurs occupés sur l'utilisation du réparateur. Pour comprendre aussi l'influence du nombre de serveurs, nous avons fixé s à 1, 3, 5, 10 et 20. Ainsi, nous avons pu constater que l'utilisation du réparateur croît avec le taux de panne et le nombre de serveurs.

- Finalement, nous pouvons conclure que le taux de rappel, le taux de panne, le taux de réparation et le nombre de serveurs, sont les facteurs essentiels qui affectent la performance et la fiabilité des systèmes avec rappel, et ce d'une manière beaucoup plus significative que la discipline de panne adoptée.

5.4 Application 2 : Systèmes multi-classes avec rappel et serveurs fiables

Le but de cette application est d'étudier l'influence des différents paramètres du système, et particulièrement celui du phénomène de rappel, ainsi que la discipline de service, sur les indices de performance moyens des systèmes multi-classes. Ainsi, nous considérons quelques exemples numériques, pour montrer l'influence de ces paramètres sur le temps de réponse qui constitue un des indices de performance les plus significatifs.

Dans un premier temps, nous nous concentrons sur des systèmes avec plusieurs classes de clients (hétérogènes) et des serveurs homogènes. Pour la validation du modèle de RdPSG coloré proposé, nous considérons le cas totalement homogène où les clients des

TAB. 5.7 – Validations

	Pascal [5]	Système multi-classes
Nombre de serveurs	4	4
Taille des sources	20	$10c_1 + 6c_2 + 4c_3$
Taux de génération des appels primaires	0.1	0.1 (c_1, c_2, c_3)
Taux de service	1	1
Taux de rappel	1.2	1.2 (c_1, c_2, c_3)
Nombre moyen de serveurs occupés	1.800 748	1.800 750
Nombre moyen de sources d'appels répétés	0.191 771	c_1 : 0.095 887 c_2 : 0.057 532 c_3 : 0.038 355 Total : 0.191 773
Taux moyen de génération des appels primaires	1.800 748	c_1 : 0.900 374 c_2 : 0.540 224 c_3 : 0.360 150 Total : 1.800 748
Temps d'attente moyen	0.106 495	0.106 496 (c_1, c_2, c_3)

différentes classes ont les mêmes paramètres (taux d'arrivée, taux de service et taux de rappel). Les résultats obtenus sont validés par le programme Pascal donné dans [5], puisque si tous les clients et les serveurs sont homogènes, les mesures devraient être proches de celles correspondantes au cas de FAR multi-serveurs à source fine avec clients homogènes et des serveurs identiques [5].

À partir de la table 5.7, nous pouvons constater que les résultats du modèle de réseau de Petri stochastique coloré correspondant au système multi-classes, convergent avec ceux du modèle de file d'attente avec rappel, et ce dans le cas de clients homogènes.

Nous présentons dans ce qui suit, des graphes qui illustrent l'effet de quelques paramètres importants, sur le temps de réponse moyen des systèmes avec serveurs homogènes et plusieurs classes de clients (hétérogènes). Pour une représentation plus facile des résultats et des graphes, nous considérons des systèmes à 3 classes de clients.

Dans la figure 5.5 qui représente le temps de réponse moyen en fonction du taux d'arrivée avec service homogène et rappel hétérogène, nous considérons un système à 4 serveurs homogènes et 3 classes de clients, dont chaque classe lui correspond un taux de rappel particulier. À partir de cette figure, nous pouvons constater que le temps de réponse moyen croît en fonction de l'intensité du flux d'arrivée des requêtes primaires, et il décroît quand le taux d'arrivée des appels répétés augmente. Ainsi, les meilleures performances correspondent aux clients de la troisième classe dont le taux de rappel est le plus élevé.

D'autre part, ce temps de réponse moyen est beaucoup plus affecté par le taux d'arrivée quand le taux de rappel est faible. Par contre, pour des taux de rappel importants, la différence est moins significative. En effet, pour les clients de la classe 1 ($\nu = 0.1$), le temps de réponse est 16 fois plus important quand le taux d'arrivée passe de 0.1 à 0.5, alors que pour les clients de la classe 3 ($\nu = 10$), l'évolution est de 50% seulement.

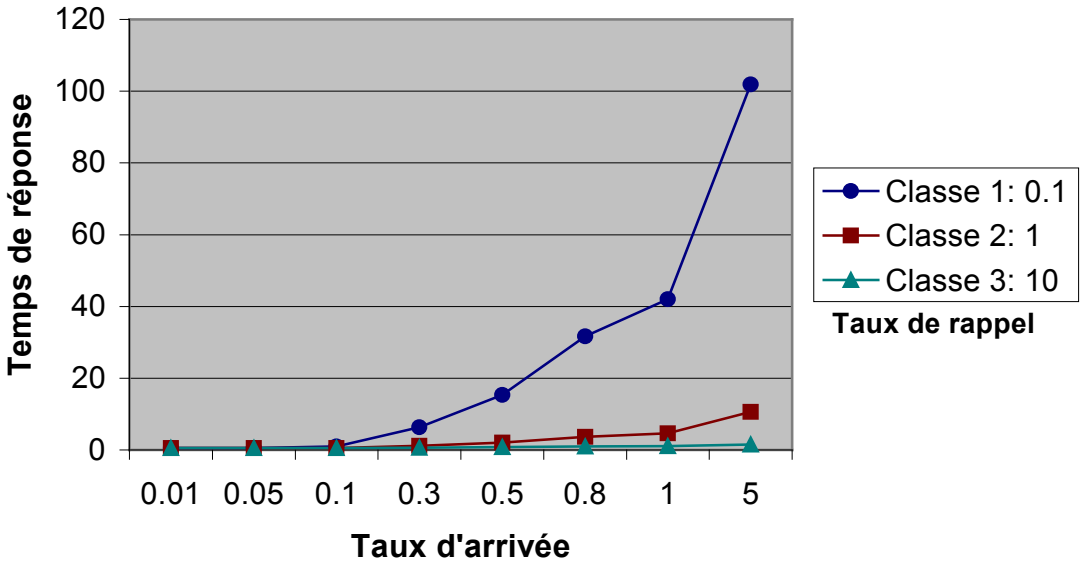

FIG. 5.5 – Temps de réponse en fonction des taux d'arrivée et de rappel

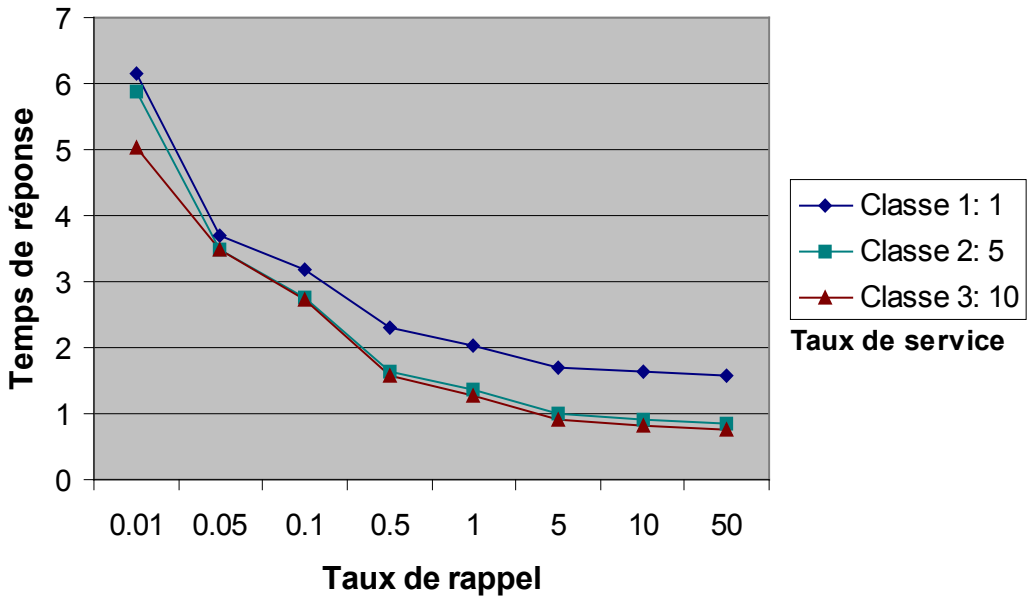

FIG. 5.6 – Temps de réponse en fonction des taux de rappel et de service

TAB. 5.8 – Paramètres des systèmes avec serveurs hétérogènes

	N	λ	ν	Classe 1	Classe 2	Classe 3
Fig. 5.7	50	abcisses	1	1 :$\mu_1 = 2$	2 :$\mu_2 = 5$	3 :$\mu_3 = 8$
Fig. 5.8	50	1	abcisses	1 :$\mu_1 = 2$	2 :$\mu_2 = 5$	3 :$\mu_3 = 8$

À partir de la figure 5.6 qui illustre la variation du temps de réponse moyen en fonction du taux de rappel, avec service hétérogène (dépendant du type de clients) et un processus d'arrivée des requêtes primaires homogène, nous pouvons déduire que les meilleures performances correspondent aux clients de la troisième classe, dont le taux de service est le plus élevé. D'autre part, on voit bien que les variations du taux de rappel ont une influence significative sur les performances du système, quand ce taux est faible. Par contre, quand de plus en plus de requêtes répétées arrivent, la décroissance n'est pas considérable.

Dans un second temps, nous considérons des systèmes avec clients homogènes et 3 classes de serveurs hétérogènes. La première classe qui correspond aux serveurs les moins rapides, comprend un serveur unique de taux μ_1, la seconde classe comprend 2 serveurs de taux μ_2, et la troisième classe regroupe les 3 serveurs les plus rapides de taux μ_3. Ainsi, nous étudions l'effet des paramètres du système et de la discipline de service (service aléatoire ou service le plus rapide SPR) sur le temps de réponse moyen.

Les paramètres des systèmes étudiés sont résumés dans la table 5.8.

Dans les figures 5.7 et 5.8, nous pouvons voir l'évolution du temps de réponse moyen en fonction du taux d'arrivée des requêtes primaires et du taux de rappel resp., ainsi que la différence entre les deux disciplines de service.

À partir de ces figures, nous voyons bien que le temps de réponse moyen croît avec la croissance du flux des arrivées primaires, et décroît avec la croissance du flux des appels répétés. D'autre part, les résultats obtenus pour la discipline du service le plus rapide (SPR), sont toujours meilleurs que ceux du service aléatoire, particulièrement quand le flux des arrivées est important. En effet, l'écart entre les deux disciplines est plus significatif quand le taux des arrivées primaires est important.

À présent, nous considérons un système complètement hétérogène, avec 3 classes de clients, chacune caractérisée par un taux de rappel particulier, et 2 classes de serveurs (serveurs rapides et serveurs lents). Nous commençons d'abord par la validation du modèle de RdPSG coloré proposé, et ce dans le cas de clients et serveurs homogènes. Les résultats de cette validation sont présentés dans la table 5.9, à partir de laquelle nous pouvons voir que les indices de performance obtenus pour ce modèle à l'aide du GreatSPN, sont très proches (à 10^{-5} près), de ceux correspondants à la FAR multi-serveurs à source finie [5]. D'autre part, ces résultats sont identiques pour les deux disciplines de service, ce qui est logique du fait que tous les serveurs sont supposés avoir la même vitesse de traitement.

Nous étudions ensuite, l'effet des paramètres du système et la discipline de service sur le temps de réponse moyen de ces systèmes totalement hétérogènes. Ainsi, nous considé-

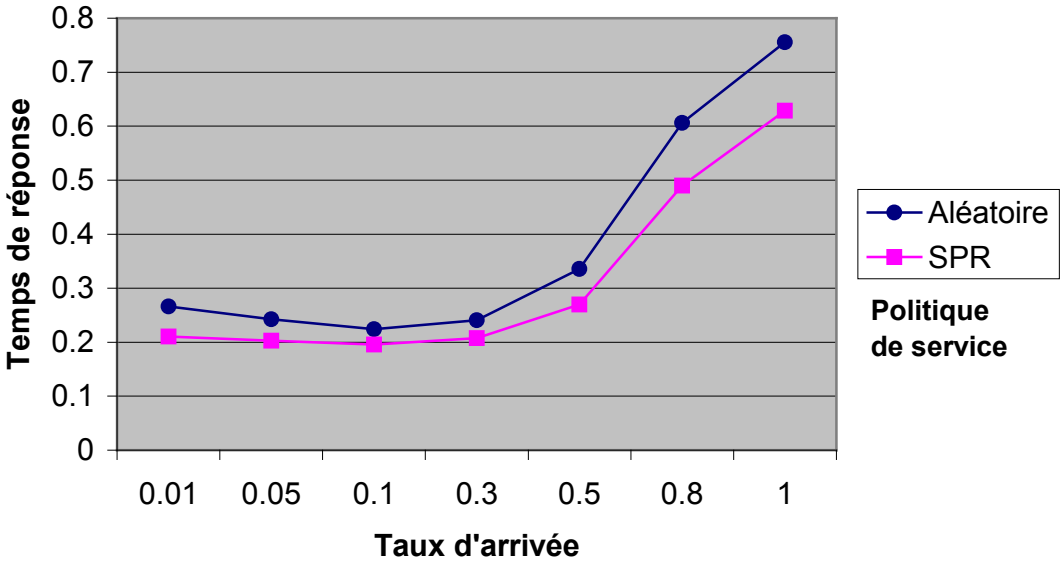

FIG. 5.7 – Temps de réponse en fonction du taux d'arrivée

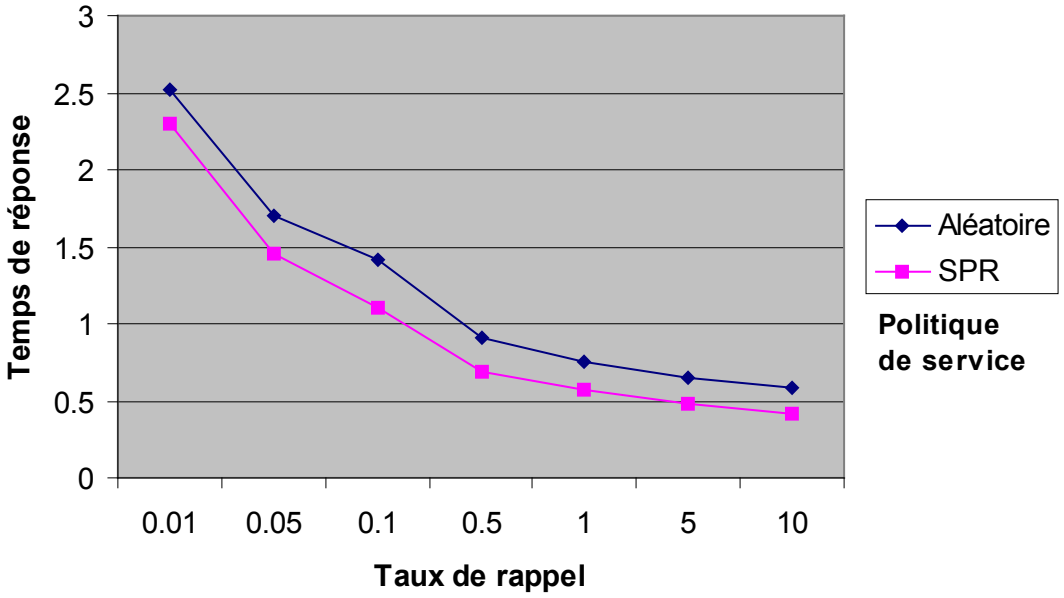

FIG. 5.8 – Temps de réponse en fonction du taux de rappel

TAB. 5.9 – Validations

	Pascal [5]	Aléatoire	SPR
Nombre de serveurs	4	$2s_1 + 2s_2$	$2s_1 + 2s_2$
Taille des sources	20	$10c_1 + 6c_2 + 4c_3$	$10c_1 + 6c_2 + 4c_3$
Taux de génération des appels primaires	0.1	0.1 (c_1, c_2, c_3)	0.1 (c_1, c_2, c_3)
Taux de service	1	1 (s_1, s_2)	1 (s_1, s_2)
Taux de rappel	1.2	1.2 (c_1, c_2, c_3)	1.2 (c_1, c_2, c_3)
Nombre moyen de serveurs occupés	1.800 748	1.800 750	1.800 750
Nombre moyen de sources d'appels répétés	0.191 771	0.191 773	0.191 773
Taux moyen de génération des appels primaires	1.800 748	1.800 748	1.800 748
Temps d'attente moyen	0.106 495	0.106 496	0.106 496

TAB. 5.10 – Temps de réponse moyen en fonction du taux d'arrivée avec clients et serveurs hétérogènes

	Classe 1 : $\nu_1 = 0.1$		Classe 2 : $\nu_2 = 1$		Classe 3 : $\nu_3 = 10$	
λ	Aléatoire	SPR	Aléatoire	SPR	Aléatoire	SPR
0.01	0.355 301	0.308 448	0.354 009	0.307 421	0.353 878	**0.307 320**
0.05	0.493 347	0.441 976	0.376 320	0.340 948	0.364 160	**0.330 612**
0.1	1.100 978	1.016 795	0.446 152	0.417 840	0.377 183	**0.355 165**
0.5	15.443 962	15.267 005	1.925 974	1.908 168	0.590 826	**0.587 176**
1	41.000 866	40.879 087	4.426 307	4.414 946	0.843 833	**0.842 541**
5	100.233 382	100.207 120	10.288 084	10.285 549	1.337 306	**1.336 992**
10	109.556 050	109.536 685	11.218 722	11.216 850	1.417 344	**1.417 103**

rons un système à 30 clients répartis en 3 classes, un serveur rapide ($\mu_1 = 5$) et deux serveurs lents ($\mu_2 = 2$).

Dans la table 5.10, nous donnons les résultats obtenus avec chacune des deux politiques de service pour chaque type de clients. Ainsi, nous pouvons constater que la politique du serveur le plus rapide (SPR) donne de meilleurs résultats que celle du service aléatoire, et ce pour les différentes classes de clients. Par ailleurs, les meilleures performances sont obtenues par les clients de la troisième classe, dont le taux de rappel est le plus élevé, ce qui coïncide avec les résultats des expériences précédentes.

Par ailleurs, nous remarquons que les temps de réponse des clients des 3 classes, convergent quand le taux de génération des appels primaires est faible. Par contre, plus l'intensité des arrivées primaires augmente, plus l'écart entre les 3 classes est important. En effet, dans ce cas, les clients de la classe 3 dont le taux de rappel est le plus important, sont nettement plus avantagés, particulièrement sous la discipline du serveur le plus rapide.

5.5 Conclusion

Un des avantages de l'approche proposée dans cette thèse, est la possibilité d'application des différents outils développés par la communauté des réseaux de Petri stochastiques et stochastiques colorés, pour l'analyse des systèmes complexes avec phénomène de rappel.

Divers calculs numériques ont été accomplis en utilisant l'outil GreatSPN version 2.0.2, pour montrer l'effet des différents paramètres et des différentes politiques de panne et disciplines de service, sur les performances des systèmes mono-classe avec serveurs non-fiables et des systèmes multi-classe.

En effet, sur la base d'une série d'expérimentations et de résultats numériques exacts obtenus, nous avons pu étudier les possibilités qui permettent l'optimisation et l'amélioration des performances des systèmes complexes avec rappel. D'ailleurs, les résultats obtenus nous ont confirmé qu'une conception réfléchie pour choisir précisément les bons paramètres du système et les disciplines adéquates, sans négliger l'aspect économique, est très importante pour atteindre les meilleures performances possibles.

Enfin, les différents résultats numériques exacts dérivés de cet outil, illustrent l'intérêt et prouve la viabilité de l'approche proposée dans cette thèse, tout en montrant l'influence des différents paramètres sur les mesures de performance des systèmes mono-classe et systèmes multi-classes avec rappel.

Conclusion Générale

La complexité et l'expansion croissante des systèmes avec rappel, tels que les systèmes téléphoniques, où les abonnés recomposent le numéro après réception du signal occupé, ou bien les systèmes informatiques où plusieurs terminaux font des rappels pour recevoir un service d'un processeur central, requièrent un puissant modèle pour la description, la vérification de la correction et l'évaluation des performances et de la fiabilité de ces systèmes.

Parmi les différents modèles existants, le modèle conventionnel habituellement utilisé pour l'analyse de ces systèmes, est celui des files d'attente avec rappel. Durant ces deux dernières décennies, il y a eu un intérêt croissant dans l'étude de ce modèle, ce qui est essentiellement dû, aux avancées technologiques dans les domaines de l'informatique et des télécommunications.

Cependant, la prise en compte du phénomène d'appels répétés, a rendu les résultats de la théorie des files d'attente standards inadéquats, et a introduit de grandes difficultés analytiques. En fait, les résultats explicites détaillés existent pour certaines files d'attente avec rappel particulières, avec des hypothèses contraignantes sur certains paramètres, tel que le nombre de serveurs, la taille de la population, la fiabilité des serveurs, l'homogénéité des clients et des serveurs, etc., alors que pour beaucoup d'autres modèles compliqués, les résultats sont obtenus par des algorithmes numériques, des méthodes d'approximation ou par simulation.

En réalité, l'intérêt essentiel des expressions analytiques réside dans le fait qu'elles fournissent rapidement et à moindre coût des indices de performance du système modélisé. Malheureusement, ces formules sont en général difficiles et même impossible à obtenir. Pour d'autres cas, elles peuvent être très compliquées et difficiles à manipuler. C'est pour cela, que nous avons proposé dans le cadre de cette thèse, une approche qui permet l'obtention de résultats exacts des indices de performance et de fiabilité moyens, ce qui est évidemment meilleur qu'une solution d'approximation ou de simulation.

Par ailleurs, la combinaison du phénomène de rappel avec la limitation de la source de clients et la non-fiabilité des serveurs d'une part, et d'autre part la multiplicité des classes de clients et des classes de serveurs dans un même système, rend sa structure assez complexe et introduit de multiple problèmes de synchronisation, en plus des phénomènes de blocage liés aux appels répétés. Ceci complique énormément l'analyse des performances du système et rend les approches de modélisation traditionnelles difficiles à utiliser et évoque par conséquent, l'adoption d'une approche descriptive plus puissante, telles que celles basées sur les réseaux de Petri stochastiques généralisés et les réseaux de Petri sto-

chastiques généralisés colorés.

Ainsi, l'intérêt de la démarche présentée dans cette thèse, réside essentiellement dans la proposition d'un modèle d'analyse des systèmes avec rappel, autre que le modèle conventionnel des files d'attente avec rappel, et donc d'adapter les méthodes existantes dans ce domaine, pour obtenir des résultats exacts pour des systèmes complexes avec rappel.

Les RdPSG constituent un important modèle graphique et mathématique, qui offre une grande puissance descriptive, ce qui nous a permis la modélisation et l'analyse des systèmes avec rappel incluant diverses caractéristiques (source finie, multiplicité et non-fiabilité des serveurs). Par opposition aux modèles de files d'attente, les RdPSG offrent la possibilité d'inclure facilement et de façon très élégante, les différentes structures de synchronisation, et de représenter très aisément les différents mécanismes de blocage.

Une extension importante de ce modèle correspond aux RdPSG colorés. Ce formalisme de haut-niveau est adapté à la description et l'analyse des systèmes avec des composants hétérogènes. En effet, la possibilité d'associer à chaque jeton, une couleur qui dénote son type nous a permis la représentation et l'analyse des systèmes avec plusieurs classes de clients et classes de serveurs, d'une manière concise et détaillée.

La présentation des différentes caractéristiques qui peuvent être facilement modélisés et analysés en se basant sur les RdPSG et les RdPSG colorés, permet d'illustrer l'intérêt, la simplicité, la concision et la puissance de ces modèles pour la conception des systèmes avec rappel, et de montrer la force et l'élégance de la sémantique correspondante. Ainsi, la puissance d'expression de ces formalismes de haut-niveau constitue une des principales raisons de leur application pour les systèmes avec rappel. D'ailleurs, leur flexibilité nous a permis une construction simple de modèles compacts et détaillés, ainsi qu'une modification facile de ces modèles pour prendre en compte d'autres caractéristiques supplémentaires ayant un sens et un intérêt pratique telles que les pannes et les réparations. C'est ainsi, que nous avons pu analyser des modèles avec différentes politiques de panne et introduire même une nouvelle discipline de panne plus générale que celles définies dans la littérature des files d'attente avec rappel.

En effet, l'utilisation des RdPSG nous permet d'incorporer des caractéristiques qui pourraient être difficiles à modéliser et par conséquent à analyser par des méthodes plus conventionnelles. Par exemple, si les serveurs sont sujets à des périodes de maintenance préventive, en plus des pannes aléatoires, nous pouvons facilement modéliser les deux phénomènes par un RdPSG d'une manière hiérarchique. De la même façon, pour des systèmes particuliers, avec des pannes de la station (pannes générales), la modélisation et l'analyse peuvent être facilement adaptées des modèles avec pannes des serveurs, contrairement aux modèles de FAR. Par conséquent, une classe importante de systèmes complexes avec rappel sont simples à modéliser et donc à analyser en se basant sur la théorie des RdPSG et RdPSG colorés.

Un autre intérêt majeur des RdPSG ordinaires et colorés, est de pouvoir combiner l'analyse qualitative et l'analyse quantitative (i.e. l'évaluation des performances). Par ailleurs, ils offrent aussi un riche moyen d'expression des indices de performance et de

fiabilité exacts en régime stationnaire. En effet, les formules permettant le calcul numérique exact de ces indices s'expriment en fonction des éléments de base du réseau (places, transitions, marquages, couleurs) et des probabilités stationnaires. D'autre part, ces formules sont assez élégantes et d'une manipulation très aisée. Ainsi, l'approche proposée rend l'analyse des performances des systèmes avec rappel plus simple qu'avec les modèles markoviens directs. En effet, les formules exacts de performance et de fiabilité obtenues illustre l'adéquation des modèles de RdPSG pour l'évaluation des performances de ce type de systèmes.

Un autre avantage important, est la possibilité d'application des différentes techniques et outils définis dans le domaine des RdPSG ordinaires et colorés pour l'analyse des systèmes avec appels répétés. Dans le cadre de notre travail, nous avons proposé l'application du logiciel GreatSPN pour l'analyse de ces systèmes. En effet, sur la base d'une série d'expérimentations et de résultats numériques exacts obtenus par cet outil, nous avons pu étudier les possibilités qui permettent l'optimisation et l'amélioration des performances de systèmes complexes. Par ailleurs, la présentation des résultats numériques générés par cet outil prouve la viabilité de l'approche proposée.

Enfin, comme perspectives, il serait intéressant d'orienter des études qui consistent à appliquer les théories et les techniques basées sur le formalisme des RdP stochastiques pour l'analyse d'autres systèmes avec rappel compliqués. En premier lieu, il serait intéressant de considérer les systèmes non-markoviens avec une distribution des temps d'inter-arrivées et/ou des temps de rappel non exponentiels, pour lesquels les résultats restent limités aux méthodes d'approximation et de simulation [36, 208, 38, 209, 210, 211]. En effet, un des problèmes les plus importants et qui est toujours ouvert est de développer des résultats (analytiques aussi bien que numériques) dans le cas des systèmes non-markoviens. Un autre problème intéressant est de considérer le régime transitoire dont les travaux restent à ce jour assez rares [212, 213]. Un autre espace de recherche serait de considérer l'analyse des réseaux de files d'attente avec rappel. D'ailleurs, il a été prouvé récemment dans [214], que les solutions stationnaires en forme-produit sont non-existantes pour ce type de réseaux. D'autre part, la conception et l'élaboration d'un modèle de RdPSG coloré non-borné serait d'un grand intérêt pour l'étude des systèmes multi-classes à population infinie. Enfin, il serait bénéfique d'implémenter l'approche proposée dans cette thèse en orientant une étude vers la conception d'un outil qui permet d'automatiser la modélisation des systèmes avec rappel en des réseaux de Petri stochastiques généralisés ordinaires ou colorés, et puis d'intégrer cet outil comme module dans un progiciel d'analyse des RdPSG, tel que le GreatSPN ou le SPNP. D'autre part, il serait intéressant de développer une nouvelle version du GratSPN, dans laquelle les taux des transitions temporisées et les poids des transitions immédiates pourraient être dépendants des couleurs (non-constants).

Annexe

Définition 1 : Un multiensemble sur un ensemble fini et non vide X est une application de X vers $\mathbb{N}$.

Intuitivement, le multiensemble est un ensemble qui peut contenir plusieurs occurrences d'un même élément. Nous notons par $Bag(X)$ *l'ensemble des multiensembles sur* X. Chaque multiensemble $a \in Bag(X)$ peut être représenté par la somme formelle $a = \sum_{x \in X} a(x).x$, où l'entier $a(x)$, positif ou nul, désigne le nombre d'occurrences de l'élément x dans le multiensemble a.

La somme de deux éléments de $Bag(X)$ et le produit d'un élément de $Bag(X)$ par un entier se définissent naturellement.

Définition 2 : Soit a un élément de $Bag(X)$:
- Si b est un élément de $Bag(X)$, alors la somme $a + b$ est un élément de $Bag(X)$ défini par : $a + b = \sum_{x \in X}(a(x) + b(x)).x$
- Si λ est un entier positif, alors le produit $\lambda.a$ est un élément de $Bag(X)$ défini par : $\lambda.a = \sum_{x \in X}(\lambda.a(x)).x$

L'ensemble $Bag(X)$ est muni d'une relation d'ordre qui n'est autre que l'extension naturelle de la relation sur $\mathbb{N}$, ce que formalise la définition suivante.

Définition 3 : Soit $a = \sum_{x \in X} a(x).x$ et $b = \sum_{x \in X} b(x).x$ deux multiensembles sur X. $a \geq b$ ssi : $\forall x \in X$, on a $a(x) \geq b(x)$.

Les réseaux colorés manipulent des tuples de valeurs définis comme suit.

Définition 4 : Soit $C_1, ..., C_k$ des ensembles finis. On appelle un élément de $C_1 \times ... \times C_k$ un tuple et on le note $< c_1, ..., c_k >$.

Bibliographie

[1] J.W. Cohen. Basic problems of telephone traffic theory and the influence of repeated calls. *Philips Telecommunication Review*, 18(2) :49–100, 1957.

[2] F.P. Kelly. Auto-repeat facilities and telephone networks performance. *J. Roy. Statist. Soc.*, B48 :123–132, 1985.

[3] T. Yang and J.G.C. Templeton. A survey on retrial queues. *Queueing Systems*, 2 :201–233, 1987.

[4] G.I. Falin. A survey on retrial queues. *Queueing Systems*, 7 :127–168, 1990.

[5] G.I. Falin and J.G.C. Templeton. *Retrial Queues*. Chapman and Hall, London, 1997.

[6] L. Escudero J. Barcelo and J. Artalejo, editors. *Proceedings of the* 1^{st} *International Workshop on Retrial Queues (WRQ'98)*, volume 7, Madrid, 1998. Top.

[7] J.R. Artalejo. A classified bibliography of research in retrial queues : Progress in 1990-1999. *Top*, 7 :187–211, 1999.

[8] J.R. Artalejo. Accessible bibliography on retrial queues. *Mathematical and Computer Modelling*, 30 :1–6, 1999.

[9] V.G. Kulkarni and H.M. Liang. Retrial queues revisited. In Ed. J.H. Dshalalow, editor, *Frontiers in Queueing : Models and Applications in Science and Engineering*, pages 19–34. CRC Press, Boca Raton, 1997.

[10] J.R. Artalejo and G.I. Falin. Standard and retrial queueing systems : A comparative analysis. *Revista Matematica Complutense*, 15(1) :101–129, 2002.

[11] J.R. Artalejo. Retrial queues with a finite number of sources. *Journal of Korean Mathematical Society*, 35(3) :503–525, 1998.

[12] G.I. Falin and J.R. Artalejo. A finite source retrial queue. *European Journal of Operational Research*, 108(3) :409–424, 1998.

[13] L. Bright and P.G. Taylor. Calculating the equilibrium distribution in level dependent quasi-birth-and-death processes. *Stochastic Models*, 11 :497–525, 1995.

[14] J.R. Artalejo and M. Pozo. Numerical calculation of the stationary distribution of the main multiserver retrial queue. *Annals of Operations Research*, 116 :41–56, 2002.

[15] A. Gomez-corral. Analysis of a single-server retrial queue with quasi-random input and nonpreemptive priority. *Computers and Mathematics with Applications*, 43 :767–782, 2002.

[16] G.I. Falin. A multiserver retrial queue with a finite number of sources of primary calls. *Mathematical and Computer Modelling*, 30(3-4) :33–49, 1999.

[17] H. Ohmura and Y. Takahashi. An analysis of repeated call model with a finite number of sources. *Electronics and Communications in Japan*, 68(6) :112–121, 1985.

[18] P. Tran-Gia and M. Mandjes. Modeling of customer retrial phenomenon in cellular mobile networks. *IEEE Journal on Selected Areas in Communications*, 15 :1406–1414, 1997.

[19] M. Ajmone Marsan, G.D. Carolis, E. Leonardi, R.L. Cigno, and M. Meo. Efficient estimation of call blocking probabilities in cellular mobile telephony networks with customer retrials. *IEEE Journal on Selected Areas in Communications*, 19 :332–346, 2001.

[20] D.J. Houck and W.S. Lai. Traffic modelling and analysis of hybrid fiber-coax systems. *Computer Networks and ISDN Systems*, 30 :821–834, 1998.

[21] H. Li and T. Yang. A single-server retrial queue with server vacations and a finite number of input sources. *European Journal of Operational Research*, 85 :149–160, 1995.

[22] G.K. Janssens. The quasi-random input queueing system with repeated attempts as a model for a collision-avoidance star local area network. *IEEE Transactions on Communications*, 45(3) :360–364, 1997.

[23] M.K. Mehmet-Ali, J.F. Hayes, and A.K. Elhakeem. Traffic analysis of a local area network with star topology. *IEEE Transactions on Communications*, 36 :703–712, 1988.

[24] A.I. Kalmychkov and G.A. Medvedec. Probability characteristics of Markov local-area networks with random-access protocols. *Automatic Control and Computer Science*, 24 :38–45, 1990.

[25] I.I. Khomichkov. Study of models of local area networks with multiple access protocols. *Automation and Remote Control*, 54 :1801–1811, 1993.

[26] A. Aissani and J.R. Artalejo. On the single server retrial queue subject to breakdowns. *Queueing Systems*, 30 :309–321, 1998.

[27] J.R. Artalejo. New results in retrial queueing systems with breakdowns of the servers. *Statistica Neerlandica*, 48(1) :23–36, 1994.

[28] T. Yang and H. Li. The M/G/1 retrial queue with the server subject to starting failures. *Queueing Systems*, 16 :83–96, 1994.

[29] J. Wang, J. Cao, and Q. Li. Reliability analysis of the retrial queue with server breakdowns and repairs. *Queueing Systems*, 38 :363–380, 2001.

[30] J. Roszik and J. Sztrik. The effect of server's breakdowns on the performance of finite-source retrial queueing systems. In *Proc. of 6th International Conference on Applied Informatics*, volume 2, pages 221–230, Eger, Hungary, 2004.

[31] B. Almasi, J. Roszik, and J. Sztrik. Homogeneous finite-source retrial queues with server subject to breakdowns and repairs. In *Proc. of 5th EURO/INFORMS Joint International Meeting*, Istanbul, Turkey, 2003. Mathematical and Computer Modelling, 42 :673-682, 2005.

[32] B. Almasi, J. Roszik, and J. Sztrik. Heterogeneous finite-source retrial queues with server subject to breakdowns and repairs. In *Proc. of XXIII Seminar on Stability Problems of Stochastic Models*, Pamplona, Spain, 2003. Journal of Mathematical Sciences, 132(5) :677-685, 2006.

[33] J. Roszik and J. Sztrik. Performance analysis of finite-source retrial queues with non-reliable heterogeneous servers. In *XXIV Seminar on Stability Problems of Stochastic Models*, Jurmala, Latvia, 2004. Journal of Mathematical Sciences (submitted, 2006).

[34] J. Roszik and J. Sztrik. Finite-source retrial queueing systems with heterogeneous non-reliable servers and different service policies. In *Proc. of XXVIth Hungarian Operational Research Conference*, Gyor, Hungary, 2004.

[35] M. Ajmone Marsan, G. Balbo, G. Conte, S. Donatelli, and G. Franceschinis. *Modelling with Generalized Stochastic Petri Nets*. John Wiley & Sons, New York, 1995.

[36] B.D. Choi and Y. Chang. MAP1,MAP2/M/c retrial queue with the retrial group of finite capacity and geometric loss. *Mathematical and Computer Modelling*, 30 :99–113, 1999.

[37] B.D. Choi and Y. Chang. Single server retrial queues with priority calls. *Mathematical and Computer Modelling*, 30 :7–32, 1999.

[38] B.D. Choi, Y. Chang, and B. Kim. MAP1,MAP2/M/c retrial queue with guard channels and its application to cellular networks. *TOP*, 7(2) :231–248, 1999.

[39] C. Langaris and E. Moutzoukis. Non-pre-emptive priorities and vacations in a multiclass retrial queueing system. *Commu. Statis. Stochastic Models*, 12 :455–472, 1996.

[40] G. Bolch, J. Roszik, and J. Sztrik. Heterogeneous finite-source retrial queues in the analysis of communication systems with CSMA/CD protocols. In *Proc. of the International Conference on Modern Mathematical Methods of Analysis and Optimization of Telecommunication Networks*, pages 39–45, Gomel, Belarus, 2003.

[41] B. Almasi, G. Bolch, and J. Sztrik. Heterogeneous finite-source retrial queues. *Journal of Mathematical Sciences*, 121(5) :2590–2596, 2004.

[42] J. Roszik and J. Sztrik. Retrial queues for performance modelling and evaluation of heterogeneous networks. In *Proc. of Conference on Performance Modelling and Evaluation of Heterogeneous Networks HET-NET'04*, Ilkey, England, 2004.

[43] B. Almasi, J. Roszik, and J. Sztrik. Multiserver retrial queues with finite number of heterogeneous sources. In *Proc. of 6th International Conference on Applied Informatics*, volume 2, pages 19–26, Eger, Hungary, 2004.

[44] B. Pourbabai. Analysis of a G/M/k/o queueing loss system with heterogeneous servers and retrials. *International Journal of Systems Science*, 18 :985–992, 1987.

[45] G. Chiola, D. Dutheillet, G. Franceschinis, and S. Haddad. On Well-Formed Colored nets and their symbolic reachability graph. In *In Proc. of the 11th International Conference on Application and Theory of Petri nets*, pages 106–125, Paris, France, 1990.

[46] G. Chiola, D. Dutheillet, G. Franceschinis, and S. Haddad. Stochastic well-formed colored nets for symmetric modelling applications. *IEEE Transactions on Computers*, 42(11) :1343–1360, 1993.

[47] A.K. Erlang. Solution of some problemes in the theory of probabilities of significance in automatic telephone exchanges. *Post Office Electrical Engineer's Journal*, 10 :189–197, 1917.

[48] T. Engset. The probability theory for computing the number of switching equipments in automatic telephone exchanges. *E.T.Z.*, 31 :304–305, 1918.

[49] A. Aissani. Propriétés de second ordre des modèles d'attente, application à la conception des systèmes de production. In *Proceedings de la 4ième Rencontre de Recherche Opérationnelle*, pages 5–10, USTHB, Alger, Octobre 1996.

[50] J.A. Buzacott and J.G. Shanthikumar. Design of manufacturing systems using queueing models. *Queueing Systems*, 12 :135–214, 1992.

[51] J. Pellaumail, P. Boyer, and P. Leguesdron. *Réseaux ATM et P-simulation.* 1994.

[52] E. Brockmeyer, H.L. Halstrom, and A. Jensen. The life and works of a.k. erlang. *Trans. Dan. Acad. Techn. Sci. No. 2, Copenhagen, 138*, 1948.

[53] L. Kleinrock. *Queueing systems, Volume I : Theory.* John Wiley, New York, 1975.

[54] J.H. Dshalalow. *Advances in queueing : Theory, Methods and Open problems.* CRC Press, Boca Raton, 1995.

[55] L. Kosten. On the influence of repeated calls in the theory of probabilities of blocking. *De Ingenieur*, 59(1), 1947.

[56] R.I. Wilkinson. Theories for toll traffic engineering in the U.S.A. *The Bell System Technical Journal*, 35(2) :421–514, 1956.

[57] A.A.B. Pritsker. Application of multichannel queueing results to the analysis of conveyor systems. *Ind. Engng*, 17 :14–21, 1966.

[58] D. Sonderman. An analytical model for recirculating conveyors with stochastic input and outputs. *Int. J. Prod. Res.*, 20 :591–605, 1982.

[59] W.M. Nawijn. *Stochastic conveyor systems.* Ph.d. dissertation, Twente University of Technology, Nschede, 1983.

[60] B. Pourbabai. The effect of variability in a closed loop material conveyor system. *International Journal on Mathematical Modeling*, 7 :185–196, 1986.

[61] B. Pourbabai. Performance modeling of a closed loop material handling system. *Eur. J. Opr. Res.*, 32 :340–352, 1987.

[62] J.R. Artalejo and A. Gómez-Corral. Channel idle periods in computer and telecommunication systems with customer retrials. *Telecommunication Systems*, 24 :29–46, 2003.

[63] E. Onur, H. Delic, C. Ersoy, and M.U. Caglayan. Measurement-based replanning of cell capacities in GSM networks. *Computer Networks*, 39 :749–767, 2002.

[64] J. Roszik, J. Sztrik, and C. Kim. Retrial queues in the performance modeling of cellular mobile networks unsing MOSEL. *International Journal of Simulation : Systems, Science and Technology*, 1-2 :38–47, 2005.

[65] J. Riordan. *Stochastic Service Systems.* Wiley, New York, 1962.

[66] A.M. Aleksandrov. A queueing system with repeated orders. *Engineering Cybernetics*, 12(3) :1–4, 1974.

[67] P. Le Gall. Les appels répétés et les méthodes de calcul et d'observation du trafic. In *2ème conférence bulgare sur les télécommunications*, Varna, October 1976.

[68] A. Ephremides. Analysis of protocols of multiple access. In *Proc. of the International Seminar on Modeling and Peformance Evaluation Methodology*, Berlin, 1983.

[69] B. Pourbabai. A random access telecommunication system. *Journal of Information Processing and Cybernetics*, 24 :613–625, 1988.

[70] P. Skelly, M. Schwartz, and S. Dixit. A histogram-based model for video traffic behavior in an ATM multiplexer. *IEEE Trans. on Networking*, 1 :446–459, 1993.

[71] A. Aissani. A survey on retrial queueing models. In *Journées de Statistiques Appliquées*, pages 1–11, Alger, 16-18 Avril 1994.

[72] G. Bolch, S. Greiner, H. De Meer, and K.S. Trivedi. *Queueing Networks and Markov Chains*. New York, 1998.

[73] L. Kosten. *Stochastic Theory of Service Systems*. Pergamon Press, Oxford, 1973.

[74] A. Elldin. Approach to the theoretical description of repeated call attempts. *Ericsson Technics*, 23(3) :p. 345, 1967.

[75] J. Keilson, J. Cozzolino, and H. Young. A service system with unfilled requests repeated. *Oper. Res.*, 16 :1126–1137, 1968.

[76] G.L. Jonin and J.J. Sedol. Investigation of telephone systems in the case of repeated calls. *Latvian Mathematical Yearbook*, 7 :71–82, 1970.

[77] G.I. Falin. Double channel queueing system with repeated calls. Technical Report Paper No. 4221-84, All-Union Institute for Scientific and Technical Information, Moscow, USSR, 1984.

[78] T. Hanschke. Explicit formulas for the characteristics of the M/M/2/2 queue with repeated attempts. *Journal of Applied Probability*, 24 :486–494, 1987.

[79] J.R. Artalejo. Stationary analysis of the characteristics of the M/M/2 queue with constant repeated attempts. *Operations Research*, 33(2) :83–95, 1996.

[80] B.D. Choi, Y.C. Kim, and Y.W. Lee. The M/M/c retrial queue with geometric loss and feedback. *Computers and Mathematics with Applications*, 36 :41–52, 1998.

[81] C.E.M. Pearce. Extended continued fractions, recurrence relations and two-dimensional Markov processes. *Advances in Applied Probability*, 21 :357–375, 1989.

[82] G.I. Falin. Multichannel queueing systems with repeated calls under high intensity of repetitions. *Journal of Information Process in Cybernetics EIK*, 23 :37–47, 1987.

[83] S.N. Stepanov. Markov models with retrials : the calculation of stationary performance measures based on the concept of truncation. *Mathematical and Computer Modelling*, 30 :207–228, 1999.

[84] M.F. Neuts and B.M. Rao. Numerical investigation of a multiserver retrial model. *Queueing Systems*, 7(2) :169–190, 1990.

[85] V.V. Anisimov and J.R. Artalejo. Approximation of multiserver retrial queues by means of generalized truncated models. *Top*, 10 :51–66, 2002.

[86] M.F. Neuts. Markov chains with applications in queueing theory, which have a matrix-geometric invariant probability vector. *Advances in Applied Probability*, 10 :185–212, 1978.

[87] V.A. Malyshev. Homogeneous random walks on the product of a finite set and a half-line. In *In : Probabilistic Methods of Research*, pages 5–13, Moscow State University, 1972.

[88] N. Deul. Stationnary conditions for multi-server queueing systems with repeated calls. *Elektr. Informationsverarbeit. Kybernet.*, 16 :607–613, 1980.

[89] G.I. Falin. On sufficient conditions for ergodicity of multichannel queueing systems with repeated calls. *Advances in Applied Probability*, 16(2) :447–448, 1984.

[90] G.I. Falin. On ergodicity of multichannel queueing systems with repeated calls. *Sov. J. Comput. Syst. Sci.*, 25(1) :60–65, 1987.

[91] A.A. Fredericks and G.A. Reisner. Approximations to stochastic service systems with an application to a retrial model. *Bell system Technical Journal*, 58(3) :557–576, 1979.

[92] A. Gomez-corral and M.F. Ramalhoto. The stationary distribution of a markovian process arising in the theory of multiserveur retrial queueing systems. *Mathematical and Computer Modelling*, 30 :141–158, 1999.

[93] Y.C. Kim. On M/M/3/3 retrial queueing system. *Honam Mathematical Journal*, 17 :141–147, 1995.

[94] J.R. Artalejo and A. Gómez-Corral. Waiting time in the M/M/c queue with finite retrial group. *Bulletin of Kerala Mathematics Association*, 2 :1–17, 2005.

[95] G.I. Falin. Investigation of weakly loaded switching systems with repeated calls. *Engineering Cybernetics Rev.*, 19(3) :69–73, 1981.

[96] S.N. Stepanov. Asymptotic formulae and estimations for probabilistic characteristics of full-available group with absolutely persistent subscribers. *Problems of Control and Information Theory*, 12(5) :p. 361, 1983.

[97] B.S. Greenberg and R.W. Wolff. An upper bound on the performance of queues with returning customers. *Journal of Applied Probability*, 24 :466–475, 1987.

[98] G.I. Falin. Calculation of probability characteristics of a multiline system with repeat calls. *Computational Mathematics and Cybernetics*, 1 :43–49, 1983.

[99] M.F. Neuts. *Matrix-Geometric Solutions in Stochastic Models : An Algorithmic Approach*. The John Hopkins University Press, Baltimore, 1981.

[100] R. Syski. *Introduction to Congestion in Telephone Systems*. Elsevier Science Publishers, Amsterdam, 1986.

[101] R.B. Cooper. *Introduction to Queueing Theory*. Edward Arnold, 1981.

[102] M. Ajmone Marsan, G. De Carolis, E. Leonardi, R. Lo Cigno, and M. Meo. An approximate model for computation of blocking probabilities in cellular networks with repeated calls. *Telecommunication Systems*, 15 :53–62, 2000.

[103] J. Sztrik. Asymptotic analysis of finite-source atm system. In *Proceedings of 5th IFIP Workshop on Performance Modelling and Evaluation of ATM networks*, volume 85, pages 1–12, Ilkey, England, 1997.

[104] H. Takagi. *Queueing Analysis - A foundation of Performance Evaluation : Finite Systems*, volume 2. Elsevier Science Publishers B.V., North-Holland, Amsterdam, 1993.

[105] Y.N. Kornyshev. Design of a fully accessible switching system with repeated calls. *Telecommunications*, 23 :46–52, 1969.

[106] J.R. Artalejo and A. Gomez-Corral. Information theoretic analysis for queueing systems with quasi-random input. *Mathematical and Computer Modelling*, 22(3) :65–76, 1995.

[107] J.R. Artalejo, V. Rajagopalan, and R. Sivasamy. On finite markovian queues with repeated attempts. *Inverstigacion Operativa*, 9 :83–94, 2000.

[108] G.I. Falin and A. Gomez Corral. On a bivariate Markov process arising in the theory of single-server retrial queues. *Statistica Neerlandica*, 54 :67–78, 2000.

[109] W.J. Stewart. *Intoduction to the numerical solution of Markov chains.* Princeton University Press, 1994.

[110] I.I. Khomichkov. Calculation of the characteristics of local area network with p-persistent protocol of multiple random access. *Automation and Remote Control*, 56 :208–218, 1995.

[111] C. Tropper. *Local Computer Network Technologies.* Academic Press, New York, 1981.

[112] I.N. Kovalenko, N.Y. Kuznetsov, and P.A. Pegg. *Mathematical Theory of Reliability of Time Dependent Systems with Practical Applications.* John Wiley and Sons, Chichester, 1997.

[113] N. Ravichandran. *Stochastic Methods in Reliability Theory.* John Wiley and Sons, New York, 1990.

[114] A. Aissani. Unreliable queueing systems with repeated orders. *Microelectron. Reliab.*, 33(14) :2093–2106, 1993.

[115] A. Aissani. A retrial queue with redundancy and unreliable server. *Queueing Systems*, 17 :431–449, 1995.

[116] V.G. Kulkarni and B.D. Choi. Retrial queues with server subject to breakdowns and repairs. *Queueing Systems*, 7(2) :191–208, 1990.

[117] C.H. Yoon and C.K. Un. Performance of personal radio telephone system with and without guard channels. *IEEE J. Select Areas Commun.*, 11 :911–917, 1993.

[118] V.G. Kulkarni. On queueing systems with retrials. *J. Appl. Probab.*, 20 :380–389, 1983.

[119] G.I. Falin. The influence of inhomogeneity of the composition of subscribers on the functionning of telephone systems with repeated calls. *Eng. Cybernet. Rev.*, 21(6) :21–25, 1983.

[120] B. Pourbabai. Markovian queueing systems with retrials and heterogeneous servers. *Computers and Mathematics with Applications*, 13 :917–923, 1987.

[121] B. Pourbabai. Asymptotic analysis of G/G/k queueing-loss system with retrials and heterogeneous servers. *International Journal of Systems Science*, 19(6) :1047–1052, 1988.

[122] K. Begain, G. Bolch, and H. Herold. *Practical Performance Modeling - Application of the MOSEL Language.* Kluwer Academic Publisher, Boston, 2001.

[123] K. Begain, J. Barner, G. Bolch, and A.I. Zreikat. The performance and reliability modelling language MOSEL and its application. *International Journal of Simulation*, 3 :66–80, 2003.

[124] C.A. Petri. *Kommunikation mit Automaten.* Phd dissertation, Institut für Instrumentelle Mathematik, University of Bonn, West Germany, 1962. (traduction anglaise : Rapport technique R5AC-TR-65-377, vol. 1, suppl. 1, Griffiths air force base, New York, 1966).

[125] M. Hack. Analysis of production schemata by Petri nets. Technical report, MAC TR.94, MIT, February 1972.

[126] A.W. Holt and F. Commoner. Events and conditions, record of the project MAC. In *Proc. of the Conference on ACM*, pages 3–52, New York, 1970.

[127] J. Billington, M. Diaz, and G. Rozenberg. Application of Petri nets to communication networks. In Springer Verlag Lecture Notes in Computer Science, editor, *Advances in Petri nets*, volume 1605, 1999.

[128] M. Diaz. *Les réseaux de Petri - Modèles Fondamentaux.* Hermès Science Publications, Paris, 2001.

[129] C. Ramchandani. *Analysis of Asynchronous Concurrent Systems by Timed Petri nets.* Thèse de doctorat, Massachusetts Institute of Technology, Department of Electrical Engineering, Cambridge, Massachusetts, 1974.

[130] M.K. Molloy. *On the integration of delay and throughput measures in distributed operating models.* Ph.d. thesis, University of California, Los Angeles, 1981.

[131] M.K. Molloy. Performance analysis using stochastic Petri nets. *IEEE Transaction on Computers*, C-31(9) :913–917, Sept. 1982.

[132] S.O. Natkin. *Les réseaux de Petri stochastiques et leur application à l'évaluation des systèmes informatiques.* Ph.d. thesis, Conservatoire National des Arts et Metiers (CNAM), Paris, June 1980.

[133] M. Ajmone Marsan, G. Balbo, and G. Conte. A class of generalized stochastic petri nets for the performance evaluation of multiprocessor systems. *ACM Transactions on Computer Systems*, 2(2) :93–122, May 1984.

[134] M. Ajmone Marsan and V. Signore. Timed Petri nets performance models for fiber optic LAN architectures. In *Proc. 2nd International Workshop on Petri Nets and Performance Models (PNPM'87)*, pages 66–74, 1987.

[135] M. Ajmone Marsan, M. Meo, and M. Sereno. GSPN analysis of dual-band mobile telephony networks. In IEEE Computer Society Press, editor, *Proc. of the International Workshop on Petri Nets and Performance Models (PNPM'99)*, pages 54–63, 1999.

[136] R.Y. Al-Jaar and A.A. Desrochers. Modeling and analysis of transfer lines and production networks using generalized stochastic Petri nets. In *Proc. of UPCAEDM Conference*, pages 12–21, Atlanta, June 1988.

[137] C. Lindemann, G. Ciardo, R. German, and G. Hommel. Performability modeling of an automated manufacturing system with deterministic and stochastic Petri nets. In IEEE Press, editor, *Proc. IEEE Inter. Conf. on Robotics and Automation*, pages 576–581, Atlanta, May 1993.

[138] G.W. Brams. *Réseaux de Petri : Théorie et Pratique*, volume 1. Masson, Paris, 1983.

[139] M. Zouaoui. *Définition d'un langage de type logique temporelle pour la spécification et l'évaluation des performances.* Thèse de doctorat d'etat, Université Pierre et Marie Curie, Paris 6, Mars 1998.

[140] B. Baynat. *Théorie des files d'attente - des chaînes de Markov aux réseaux à forme produit.* Hermès Science Publications, Paris, 2000.

[141] M. Ajmone Marsan. Stochastic Petri nets : An elementary introduction. Technical report, Université de Milan, Italie, 1990.

[142] A. Aissani. Les chaînes de Markov, Cours de Modélisation et Simulation. Technical report, USTHB, 2006.

[143] S. Lipschutz. Cours et problèmes de probabilités. Technical report, Université de Temple, 1981.

[144] G. Florin and S. Natkin. Les réseaux de Petri stochastiques. *Technique et Science Informatique*, 4(1) :143–160, 1985.

[145] M. Granda, J. M. Drake, and J. A. Gregori. Performance evaluation of parallel systems by using unbounded generalized stochastic Petri nets. *IEEE Transactions on Software Engineering*, 18(1), 1992.

[146] S. Donatelli G. Chiola and G. Franceschinis. GSPN versus SPN : What is the actual role of immediate transitions ? In IEEE Computer Society, editor, *In Proceedings 4th International Workshop on Petri nets and Performance Models (PNPM'91)*, pages 20–31, Melbourne, Australia, 1991.

[147] M. Ajmone Marsan, G. Balbo, A. Bobbio, G. Ghiola, and A. Cumani. Effect of execution policies on the semantics and analysis of stochastic petri nets. *IEEE Transactions on Software Engeneering*, 15(7) :832–846, July 1989.

[148] S. Haddad, P. Moreaux, and G. Chiola. Distributions de Cox et Phase-type dans les réseaux de Petri stochastiques : Une méthodologie efficace de résolution. *Operations Research*, 32(3) :289–323, 1998.

[149] M. Ajmone Marsan, G. Balbo, G. Ciardo, and G. Conte. A software tool for the automatic analysis of generalized stochastic Petri net models. In ed.) (Inria, D. Potier, editor, *In Proceedings of International Conference on Modelling Techniques and Tools for Performance Analysis*, pages 155–170. Elsevier Science Publishers B. V. (North Holland), 1985.

[150] G. Balbo and M. Silva. *Performance Models for Discrete Event Systems with synchronisations : Formalisms and Analysis Techniques - MATCH Advanced Schools.* Univeristé de Zaragoza (Espagne), September 1998.

[151] C. Dutheillet. *Symétrie dans le réseaux colorés - Définition, analyse et application à l'évaluation des performances.* Thèse de doctorat, Université Pierre et Marie Curie, Paris VI, Mars 1992.

[152] A.A. Desrochers and R.Y. Al-Jaar. *Applications of Petri Nets in Manufacturing Systems : Modeling, Control and Performance Analysis.* The Institute of Electrical and Electronics Engineers, INC. New York, 1995.

[153] A. Zenie. *Les réseaux de Petri stochastiques colorés : Application à l'analyse des systèmes répartis en temps réel.* Thèse de doctorat d'etat, Université Pierre et Marie Curie, Paris VI, Octobre 1987.

[154] J.D.C. Little. A proof of the queueing formula : $L = \lambda.W$. *Operations Research*, 9 :383–387, 1961.

[155] G. Chiola. A simulation framework for timed and stochastic petri nets. *International Journal of Computer Simulation, Special issue on simualtion of multiple processor networks*, 1991.

[156] B.R. Haverkort. Approximate performability and dependability modelling using generalized stochastic Petri nets. *Performance Evaluation*, 18 :61–78, 1993.

[157] S. Donatelli. Superposed generalized stochastic Petri nets : Definition and efficient solution. In Springer Verlag, editor, *Proc. of 15th International Conference an Application and Theory of Petri Nets*, pages 258–277, Zaragoza, Spain, 1994.

[158] P. Ziegler and H. Szczerbicka. A structure based decomposition approach for GSPN. In *Proc. of the 6th Int. Workshop on Petri Nets & Performance Models PNPM'95*, Durham, North-Carolina USA, October 1995.

[159] C. J. Pérez-Jiménez and J. Campos. On state space decomposition for the numerical analysis of stochastic petri nets. In IEEE Computer Society, editor, *Proceedings of the 8th Int. Workshop on Petri Nets & Performance Models PNPM'99*, 1999.

[160] G. Chiola, J. Campos, J.M. Colom, M. Silva, and C. Anglano. Operational analysis of timed Petri nets and applications to the computation of performance bounds. In IEEE-CS Press, editor, *In Proc. 5th International Workshop on Petri nets and Performance Models*, Toulouse, 1993.

[161] Z. Liu. Performance analysis of stochastic timed Petri nets using linear programming approach. Technical Report 2642, INRIA, Centre Sophia-Antipolis, Août 1995.

[162] P. Buchholz. Hierarchical markovian models : symmetries and reduction. In R. Pooley and J. Hillston editors, editors, *Computer Performance Evaluation : Modelling Techniques and Tools*, pages 305–319. Edinburgh University Press, 1992.

[163] P. Buchholz. Aggregation and reduction techniques for hierarchical GCSPNs. In *Proc. of the 5th Int. Workshop on Petri Nets and Performance Models*, pages 216–225. IEEE Computer Society Press, 1993.

[164] A. Bobbio, A. Puliafito, M. Telek, and S. Trivedi. Recent developpements in stochastic petri nets. *Journal of circuits, systmes and computers*, 8(1) :119–158, 1998.

[165] S. Haddad, P. Moreaux, M. Sereno, and M. Silva. Structural characterization and behavioural properties of product form stochastic Petri nets. In *Proc. of the 22th International Conference on Application and theory of Petri Nets*, NewCastle Upon Tine, June 2001.

[166] S. Donatelli and M. Sereno. On the product-form solution for stochastic Petri nets. In Springer Verlag Lecture Notes in Computer Science, editor, *Proc. International Conference an Application and Theory of Petri Nets*, volume 616, pages 154–172, 1993.

[167] M. Sereno and G. Balbo. Computational algorithms for product form solution stochastic Petri nets. In *Proc. of 5th Int. Workshop on Petri Nets and Performance Models (PNPM'93)*. IEEE Computer Society, 1993.

[168] D.D. Deavours and W.H. Sanders. An efficient disk-based tool for solving very large Markov models. In M. Calzarossa IN R. Marie, B. Plateau and G. Rubino editors, editors, *Proc. of the 9th Int. Conf. on Modelling Techniques and Tools for Computer Performance Evaluation*, volume LNCS 1245. LNCS, 1997.

[169] D.D. Deavours and W.H. Sanders. "on-the-fly" solution techniques for stochastic Petri nets and extensions. In IEEE Computer Society Press, editor, *Proc. of the 7h Int. Workshop on Petri Nets and Performance Models PNPM'97*, pages 132–141, St. Malo, France, June 1997.

[170] G. Ciardo and M. Tilgner. On the use of Kronecker operators for the solution of generalized stochastic Petri nets. Technical Report 96-35, Institute for Computer Applications in Science and Engineering, Hampton, 1996.

[171] G. Ciardo and A.S. Miner. A data structure for the efficient Kronecker solution of GSPNs. In IEEE Computer Society, editor, *Proceedings of the 8th Int. Workshop on Petri Nets & Performance Models PNPM'99*, 1999.

[172] S.C. Allmaier, M. Kowarschik, and G. Horte. State space construction and steady-state solution of GSPNs on a shared-memory multiprocessor. In IEEE Computer-Society, editor, *In 7-th International Conference on Petri nets and Performance Models (PNPM'97)*, pages 112–121, 1997.

[173] S. Caselli, G. Conte, and P. Marenzoni. Parallel state space exploration for GSPN models. In Springer Verlag LNCS, editor, *In proc. 16th Int. Conf. on Application and Theory of Petri Nets*, volume 935, pages 181–200, 1995.

[174] P. Marenzoni, S. Caselli, and G. Conte. Analysis of large GSPN models : A distributed solution tool. In *IEEE Int. Workshop on Petri Nets Performance Models*, 1997.

[175] W.J. Stewart. *Numerical methods for computing of finite irreductible Markov chains.* USA, 1997.

[176] I.S. Duff and J.K. Reid. The design of MA48 : A code for the direct solution of sparse unsymmetric linear systems of equations. *ACM Transactions on Mathematical Software*, 22(2) :p. 187, 1996.

[177] K. Jensen. Coloured Petri nets and the invariant method. *Theoretical Computer Science*, 14 :317–336, 1981.

[178] G. Chiola, G. Bruno, and T. Demaria. Introducing a color formalism into generalized stochastic Petri nets. In *In Proc. of the 9th European Workshop on Application and Theory of Petri nets*, Venezia, Italy, 1988.

[179] D. Dutheillet and S. Haddad. Aggregation of states in colored stochastic Petri nets : Application to a multiprocessor architecture. In IEEE-CS Press, editor, *Proc. 3rd International Workshop on Petri nets and Performance Models*, pages 40–49, Kyoto, Japan, 1989.

[180] J.G. Kemeny and J.L. Snell. *Finite Markov chains.* V. Nostrand, Princeton, NJ, 1960.

[181] G. Chiola and G. Franceschinis. Colored GSPN models and automatic symmetry detection. In *In Proc. of the 3rd Int. Workshop of Petri Nets and Performance Models*, Kyoto, Japan, 1989.

[182] J.A. Carrasco. Automated construction of compound markov chains from generalized stochastic high-level Petri nets. In *Proceedings of the 3rd Int. Workshop on Petri Nets and Performance Models*, pages 93–102, Kyoto, Japan, 1989.

[183] S. Haddad and P. Moreaux. Evaluation of high level petri nets by means of aggregation and decomposition. In *Proc. of the 6th International Conference on Petri Nets and Performance Models (PNPM'95)*, pages 11–20. IEEE Computer Society, 1995.

[184] P. Moreaux. Combinaison de l'agrégation et de la décomposition tensorielle dans les réseaux de petri stochastiques bien formés. *RAIRIR*, 5 :61–76, March 1997.

[185] S. Haddad and P. Moreaux. Asynchronous composition of high level petri nets : A quantitative approach. In *In Proc. of the 17th International Conference on Application and Theory of Petri Nets*, volume 1091, pages 192–211, Osaka, Japan, 1996. Lecture Notes in Computer Science, Springer-Verlag.

[186] K. Djemame. Queueing networks versus Petri nets. Two performance case studies in distributed simulation. Technical report, Computing Science Department, University of Glasgow, Scotland, 1997.

[187] M. Vernon, J. Zahorjan, and E. Lazowska. A comparison of performance Petri nets and queueing network models. In *Proc. of 3rd Int. Workshop on Modelling Techniques and Performance Evaluation*, pages 181–192, Paris, France, 1987.

[188] S. Balsamo and V. De Nitto-Persone. A survey on product-form queueing networks with blocking and their equivalences. *Annals of Operation Research*, 48 :31–61, 1994.

[189] M. Gribaudo and M. Sereno. Gspn semantics for queueing networks with blocking. In IEEE-CS Press, editor, *Proc. of the 7th International Workshop on Petri nets and performance models*, Saint Malo, France, June 1997.

[190] W. Reisig and G. Rozenberg, editors. *Lectures on Petri Nets I : Basic Models*, number 1491 in Advances in Petri nets. LNCS, Springer-Verlag, June 1998.

[191] W. Reisig and G. Rozenberg, editors. *Lectures on Petri Nets I : Applications*, number 1492 in Advances in Petri nets. LNCS, Springer-Verlag, June 1998.

[192] M. Gribaudo and M. Sereno. On the use of structural petri net analysis for studying product form equilibrium distributions of queueing networks with blocking. In Springer Verlag Lecture Notes in Computer Science, editor, *Proc. of the 19th International Conference on Application and Theory of Petri nets (ICATPN'98)*, volume 1420, pages 247–265, Lisbon, Portugal, June 1998.

[193] G. Balbo, S.C. Bruell, and S. Ghanta. Combining queueing network and generalized stochastic petri net models for the solution of complex models of system behavior. *IEEE Transactions on Computers*, 37(10) :1251–1268, 1988.

[194] N. Gharbi and M. Ioualalen. GSPN analysis of retrial systems with servers breakdowns and repairs. *Applied Mathematics and Computation*, 174(2) :1151–1168, 2006.

[195] N. Gharbi and M. Ioualalen. Performance analysis of retrial queueing systems using Generalized Stochastic Petri nets. In *Theory and Practice of Timed Systems (TPTS'02)*, Grenoble, France, 2002. Electronic Notes in Theoretical Computer Science, 65(6), 2002.

[196] N. Gharbi and M. Ioualalen. GSPN analysis of finite-population retrial queueing systems with server vacations. In *Fourth International Symposium on Programming and Systems (ISPS'99)*, Alger, 1999.

[197] N. Gharbi and M. Ioualalen. Performance evaluation of multi-server queues with station and server vacations. In *International Industrial Simulation Conference ISC'04*, pages 397–401, Malaga, Spain, June 2004.

[198] J.R. Artalejo, A. Gomez-Corral, and M.F. Neuts. Analysis of multiserver queues with constant retrial rate. *European Journal of Operational Research*, 135 :569–581(123–135), 2001.

[199] J.R. Artalejo. On the single server retrial queue with batch arrivals. *The Indiana Journal of Statistics*, 66(1) :140–158, 2004.

[200] G.I. Falin. On a multiclass batch arrival retrial queue. *Adv. Appl. Prob.*, 20 :483–487, 1988.

[201] S.A. Grishechkin. Multiclass batch arrival retrial queues analyzed as branching processes with immigration. *Queueing Systems*, 11 :395–418, 1992.

[202] G. Chiola, C. Franceschinis, R. Gaeta, and M. Ribaudo. Greatspn 1.7 : Graphical editor and analyzer for timed and stochastic petri nets. *Performance Evaluation*, 24(1-2) :47–68, 1995.

[203] G. Ciardo, J. Muppala, and K.S. Trivedi. SPNP : Stochastic Petri net package. In IEEE Computer Society, editor, *In proceedings 3rd International Workshop on Petri Nets and Performance Models (PNPM'89)*, pages 142–151, (Kyoto,Japan), December 1989.

[204] E.M. Thurner. TOMSPIN - A tool for modelling with stochastic Petri nets and performance models. In IEEE Computer Society Press, editor, *Proc. of International Workshop on Petri Nets and Performance Models (PNPM'95)*, pages 218–219, 1995.

[205] W.H. Sanders, W.D. Obal, M.A. Qureshi, and F.K. Widjanarko. The UltraSAN : Modeling environment. *Performance Evaluation*, 1995.

[206] C. Lindemann. DSPNexpress : a software package for the efficient solution of deterministic and stochastic petri nets. *Performance Evaluation*, 1995.

[207] G. Chiola, M. Ajmone Marsan, G. Balbo, and G. Conte. Generalized stochastic petri nets : A definition at the net level and its implications. *IEEE Transactions on Software Engineering*, 19(2) :89–107, 1993.

[208] A.N. Dudin and V.I. Klimenok. Queueing system BMAP/G/1 with repeated calls. *Mathematical and Computer Modelling*, 30 :115–128, 1999.

[209] J.E. Diamond and A.S. Alfa. Matrix analytical methods for multi-server retrial queues with buffers. *Top*, 7 :249–266, 1999.

[210] A.N. Dudin and V.I. Klimenok. A retrial BMAP/SM/1 system with linear repeated requests. *Queueing systems*, 34 :47–66, 2000.

[211] J.E. Diamond and A.S. Alfa. The MAP/PH/1 retrial queue. *Stochastic Models*, 14 :1151–1177, 1998.

[212] M. Vazquez. A retrial model in a nonstationary regime. *Top*, 4 :121–133, 1996.

[213] V.V. Anisimov. Averaging methods for transient regimes in overloading retrial queueing systems. *Mathematical and Computer Modelling*, 1999.

[214] J.R. Artalejo and A. Economou. On the non-existence of product-form solutions for queueing networks with retrials. *Electronic modeling*, 27 :13–19, 2005.

Résumé

Les systèmes avec rappel (ou systèmes avec appels répétés) apparaissent dans beaucoup de domaines, tels que les réseaux informatiques et les télécommunications. La plupart des modèles de files d'attente avec rappel étudiées supposent que le flux d'entrée est homogène du point de vue des caractéristiques des clients, telles que les distributions des temps d'inter-arrivée, des temps de service et des temps de rappel. Cependant, en pratique, ces caractéristiques peuvent varier pour les différents types de clients. Ceci nous conduit aux systèmes multi-classes avec rappel, qui apparaissent dans divers domaines d'applications, tels que les réseaux mobiles cellulaires. Cependant, les modèles multi-classes sont beaucoup plus difficiles à analyser mathématiquement que les modèles à classe unique. Ainsi, les résultats explicites sont disponibles seulement pour certains modèles particuliers, avec une population infinie et une station de service qui comprend un serveur unique ou plusieurs serveurs homogènes fiables. En fait, les modèles avec serveurs hétérogènes sont très peu étudiés, malgré leur importance dans les systèmes réels. D'ailleurs, on ne trouve dans la littérature que quelques références dans lesquelles les clients sont supposés être homogènes et les serveurs sont hétérogènes. Pour ce qui est des modèles avec rappel, clients et serveurs hétérogènes à la fois, aucune étude n'a été faite à ce jour, et ce ni pour le cas de serveurs fiables ni serveurs non-fiables. Ceci est dû essentiellement à la complexité de l'analyse de ces modèles. Par ailleurs, certains composants de ces systèmes sont souvent sujets à des pannes aléatoires, qui peuvent avoir un impact négatif non négligeable sur les performances du système. Ainsi, nous proposons dans cette thèse, une approche de modélisation et d'évaluation des performances et de la fiabilité des systèmes multi-classes avec rappel et source finie, à l'aide du modèle des réseaux de Petri stochastiques généralisés colorés (RdPSGC). L'avantage de l'utilisation des RdPSGC est le fait qu'ils constituent un modèle graphique et mathématique de haut-niveau, approprié pour la description et l'analyse de systèmes complexes avec des composants hétérogènes, à l'aide de modèles compacts et précis. D'autre part, ce modèle stochastique coloré permet une analyse des propriétés qualitatives en utilisant des algorithmes efficaces et offre aussi un riche moyen d'expression des indices de performance et de fiabilité exacts en régime stationnaire.

Mots Clés : Systèmes multi-classes avec rappel, Source finie, Disciplines de panne, Modélisation, Réseau de Petri stochastique généralisé coloré, Indices de performance et de fiabilité.

Abstract

Retrial systems (or systems with repeated calls) arise in telecommunication and computer networks areas. Most retrial queueing models assume that the input flow is homogeneous from the point of view of customers characteristics such as inter-arrival time, service time and retrial time distributions. However, in practice, these characteristics may differ widely for different customers types. This leads us to multi-class retrial systems, which arise in various practical areas as digital cellular mobile networks. However, multiclass retrial models are far more difficult for mathematical analysis than single class models. So, explicit results are available only in few special queueing models, and almost works assumed that service station consists of one single server or multiple homogeneous reliable servers and the population size is infinite. Retrial models with heterogeneous servers are still an interesting topic, even in homogeneous customers case. In fact, we have found in the literature no paper on retrial systems with several classes of customers and multiple heterogeneous servers, because of complexity of the models analysis. On the other hand, some components of these systems are often subject to random breakdowns, which may have heavy negative impact on the system performances. Hence, we propose, in this thesis, an approach for modeling and analyzing finite-source retrial systems with several customers'classes and servers'classes, using the colored generalized stochastic Petri nets (CGSPNs) model. The interest of the use of CGSPNs is the fact that it is a high-level graphical and mathematical model, appropriate for describing and analyzing performances of systems with heterogeneous components, by means of compact and detailed models. On the other hand, this colored stochastic model allows verification of qualitative properties using efficient algorithms and offers the possibility of generating formulas for exact performance and reliability indices.

Keywords : Multiclass retrial systems, Finite source, Breakdowns disciplines, Modeling, Colored Generalized Stochastic Petri nets, Performance and reliability indices.