Université de Montréal

Simulation de centres de contacts

par
Eric Buist

Département d'informatique et de recherche opérationnelle
Faculté des arts et des sciences

Thèse présentée à la Faculté des études supérieures
en vue de l'obtention du grade de Philosophiæ Doctor (Ph.D.)
en informatique

Janvier, 2009

Université de Montréal
Faculté des études supérieures

Cette thèse intitulée:

Simulation de centres de contacts

présentée par:

Eric Buist

a été évaluée par un jury composé des personnes suivantes:

Fabian Bastin,	président-rapporteur
Pierre L'Écuyer,	directeur de recherche
Patrice Marcotte,	membre du jury
Benoît Montreuil,	examinateur externe
Christopher R. Bryant,	représentant du doyen de la FES

Thèse acceptée le: 12 mai 2009

RÉSUMÉ

Un *centre de contacts* est un ensemble de ressources formant une interface entre un organisme et ses usagers. Plusieurs entreprises disposent d'un tel centre pour offrir des services à leurs clients tandis que des organismes gouvernementaux en possèdent pour les services de renseignements, d'urgence, etc. Les centres de contacts revêtent ainsi une grande importance économique, d'où le besoin de les analyser et d'en optimiser le rendement. Cette analyse consiste à construire un modèle du centre de contacts et à l'utiliser pour évaluer la performance du centre pour plusieurs configurations.

Avec l'accroissement de la complexité de ces centres, la simulation devient progressivement le seul outil capable de prendre tous les éléments en compte. Mais les outils disponibles pour la simulation ne sont pas suffisamment performants pour effectuer des analyses et de l'optimisation efficacement. Pour simuler des centres de contacts plus facilement, nous avons alors, dans le cadre de notre projet de maîtrise, développé la bibliothèque *ContactCenters* qui permet de construire des simulateurs de centres de contacts dans le langage de programmation Java. En utilisant cette bibliothèque, nous avons également construit divers exemples de simulateurs dont un logiciel permettant de simuler, sans programmation Java, la plupart des centres d'appels que nous avons eu à traiter. ContactCenters est déjà plus rapide que tous les outils de simulation commerciaux équivalents que nous connaissons, mais le logiciel n'est pas encore suffisamment performant pour effectuer de l'optimisation efficacement et son utilisation pose des difficultés aux gestionnaires.

Dans cette thèse, nous proposons des améliorations aux outils existants pour simuler les centres de contacts. En premier lieu, nous explorons des techniques pour augmenter l'efficacité en réduisant la variance ou le travail de simulation. Dans le premier cas, pour un temps de calcul identique ou légèrement supérieur, nous obtenons une variance beaucoup plus petite. Dans le second cas, nous obtenons une variance identique ou légèrement plus grande pour un temps de calcul significativement plus petit. Pour cela,

nous étudions des techniques telles que la stratification, les variables de contrôle et les variables aléatoires communes. Nous examinons aussi la possibilité d'utiliser un modèle simplifié de chaîne de Markov en temps continu avec l'uniformisation et la conversion en temps discret. De plus, nous adaptons une technique de scission et de recombinaison à ce modèle pour réutiliser du travail de simulation. En second lieu, nous améliorons l'architecture de notre logiciel de simulation pour le rendre plus flexible, extensible et facile à utiliser.

Mots clés : Centres d'appels, réduction de la variance, stratification, variable de contrôle, variables aléatoires communes, chaîne de Markov, uniformisation, scission et recombinaison.

ABSTRACT

A *contact center* is a set of resources for communication between an organization and its users. Many companies have such a center to provide services to their customers while government uses them for information, emergency services, etc. Contact centers clearly have a great economical importance which justifies the need to analyze them, and optimize their performance. This analysis is performed by building a model of a contact center, and by using the model to evaluate the performance of the center for multiple configurations.

With the increase in complexity of contact centers, simulation is progressively becoming the only tool capable of taking every detail into account. But currently available simulation tools are not fast enough to perform analysis and optimization efficiently. We have thus developed the *ContactCenters* library during our master's thesis to ease the simulation of contact centers. This library can be used to construct simulators of contact centers in the Java programming language. Using ContactCenters, we have developed some example programs, including a generic tool for simulating most common call centers without programming in Java. ContactCenters is faster than all equivalent commercial tools we know of, but it is still not efficient enough for optimization, and its use is often hard for managers.

In this thesis, we propose improvements to existing tools for simulating contact centers. Firstly, we explore techniques for improving efficiency by reducing the variance, or the simulation work. In the first case, for an equal or slightly larger computing time, we obtain an estimator with a significantly smaller variance. In the second case, for an equal or slightly larger variance, we obtain an estimate after a smaller computing time. For this, we explore techniques such as stratification, control variates, and common random numbers. We also consider the possibility of using a simplified continuous-time Markov chain model with uniformization and discrete-time conversion. Moreover, we adapt a split and merge technique to this model in order to reuse simulation work. Sec-

ondly, we improve the design of our simulation tool to make it more flexible, extensible, and user-friendly.

Keywords: Call centers, variance reduction, stratification, control variate, common random numbers, Markov chain, uniformization, split and merge.

TABLE DES MATIÈRES

LISTE DES TABLEAUX

LISTE DES FIGURES

LISTE DES SIGLES

CMTC Chaîne de Markov en temps continu, voir section 5.1

CMTD Chaîne de Markov en temps discret, voir section 5.1

FTE *Full time equivalent* ou équivalent à temps plein, quotient du temps total de branchement des agents et de la durée moyenne d'un quart de travail

i.i.d. Indépendant et identiquement distribué, caractérise un ensemble de variables aléatoires conjointement indépendantes dont la loi de probabilité de chaque élément est la même.

RPC *Right party connect*, communication sortante réussie, c'est-à-dire que le contact a été établi et la bonne personne a été rejointe

SSJ Stochastic Simulation in Java, bibliothèque utilisée par ContactCenters pour effectuer la simulation

VAC Variables aléatoires communes, voir section 4.1

VAI Variables aléatoires indépendantes, voir section 4.1

VC Variable de contrôle, voir section 3.2

XML eXtensible Markup Language, méta-langage permettant de représenter des données complexes et hiérarchiques sous la forme d'un fichier textuel

NOTATION

La plupart des symboles récapitulés ici sont définis dans le chapitre 2 de ce document.

$A(t_1,t_2)$ Nombre d'arrivées pendant l'intervalle $[t_1,t_2]$

$g_1(s,t_1,t_2), g_2(s,t_1,t_2)$ Niveau de service, c'est-à-dire fraction des contacts qui ont attendu moins de s secondes pendant l'intervalle $[t_1,t_2]$.

i Indice d'un groupe d'agents

I Nombre de groupes d'agents

k Indice d'un type de contact

K Nombre de types de contacts

$L(t_1,t_2)$ Nombre de contacts arrivés pendant l'intervalle $[t_1,t_2]$ et ayant abandonné par la suite

$L_{\mathrm{B}}(s,t_1,t_2)$ Nombre de contacts arrivés pendant l'intervalle $[t_1,t_2]$ et ayant abandonné par la suite après un temps d'attente supérieur à s

$L_{\mathrm{G}}(s,t_1,t_2)$ Nombre de contacts arrivés pendant l'intervalle $[t_1,t_2]$ et ayant abandonné par la suite après un temps d'attente inférieur ou égal à s

$\ell(t_1,t_2)$ Proportion des contacts arrivés pendant l'intervalle $[t_1,t_2]$ et ayant abandonné par la suite

n Taille d'un échantillon de nombres aléatoires

$N_i(t)$ Nombre total d'agents dans un groupe i au temps t

$N_{\mathrm{B},i}(t)$ Nombre d'agents occupés dans un groupe i au temps t

$N_{\mathrm{F},i}(t)$ Nombre d'agents libres pour servir un contact dans un groupe i au temps t

$N_{\mathrm{G},i}(t)$ Nombre d'agents fantômes du groupe i, c'est-à-dire nombre d'agents devant quitter le système après avoir terminé le service d'un contact en cours, au temps t

$N_{\mathrm{I},i}(t)$ Nombre d'agents inoccupés dans un groupe i, pouvant ou non servir des contacts au temps t

$o(t_1,t_2)$ Taux d'occupation des agents du groupe i pendant l'intervalle de temps $[t_1,t_2]$

p Indice d'une période, entre 0 et $P+1$

P Nombre de périodes principales

$q_{i,j}$ Taux de transition de l'état i vers l'état j pour une CMTC

q Taux de transition utilisé pour une CMTC uniformisée

Q_k Nombre de contacts de type k en attente

$Q(t)$ Taille de la file d'attente au temps t

$\mathcal{S}$ Ensemble dénombrable représentant un espace d'états

$S_{k,i}$ Nombre de contacts de type k en service auprès d'agents du groupe i

$S(t_1,t_2)$ Nombre de contacts arrivés pendant l'intervalle $[t_1,t_2]$ et servis à un temps $t \geq t_1$

$S_{\mathrm{B}}(s,t_1,t_2)$ Nombre de contacts arrivés pendant l'intervalle $[t_1,t_2]$ et servis à un temps $t \geq t_1$ après un temps d'attente supérieur à s

$S_{\mathrm{G}}(s,t_1,t_2)$ Nombre de contacts arrivés pendant l'intervalle $[t_1,t_2]$ et servis à un temps $t \geq t_1$ après un temps d'attente inférieur ou égal à s

t Utilisé pour représenter un temps

t_p Temps de fin de la période p

T Temps de départ du dernier contact

θ Représente un paramètre d'un modèle

U Variable aléatoire suivant la loi uniforme sur l'intervalle $[0,1)$

Υ Représente un ensemble de valeurs pour un paramètre d'un modèle

$w(t_1,t_2)$ Temps d'attente moyen des contacts arrivés pendant l'intervalle $[t_1,t_2]$

$W(t_1,t_2)$ Somme des temps d'attente pour tous les contacts arrivés pendant l'intervalle $[t_1,t_2]$

$W_{\mathrm{L}}(t_1,t_2)$ Somme des temps d'attente pour tous les contacts arrivés pendant l'intervalle $[t_1,t_2]$ et ayant abandonné par la suite

$W_{\mathrm{S}}(t_1,t_2)$ Somme des temps d'attente pour tous les contacts arrivés pendant l'intervalle $[t_1,t_2]$ et servis par la suite

$\{X(t), t \geq 0\}$ Processus stochastique en temps continu

$\{X_n, n \geq 0\}$ Processus stochastique en temps discret

REMERCIEMENTS

Je remercie tout d'abord Pierre L'Écuyer pour m'avoir proposé ce projet et m'avoir soutenu en tant que directeur de recherche tout au long de sa réalisation. Je le remercie pour ses nombreuses suggestions qui ont contribué à orienter ma recherche et améliorer la bibliothèque ContactCenters, sa documentation et cette thèse.

Je remercie Bell Canada et le Conseil de Recherche en Sciences Naturelles et en Génie (CRSNG) pour avoir contribué au financement de ce projet par le biais d'une bourse d'études à incidence industrielle et d'un projet de recherche sur les centres d'appels dans lequel s'inscrit cette thèse.

Je remercie Athanassios Avramidis, Mehmet Tolga Ceşik, Wyean Chan, Naoufel Thabet, Raphaël Bean, Mohamed Nassim et Vincent Béchard pour avoir utilisé la bibliothèque pendant son développement et proposé diverses améliorations.

Je remercie également mes parents, Pauline et Réal, pour avoir pris soin de mon éducation sans laquelle cette thèse n'aurait pas pu être écrite. Je les remercie également, ainsi que mon frère David et ma sœur Eve, pour m'avoir soutenu et encouragé à mener ce projet jusqu'au bout malgré les embûches rencontrées.

CHAPITRE 1

INTRODUCTION

Un *centre de contacts* est un ensemble de ressources telles que des lignes téléphoniques, des commutateurs, des routeurs, des employés et des ordinateurs servant d'interface de communication entre un organisme et ses usagers. Un *contact* consiste en une requête de communication entre un usager et un organisme. La communication peut être effectuée via le téléphone, la télécopie, le courrier électronique, etc. Un centre de contacts ne traitant que des appels téléphoniques est appelé un *centre d'appels*.

De tels centres doivent traiter un grand nombre de requêtes de divers types, nécessitent une infrastructure technologique importante et emploient plusieurs préposés, aussi appelés *agents*, d'où un coût de gestion élevé. D'un autre côté, la qualité du service offert affecte l'image de marque de l'organisme possédant un centre de contacts. Certains centres de contacts qui effectuent de la vente à distance peuvent même devenir une source de revenus pour une entreprise. L'importance économique des centres de contacts a déjà clairement été démontrée [25]. Les gestionnaires doivent donc établir un équilibre entre la réduction du coût et la qualité du service en effectuant des analyses de sensibilité et de l'optimisation.

Ce chapitre présente les genres de problèmes peuvent poser les centres de contacts. Nous y traitons également des types d'outils disponibles pour leur analyse pour ensuite nous concentrer sur notre solution, *ContactCenters*, que nous avons améliorée dans le cadre de ce projet. Nous présentons ensuite les objectifs de notre projet de recherche et le plan du reste de cette thèse.

1.1 Problèmes posés par les centres de contacts

Pour réduire les coûts et améliorer la qualité de service, les gestionnaires doivent considérer divers scénarios pour effectuer des analyses de sensibilité. Ils sont également

amenés à résoudre des problèmes d'optimisation, la plupart du temps sans disposer d'outils adéquats. De plus, ils doivent faire de la planification à court, moyen et long termes du comportement de centres de contacts. Ces problèmes sont présentés dans [1, 7, 34, 71] et examinés plus en profondeur dans les références données par ces articles. Dans cette thèse, nous développons des outils pour aider à effectuer l'analyse de sensibilité et l'optimisation.

L'analyse de sensibilité consiste à évaluer la performance d'un système sous diverses conditions et utiliser les résultats obtenus pour étudier l'effet de ces conditions sur la performance. Une telle analyse permet par exemple d'évaluer l'impact d'une diminution des durées de service sur la proportion des contacts répondus après un temps d'attente inférieur à un certain seuil, appelée *niveau de service*. L'analyse de sensibilité permet aussi d'évaluer la pertinence de changements opérationnels dans un centre de contacts, notamment la mise en place d'une nouvelle politique de routage des contacts, la formation d'agents pour servir de nouveaux types de contacts, l'acheminement d'une portion des contacts vers un centre géré par un fournisseur tiers, la construction d'un centre de contacts virtuel en mettant en réseau deux ou plusieurs centres réels, etc.

Outre le niveau de service, la performance du système est souvent évaluée en mesurant le pourcentage d'abandons des contacts, leur temps d'attente moyen et le taux d'occupation des agents. Les contacts sont habituellement partitionnés en plusieurs types tandis que les agents sont divisés en plusieurs groupes. Il est courant d'observer, en plus de la performance globale, celle pour chaque type de contact et groupe d'agents.

Parfois, les agents ne servent pas tous les contacts avec le même rendement. Si un agent préfère les contacts d'un certain type k_1 mais peut aussi servir ceux d'autres types moins efficacement, il est souhaitable que cet agent reçoive des contacts de type k_1 la plupart du temps. Dans un tel contexte, les gestionnaires s'intéressent à la qualité de l'affectation des contacts aux agents, c'est-à-dire à la proportion des contacts d'un certain type servis par des agents d'un certain groupe.

Souvent, le nombre de scénarios à considérer lors d'une analyse est très grand et

les gestionnaires recherchent le scénario optimal. Par exemple, un problème courant consiste à trouver le nombre minimal d'agents dans chaque groupe pour atteindre un niveau de service donné. L'optimisation demande ainsi d'évaluer la performance du centre de contacts plusieurs fois avec des paramètres différents. Elle vise en premier lieu le nombre d'agents par intervalle de temps et par groupe (aussi appelé *staffing* en anglais [4]), la construction d'horaires pour les agents [6], mais elle s'étend également au niveau du routage [56] et de la composition automatique d'appels destinés par exemple à la vente de produits, le rappel de clients, etc.

L'optimisation peut aussi se concentrer sur les recours, c'est-à-dire les décisions prises par les gestionnaires pendant la journée si la performance du système diffère de celle prévue ou désirée. Ces décisions peuvent inclure l'appel de nouveaux agents en réserve, la fin prématurée du quart de travail pour certains agents en service, la redirection d'une fraction des contacts vers un fournisseur tiers, etc. L'optimisation de ces recours est une importante avenue de recherche et nécessite beaucoup de simulation.

1.2 Outils d'analyse

1.2.1 Modélisation

Examinons maintenant les outils disponibles pour analyser les centres de contacts. Avant d'entreprendre toute analyse, les gestionnaires doivent construire un modèle mathématique approximant le centre de contacts considéré. Ce modèle peut ensuite être utilisé pour prédire le comportement du système lorsque ses paramètres sont modifiés. Le modèle le plus courant, sur lequel nous nous concentrons dans cette thèse, est un réseau de files d'attente semblables à celle de la figure 1.1. Dans ce modèle, les contacts sont partitionnés en K types. Les contacts des K_{I} premiers types sont dits *entrants*, c'est-à-dire qu'ils correspondent à des requêtes initiées par des usagers communiquant avec le centre. Les contacts des autres types sont dits *sortants*, c'est-à-dire que les requêtes leur correspondant sont initiées par des agents du centre ou un système de composition

automatique.

Les agents servant les contacts sont séparés en I groupes. Le groupe d'un agent détermine l'ensemble des types de contacts qu'il peut servir si bien que tous les agents ne peuvent pas servir n'importe quel contact. Un tel centre est alors dit *multi-skill* en anglais. Nous reviendrons sur ces notions dans la première section du chapitre suivant.

Déterminer les bonnes partitions pour les contacts et les agents est un problème de modélisation. En général, chaque contact et chaque agent possède ses propres attributs qui le rend unique si bien que former des types de contacts et des groupes d'agents exige habituellement de négliger certaines caractéristiques. Ainsi, plus il y a de types de contacts ou de groupes d'agents, plus le modèle est réaliste, mais plus il est difficile d'obtenir les données nécessaires pour en fixer les paramètres et d'interpréter les résultats obtenus. En revanche, s'il y a trop peu de types de contacts ou de groupes d'agents, le modèle devient trop irréaliste et ne permet pas de mener à bien l'analyse désirée.

Ces partitions établies, il faut modéliser le routage, c'est-à-dire les règles déterminant vers quels agents sont acheminés les contacts qui arrivent et quels contacts en attente servent les agents devenus libres. De plus, les gestionnaires doivent estimer des lois de probabilité pour les durées de service des contacts, modéliser les abandons, déterminer le processus d'arrivée des contacts, etc. Ils doivent également garder à l'esprit que tous les paramètres sont susceptibles de varier dans le temps. L'estimation de lois est traitée dans bon nombre de livres de simulation, comme [53, 58].

Mais tout cela nécessite des données qui sont difficiles à obtenir si les systèmes en place dans les centres de contacts ne les collectent ou ne les conservent pas. En particulier, il arrive que seules des données agrégées par demi-heures ou même pour toute la journée soient disponibles. De plus, modéliser les abandons est difficile, car nous ne pouvons pas observer le temps de patience des contacts qui n'ont pas abandonné. Ce problème est abordé et traité dans [34, 35]. Pour des exemples de modélisations, voir [11, 25].

Le modèle construit est ensuite exécuté pour tenter de reproduire le comportement

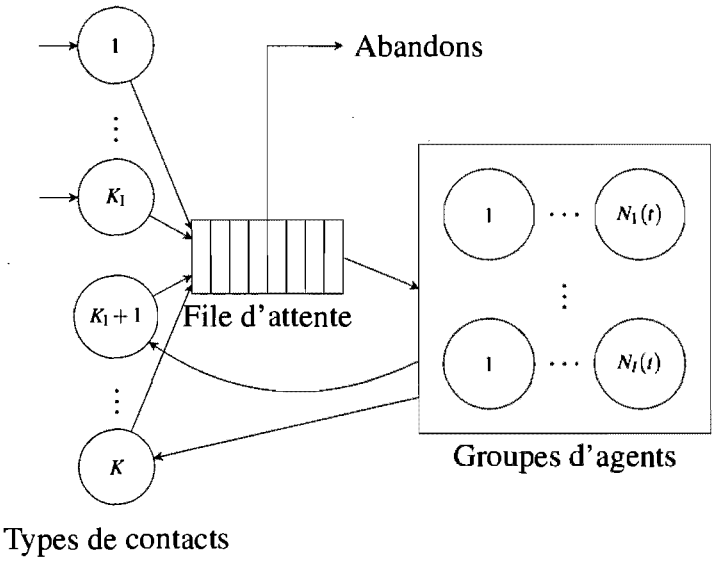

Figure 1.1 – Modèle de file d'attente représentant un centre de contacts usuel

du centre de contacts réel. Les résultats obtenus sont comparés avec les données réelles afin de valider le modèle et le corriger si nécessaire. Ce problème de modélisation et de validation est couvert par tous les ouvrages de base en simulation, comme [53].

1.2.2 Formules analytiques

Il existe plusieurs façons différentes d'exécuter un modèle de centre de contacts. Avec les premiers centres ne traitant qu'un seul type d'appel téléphonique, des formules analytiques fondées sur la théorie des files d'attente étaient utilisées pour effectuer l'analyse sous des hypothèses simplificatrices fortes. Par exemple, le modèle Erlang C, couramment utilisé, considère que les arrivées suivent un processus de Poisson, que les temps de service sont indépendants et suivent la loi exponentielle et qu'aucun abandon n'est autorisé, ce qui est plutôt irréaliste. Sous ces hypothèses, il est possible de dériver une formule donnant la probabilité à long terme que tous les agents sont occupés, ce qui correspond à la probabilité à long terme qu'un appel ait à attendre. À partir de cette probabilité, il est possible de déterminer d'autres mesures de performance telles que le

temps d'attente moyen et le niveau de service.

D'un autre côté, le modèle Erlang B suppose que tout appel qui ne peut pas être servi immédiatement est bloqué ; aucune attente en file n'est autorisée. La formule d'Erlang B, qui permet d'obtenir la probabilité à long terme qu'un appel soit bloqué, est valide peu importe la loi de probabilité du temps de service. Le modèle Erlang A autorise quant à lui les abandons, mais il suppose que les temps de patience des contacts sont i.i.d. exponentiels.

Ces formules analytiques sont présentées dans bon nombre d'ouvrages sur la théorie des files d'attente, notamment [20] et reprises dans certains articles comme [34, 70]. Elles sont suffisamment simples pour être implantées par les gestionnaires sous forme de macros dans des tableurs comme Microsoft Excel. Il existe aussi des logiciels dédiés à leur calcul, comme par exemple [16].

Dans le cas d'un centre avec plusieurs types de contacts et groupes d'agents, des approximations existent, mais elles imposent des hypothèses additionnelles sur le routage et les abandons. Par exemple, plusieurs approximations imposent qu'à l'arrivée d'un appel, une liste de groupes d'agents dépendant de son type soit visitée et que le premier groupe comportant au moins un agent libre soit sélectionné. En particulier, [50] propose une approximation qui est valide seulement si tout appel ne pouvant être servi immédiatement est bloqué. L'approximation plus générale développée dans [4] permet l'attente en file, mais elle restreint la façon qu'un agent devenant libre choisit un contact en attente.

Ces formules considèrent habituellement que le centre d'appels fonctionne dans des conditions identiques depuis un temps infini. En réalité, les conditions changent régulièrement et nous souhaitons estimer la performance sur un horizon fini, par exemple une journée ou un mois. Les principales approximations du niveau de service sur horizon fini sont reprises dans [45], mais elles supposent toujours que les arrivées suivent un processus de Poisson, les temps de service sont exponentiels et aucun abandon n'a lieu. En pratique, ces hypothèses ne sont pas satisfaites. Par exemple, [5] donne un exemple

dans lequel des processus d'arrivées doublement stochastiques reproduisent mieux les arrivées qu'un processus de Poisson ordinaire. De même, [11] présente un cas pour lequel les durées de service observées s'ajustent beaucoup mieux à une loi lognormale qu'à une loi exponentielle. De plus, il est la plupart du temps trop coûteux de former tous les employés pour servir tous les contacts, même si [81] illustre que cela pourrait augmenter la qualité de service en théorie. Ainsi, agréger tous les types de contacts et les groupes d'agents pour utiliser une approximation adaptée à un modèle simple donne souvent des résultats irréalistes.

1.2.3 Simulation

Bien que les centres de contacts sont encore modélisés par un système de files d'attente, seule la simulation peut fournir des évaluations précises tenant compte de toute la complexité. Par contre, évaluer la performance pour plusieurs scénarios par simulation demande d'effectuer des milliers, voire des millions de réplications pour aboutir à un résultat précis. Il est alors important, pour y parvenir, de disposer d'outils très rapides.

Pour donner une idée du temps pris par la simulation pendant l'optimisation, prenons par exemple la technique décrite dans [6]. Celle-ci considère le problème de construction d'horaires comme un programme mathématique utilisant comme fonction objectif un coût linéaire par rapport au nombre d'agents sur chaque quart de travail et le niveau de service des contacts pour les contraintes. La technique résout un programme linéaire correspondant à une version simplifiée du programme mathématique sans les contraintes (stochastiques) sur les niveaux de service. Elle utilise ensuite la simulation pour évaluer la réalisabilité de la solution et ajoute des contraintes au programme linéaire simplifié en utilisant des coupes établies par un estimé du sous-gradient du niveau de service par rapport au vecteur d'affectation des agents, lui aussi obtenu par simulation. Ce processus de coupe est répété jusqu'à obtenir une solution réalisable. Si cette méthode est appliquée sur un centre d'appels avec 20 types d'appels, 35 groupes d'agents et 52 périodes, avec seulement 300 réplications (correspondant à des jours dans le modèle) pour le test de

réalisabilité et 20 réplications pour chaque composante de sous-gradient, il faut malgré tout $300 + 20 * 35 * 52 = 36\ 700$ réplications par itération. Avec ces paramètres et notre outil de simulation qui est le plus rapide parmi ceux que nous connaissons, l'optimisation a exigé cinq heures de temps de calcul.

Pour améliorer la vitesse de simulation, nous pouvons simplifier le modèle utilisé, optimiser le programme exécutant le modèle ou encore appliquer des techniques de réduction de la variance. La première idée est souvent inacceptable tandis que la seconde réduit le temps d'exécution jusqu'à un certain point seulement. La troisième idée, par contre, a un grand potentiel pour améliorer l'efficacité, car il existe un très grand nombre de techniques différentes qu'il est possible d'appliquer et de combiner. Ces techniques sont décrites dans la plupart des ouvrages de base en simulation, notamment [10, 28, 36, 53, 58]. Par contre, leur application n'est souvent pas directe et demande des adaptations. Plusieurs articles appliquent ou combinent ce genre de techniques, notamment [8, 9, 18]. Par contre, aucun de ces articles ne concerne spécifiquement les centres de contacts.

Un centre de contacts peut bien entendu être modélisé grâce à un logiciel de simulation générique, mais cette tâche nécessite un énorme travail de conception et même de programmation. Il existe heureusement des logiciels spécialisés qui supportent la simulation de la plupart des centres de contacts d'aujourd'hui. Arena Contact Center Edition de Rockwell [75] et ccProphet de NovaSim [73] sont des exemples de tels logiciels. Toutefois, de nouveaux cas qui n'étaient pas prévus initialement peuvent survenir à n'importe quel moment et s'avérer difficiles à traiter sans recourir à des mécanismes de bas niveau qui dépendent du logiciel choisi et qui peuvent nécessiter la mise à jour vers une version plus complète (et plus coûteuse) du produit. De telles extensions sont aussi nécessaires pour appliquer des techniques de réduction de la variance. Les logiciels commerciaux sont également formés d'un grand nombre de couches superposées, interconnectées et difficiles à séparer qui peuvent diminuer la performance. Ainsi, faute d'outils adéquats, plusieurs gestionnaires de centres de contacts emploient encore les formules analytiques

même lorsque leurs hypothèses ne sont pas vérifiées.

1.3 Éléments de ContactCenters

De notre côté, nous avons développé la bibliothèque ContactCenters [12, 14, 15] en Java. Ce langage est puissant, largement utilisé et très bien supporté. Nous avons employé la bibliothèque SSJ [55, 59, 64] comme système de simulation en Java pour la génération des nombres aléatoires, la gestion de la liste d'événements et la collecte statistique. Grâce à l'héritage, les classes de ContactCenters peuvent facilement être étendues sans les réécrire en entier. Un simulateur peut tirer parti de Java pour accéder à un grand nombre de bibliothèques d'optimisation, d'analyse statistique, ainsi qu'à des outils de construction d'interfaces graphiques. Grâce aux optimisations des récentes machines virtuelles Java, un simulateur écrit en Java s'exécute beaucoup plus rapidement qu'un modèle conçu grâce aux outils commerciaux les plus utilisés et fondés sur un langage complètement interprété et peu répandu.

ContactCenters est formée de composantes indépendantes qui sont reliées entre elles au moment de construire un programme simulant un modèle précis et détaillé de centre de contacts. Un tel programme peut également intégrer des techniques de réduction de la variance. Ces composantes représentent les contacts (appels, télécopies, etc.), les processus d'arrivées, le composeur d'appels sortants, les groupes d'agents, files d'attente et le routeur. Chaque contact est représenté par une entité, c'est-à-dire un objet, avec son propre ensemble d'attributs prédéfinis que l'utilisateur peut étendre si nécessaire. Les sources de contacts (processus d'arrivées et composeurs d'appels sortants) construisent de tels contacts et les envoient au routeur qui se charge de mettre les contacts en service auprès d'agents ou les insérer dans des files d'attente pour les traiter plus tard. Le routeur annonce aussi les contacts sortants du centre à un système de collecte statistique.

Le programmeur peut facilement construire un observateur et l'enregistrer auprès de ces composantes pour, par exemple, connaître les contacts qui sont créés, ceux qui sortent

du système, suivre l'état des agents, etc. En fait, tout le couplage entre les composantes du système est effectué à l'aide d'observateurs [33], ce qui donne un maximum de flexibilité. Dans ce modèle, chaque objet *observable* comporte une liste d'*observateurs* dont le contenu exact n'est connu qu'à l'exécution. Un observateur est un objet implantant une interface spécifique. Lorsque de l'information doit être diffusée, la liste est parcourue et une méthode spécifiée par l'interface est appelée pour chaque observateur. De cette façon, les composantes peuvent être testées, améliorées et remplacées indépendamment des autres.

Par contre, écrire un programme Java pour simuler un centre de contacts peut être long et n'est pas à la portée de tous les gestionnaires. Un tel programme doit aussi répondre à un certain nombre de normes pour pouvoir interagir de façon générale avec d'autres programmes tels qu'un optimiseur. Il est alors important de disposer d'un simulateur précompilé le plus général possible permettant de traiter les cas les plus courants et d'interagir facilement avec d'autres outils. En utilisant les composantes de ContactCenters, nous avons construit un tel simulateur qui permet, en utilisant des fichiers de configuration dans le format XML [83], de traiter la plupart des centres d'appels courants.

XML est un langage permettant de décrire du contenu hiérarchique sous la forme d'une chaîne de caractères, en employant des éléments auxquels sont associés des attributs et qui peuvent contenir du texte ou d'autres éléments. Bien entendu, une application XML particulière spécifie quelles informations sont permises dans les fichiers qui lui sont propres.

Ce simulateur générique précompilé se restreint à un modèle particulier mais assez général de centre d'appels mixte supportant K_I types d'appels entrants et K_O types sortants, avec I groupes d'agents et un horizon séparé en P périodes principales de durée fixe. L'usager peut employer n'importe quelle loi de probabilité de SSJ pour les durées de patience et les durées de service, mais il doit choisir les processus d'arrivées, les politiques de composition d'appels sortants et la politique de routage parmi des listes

de politiques prédéfinies. Il peut par contre paramétrer les différents processus avec des valeurs numériques.

Le simulateur calcule différentes statistiques comme le nombre d'appels produits, servis, bloqués ou ayant abandonné. Ces statistiques sont utilisées entre autres pour estimer les mesures de performance qui intéressent les gestionnaires, comme le niveau de service, le pourcentage d'abandons, etc. Chaque mesure est estimée pour chaque période séparément ainsi que pour tout l'horizon.

La simulation peut être effectuée de façon stationnaire pour une seule période de durée supposément infinie dans le modèle, en utilisant la méthode des moyennes par lots [53] pour obtenir des intervalles de confiance, ou pour tout l'horizon, avec un nombre donné de réplications indépendantes. Dans le cas stationnaire (horizon infini), le simulateur est initialisé avec les paramètres pour une période et ces paramètres demeurent fixes tout au long de l'expérience. Dans le cas non stationnaire (horizon fini), les paramètres peuvent changer d'une période à l'autre. Simuler sur horizon infini ne semble pas naturel pour des centres de contacts, mais il peut s'avérer utile de le faire pour comparer les résultats de simulation avec des approximations considérant le système comme stationnaire.

1.4 Plan de la thèse

Dans cette thèse, nous proposons des améliorations aux outils existants pour simuler les centres de contacts. En particulier, nous améliorons l'efficacité à l'aide de techniques de réduction de la variance, nous proposons un modèle simplifié plus rapide à simuler mais moins restrictif que ceux employés par les approximations courantes et nous améliorons la flexibilité et la facilité d'utilisation de notre logiciel de simulation.

D'abord, le chapitre suivant explique de façon plus détaillée ce que sont les centres de contacts et introduit la notation mathématique utilisée tout au long de cette thèse. Ensuite, nous examinons des techniques réduisant la variance pour un scénario particulier.

Dans cette optique, le chapitre 3, publié en partie dans [61], étudie la combinaison de la stratification sur une variable continue avec une variable de contrôle linéaire et montre que l'interaction entre ces deux techniques soulève des problèmes intéressants. La stratification [19] consiste à séparer les réalisations d'une ou plusieurs variables aléatoires en strates et à échantillonner séparément dans chaque strate pour ensuite calculer un estimateur combinant les strates. Une variable de contrôle linéaire [37] consiste quant à elle en une variable aléatoire d'espérance nulle multipliée par un coefficient et ajoutée à un estimateur. La combinaison des deux techniques pose des problèmes, car le coefficient optimal de la variable de contrôle dépend de la variable sur laquelle nous stratifions. Il existe également plusieurs façons d'effectuer la combinaison et aucune technique ne surpasse toutes les autres en général. En particulier, nous pourrions croire qu'utiliser un coefficient variant en fonction d'une variable de stratification continue réduit davantage la variance qu'un coefficient dépendant uniquement de la strate puisque la première méthode utilise davantage d'informations que la seconde. Mais souvent, utiliser un coefficient par strate est plus simple à implanter et réduit davantage la variance. Après avoir examiné la synergie entre ces deux techniques, nous évaluons l'effet de certaines variables de contrôle sur un exemple de centre de contacts avant d'appliquer la combinaison et examiner les résultats.

Comme expliqué dans la section 1.1, les gestionnaires sont souvent amenés à simuler plusieurs scénarios pour un même modèle. Bien que simuler chaque scénario plus rapidement réduit le temps total pris par toute l'expérience, il est judicieux d'étudier des techniques tirant parti de la similarité existante entre les configurations. Le chapitre 4 examine pour cela l'application des variables aléatoires communes [10] pour estimer la différence de mesures de performance par rapport à un paramètre d'un centre de contacts. Cette technique consiste à utiliser les mêmes nombres aléatoires aux mêmes endroits pour chaque configuration simulée. Dans ce chapitre, publié en partie dans [60], nous comparons différentes méthodes de synchronisation et développons pour certains cas des preuves de convergence de la variance de la différence par rapport à la variation

d'un paramètre continu.

Puisque les techniques précédentes ne sont pas efficaces dans tous les cas, nous avons pensé à simplifier notre modèle. Dans cette optique, le chapitre 5, publié en partie dans [13], développe un modèle de centre de contacts utilisant une chaîne de Markov en temps continu et permettant dans certains cas d'améliorer la vitesse de simulation en utilisant l'uniformisation [40] et la conversion en temps discret [31]. La technique consiste à simuler la chaîne de Markov en temps discret imbriquée et à calculer des espérances conditionnelles à la séquence des états visités et au nombre de transitions effectuées. Si nous appliquons cette idée aux centres de contacts, la chaîne de Markov utilisée possède un état à plusieurs dimensions. Nous traitons ce problème en appliquant la recherche indexée pour générer les transitions dans les modèles avec plusieurs types de contacts et groupes d'agents. L'estimation du niveau de service avec la conversion en temps discret n'est pas triviale étant donné qu'elle fait appel aux temps d'attente de tous les contacts qui ne sont pas générés avec cette méthode. Nous résolvons ce problème en calculant pour chaque contact une probabilité conditionnelle au nombre de transitions passées en attente et au nombre total de transitions. De plus, l'uniformisation fait parfois en sorte que le taux de transition utilisé est élevé, provoquant beaucoup de transitions fictives. Pour remédier à cela, nous développons une technique de partitionnement de l'ensemble d'états permettant de bénéficier d'un taux de transition adaptatif sans briser l'uniformisation. Si le taux d'arrivée et le nombre d'agents sont élevés et s'il y a un nombre modéré de types de contacts et de groupes d'agents, la simulation avec ce modèle est plus rapide qu'avec un modèle tenant compte de tous les détails et notre chaîne de Markov se fonde sur des hypothèses moins restrictives que les formules d'approximation couramment utilisées. Cette technique est ainsi très utile pour les premières étapes d'algorithmes d'optimisation, la simulation par événements discrets étant utilisée à la fin pour raffiner la solution trouvée.

Lorsque nous simulons plusieurs configurations, en plus d'augmenter l'efficacité pour simuler chaque configuration séparément, nous souhaitons éviter de répéter du tra-

vail inutilement. Dans cette optique, le chapitre 6 généralise une technique de scission et de recombinaison (*split and merge* en anglais) présentée dans [68], pour l'adapter à un modèle de centre de contacts. Avec cette technique, des copies du modèle simulé se scindent au moment où des décisions impliquant les paramètres à faire varier sont prises et peuvent fusionner lorsque l'état du système est le même. Par contre, avec un modèle trop complexe, le nombre de points de décision est trop grand ou les tests pour la scission et la recombinaison sont si coûteux que simuler chaque configuration séparément est plus rapide. Nous proposons un modèle de centre de contacts ne présentant pas ces problèmes et sur lequel nous avons appliqué la technique avec succès pour estimer des sous-gradients du niveau de service. Dans ce modèle, chaque point de décision correspond à l'arrivée d'un contact et la scission se produit en fonction du routage du nouveau contact.

Enfin, le chapitre 7 présente les grandes lignes de l'architecture de ContactCenters, ses principales fonctionnalités ainsi que les problèmes principaux rencontrés pendant son développement. En particulier, nous examinons plus en profondeur les possibilités du simulateur générique précompilé. Nous découvrons aussi que le mode de gestion des groupes d'agents affecte dans certains cas les résultats de simulation. Nous comparons différentes heuristiques pour prédire les temps d'attente dans le cas où le modèle comporte plusieurs types de contacts. Nous examinons aussi comment nous avons augmenté l'extensibilité du simulateur générique nécessitant peu ou pas de programmation Java en permettant à l'usager de personnaliser les lois de probabilité, les politiques de routage et de composition d'appels sortants et les processus d'arrivées utilisés. Nous décrivons aussi comment nous avons simplifié l'entrée des paramètres en construisant un schéma XML pour notre format de données et comment nous avons amélioré la clarté des messages d'erreur en tirant parti du mécanisme de propagation des exceptions de Java. Voir la documentation de ContactCenters [14] pour des détails supplémentaires sur l'architecture, les fonctionnalités et l'utilisation du logiciel de simulation.

CHAPITRE 2

CONCEPTS ET NOTATION MATHÉMATIQUE DE BASE

Dans ce chapitre, nous présentons les concepts et la notation mathématique de base qui seront utilisés tout au long de cette thèse. La prochaine section décrit de façon plus détaillée ce qu'est un centre de contacts. La section 2.2 décrit de façon mathématique les mesures de performance abordées au chapitre précédent. Enfin, la section 2.3 rappelle la notion d'ordre asymptotique qui sera utilisée à certains moments dans cette thèse.

2.1 Les centres de contacts

Comme indiqué au chapitre précédent, un centre de contacts est habituellement modélisé par un réseau de files d'attente semblables à celle de la figure 1.1. Un *contact* consiste en une requête de communication entre un usager et un organisme. Les contacts *entrants* sont générés par des usagers tentant d'entrer en communication pour obtenir un service tel qu'une réservation ou du support technique. Les contacts *sortants* sont initiés de façon proactive par les employés ou, dans le cas des appels téléphoniques, par un système spécialisé appelé *composeur*. Ils permettent par exemple la vente à distance ainsi que le rappel de clients. Les centres capables de traiter les deux types de contacts sont dits *mixtes*.

Afin de simplifier le traitement, chaque contact est classé selon un type représenté sous la forme d'un entier k entre 1 et K, où K est le nombre total de types de contacts supportés par un système particulier. Un contact est entrant si son type $k \in \{1,\ldots,K_{\mathrm{I}}\}$, où $K_{\mathrm{I}} \leq K$ est le nombre de types de contacts entrants tandis qu'il est sortant si $k \in \{K_{\mathrm{I}}+1,\ldots,K\}$. Le type de contact peut être déterminé en utilisant sa provenance (numéro de téléphone de l'appelant, site Web utilisé, etc.), les choix de l'usager dans des menus, etc. Il peut représenter la raison de la communication, l'importance du client, etc.

Les contacts sortants sont pour leur part produits par des agents ou un appareil ap-

pelé composeur qui puisent les informations sur les usagers à contacter depuis une liste. La composition de cette liste ainsi que le nombre de contacts à tenter simultanément par le composeur dépend d'une *politique de composition*. Un contact sortant peut échouer si, par exemple, le numéro de téléphone du client ou son adresse électronique sont invalides, si l'usager appelé est occupé ou absent, si une boîte vocale est atteinte, etc. Si un composeur est utilisé pour appeler des clients, il se peut aussi qu'au moment où la communication est complètement établie, aucun agent n'est disponible pour la traiter ; on appelle cela un *mismatch*. Les clients victimes d'un tel problème peuvent soit attendre en file ou être bloqués. Bien entendu, le temps de patience d'un client ainsi traité sera habituellement très petit. Si la communication réussit, un agent doit vérifier que la bonne personne est contactée avant de pouvoir traiter la communication sortante. Une communication ainsi réussie est appelée *right party connect*.

Le *service* d'un contact consiste en un traitement destiné à satisfaire la requête d'un usager. De nos jours, plusieurs requêtes peuvent être traitées entièrement par des systèmes automatisés, mais parfois, un usager peut manifester le besoin ou le désir de parler à un être humain. Dans ce contexte, le service comprend la phase de traitement automatique, le travail d'un employé pendant la communication avec le client et le travail que l'employé doit parfois effectuer après le dialogue.

Chaque employé, aussi appelé *agent*, fait partie d'un groupe $i \in \{1,\ldots,I\}$ définissant ses compétences. Il possède également certaines particularités qui peuvent affecter son efficacité et son horaire de travail. Un agent peut aussi correspondre à une ressource informatique dans le cas où on voudrait modéliser un système de traitement automatisé. Au temps t de la journée, le groupe i contient $N_i(t)$ membres dont $N_{\mathrm{B},i}(t)$ sont en train de servir des contacts et $N_{\mathrm{I},i}(t)$ sont inoccupés. Il se peut que $N_i(t) < N_{\mathrm{B},i}(t) + N_{\mathrm{I},i}(t)$ si $N_{\mathrm{G},i}(t)$ agents terminent leur quart de travail (quittent le groupe) après avoir terminé le service en cours. Parmi les agents inoccupés, seuls $N_{\mathrm{F},i}(t) \leq N_{\mathrm{I},i}(t)$ sont connectés et disponibles pour de nouveaux services. Le nombre d'agents $N_i(t)$ est souvent plus petit que le nombre planifié en raison de retards, de pauses prolongées, etc. La figure 2.1 pré-

sente une vue schématique de ces différentes quantités. Nous pouvons également définir $N(t)$, $N_{\mathrm{G}}(t)$, $N_{\mathrm{B}}(t)$, $N_{\mathrm{I}}(t)$ et $N_{\mathrm{F}}(t)$ comme les équivalents des quantités précédentes pour tous les groupes d'agents du système.

Les agents du groupe i ne peuvent servir que des contacts dont le type fait partie d'un ensemble $S_i \subseteq \{1,\ldots,I\}$. On parle d'un groupe d'agents *généralistes* si $S_i = \{1,\ldots,I\}$ et d'un groupe de *spécialistes* si $|S_i|$ est proche de 1. On parle d'un centre de contacts *multi-skills* si $I > 1$, $K > 1$ et S_i diffère d'un groupe d'agents à l'autre.

Pendant leurs quarts de travail, les agents sont enregistrés auprès d'un *routeur* chargé d'acheminer les nouveaux contacts vers des agents libres et d'affecter des contacts en attente à des agents devenus libres. Les règles de routage peuvent être très complexes, allant d'une simple liste de groupes d'agents spécifique à chaque type de contacts à une politique dynamique tenant compte de tout l'état du système pour prendre chaque décision. Évidemment, le routeur est une composante centrale dans un centre de contacts.

Un usager peut être servi par plusieurs agents avant d'obtenir satisfaction. Par exemple, un utilisateur éprouvant des problèmes techniques avec un logiciel pourrait parler à différents techniciens (simultanément ou séquentiellement) avant d'obtenir une solution. Un *retour* se produit lorsqu'un usager servi doit recontacter l'organisme pour obtenir un nouveau service ou tenter de nouveau de satisfaire sa requête initiale.

Dans le cas de communications différées comme les courriers électroniques, le service peut même être *préemptif*, c'est-à-dire qu'un agent peut interrompre une tâche consistant par exemple à répondre à un message pour se charger d'une tâche plus prioritaire comme traiter un appel téléphonique. Ainsi, en raison des retours et du service préemptif, les agents traitent parfois plusieurs usagers simultanément.

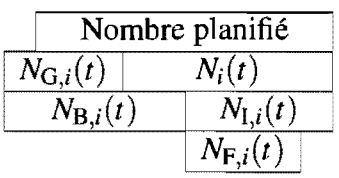

Figure 2.1 – Vue schématique des différentes quantités reliées à un groupe d'agents

Un usager qui ne peut être servi immédiatement doit attendre en file. Il peut alors devenir impatient et décider d'abandonner, quittant le système sans recevoir de service. Dans ce cas, il peut tenter de recontacter le centre plus tard (on appelle cela *retrial* en anglais) ou abandonner bel et bien, selon l'importance du service qu'il désire recevoir. Certains centres de contacts permettent également de laisser un message dans le but d'être recontacté ultérieurement. Cela forme des *files d'attente virtuelles* [82] qui sont traitées différemment des véritables files d'attente puisque le service des contacts dans de telles files est moins prioritaire que celui d'appels téléphoniques en attente. Dans le cas d'un appel téléphonique, un usager arrivé au moment où aucune ligne n'est disponible peut également recevoir un signal occupé et être *bloqué* sans pouvoir attendre en file ou être servi.

Afin de simplifier la modélisation et l'estimation des paramètres des centres de contacts, l'horizon considéré (jour, semaine, mois, etc.) est habituellement divisé en périodes de quinze à soixante minutes pendant lesquelles les paramètres du modèle demeurent constants. Parfois, des statistiques sont également recueillies pour chacune des périodes séparément.

Plus précisément, comme le montre la figure 2.2, l'horizon est divisé de façon à comprendre P périodes dites *principales* représentant les heures d'ouverture. Chaque période principale $p = 1,\ldots,P$ correspond à l'intervalle de temps $[t_{p-1}, t_p)$, où $t_0 < \cdots < t_P$. Dans le cas fréquent où chacune de ces périodes a une durée fixe d, $t_p = t_0 + pd$ pour $p = 1,\ldots,P$. Souvent, des contacts arrivent et se mettent en file avant l'ouverture du centre de contacts et le centre demeure actif après la fermeture pour traiter les services en cours et vider les files d'attente. C'est pourquoi nous définissons deux périodes additionnelles : la *période préliminaire* $[0, t_0)$ pendant laquelle le centre de contacts n'est pas encore ouvert et la *période de fermeture* $[t_P, T]$ pendant laquelle aucune arrivée ne se produit et les agents terminent leurs services.

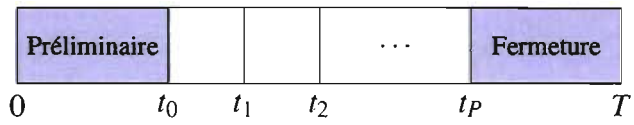

Figure 2.2 – Partitionnement de l'horizon du modèle en périodes

2.2 Mesures de performance considérées

Les mesures de performance auxquelles s'intéressent les gestionnaires sont entre autres le niveau de service, le pourcentage d'abandons, le temps d'attente moyen, le taux d'occupation des agents et le nombre d'équivalents à temps plein. Dans certains modèles, ils tentent également d'examiner la distribution des contacts entre les groupes d'agents. Toutes ces mesures sont définies sur un intervalle de temps habituellement constant $[t_1, t_2]$ correspondant à une demi-heure, une journée, un mois, etc.

Soit $S(t_1, t_2)$ le nombre de contacts servis dont le temps d'arrivée se situe dans cet intervalle $[t_1, t_2]$ et soit $L(t_1, t_2)$ le nombre d'abandons pour les contacts arrivés pendant ce même intervalle. Ces deux quantités sont des variables aléatoires suivant une loi de probabilité inconnue en général. Pour analyser le comportement de telles variables, nous devons d'abord en recueillir un certain nombre d'observations sur lesquelles nous pouvons ensuite calculer des statistiques. En particulier, la moyenne de plusieurs observations de $S(t_1, t_2)$ permet d'estimer l'espérance notée $\mathbb{E}[S(t_1, t_2)]$ correspondant à la moyenne si nous disposions d'un nombre infini d'observations. Nous pouvons également estimer la variance de $S(t_1, t_2)$ qui est notée $\mathrm{Var}[S(t_1, t_2)]$. Le nombre de contacts servis durant toute la journée est noté $S = S(0, T)$ et son espérance est $\mathbb{E}[S(0, T)]$. De façon semblable, nous définissons $L = L(0, T)$.

Soit maintenant $S_G(s, t_1, t_2)$ le nombre de contacts arrivés durant l'intervalle $[t_1, t_2]$ et qui ont été servis après avoir attendu au plus s unités de temps. En particulier, $S_G(0, t_1, t_2)$ correspond au nombre de contacts servis dès leur arrivée. Nous définissons également $S_B(s, t_1, t_2) = S(t_1, t_2) - S_G(s, t_1, t_2)$ comme étant le nombre de contacts servis après avoir

attendu plus de s unités de temps. Contrairement à $S_G(s,t_1,t_2)$ et $S_B(s,t_1,t_2)$ qui sont des variables aléatoires, s est une constante correspondant à un seuil aussi appelé *temps d'attente acceptable*. De la même façon, nous pouvons définir $L_G(s,t_1,t_2)$ comme le nombre d'abandons avant le seuil et $L_B(s,t_1,t_2) = L(t_1,t_2) - L_G(s,t_1,t_2)$. Encore une fois, nous avons $S_G(s) = S_G(s,0,T)$, $L_G(s) = L_G(s,0,T)$, etc. Cette notation s'applique pour toute autre variable aléatoire définie sur un intervalle de temps.

Le *niveau de service* sur lequel Bell Canada et le CRTC [22] se sont entendus est

$$\frac{S_G(s,t_1,t_2)}{S(t_1,t_2)+L_B(s,t_1,t_2)}$$

où $[t_1,t_2]$ correspond à un mois. Souvent, l'objectif visé pour ce niveau de service est 80% avec un temps d'attente de 20 secondes. Pour analyser cette nouvelle variable aléatoire, nous devons encore une fois calculer des statistiques sur un certain nombre d'observations. Il est d'usage d'utiliser le rapport d'espérances

$$g_1(s,t_1,t_2) = \frac{\mathbb{E}[S_G(s,t_1,t_2)]}{\mathbb{E}[S(t_1,t_2)+L_B(s,t_1,t_2)]}, \tag{2.1}$$

car cela correspond au niveau de service à long terme, sur un nombre infini de mois. De plus, cette mesure ne dépend pas de la longueur de l'horizon si bien que nous pouvons approximer le niveau de service mensuel en simulant des journées indépendantes au lieu de mois entiers.

D'autres définitions du niveau de service sont possibles, par exemple

$$g_2(s,t_1,t_2) = \frac{\mathbb{E}[S_G(s,t_1,t_2)+L_G(s,t_1,t_2)]}{\mathbb{E}[S(t_1,t_2)+L(t_1,t_2)]}. \tag{2.2}$$

Nous pouvons aussi définir le niveau de service sur tout l'horizon, $g_1(s)$ et $g_2(s)$, en remplaçant $S_G(s,t_1,t_2)$ par $S_G(s,0,T)$, $S(t_1,t_2)$ par $S(0,T)$, etc. dans les formules. L'égalité $g_l(s) = g_l(s,0,T)$, pour $l = 1,2$, ne tient que si T est déterministe. Cette même idée s'applique pour définir toute autre fonction d'espérances sur l'horizon entier.

Soit $A(t_1,t_2)$ le nombre d'arrivées pendant l'intervalle considéré. Si aucun contact n'est bloqué, $A(t_1,t_2) = S(t_1,t_2) + L(t_1,t_2)$. Le *pourcentage d'abandons* à long terme est alors

$$\ell(t_1,t_2) = \frac{\mathbb{E}[L(t_1,t_2)]}{\mathbb{E}[A(t_1,t_2)]}. \tag{2.3}$$

Une variante de ce taux consiste à remplacer $L(t_1,t_2)$ par $L_B(s,t_1,t_2)$ pour omettre les abandons se produisant trop rapidement et qui peuvent être causés par des défaillances du système ou une quantité inévitable d'usagers impatients.

Le *temps d'attente moyen* à long terme est quant à lui défini comme

$$w(t_1,t_2) = \frac{\mathbb{E}[W(t_1,t_2)]}{\mathbb{E}[A(t_1,t_2)]}, \tag{2.4}$$

où $W(t_1,t_2)$ est la somme des temps d'attente pour tous les contacts arrivés pendant l'intervalle $[t_1,t_2]$. Nous pouvons décomposer cette somme comme $W(t_1,t_2) = W_S(t_1,t_2) + W_L(t_1,t_2)$ où $W_S(t_1,t_2)$ est la somme des temps d'attente des contacts servis après être arrivés durant l'intervalle $[t_1,t_2]$ et $W_L(t_1,t_2)$ est la somme pour les contacts ayant abandonné. Deux variantes importantes du temps d'attente moyen sont alors le *temps de réponse moyen* $w_S(t_1,t_2) = \mathbb{E}[W_S(t_1,t_2)]/\mathbb{E}[S(t_1,t_2)]$ et le *temps moyen avant abandon* $w_L(t_1,t_2) = \mathbb{E}[W_L(t_1,t_2)]/\mathbb{E}[L(t_1,t_2)]$.

De plus, nous pouvons décomposer la somme des temps d'attente comme $W(t_1,t_2) = \sum_{j=1}^{A(t_1,t_2)} W_j$, où W_j est le temps d'attente observé pour le contact j. Cela signifie que si le contact j est servi, W_j correspond à son temps de réponse tandis que s'il abandonne, W_j est son temps avant abandon. Dans cette décomposition, nous pouvons remplacer chaque temps d'attente W_j par un temps d'excès $\max(W_j - s, 0)$. Cela permet d'ignorer les temps d'attente très courts et d'obtenir le *temps moyen d'excès*, qui correspond au temps moyen passé en file d'attente après avoir attendu le temps acceptable s. Le temps moyen d'excès est une mesure alternative de qualité de service fournissant parfois plus d'informations que le niveau de service [49]. Toutes ces mesures peuvent être aussi définies pour un type de contact k particulier en ne comptant que les contacts de ce type plutôt que tous

les contacts.

Une autre mesure de performance apparentée au temps d'attente est la taille moyenne de la file définie comme

$$q(t_1,t_2) = \frac{\mathbb{E}[Q(t_1,t_2)]}{t_2 - t_1} \tag{2.5}$$

où

$$Q(t_1,t_2) = \int_{t_1}^{t_2} Q(t)\, dt$$

et $Q(t)$ est la taille de la file d'attente au temps t. Pour définir $q(t_1,t_2)$ sur l'horizon entier, nous pouvons remplacer $t_2 - t_1$ par $\mathbb{E}[T]$ au dénominateur, mais il peut en résulter une taille moyenne de file très petite si le temps T de la dernière fin de service est beaucoup plus grand que le temps de fermeture t_P du centre de contacts. Pour éviter ce problème, il vaut mieux estimer la taille moyenne de la file seulement pendant les heures d'ouverture du centre de contacts.

Le *taux d'occupation des agents* à long terme est quant à lui défini par

$$o(t_1,t_2) = \frac{\text{Nombre moyen d'agents occupés}}{\text{Nombre moyen d'agents connectés}} = \frac{\mathbb{E}[N_B(t_1,t_2)]}{\mathbb{E}[N_B(t_1,t_2) + N_F(t_1,t_2)]} \tag{2.6}$$

où

$$N_B(t_1,t_2) = \int_{t_1}^{t_2} N_B(t)\, dt$$

est le temps total d'activité des agents pendant l'intervalle $[t_1,t_2]$, c'est-à-dire la somme de tout le temps passé par les agents à servir des contacts. Il se peut que $N_B(t_1,t_2) > t_2 - t_1$; par exemple, si $n > 1$ agents sont occupés pendant tout l'intervalle $[t_1,t_2]$, nous avons $N_B(t_1,t_2) = n(t_2 - t_1)$. De façon analogue, nous pouvons définir $N_F(t_1,t_2)$ comme le temps total de disponibilité des agents, $N_I(t_1,t_2)$ comme le temps total d'inactivité des agents (rappelons qu'un agent inactif peut être disponible ou non pour servir des contacts), $N_G(t_1,t_2)$ comme le temps total passé par des agents devant quitter le groupe après leur service et $N(t_1,t_2)$ comme le temps total de présence des agents dans le système.

Avec cette notation, nous pouvons définir

$$\tilde{N}(h,t_1,t_2) = \frac{\mathbb{E}[N(t_1,t_2)+N_G(t_1,t_2)]}{h} \tag{2.7}$$

comme le nombre d'*équivalents à temps plein* (*full-time equivalents* ou FTE en anglais) si chaque agent a un quart de travail de durée h pendant l'intervalle $[t_1,t_2]$. Soit également $\bar{N}(t_1,t_2) = \tilde{N}(t_2-t_1,t_1,t_2)$ le nombre moyen d'agents pendant l'intervalle $[t_1,t_2]$. Étant donné que $N(t_1,t_2)$ est habituellement déterministe, il arrive souvent que $\tilde{N}(h,t_1,t_2)$ est approximé en négligeant $N_G(t_1,t_2)$. Le taux d'occupation et le nombre d'agents moyens peuvent aussi être définis pour un groupe d'agents i particulier en ne comptant que les agents de ce groupe plutôt que tous les agents.

Dans certains modèles, nous souhaitons favoriser le service des contacts de type k par des agents de certains groupes tout en permettant le débordement vers des groupes secondaires. Nous nous intéressons alors à la qualité de l'affectation des contacts aux agents. Soit pour cela $S_{k,i}(t_1,t_2)$ le nombre de contacts de type k servis par des agents du groupe i et soit $S_k(t_1,t_2)$ le nombre de contacts de type k servis par n'importe quel agent. Pour chaque paire (k,i), nous pouvons estimer

$$s_{k,i}(t_1,t_2) = \frac{\mathbb{E}[S_{k,i}(t_1,t_2)]}{\mathbb{E}[S_k(t_1,t_2)]}, \tag{2.8}$$

la fraction à long terme des contacts de type k servis par des agents du groupe i. Pour le groupe d'agents primaire $i(k)$ des contacts de type k, nous souhaitons que le rapport $s_{k,i(k)}(t_1,t_2)$ soit élevé.

Par contre, il peut arriver qu'un contact de type k ait plusieurs groupes d'agents primaires, ce qui nous mène à généraliser la mesure de performance précédente de la façon suivante. Soit alors $0 \leq w_{k,i} \leq 1$ une pondération fixant l'importance pour un contact de

type k d'être servi par un agent du groupe i et supposons que, pour tout $k = 1, \ldots, K$,

$$\sum_{i=1}^{I} w_{k,i} = 1.$$

Nous supposons également que $w_{k,i} = 0$ si $k \notin S_i$. Dans un tel cas, la qualité de l'affectation des contacts aux agents peut être mesurée par la fonction d'espérances

$$s_k(w_{k,1}, \ldots, w_{k,I}, t_1, t_2) = \frac{\sum_{i=1}^{I} \mathbb{E}[w_{k,i} S_{k,i}(t_1, t_2)]}{\mathbb{E}[S_k(t_1, t_2)]} \tag{2.9}$$

pour le type de contact k et

$$s(\mathbf{w}, t_1, t_2) = \frac{\sum_{k=1}^{K} \sum_{i=1}^{I} \mathbb{E}[w_{k,i} S_{k,i}(t_1, t_2)]}{\mathbb{E}[S(t_1, t_2)]} \tag{2.10}$$

de façon globale, où $\mathbf{w}$ est la matrice de $K \times I$ regroupant les $w_{k,i}$. Si $K = I$ et le groupe d'agents k est le seul groupe primaire pour les contacts de type k, $\mathbf{w}$ correspond à la matrice identité.

Du côté des contacts sortants, les gestionnaires s'intéressent surtout au pourcentage de *mismatch* observé, c'est-à-dire la proportion d'usagers rejoints à un moment où aucun agent n'est disponible pour les servir par rapport au nombre total d'usagers rejoints. Ils examinent aussi le nombre de *right party connects* et l'impact des contacts sortants sur la qualité de service des contacts entrants.

Lors de l'estimation des mesures de performance correspondant à des rapports d'espérances, chaque événement relatif à un contact, par exemple son abandon ou la fin de son service, doit être compté dans la période de son arrivée et non dans celle où l'événement se produit. Ceci est nécessaire pour éviter d'introduire un biais dans les estimateurs. Par exemple, si plusieurs contacts arrivaient pendant la période p et étaient servis pendant la période $p+1$, la valeur du niveau de service dans la période $p+1$ pourrait dépasser 100% si tous les événements étaient comptés dans la période où ils se produisent.

2.3 Ordre asymptotique

La notion d'ordre asymptotique est utilisée pour comparer des fonctions entre elles. La notation pour cela, que nous rappelons ici, est présentée dans bon nombre de livres d'algorithmique ou de simulation, notamment [21, 58]. Pour toute fonction $g(n)$ dont le domaine correspond aux nombres réels non négatifs, nous pouvons définir l'ensemble

$$\begin{aligned}\mathcal{O}(g(n)) \quad = \quad & \{f(n) : \text{Il existe des constantes } c > 0 \text{ et } n_0 > 0 \text{ telles que } 0 \leq f(n) \leq cg(n) \\ & \text{pour tous } n \geq n_0 \text{ si } n \to \infty \text{ ou} \\ & \text{pour tous } n \leq n_0 \text{ si } n \to 0\}.\end{aligned}$$

Historiquement, la notation $f(n) = \mathcal{O}(g(n))$ est utilisée pour signifier que $f(n) \in \mathcal{O}(g(n))$. Cette appartenance signifie que si n est suffisamment grand ou suffisamment petit, selon le contexte, la fonction $f(n)$ est bornée à une constante près par la fonction $g(n)$. Par exemple, nous avons $2n^2 = \mathcal{O}(n^2)$ et $5n+1 = \mathcal{O}(n) = \mathcal{O}(n^2)$.

De façon semblable, nous définissons l'ensemble

$$\begin{aligned}o(g(n)) \quad = \quad & \{f(n) : \text{Pour tous } c > 0, \text{ il existe une constante } n_0 > 0 \text{ telle que } 0 \leq f(n) < cg(n) \\ & \text{pour tous } n \geq n_0 \text{ si } n \to \infty \text{ ou} \\ & \text{pour tous } n \leq n_0 \text{ si } n \to 0\}.\end{aligned}$$

Par exemple, nous avons $5n+1 = o(n^2)$, mais $2n^2 \neq o(n^2)$. La notation $f(n) = o(g(n))$ signifie aussi que $f(n)/g(n) \to 0$ si $n \to \infty$ ou $n \to 0$ selon le contexte, ce qui indique que $f(n)$ est asymptotiquement négligeable par rapport à $g(n)$.

CHAPITRE 3

COMBINAISON DE LA STRATIFICATION ET DES VARIABLES DE CONTRÔLE

Nous essayons d'abord et avant tout de réduire le temps de simulation d'une configuration particulière d'un modèle donné, le plus indépendamment possible du niveau de détails du modèle. Pour cela, il existe un très grand nombre de techniques pour réduire la variance dans les simulations. Par contre, la plupart de ces méthodes doivent être personnalisées pour les adapter à l'application donnée et il est parfois difficile de les utiliser pour un système complexe. Une autre avenue consiste à combiner plusieurs techniques, ce qui fait souvent surgir des problèmes insoupçonnés. De telles combinaisons ont été étudiées dans [8, 9, 18] pour certains cas particuliers. Nous souhaitons également appliquer les techniques sur des fonctions de plusieurs moyennes, par exemple le niveau de service dans un centre de contacts.

Dans ce chapitre, nous étudions la combinaison de la stratification sur une variable continue avec une variable de contrôle linéaire. Pour cela, nous exposons d'abord dans la section suivante la notation que nous avons utilisée ainsi que l'estimateur habituel de la variance dans le cas d'une fonction de plusieurs moyennes. Ensuite, dans les sections 3.2 et 3.3, nous présentons les variables de contrôle linéaires et la stratification séparément. L'adaptation de la première technique aux fonctions de plusieurs moyennes est connue tandis que celle de la seconde est innovatrice. Dans la section 3.4, nous expliquons ensuite comment combiner ces deux techniques et étudions la synergie qui existe entre elles. En dernier lieu, dans la section 3.5, nous appliquons ces idées sur des exemples de centres de contacts, en testant quelques variables de contrôle et de stratification ainsi que les techniques de combinaison.

Une partie de ce chapitre est publiée dans [61]. Ce dernier article est une extension de l'article préliminaire [60], qui applique également les variables aléatoires communes

dans le but d'estimer une différence. Nous reviendrons sur ce volet dans le chapitre suivant.

3.1 Variance sur une fonction de plusieurs moyennes

Cette section rappelle les formules usuelles concernant la variance d'une fonction de plusieurs moyennes, qui seront utilisées par la suite pour expliquer les variables de contrôle et la stratification. Nous nous intéressons à l'espérance $\boldsymbol{\mu} = \mathbb{E}[\mathbf{X}]$ d'un vecteur aléatoire $\mathbf{X} = (X_1, \ldots, X_d)$. Soit $g : \mathbb{R}^d \to \mathbb{R}$ une fonction continue et dérivable dans le voisinage de $\boldsymbol{\mu}$. La constante $v = g(\boldsymbol{\mu})$ est estimée par $\hat{v}_n = g(\hat{\boldsymbol{\mu}}_n)$, où $\hat{\boldsymbol{\mu}}_n$ est un estimateur asymptotiquement sans biais de $\boldsymbol{\mu}$, c'est-à-dire que $\mathbb{E}[\hat{\boldsymbol{\mu}}_n] \to \boldsymbol{\mu}$ si $n \to \infty$. Nous supposons également que $\hat{\boldsymbol{\mu}}_n \Rightarrow \boldsymbol{\mu}$, où $\Rightarrow$ dénote la convergence en loi. Cela signifie que quand $n \to \infty$, la fonction de répartition de $\hat{\boldsymbol{\mu}}_n$ tend vers celle d'une loi dégénérée générant $\boldsymbol{\mu}$ avec probabilité 1.

En général, $\hat{v}_n$ est un estimateur biaisé de v, à moins que $g(\boldsymbol{\mu})$ soit une fonction linéaire. Un cas important de fonction linéaire est $g(\boldsymbol{\mu}) = \mu_i$, c'est-à-dire une composante individuelle de $\boldsymbol{\mu}$.

L'expression de $\mathrm{Var}[g(\mathbf{X})]$ est habituellement inconnue, mais nous pouvons obtenir une expression relativement simple pour la variance de $\sqrt{n}\hat{v}_n$ quand $n \to \infty$, appelée *variance asymptotique*. Nous utiliserons cette variance asymptotique comme approximation de la variance en supposant que n est grand. Pour cela, le théorème de Taylor [76] indique que si $\hat{\boldsymbol{\mu}}_n$ est près de $\boldsymbol{\mu}$,

$$g(\hat{\boldsymbol{\mu}}_n) = g(\boldsymbol{\mu}) + (\nabla g(\hat{\boldsymbol{\xi}}_n))^{\mathrm{t}}(\hat{\boldsymbol{\mu}}_n - \boldsymbol{\mu}) \approx g(\boldsymbol{\mu}) + (\nabla g(\boldsymbol{\mu}))^{\mathrm{t}}(\hat{\boldsymbol{\mu}}_n - \boldsymbol{\mu}) \tag{3.1}$$

où $\hat{\boldsymbol{\xi}}_n \in \mathbb{R}^d$ se trouve sur le segment reliant $\boldsymbol{\mu}$ et $\hat{\boldsymbol{\mu}}_n$ et $\nabla g(\boldsymbol{\mu})$ est le gradient de $g(\boldsymbol{\mu})$ évalué à $\boldsymbol{\mu}$. Si nous supposons que le gradient est continu dans le voisinage de $\boldsymbol{\mu}$ et puisque la fonction est évaluée à $\hat{\boldsymbol{\mu}}_n$ avec $\hat{\boldsymbol{\mu}}_n \Rightarrow \boldsymbol{\mu}$ quand $n \to \infty$, alors $\hat{\boldsymbol{\xi}}_n \Rightarrow \boldsymbol{\mu}$ si bien que $\nabla g(\hat{\boldsymbol{\xi}}_n) \Rightarrow \nabla g(\boldsymbol{\mu})$. Nous pouvons alors supposer que l'approximation (3.1) devient

exacte à la limite, ce qui nous donne une expression pour la variance :

$$\begin{aligned}\sigma^2 &\stackrel{\text{def}}{=} \lim_{n\to\infty} n\text{Var}[g(\hat{\boldsymbol{\mu}}_n)] = \lim_{n\to\infty} n\text{Var}[g(\boldsymbol{\mu}) + (\nabla g(\boldsymbol{\mu}))^{\mathrm{t}}(\hat{\boldsymbol{\mu}}_n - \boldsymbol{\mu})] \\ &= \lim_{n\to\infty} n\text{Var}[(\nabla g(\boldsymbol{\mu}))^{\mathrm{t}}\hat{\boldsymbol{\mu}}_n] = \lim_{n\to\infty} n(\nabla g(\boldsymbol{\mu}))^{\mathrm{t}}\text{Cov}[\hat{\boldsymbol{\mu}}_n]\nabla g(\boldsymbol{\mu}) \\ &= (\nabla g(\boldsymbol{\mu}))^{\mathrm{t}}\boldsymbol{\Sigma}_{\mathrm{XX}}\nabla g(\boldsymbol{\mu}) \qquad (3.2)\end{aligned}$$

où $\boldsymbol{\Sigma}_{\mathrm{XX}} = \text{Cov}[\mathbf{X}] = n\text{Cov}[\hat{\boldsymbol{\mu}}_n]$ est une matrice de covariances de dimensions $d \times d$, définie positive. Si $g(\boldsymbol{\mu})$ est linéaire, le développement en série de Taylor correspond à la fonction et l'approximation devient exacte si bien que $\sigma^2 = n\text{Var}[\hat{v}_n] = \text{Var}[g(\mathbf{X})]$. En pratique, σ^2 est estimée en remplaçant $\boldsymbol{\mu}$ par $\hat{\boldsymbol{\mu}}_n$ et $\boldsymbol{\Sigma}_{\mathrm{XX}}$ par la matrice de covariances empiriques notée $\hat{\boldsymbol{\Sigma}}_{\mathrm{XX},n}$.

Nous pourrions croire que réduire la variance sur chaque composante du vecteur $\mathbf{X}$ devrait réduire celle sur toute la fonction $g(\mathbf{X})$, mais en général, tel n'est pas le cas à moins que $\boldsymbol{\Sigma}_{\mathrm{XX}}$ corresponde à la matrice diagonale. Prenons comme exemple concret un rapport de deux moyennes. Soit dans ce cas $\mathbf{X} = (X_1, X_2)$, $\boldsymbol{\mu} = (\mu_1, \mu_2)$,

$$\boldsymbol{\Sigma}_{\mathrm{XX}} = \begin{pmatrix} \sigma_1^2 & \sigma_{12} \\ \sigma_{12} & \sigma_2^2 \end{pmatrix},$$

$\sigma_1^2 = \text{Var}[X_1]$, $\sigma_2^2 = \text{Var}[X_2]$ et $\sigma_{12} = \text{Cov}[X_1, X_2]$. Nous avons $g(\boldsymbol{\mu}) = \mu_1/\mu_2$ et le gradient est $\nabla g(\mu_1, \mu_2) = (1/\mu_2, -\mu_1/\mu_2^2)^{\mathrm{t}}$. La variance, quant à elle, est

$$\sigma^2 = (\sigma_1^2 + \sigma_2^2 v^2 - 2\sigma_{12} v)/\mu_2^2. \qquad (3.3)$$

Pour réduire cette variance, il faut, d'après cette formule, réduire la variance au numérateur et au dénominateur ou augmenter la covariance entre les deux termes. Par contre, simplement réduire la variance au numérateur et au dénominateur séparément risque aussi de réduire la corrélation entre les deux parties du rapport, ce qui contribuerait à augmenter sa variance.

Nous souhaitons en général augmenter l'*efficacité* de $\hat{v}_n$ définie par

$$\text{Eff}[\hat{v}_n] = \frac{1}{\text{MSE}[\hat{v}_n]C[\hat{v}_n]} \tag{3.4}$$

où

$$\text{MSE}[\hat{v}_n] = \beta^2[\hat{v}_n] + \text{Var}[\hat{v}_n]$$

est l'*erreur quadratique moyenne*,

$$\beta[\hat{v}_n] = \mathbb{E}[\hat{v}_n] - v$$

est son *biais* et $C[\hat{v}_n]$ est son coût de calcul moyen par l'ordinateur. Selon cette définition, nous pouvons accroître l'efficacité d'un estimateur en réduisant son biais, sa variance ou son temps de calcul. Dans ce chapitre, nous nous concentrons sur la réduction de variance.

3.2 Variables de contrôle linéaires

Cette technique consiste à ajouter une somme pondérée de variables aléatoires d'espérance nulle à l'estimateur qui nous intéresse. Elle est étudiée pour le cas d'une seule variable aléatoire dans [37, 52]. Dans cette section, nous reprenons le cas d'une fonction de plusieurs moyennes, présenté dans [36, 38, 58].

Soit $\mathbf{C} \in \mathbb{R}^q$ un vecteur aléatoire dont $\mathbf{c} = \mathbb{E}[\mathbf{C}]$ est connue et pour lequel nous disposons d'un estimateur $\hat{\mathbf{c}}_n$ asymptotiquement sans biais. L'estimateur de v avec variables de contrôle, qui remplace simplement $g(\mathbf{X})$ par une autre fonction, est alors

$$\hat{v}_{\text{c},n} = g(\hat{\boldsymbol{\mu}}_n) - \boldsymbol{\beta}^{\text{t}}(\hat{\mathbf{c}}_n - \mathbf{c}) \tag{3.5}$$

où $\boldsymbol{\beta} \in \mathbb{R}^q$ est un vecteur colonne de coefficients à déterminer. Cet estimateur $\hat{v}_{\text{c},n}$, en général biaisé à moins que $g(\boldsymbol{\mu})$ soit linéaire, est une généralisation de l'estimateur plus

connu s'appliquant sur un simple scalaire.

Soit

$$\begin{aligned} \sigma_c^2 \stackrel{\text{def}}{=} \lim_{n\to\infty} n\text{Var}[\hat{v}_{c,n}] &= (\nabla h(\boldsymbol{\mu},\mathbf{c}))^t \boldsymbol{\Sigma}_c \nabla h(\boldsymbol{\mu},\mathbf{c}), \\ \text{où } \nabla h(\boldsymbol{\mu},\mathbf{c}) &= \begin{pmatrix} \nabla g(\boldsymbol{\mu}) \\ -\boldsymbol{\beta} \end{pmatrix} \\ \text{et } \boldsymbol{\Sigma}_c &= \text{Cov}\begin{bmatrix} \mathbf{X} \\ \mathbf{C} \end{bmatrix} = \begin{pmatrix} \boldsymbol{\Sigma}_{XX} & \boldsymbol{\Sigma}_{XC} \\ \boldsymbol{\Sigma}_{CX} & \boldsymbol{\Sigma}_{CC} \end{pmatrix}. \end{aligned}$$

Ici, $\boldsymbol{\Sigma}_{XX} = \text{Cov}[\mathbf{X}]$ est une matrice $d \times d$, $\boldsymbol{\Sigma}_{CC} = \text{Cov}[\mathbf{C}]$ est de dimensions $q \times q$ tandis que $\boldsymbol{\Sigma}_{XC}^t = \boldsymbol{\Sigma}_{CX} = \text{Cov}[\mathbf{C},\mathbf{X}]$ est de dimensions $q \times d$. Nous considérons que toutes ces matrices sont définies positives. Alors,

$$\sigma_c^2 = \sigma^2 - 2\boldsymbol{\beta}^t \boldsymbol{\Sigma}_{CX} \nabla g(\boldsymbol{\mu}) + \boldsymbol{\beta}^t \boldsymbol{\Sigma}_{CC} \boldsymbol{\beta}. \tag{3.6}$$

Selon le choix du vecteur $\boldsymbol{\beta}$, la variance asymptotique peut être réduite ou augmentée. Avec $\boldsymbol{\beta} = \mathbf{0}$, $\sigma_c^2 = \sigma^2$ (aucun changement de la variance). La valeur de $\boldsymbol{\beta}$ minimisant la variance asymptotique est donnée par

$$\boldsymbol{\beta}^* = (\boldsymbol{\Sigma}_{CC})^{-1} \boldsymbol{\Sigma}_{CX} \nabla g(\boldsymbol{\mu}). \tag{3.7}$$

Avec ce vecteur optimal, la variance devient

$$\sigma_c^2 = (1 - R_{CX}^2)\sigma^2 \tag{3.8}$$

où

$$R_{CX}^2 = \frac{(\nabla g(\boldsymbol{\mu}))^t \boldsymbol{\Sigma}_{CX}^t (\boldsymbol{\Sigma}_{CC})^{-1} \boldsymbol{\Sigma}_{CX} \nabla g(\boldsymbol{\mu})}{\sigma^2}$$

est le *coefficient de détermination* (ou le carré de la corrélation multiple) de $g(\mathbf{X})$ et $\mathbf{C}$. Si $\boldsymbol{\Sigma}_{CC}$ est définie positive, ce que nous avons supposé, alors $\sigma_c^2 \leq \sigma^2$. Plus cette valeur

est élevée, plus la variable de contrôle est efficace.

En pratique, $\boldsymbol{\mu}$ et $\boldsymbol{\Sigma}_{CX}$ sont inconnues et remplacées par leurs équivalents empiriques $\hat{\boldsymbol{\mu}}_n$ et $\hat{\boldsymbol{\Sigma}}_{CX,n}$ estimés soit par des expériences pilotes, soit avec les observations employées pour calculer $\hat{v}_{c,n}$. Même si $\boldsymbol{\Sigma}_{CC}$ est parfois connue, l'estimer elle aussi ajoute un contrôle non linéaire à l'estimateur et réduit davantage la variance [58, 72]. Si les matrices sont estimées avec les observations utilisées pour estimer v, cela ajoute du biais à l'estimateur $\hat{v}_{c,n}$, mais ce biais est négligeable si n est très grand.

3.3 Stratification

Cette technique, qui existe déjà depuis plusieurs années, consiste à subdiviser l'échantillon servant à estimer une moyenne en plusieurs strates dans lesquelles la variance est petite. Dans cette section, nous reprenons la description donnée dans [19, 58], mais nous la généralisons pour le cas d'une fonction de plusieurs moyennes, ce qui est innovateur. Nous traitons aussi le cas où nous stratifions par rapport à un vecteur de variables aléatoires continues.

Soit S une variable aléatoire ayant une grande corrélation avec $\mathbf{X}$ ou, mieux encore, avec $g(\mathbf{X})$. En supposant que S est discrète sur un support fini $\{1,\ldots,m\}$ et a une fonction de masse $p_s = \mathbb{P}[S = s]$, nous pouvons utiliser la probabilité totale pour récrire $\boldsymbol{\mu}$ comme

$$\boldsymbol{\mu} = \mathbb{E}[\mathbf{X}] = \sum_{s=1}^{m} p_s \boldsymbol{\mu}_s \qquad \text{avec} \qquad \boldsymbol{\mu}_s = \mathbb{E}[\mathbf{X} \mid S = s]$$

l'espérance de $\mathbf{X}$ étant donné que $S = s$. Soit également $\mu_{s,i}$ la composante i du vecteur $\boldsymbol{\mu}_s$. La *stratification* consiste à estimer $\boldsymbol{\mu}_s$ de façon indépendante pour chaque valeur de $S = s$, appelée *strate*, pour ensuite obtenir une estimation de $\boldsymbol{\mu}$. Cette technique fonctionne bien s'il est facile de générer une observation dans une strate s donnée.

Parfois, il n'est pas évident de trouver une telle variable de stratification S. Dans ce cas, nous pouvons utiliser la technique générale suivante pour stratifier. Sachant que tous les nombres aléatoires nécessaires pour la simulation proviennent d'un vecteur d'uni-

formes, nous pouvons choisir t de ces uniformes et les utiliser pour stratifier. Nous pouvons dans ce cas partitionner l'hypercube $[0,1)^t$ en m régions, avec p_s le volume de la région s. Chaque région est choisie de façon à ce que la variance induite par la variation de $\mathbf{U}$ dans la région soit petite, où $\mathbf{U}$ est un vecteur aléatoire uniforme sur $[0,1)^t$. Si aucune information sur la variance en fonction de $\mathbf{U}$ n'est disponible à priori, nous pouvons partitionner l'hybercube en m régions de volume identique et ainsi fixer $p_s = 1/m$. Habituellement, les régions correspondent à des boîtes rectangulaires et t est petit, par exemple 1 ou 2. Lorsque $t = 1$, les régions correspondent à des intervalles sur $[0,1)$.

Pour échantillonner dans la strate s, il suffit alors de générer $\mathbf{U}$ uniformément dans la région s, de fixer $S = s$ et d'utiliser $\mathbf{U}$ pour la simulation. Les autres uniformes utilisées pour générer $\mathbf{X}$ sont générées de la façon habituelle.

Soit n_s le nombre d'observations disponibles pour $S = s$ et

$$n = \sum_{s=1}^{m} n_s$$

le nombre total d'observations. Soit $\mathbf{X}_{s,j} \in \mathbb{R}^d$ l'observation j dans la strate s, pour $j = 1,\ldots,n_s$ et $s = 1,\ldots,m$. Soit également $X_{s,i,j}$ la composante i de $\mathbf{X}_{s,j}$. L'estimateur stratifié de $\boldsymbol{\mu}$ est alors

$$\bar{\mathbf{X}}_{\text{strat},n,m} = \sum_{s=1}^{m} p_s \hat{\boldsymbol{\mu}}_{s,n_s} \qquad \text{où} \qquad \hat{\boldsymbol{\mu}}_{s,n_s} = \frac{1}{n_s} \sum_{j=1}^{n_s} \mathbf{X}_{s,j} \tag{3.9}$$

est un estimateur de $\boldsymbol{\mu}_s$. La composante i de ce vecteur est notée $\bar{X}_{\text{strat},i,n,m}$. Soit $\boldsymbol{\Sigma}_{\text{strat}} \stackrel{\text{def}}{=} n\text{Cov}[\bar{\mathbf{X}}_{\text{strat},n,m}]$ la matrice de covariances de l'estimateur stratifié et $\boldsymbol{\Sigma}_s = \text{Cov}[\mathbf{X} \mid S = s]$ la matrice conditionnelle à la strate s. Alors,

$$\boldsymbol{\Sigma}_{\text{strat}} = n \sum_{s=1}^{m} \frac{p_s^2 \boldsymbol{\Sigma}_s}{n_s}. \tag{3.10}$$

Nous avons défini $\boldsymbol{\Sigma}_{\text{strat}}$ de cette façon afin qu'elle soit comparable à $\boldsymbol{\Sigma}_{\text{XX}}$.

Pour estimer $g(\boldsymbol{\mu})$ avec la stratification, nous calculons $\hat{v}_{\text{strat},n,m} = g(\bar{\mathbf{X}}_{\text{strat},n,m})$, c'est-à-dire la fonction avec l'estimateur stratifié (3.9). La variance asymptotique de cet estimateur est

$$\begin{aligned} \sigma^2_{\text{strat}} &\stackrel{\text{def}}{=} \lim_{n\to\infty} n\text{Var}[g(\bar{\mathbf{X}}_{\text{strat},n,m})] && (3.11)\\ &= (\nabla g(\boldsymbol{\mu}))^{\text{t}}\boldsymbol{\Sigma}_{\text{strat}}\nabla g(\boldsymbol{\mu}) && (3.12)\\ &= n\sum_{s=1}^{m}\frac{p_s^2\sigma_s^2}{n_s} \end{aligned}$$

où

$$\sigma_s^2 \stackrel{\text{def}}{=} (\nabla g(\boldsymbol{\mu}))^{\text{t}}\boldsymbol{\Sigma}_s\nabla g(\boldsymbol{\mu})$$

est la contribution de la strate s à la variance globale. Avec cette définition, si $g(\boldsymbol{\mu}) = \mu_i$, $\sigma^2_{\text{strat}} = n\text{Var}[\bar{X}_{\text{strat},i,n,m}]$, $\sigma_s^2 = \text{Var}[X_i \mid S = s] = \text{Var}[X_{s,i,j}]$ et nous retrouvons les formules habituelles pour la stratification.

Pour que $\sigma^2_{\text{strat}} < \infty$, nous supposons que si $n \to \infty$, $n/n_s \to k_s$ avec $0 < k_s < \infty$ pour $s = 1,\ldots,m$. Cette condition est satisfaite si, pour $s = 1,\ldots,m$, $n_s = f_s(n) = \mathcal{O}(n)$.

Notre variable de stratification S est choisie de façon à ce que plus le nombre m de strates est grand, plus σ_s^2 pour $s = 1,\ldots,m$ est petit. Par contre, avec un petit nombre de strates, l'estimation des matrices de covariances est plus précise pour un budget fixe de simulation puisque nous pouvons placer davantage d'observations dans les strates. Le choix de m est ainsi un compromis entre la réduction de la variance et la précision d'estimation de cette variance.

En supposant qu'il est possible de générer $\mathbf{X}$ conditionnellement à $S = s$ pour un s choisi, nous pouvons ajuster les valeurs de n_s avant la simulation. Il existe pour cela plusieurs mécanismes d'allocation. En particulier, avec l'*allocation proportionnelle* qui fixe $n_s = np_s$, la matrice de covariances $\boldsymbol{\Sigma}_{\text{strat}}$ et la variance σ^2_{strat} deviennent respectivement

$$\boldsymbol{\Sigma}_{\text{stratp}} \stackrel{\text{def}}{=} n\sum_{s=1}^{m}\frac{p_s^2\boldsymbol{\Sigma}_s}{np_s} = \sum_{s=1}^{m}p_s\boldsymbol{\Sigma}_s \quad \text{et} \quad \sigma^2_{\text{stratp}} = n\sum_{s=1}^{m}\frac{p_s^2\sigma_s^2}{np_s} = \sum_{s=1}^{m}p_s\sigma_s^2. \qquad (3.13)$$

Nous pouvons également rechercher l'*allocation optimale* visant à minimiser σ^2_{strat}. En utilisant un Lagrangien pour traiter la contrainte $n = \sum_{s=1}^{m} n_s$, nous obtenons [58]

$$n_s = \frac{n p_s \sigma_s}{\bar{\sigma}} \qquad \text{avec} \qquad \bar{\sigma} = \sum_{s=1}^{m} p_s \sigma_s. \tag{3.14}$$

La variance avec cette allocation est

$$\sigma^2_{\text{strato}} = n\bar{\sigma} \sum_{s=1}^{m} \frac{p_s^2 \sigma_s^2}{n p_s \sigma_s} = \bar{\sigma} \sum_{s=1}^{m} p_s \sigma_s = \bar{\sigma}^2. \tag{3.15}$$

Les valeurs de σ_s^2, inconnues au départ, doivent être estimées avant de pouvoir estimer les valeurs de n_s optimales. La méthode d'estimation la plus courante consiste à effectuer des expériences pilotes. Chaque valeur de n_s obtenue est ensuite arrondie à l'entier le plus près et fixée à 2 si elle est inférieure à ce nombre.

En combinant les résultats obtenus jusqu'à présent, nous pouvons décomposer la matrice de covariances de la façon suivante.

$$\begin{aligned}
& \boldsymbol{\Sigma}_{\mathbf{XX}} \\
= \; & \mathbb{E}[\mathrm{Cov}[\mathbf{X} \mid S]] + \mathrm{Cov}[\mathbb{E}[\mathbf{X} \mid S]] \\
= \; & \sum_{s=1}^{m} p_s \mathrm{Cov}[\mathbf{X} \mid S = s] \\
& + \mathbb{E}[(\mathbb{E}[\mathbf{X} \mid S = s] - \mathbb{E}[\mathbb{E}[\mathbf{X} \mid S = s]])(\mathbb{E}[\mathbf{X} \mid S = s] - \mathbb{E}[\mathbb{E}[\mathbf{X} \mid S = s]])^{\mathrm{t}}] \\
= \; & \sum_{s=1}^{m} p_s \boldsymbol{\Sigma}_s + \sum_{s=1}^{m} p_s (\boldsymbol{\mu}_s - \boldsymbol{\mu})(\boldsymbol{\mu}_s - \boldsymbol{\mu})^{\mathrm{t}} \qquad (3.16) \\
= \; & \boldsymbol{\Sigma}_{\text{stratp}} + \sum_{s=1}^{m} p_s (\boldsymbol{\mu}_s - \boldsymbol{\mu})(\boldsymbol{\mu}_s - \boldsymbol{\mu})^{\mathrm{t}}.
\end{aligned}$$

La dernière ligne montre que l'allocation proportionnelle supprime la covariance induite par la variance de $\boldsymbol{\mu}_s$ à l'intérieur des strates.

En remplaçant $\boldsymbol{\Sigma}_{\mathbf{XX}}$ par (3.16) dans l'expression (3.2) de σ^2, nous pouvons décom-

poser la variance asymptotique de la façon suivante.

$$\begin{aligned}
\sigma^2 &= (\nabla g(\boldsymbol{\mu}))^{\mathrm{t}} \boldsymbol{\Sigma}_{\mathrm{XX}} \nabla g(\boldsymbol{\mu}) \\
&= \sum_{s=1}^{m} p_s (\nabla g(\boldsymbol{\mu}))^{\mathrm{t}} \boldsymbol{\Sigma}_s \nabla g(\boldsymbol{\mu}) + \sum_{s=1}^{m} p_s (\nabla g(\boldsymbol{\mu}))^{\mathrm{t}} (\boldsymbol{\mu}_s - \boldsymbol{\mu})(\boldsymbol{\mu}_s - \boldsymbol{\mu})^{\mathrm{t}} \nabla g(\boldsymbol{\mu}) \\
&= \sum_{s=1}^{m} p_s \sigma_s^2 + \sum_{s=1}^{m} p_s (\nabla g(\boldsymbol{\mu}))^{\mathrm{t}} (\boldsymbol{\mu}_s - \boldsymbol{\mu})(\boldsymbol{\mu}_s - \boldsymbol{\mu})^{\mathrm{t}} \nabla g(\boldsymbol{\mu}) \\
&= \sigma^2_{\text{stratp}} + \sum_{s=1}^{m} p_s (\nabla g(\boldsymbol{\mu}))^{\mathrm{t}} (\boldsymbol{\mu}_s - \boldsymbol{\mu})(\boldsymbol{\mu}_s - \boldsymbol{\mu})^{\mathrm{t}} \nabla g(\boldsymbol{\mu}) \qquad (3.17) \\
&= \sigma^2_{\text{strato}} + \sum_{s=1}^{m} p_s (\sigma_s - \bar{\sigma})^2 \\
&\quad + \sum_{s=1}^{m} p_s (\nabla g(\boldsymbol{\mu}))^{\mathrm{t}} (\boldsymbol{\mu}_s - \boldsymbol{\mu})(\boldsymbol{\mu}_s - \boldsymbol{\mu})^{\mathrm{t}} \nabla g(\boldsymbol{\mu}). \qquad (3.18)
\end{aligned}$$

L'avant-dernière ligne de cette décomposition montre que l'allocation proportionnelle élimine la variance causée par la différence entre les moyennes dans les strates. La dernière ligne montre que l'allocation optimale élimine en plus la différence de variances entre les strates. De plus, si $g(\boldsymbol{\mu}) = \mu_i$, nous retrouvons la décomposition habituelle.

3.4 Combinaison de la stratification et d'une variable de contrôle

Combiner stratification et variables de contrôle n'est pas trivial et nous semble innovateur, car l'espérance de $\mathbf{C}$ et le vecteur de coefficients $\boldsymbol{\beta}$ peuvent varier d'une strate à l'autre ou même être des fonctions de $\mathbf{U}$ si nous stratifions par rapport à des variables continues. Plusieurs méthodes sont envisageables pour effectuer cette combinaison et le choix de la bonne méthode dépend de la situation. Dans cette section, nous analysons différentes méthodes de combinaison et discutons de la procédure à suivre pour implanter la combinaison en pratique. Nous testerons la combinaison sur des exemples de centres de contacts dans la section suivante.

Pour simplifier la notation, nous nous concentrons sur le cas unidimensionnel où $\mathbf{X} = (X)$, $\boldsymbol{\mu} = (\mu)$ et $g(X) = X$. Nous utilisons également une seule variable de contrôle et

une dimension de stratification si bien que $\mathbf{C} = (C)$, $\mathbf{c} = (c)$ et $\mathbf{U} = (U)$. Si nous utilisons plusieurs variables de contrôle, les coefficients et les espérances deviennent des vecteurs, les constantes sont estimées à l'aide de matrices de covariances et de vecteurs comme dans la section 3.2 et les corrélations deviennent des coefficients de détermination.

Comme nous l'avons vu précédemment, la variable de contrôle C choisie doit avoir une corrélation élevée avec X. Elle doit aussi dépendre de la variable de stratification S pour qu'une synergie s'établisse entre les deux méthodes de réduction de la variance. Par contre, si la corrélation entre C et S est trop élevée, l'effet de C risque d'être annulé par le conditionnement $S = s$ induit par la stratification. En particulier, si nous utilisons S comme variable de contrôle, la variable C devient constante dès que S est fixée si bien que nous devons choisir $\beta = 0$; la variable de contrôle n'a alors plus d'effet.

Toutes les méthodes que nous proposons dans cette section reviennent à remplacer les observations $X_{s,j}$ par

$$X_{\mathrm{sc},s,j} = X_{s,j} - b_{s,j}(C_{s,j} - e_{s,j})$$

avec différents choix de $b_{s,j}$ et de $e_{s,j}$ et où $C_{s,j}$ est la variable de contrôle associée à l'observation $X_{s,j}$. La valeur de $e_{s,j}$ peut être $c = \mathbb{E}[C]$, $c_s = \mathbb{E}[C \mid S = s] = \mathbb{E}[X_{s,j}]$ ou encore $c(U_{s,j}) = \mathbb{E}[C \mid U = U_{s,j}]$ si nous stratifions par rapport à une variable continue. Dans ce dernier cas, $U_{s,j}$ est l'uniforme utilisée pour générer les observations $X_{s,j}$ et $C_{s,j}$ correspondantes. La valeur de $b_{s,j}$ peut quant à elle être une constante globale β pour toutes les strates, une constante β_s locale à chaque strate s ou encore, dans le cas de la stratification sur une variable continue, une constante $\beta(U_{s,j})$ dépendant de la variable de stratification.

Le tableau 3.I présente les différentes possibilités de combinaisons. Chaque cellule donne le coefficient de correction de $X_{s,j}$ pour une valeur précise de $b_{s,j}$ et $e_{s,j}$. Les cellules marquées d'un trait indiquent des cas où l'estimateur peut être biaisé.

L'estimateur $X_{\mathrm{sc},s,j}$ est toujours sans biais si $b_{s,j}$ ne dépend pas de davantage d'infor-

Tableau 3.I – Possibilités pour $b_{s,j}(C_{s,j} - e_{s,j})$

	β	β_s	$\beta(U_{s,j})$
c	$\beta(C_{s,j} - c)$	—	—
c_s	$\beta(C_{s,j} - c_s)$	$\beta_s(C_{s,j} - c_s)$	—
$c(U_{s,j})$	$\beta(C_{s,j} - c(U_{s,j}))$	$\beta_s(C_{s,j} - c(U_{s,j}))$	$\beta(U_{s,j})(C_{s,j} - c(U_{s,j}))$

mations que $e_{s,j}$. Par exemple, si nous fixons $e_{s,j} = c$, nous devons utiliser une constante globale β pour obtenir un estimateur sans biais. Si $e_{s,j} = c_s$, nous pouvons utiliser une constante globale β ou une constante locale β_s. En particulier, si $e_{s,j} = c_s$ et $b_{s,j} = \beta_s$, $\mathbb{E}[b_{s,j}(C_{s,j} - e_{s,j})] = \mathbb{E}[\mathbb{E}[\beta_s(C_{s,j} - c_s) \mid S = s]] = \mathbb{E}[\beta_s(\mathbb{E}[C_{s,j} \mid S = s] - c_s)] = 0$. Par contre, cela ne fonctionne pas si $b_{s,j} = \beta(U_{s,j})$. Si $e_{s,j} = c(U_{s,j})$, nous pouvons alors utiliser l'une des trois possibilités pour $b_{s,j}$ et pouvons démontrer que $\mathbb{E}[b_{s,j}(C_{s,j} - c(U_{s,j}))] = 0$ en prenant l'espérance conditionnelle à $U_{s,j} = u_{s,j}$.

Si $e_{s,j} = c_s$ et la constante optimale est utilisée pour chaque strate, fixer $b_{s,j} = \beta_s$ réduit davantage la variance qu'avec la constante globale optimale $b_{s,j} = \beta$ qui impose la contrainte additionnelle $\beta = \beta_1 = \cdots = \beta_m$. De la même façon, si $e_{s,j} = c(U_{s,j})$ et la constante optimale $\beta^*(U_{s,j})$ est utilisée pour chaque valeur de $U_{s,j}$, il est impossible de réduire davantage la variance avec $b_{s,j} = \beta_s$ ou $b_{s,j} = \beta$ qu'avec $b_{s,j} = \beta(U_{s,j})$. Si nous éliminons les estimateurs qui peuvent être biaisés ou que nous savons d'avance moins efficaces que d'autres, il nous reste trois candidats qui correspondent aux cellules sur la diagonale du tableau 3.I. Comparer ces trois estimateurs nécessite une analyse plus approfondie que nous présentons maintenant.

3.4.1 Coefficient global pour toutes les strates

Si $e_{s,j} = c$ et $b_{s,j} = \beta$, nous obtenons l'estimateur

$$\bar{X}_{\text{stratcl},n,m} = \bar{X}_{\text{strat},n,m} - \beta(\bar{C}_{\text{strat},n,m} - c)$$

où

$$\bar{X}_{\text{strat},n,m} = \sum_{s=1}^{m} p_s \hat{\mu}_{s,n_s} = \sum_{s=1}^{m} p_s \sum_{j=1}^{n_s} X_{s,j}/n_s$$

est la version scalaire de $\bar{\mathbf{X}}_{\text{strat},n,m}$ de l'équation (3.9) et

$$\bar{C}_{\text{strat},n,m} = \sum_{s=1}^{m} p_s \hat{C}_{s,n_s} = \sum_{s=1}^{m} p_s \sum_{j=1}^{n_s} C_{s,j}/n_s$$

est la somme pondérée des observations $C_{s,j}$.

Puisque

$$\mathbb{E}[\bar{C}_{\text{strat},n,m}] = \sum_{s=1}^{m} p_s \mathbb{E}[\hat{C}_{s,n_s}] = \sum_{s=1}^{m} p_s c_s = c$$

et

$$\bar{X}_{\text{stratc1},n,m} = \sum_{s=1}^{m} p_s(\hat{\mu}_{s,n_s} - \beta(\hat{C}_{s,n_s} - c_s)),$$

ce mode de combinaison est équivalent à fixer $e_{s,j} = c_s$ et $b_{s,j} = \beta$.

Comme constante β, nous pouvons naïvement utiliser $\beta^* = \text{Cov}[X,C]/\text{Var}[C]$, qui correspond au coefficient optimal sans stratification. Par contre, comme nous le verrons, ce coefficient n'est pas optimal lorsque la stratification est mise en œuvre. La variance de l'estimateur stratifié est

$$\begin{aligned} \sigma^2_{\text{stratc1}} &\stackrel{\text{def}}{=} n\text{Var}[\bar{X}_{\text{stratc1},n,m}], \\ \text{Var}[\bar{X}_{\text{stratc1},n,m}] &= \text{Var}[\bar{X}_{\text{strat},n,m}] + \beta^2 \text{Var}[\bar{C}_{\text{strat},n,m}] - 2\beta \text{Cov}[\bar{X}_{\text{strat},n,m}, \bar{C}_{\text{strat},n,m}] \end{aligned}$$

si bien que la constante optimale permettant de minimiser $\sigma^2_{\text{stratc1}}$ est

$$\beta^*_{\text{sc}} = \frac{\text{Cov}[\bar{X}_{\text{strat},n,m}, \bar{C}_{\text{strat},n,m}]}{\text{Var}[\bar{C}_{\text{strat},n,m}]} = \frac{\sum_{s=1}^{m} p_s^2 \text{Cov}[\hat{\mu}_{s,n_s}, \hat{C}_{s,n_s}]}{\sum_{s=1}^{m} p_s^2 \text{Var}[\hat{C}_{s,n_s}]} = \frac{\sum_{s=1}^{m} p_s^2 \text{Cov}[X, C \mid S = s]/n_s}{\sum_{s=1}^{m} p_s^2 \text{Var}[C \mid S = s]/n_s}.$$

Avec cette technique, utiliser l'allocation optimale demande d'estimer β^*_{sc} et les n_s simultanément, ce qui n'est pas nécessairement facile. Si nous nous restreignons à l'allocation

proportionnelle, la constante optimale devient

$$\beta^*_{\text{scp}} = \frac{\sum_{s=1}^{m} p_s \text{Cov}[X, C \mid S = s]}{\sum_{s=1}^{m} p_s \text{Var}[C \mid S = s]}.$$

La variance avec cette constante optimale devient

$$\begin{aligned} \sigma^2_{\text{stratc1},n,m} &= (1 - \rho^2[\bar{X}_{\text{strat},n,m}, \bar{C}_{\text{strat},n,m}])\, \sigma^2_{\text{strat}} \\ &= \sigma^2_{\text{strat}} - \rho^2[\bar{X}_{\text{strat},n,m}, \bar{C}_{\text{strat},n,m}]\sigma^2_{\text{strat}} \end{aligned} \tag{3.19}$$

où $\rho^2[X,C]$ représente le carré de la corrélation linéaire entre X et C et $\sigma^2_{\text{strat}} = n\text{Var}[\bar{X}_{\text{strat},n,m}]$ est la variance sans variable de contrôle.

3.4.2 Coefficient local pour chaque strate

Si nous fixons $e_{s,j} = c_s$, nous pouvons utiliser un coefficient β_s différent pour chaque strate. Cela revient à remplacer l'estimateur $\hat{\mu}_{s,n_s}$ de chaque strate par

$$\hat{\mu}_{\text{c},s,n_s} = \hat{\mu}_{s,n_s} - \beta_s(\hat{C}_{s,n_s} - c_s).$$

La meilleure réduction de variance obtenue avec ce mode de combinaison est au moins la même que celle avec $e_{s,j} = c_s$ et $b_{s,j} = \beta$ puisque fixer $b_{s,j} = \beta_s$ permet une plus grande liberté dans le choix des constantes. L'estimateur stratifié devient

$$\bar{X}_{\text{stratc2},n,m} = \sum_{s=1}^{m} p_s \hat{\mu}_{\text{c},s,n_s}$$

et sa variance est

$$\sigma^2_{\text{stratc2}} = n \sum_{s=1}^{m} \frac{p_s^2 \sigma^2_{\text{sc},s}}{n_s}$$

où

$$\sigma^2_{\text{sc},s} = \sigma^2_s + \beta_s^2 \text{Var}[C \mid S = s] - 2\beta_s \text{Cov}[X, C \mid S = s]$$

est la variance avec variable de contrôle conditionnelle à la strate s. La constante optimale qui minimise la variance $\sigma^2_{\text{sc},s}$ est

$$\beta^*_{\text{sc},s} = \frac{\text{Cov}[X, C \mid S = s]}{\text{Var}[C \mid S = s]}.$$

Avec cette constante optimale, la variance conditionnelle à la strate s devient

$$\sigma^2_{\text{sc},s} = (1 - \rho^2[X, C \mid S = s])\,\sigma^2_s = \sigma^2_s - \rho^2[X, C \mid S = s]\sigma^2_s.$$

La variance globale devient quant à elle

$$\sigma^2_{\text{stratc2}} = \sigma^2_{\text{strat}} - n \sum_{s=1}^{m} \frac{p_s^2 \rho^2[X, C \mid S = s]\sigma^2_s}{n_s}. \tag{3.20}$$

L'espérance

$$c_s = \frac{1}{p_s} \int_{I(s)} c(u)du,$$

où $I(s)$ est le sous-intervalle de $[0, 1)$ correspondant à la strate s et $c(u) = \mathbb{E}[C \mid U = u]$, est souvent inconnue. Nous pouvons alors estimer c_s par intégration numérique si nous connaissons $c(u)$.

Avec cette méthode, nous pouvons utiliser l'allocation optimale en fixant n_s proportionnellement à $\sigma_{\text{sc},s}$ plutôt qu'à σ_s. Des expériences pilotes sont nécessaires pour estimer les variances ainsi que les coefficients $\beta^*_{\text{sc},s}$. Cette allocation diffère en général de l'allocation visant à minimiser la variance sans variables de contrôle.

3.4.3 Coefficient dépendant de la variable de stratification continue

Si nous connaissons $U_{s,j}$, l'uniforme à partir de laquelle S est générée, nous pouvons utiliser cette information pour appliquer la variable de contrôle $C_{s,j} - c(U_{s,j})$ sur chaque

observation $X_{s,j}$ qui devient alors

$$X_{\mathrm{sc},s,j} = X_{s,j} - \beta(U_{s,j})(C_{s,j} - c(U_{s,j})).$$

Nous avons alors la variance $\sigma^2(u) = \mathrm{Var}[X \mid U = u]$ et

$$\sigma^2_{\mathrm{sc}}(u) = \sigma^2(u) + \beta^2(u)\mathrm{Var}[C \mid U = u] - 2\beta(u)\mathrm{Cov}[X, C \mid U = u]$$

la variance conditionnelle avec variable de contrôle. Le coefficient optimal permettant de minimiser $\sigma^2_{\mathrm{sc}}(u)$ est donné par

$$\beta^*_{\mathrm{sc}}(u) = \frac{\mathrm{Cov}[X, C \mid U = u]}{\mathrm{Var}[C \mid U = u]}$$

et est une fonction de u. Avec cette constante optimale, la variance quand $U = u$ est

$$\sigma^2_{\mathrm{sc}}(u) = (1 - \rho^2[X, C \mid U = u])\,\sigma^2(u).$$

Pour analyser la variance conditionnelle à la strate s, nous devons utiliser la décomposition

$$\begin{aligned} \sigma^2_{\mathrm{scu},s} = \mathrm{Var}[X_{\mathrm{sc},s,j}] &= \mathbb{E}[\mathrm{Var}[X \mid U_{s,j}]] + \mathrm{Var}[\mathbb{E}[X \mid U_{s,j}]] \\ &= \mathbb{E}[\sigma^2_{\mathrm{sc}}(U_{s,j})] + \mathrm{Var}[\mu(U_{s,j})]. \end{aligned}$$

Ici, $U_{s,j}$ est une variable aléatoire uniforme sur l'intervalle correspondant à la strate s tandis que $\mu(u) = \mathbb{E}[X \mid U = u]$. Le premier terme de cette décomposition correspond à la moyenne de la variance restante lorsque $U_{s,j}$ est fixée tandis que le second représente la variance due au fait que $U_{s,j}$ est aléatoire conditionnellement à la strate s. Nous pouvons développer cette décomposition de la façon suivante :

$$\sigma^2_{\mathrm{scu},s} = \mathbb{E}[(1 - \rho^2[X, C \mid U_{s,j}])\,\sigma^2(U_{s,j})] + \mathrm{Var}[\mu(U_{s,j})]$$

$$
\begin{aligned}
&= \mathbb{E}[\sigma^2(U_{s,j})] + \mathrm{Var}[\mu(U_{s,j})] - \mathbb{E}[\rho^2[X,C \mid U_{s,j}]\sigma^2(U_{s,j})] \\
&= \sigma_s^2 - \mathbb{E}[\rho^2[X,C \mid U_{s,j}]\sigma^2(U_{s,j})].
\end{aligned}
$$

La variance globale devient enfin

$$
\sigma^2_{\mathrm{stratc3},n,m} = \sigma^2_{\mathrm{strat},n,m} - n\sum_{s=1}^{m} \frac{p_s^2 \mathbb{E}[\rho^2[X,C \mid U_{s,j}]\sigma^2(U_{s,j})]}{n_s}. \tag{3.21}
$$

Cette technique exige d'estimer la fonction $\beta^*_{\mathrm{sc}}(u)$ pour tout $u \in [0,1)$. Pour y parvenir, nous pouvons simuler avec différentes valeurs $U = u_1, u_2, \ldots$. Pour chaque valeur u_i choisie, nous estimons par simulation $\mathrm{Cov}[X,C \mid U = u_i]$ et $\mathrm{Var}[C \mid U = u_i]$ si elle est inconnue. Nous pouvons par la suite utiliser n'importe quelle méthode d'ajustement, par exemple les moindres carrés ou les splines lissées, pour estimer les deux courbes $q_1(u) = \mathrm{Cov}[X,C \mid U = u]$ et $q_2(u) = \mathrm{Var}[C \mid U = u]$. Cela nous fournit un estimateur pour la fonction $\beta^*_{\mathrm{sc}}(u) = q_1(u)/q_2(u)$.

Pour utiliser l'allocation optimale, il nous faut estimer $\sigma^2_{\mathrm{scu},s}$, ce qui requiert un estimateur pour $\mu(u)$. Sachant que $\mu = \int_0^1 \mu(u)du$, nous nous rendons compte que cela demande plus d'informations qu'estimer μ ! Une façon heuristique de contourner ce problème consiste à utiliser $\sigma^2_{\mathrm{sc},s}$, c'est-à-dire la variance dans le cas où on utilise une constante β_s pour chaque strate, pour établir l'allocation optimale. En pratique, cette allocation ne devrait pas trop différer de l'allocation optimale obtenue avec les variances $\sigma^2_{\mathrm{scu},s}$.

Si nous restreignons la fonction $\beta(u)$ de façon à la rendre constante pour chaque strate, la constante par strate sera donnée par

$$
\beta^*_{\mathrm{sc3},s} = \frac{\mathrm{Cov}[X, C_{s,j} - c(U_{s,j}) \mid S = s]}{\mathrm{Var}[C_{s,j} - c(U_{s,j}) \mid S = s]}.
$$

Cette constante est différente de $\beta^*_{\mathrm{sc},s}$ en général. Avec cette variante, on ne peut pas faire mieux qu'avec une fonction $\beta(u)$ non contrainte. Avec une constante globale β et

$e_{s,j} = c(U_{s,j})$, la contrainte sur $\beta(u)$ est encore plus forte si bien que la réduction de variance sera encore moindre.

3.4.4 Comparaison des trois meilleures techniques

Nous avons déjà vu que fixer une constante locale pour chaque strate fait au moins aussi bien, et souvent mieux, que fixer une constante globale. Par contre, il n'est pas certain qu'utiliser une constante dépendante de U est mieux que choisir une constante par strate. Intuitivement, nous pourrions croire que tel est le cas puisque la seconde méthode tient compte de davantage d'informations que la première. Mais le contre-exemple suivant suggéré par Roberto Szechtman (communication privée) montre qu'il est possible d'observer les deux comportements.

Exemple 1 Si nous prenons $X = C$ et utilisons $C - c_s$ comme variable de contrôle, nous avons l'estimateur $X_{\mathrm{sc},s,j} = X_{s,j} - \beta_s(C_{s,j} - c_s)$, avec $\beta_s = \beta^*_{\mathrm{sc},s} = 1$. Dans ce cas, $\mathrm{Var}[X_{\mathrm{sc},s,j}] = 0$ puisque $\rho[X_{s,j}, C_{s,j}] = 1$. Par contre, si nous prenons $C - c(U)$ comme variable de contrôle, nous avons $\beta(u) = \beta^*_{\mathrm{sc}}(u) = 1$ pour $u \in [0,1)$ et la variance devient $\mathrm{Var}[X_{s,j} - \beta(U_{s,j})(C_{s,j} - c(U_{s,j}))] = \mathrm{Var}[c(U_{s,j})] > 0$ si $c(U_{s,j})$ varie dans la strate s. Avec cet exemple, la première méthode réduit donc davantage la variance que la seconde.

Le cas inverse se produit si $X = C - c(U)$. Alors, on a toujours $\beta(u) = \beta^*_{\mathrm{sc}}(u) = 1$, mais la variance avec la variable de contrôle $C - c(U)$ est 0 puisque $X = 0$ pour n'importe quel U fixé. Par contre, $\mathrm{Var}[X_{\mathrm{sc},s,j}] > 0$ lorsque la variable de contrôle $C - c_s$ est employée. □

Pour comparer les deux estimateurs de façon générale, examinons la variance dans la strate s quand $e_{s,j} = c_s$.

$$\begin{aligned}
\sigma^2_{\mathrm{sc},s} &= \mathbb{E}[\mathrm{Var}[X_{s,j} - \beta^*_{\mathrm{sc},s}(C_{s,j} - c_s) \mid U_{s,j}]] + \mathrm{Var}[\mathbb{E}[X_{s,j} - \beta^*_{\mathrm{sc},s}(C_{s,j} - c_s) \mid U_{s,j}]] \\
&= \mathbb{E}[\mathrm{Var}[X_{s,j} - \beta^*_{\mathrm{sc},s}(C_{s,j} - c(U_{s,j})) \mid U_{s,j}]] \\
&\quad + \mathrm{Var}[\mu(U_{s,j}) - \beta^*_{\mathrm{sc},s}(c(U_{s,j}) - c_s)]. \qquad (3.22)
\end{aligned}$$

Dans le cas où $e_{s,j} = c(U_{s,j})$, la variance dans la strate s est

$$\sigma^2_{\text{scu},s} = \mathbb{E}[\sigma^2_{\text{sc}}(U_{s,j})] + \text{Var}[\mu(U_{s,j})]. \tag{3.23}$$

Si nous comparons les deux termes correspondants de (3.22) et (3.23), nous pouvons voir que

$$\begin{aligned} & \mathbb{E}[\text{Var}[X_{s,j} - \beta^*_{\text{sc},s}(C_{s,j} - c(U_{s,j})) \mid U_{s,j}]] \\ \geq\ & \mathbb{E}[\text{Var}[X_{s,j} - \beta^*_{\text{sc}}(U_{s,j})(C_{s,j} - c(U_{s,j})) \mid U_{s,j}]] \\ =\ & \mathbb{E}[\sigma^2_{\text{sc}}(U_{s,j})], \end{aligned}$$

mais il se peut que $\text{Var}[\mu(U_{s,j})] > \text{Var}[\mu(U_{s,j}) - \beta^*_{\text{sc},s}(c(U_{s,j}) - c_s)]$. En résumé, la technique utilisant des constantes β_s a pour avantage de réduire la variance de l'espérance conditionnelle en plus de l'espérance de la variance conditionnelle, mais la technique utilisant une fonction $\beta(u)$ se concentre sur l'espérance de la variance conditionnelle. Si $m \to \infty$, $\text{Var}[\mu(U_{s,j})] \to 0$, $\beta_s \to \beta(u)$ et $c_s \to c(u)$ si bien que les deux méthodes convergent vers la même variance.

Pour comparer les trois méthodes de façon plus générale, nous pouvons examiner les équations (3.19), (3.20) et (3.21) qui montrent que les trois méthodes que nous avons étudiées retranchent de la variance un coefficient de correction dépendant de corrélations entre X et C. Le tableau 3.II regroupe ces coefficients pour les trois méthodes. Plus le coefficient de correction donné dans le tableau est élevé, plus la méthode est efficace pour réduire la variance. Ainsi, pour que la méthode utilisant $\beta(u)$ soit efficace, il faut que X soit plus fortement corrélé avec $C - c(U)$ qu'avec $C - c_s$ et ce, pour la majorité des valeurs de U.

Tableau 3.II – Facteur retranché à la variance par les différents modes de combinaison de la stratification et des variables de contrôle

		Facteur retranché à	
$e_{s,j}$	$b_{s,j}$	σ_s^2	$\sigma^2_{\text{strat},n,m}$
c ou c_s	β^*_{sc}	—	$\rho^2[\bar{X}_{\text{strat},n,m}, \bar{C}_{\text{strat},n,m}]\sigma^2_{\text{strat},n,m}$
c_s	$\beta^*_{\text{sc},s}$	$\rho^2[X,C \mid S=s]\sigma_s^2$	$n\sum_{s=1}^m p_s^2\rho^2[X,C \mid S=s]\sigma_s^2/n_s$
$c(U_{s,j})$	$\beta^*_{\text{sc}}(U_{s,j})$	$\mathbb{E}[\rho^2[X,C \mid U_{s,j}]\sigma^2(U_{s,j})]$	$n\sum_{s=1}^m p_s^2\mathbb{E}[\rho^2[X,C \mid U_{s,j}]\sigma^2(U_{s,j})]/n_s$

3.4.5 Implantation de la combinaison

Nous décrivons ici comment il est possible d'implanter la technique expliquée dans cette section. Nous nous concentrons sur la méthode où $e_{s,j} = c_s$ et $b_{s,j} = \beta_s$ puisque cette méthode est la meilleure dans la plupart des cas, mais l'implantation des autres méthodes ne diffère pas beaucoup de celle-ci.

L'application de la combinaison nécessite les étapes suivantes.

1. Choix des variables de stratification ;
2. Choix des variables de contrôle ;
3. Sélection du nombre de strates ;
4. (Optionnel) Expériences pilotes pour estimer les quantités suivantes :
   - Les variances σ_s^2 dans les strates ;
   - Les constantes locales $\beta^*_{\text{sc},s}$;
   - Les variances $\sigma^2_{\text{sc},s}$ dans les strates avec variables de contrôle.
5. Simulation des réplications de production.

D'abord, il nous faut choisir des variables de stratification (souvent une ou deux) qui ont un impact important pendant la simulation. Dans le cas de variables continues, nous devons également déterminer une partition pour l'hybercube comme discuté à la section 3.3.

Nous choisissons ensuite une ou plusieurs variables de contrôle fortement corrélées avec X. Ces variables ne doivent pas être trop coûteuses à calculer et leur espérance doit être connue.

Déterminer le nombre de strates demande d'établir un équilibre entre une bonne réduction de la variance (obtenue avec beaucoup de strates) et un bon estimateur de la variance (obtenu avec peu de strates contenant beaucoup d'observations). Il faut également garder à l'esprit que plus le nombre de strates est élevé, plus il faut estimer de coefficients σ_s^2, $\beta_{\text{sc},s}^*$ et $\sigma_{\text{sc},s}^2$.

Avant de simuler les réplications servant à estimer μ, il est souvent nécessaire de consacrer quelques réplications pilotes à l'estimation des constantes $\beta_{\text{sc},s}^*$ et des variances $\sigma_{\text{sc},s}^2$ afin de pouvoir établir l'allocation optimale. Ces expériences ne servent qu'à estimer les constantes ; les observations obtenues sont exclues de l'estimateur de μ.

Si nous nous restreignons à l'allocation proportionnelle, il est aussi possible d'estimer les constantes $\beta_{\text{sc},s}^*$ avec les mêmes réplications que celles utilisées pour estimer μ, mais cela ajoute du biais à l'estimateur. Par contre, ce biais est faible si n est grand si bien que consacrer tout le budget aux réplications de production permet de réduire davantage la variance.

Dans certains modèles, nous pouvons estimer σ_s^2 sans expérience pilote. Par exemple, pour estimer le niveau de service dans un centre de contacts, nous pouvons utiliser une approximation telle que Erlang A. En multipliant ce niveau de service par le nombre moyen d'arrivées, nous obtenons une approximation du nombre de contacts servis ayant attendu moins de s_0 unités de temps qui peut être considéré comme une variable aléatoire binomiale dont la variance est connue. Cela nous donne une (grossière) estimation de l'écart-type qui peut être utilisée pour établir l'allocation optimale.

Si nous souhaitons combiner cela avec la variable de contrôle, toujours sans expérience pilote, nous pouvons supposer pour simplifier que $\sigma_{\text{sc},s} \approx \kappa\sigma_s$, c'est-à-dire que la variable de contrôle réduit la variance par un facteur semblable d'une strate à l'autre. Dans ce cas, l'allocation optimale établie avec les $\sigma_{\text{sc},s}^2$ est à peu près la même que celle établie avec les σ_s^2 si bien qu'il suffit d'estimer σ_s^2 comme dans le paragraphe précédent.

Lorsque le nombre de strates et le nombre d'observations par strate sont connus, nous pouvons effectuer les réplications de production. Ainsi, pour chaque strate $s = 1, \ldots, m$,

nous simulons n_s réplications afin de générer les observations $X_{s,j}$ et $C_{s,j}$. Nous obtenons alors l'estimé $\hat{\mu}_{s,n_s}$ puis $\hat{\mu}_{c,s,n_s}$ avec variable de contrôle.

3.5 Applications aux centres de contacts

Appliquons maintenant les techniques décrites dans la section précédente à un exemple concret de centre d'appels. Pour cela, nous présentons d'abord le modèle sur lequel nous avons effectué les expérimentations numériques. Ensuite, nous expérimentons avec quelques variables de contrôle et de stratification. Nous tentons enfin de combiner les meilleurs candidats et comparons les résultats obtenus.

3.5.1 Modèle considéré

Comme nous l'avons vu au chapitre 2, les centres de contacts sont des systèmes parfois très complexes comportant plusieurs types de contacts différents, des agents avec des niveaux de compétence variés, etc. Nous nous concentrons ici sur un exemple avec un seul type de contact et un seul groupe d'agents pour simplifier l'analyse des résultats. Par contre, outre cette simplification, l'exemple s'inspire d'un centre d'appels réel et les techniques que nous essayons peuvent s'appliquer sur des modèles plus complexes. Cet exemple est tiré du guide d'utilisation de SSJ [55] et utilisé dans [60, 61].

Les contacts arrivent selon un processus de Poisson à un taux aléatoire $B\lambda_p$ durant la période p. Les taux de base λ_p sont déterministes tandis que B est une variable aléatoire dont la loi de probabilité est gamma avec paramètres (α_0, α_0). Générée au début de chaque jour, la variable B a une moyenne de 1, une variance de $1/\alpha_0$ et représente le facteur d'achalandage du système [5]. Si $B > 1$, le taux d'arrivée des contacts est plus élevé que d'habitude. Si $B < 1$, il est inférieur à la moyenne. Ce processus permet d'augmenter la variance du nombre d'arrivées A puisque

$$\mathrm{Var}[A] = \mathrm{Var}[\mathbb{E}[A \mid B]] + \mathbb{E}[\mathrm{Var}[A \mid B]] = a^2\mathrm{Var}[B] + a = a^2/\alpha_0 + a,$$

où $a = \mathbb{E}[A]$ est le nombre moyen d'arrivées. Si $\alpha_0 \to \infty$, B prend toujours la valeur 1 puisque $\text{Var}[B] = 0$ et nous revenons au processus de Poisson non homogène.

Les contacts ne pouvant être servis immédiatement sont mis dans une file de type FIFO (*First in First Out* ou premier arrivé, premier servi) et abandonnent après un certain temps de patience. Les temps de patience sont i.i.d. et sont générés de la façon suivante : avec probabilité ρ, le temps de patience est 0, c'est-à-dire que le contact concerné abandonne immédiatement s'il ne peut pas être servi dès son arrivée. Avec probabilité $1-\rho$, le temps de patience est exponentiel de moyenne $1/\nu$. Les temps de service sont exponentiels avec moyenne $1/\mu$.

Pendant la période p, N_p agents sont disponibles pour servir des contacts. Si, à la fin de la période p, le nombre d'agents occupés est plus grand que N_{p+1}, les services en cours sont complétés et de nouveaux contacts ne sont acceptés que lorsque le nombre d'agents occupés devient inférieur à N_{p+1}. Pendant la période préliminaire, aucun agent n'est disponible tandis que $N_{P+1} = N_P$. Le tableau 3.III présente les paramètres initiaux avec lesquels nous avons testé cet exemple. Avec ces paramètres, le nombre total espéré d'arrivées est 1 660.

Nous nous intéressons ici à quatre mesures de performance : le niveau de service $g(s_0) = \mathbb{E}[G(s_0)]/\mathbb{E}[A]$, le nombre d'abandons $\ell = \mathbb{E}[L]/\mathbb{E}[A]$, le temps d'attente moyen $w = \mathbb{E}[W]/\mathbb{E}[A]$ ainsi que le taux d'occupation des agents $o = \mathbb{E}[N_B]/\mathbb{E}[N]$. Ici, $G(s_0) =$

Tableau 3.III – Paramètres de l'exemple avec un type de contact et un groupe d'agents

p	1	2	3	4	5	6	7	8	9	10	11	12	13
t_{p-1}	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00	19:00	20:00
N_p	4	6	8	8	8	7	8	8	6	6	4	4	4
λ_p	100/h	150/h	150/h	180/h	200/h	150/h	150/h	150/h	120/h	100/h	80/h	70/h	60/h

Périodes	Treize périodes d'une heure
α_0 pour B	10
Probabilité ρ d'abandon immédiat	0,1
Temps de patience moyen $1/\nu$	16min40s
Temps de service moyen $1/\mu$	1min40s

$S_G(s_0,0,T)+L_G(s_0,0,T)$ correspond au nombre de contacts ayant attendu moins de s_0 avant d'être servis ou d'abandonner (voir section 2.2), $L = L(0,T)$ représente le nombre d'abandons, $W = W_S(0,T)+W_L(0,T)$, la somme des temps d'attente, $A = S(0,T)+L(0,T)$ est le nombre d'arrivées, $N_B = N_B(0,T)$ est le temps total passé par tous les agents à servir des contacts et $N = N(0,T)$ est le temps total passé par tous les agents dans le système, pendant l'intervalle $[0,T]$. Ici, T correspond au temps de fin de la simulation. Nous avons défini les mesures de performance de telle sorte que les dénominateurs sont connus si bien que nous avons seulement estimé $\mathbb{E}[G(s_0)]$, $\mathbb{E}[L]$, $\mathbb{E}[W]$ et $\mathbb{E}[N_B]$ par $\bar{G}_n(s_0)$, $\bar{L}_n$, $\bar{W}_n$ et $\bar{N}_{Bn}$, respectivement. Nous avons utilisé $s_0 = 20$ secondes comme temps d'attente acceptable.

3.5.2 Test de variables de contrôle

Nous avons testé les variables de contrôle suivantes sur notre exemple :

- Le nombre $A - a$ d'arrivées ;
- Le facteur d'achalandage $B - 1$;
- Le temps de patience moyen $V - \mathbb{E}[V] = V - (1-\rho)/\nu$;
- Le temps de service moyen $1/\bar{\mu} - 1/\mu$;
- La somme des temps de service (*load*) $A/\bar{\mu} - a/\mu$;

Le tableau 3.IV présente les résultats pour $n = 10\,000$ avec les paramètres du tableau 3.III. Chacune des quatre parties du tableau se consacre à l'estimation d'une de mesure de performance : le nombre moyen de contacts ayant attendu moins de s_0 estimé par $\bar{G}_n(s_0)$, le nombre moyen d'abandons $\bar{L}_n$, le temps d'attente moyen $\bar{W}_n/a$ et le taux d'occupation des agents $\bar{N}_{Bn}/N$. La première colonne de chaque zone donne la moyenne obtenue par simulation. Les autres colonnes donnent respectivement la variable de contrôle testée (VC), l'estimé du coefficient de correction optimal $\hat{\beta}^*$, la variance empirique S_n^2 avec la variable de contrôle testée et le facteur de réduction de la variance. Ce facteur est le quotient des variances sans et avec variable de contrôle. Étant donné que ces variables de contrôle n'augmentent pas le temps de calcul de façon significative,

nous pouvons utiliser les facteurs de réduction de la variance comme mesure d'efficacité : plus le facteur est élevé, plus la variable correspondante est efficace. Chaque ligne du tableau est dédiée à une variable de contrôle, à l'exception de la première de chaque zone qui donne la variance sans variable de contrôle.

Ce tableau montre d'abord que le temps de patience et le temps de service moyens n'ont pratiquement aucun effet comme variables de contrôle. Chacun de ces temps n'a un effet direct que sur le contact concerné. Puisqu'il y a un nombre substantiel de contacts, la moyenne du temps de service est souvent très près de l'espérance et par conséquent faiblement corrélée avec le nombre $G(s_0)$ de contacts ayant attendu moins de s_0. Dans tous les exemples que nous avons testés, nous avons observé ce même résultat si bien que nous n'utiliserons plus ces variables de contrôle.

Par contre, A et B réduisent la variance d'un facteur 2 pour le nombre moyen $\bar{G}_n(s_0)$ de contacts ayant attendu moins de s_0 et le temps d'attente moyen $\bar{W}_n/a$, et d'un facteur 3 pour le nombre moyen $\bar{L}_n$ d'abandons. Pour ces trois mesures, les variables B et A sont à peu près équivalentes. Ces résultats montrent que lorsque B est fixée, la connaissance de A n'apporte pas beaucoup d'informations supplémentaires sur les mesures de performance testées. Combiner B avec A n'apporte aucune réduction additionnelle de la variance par rapport à utiliser A seule, car l'information de B est entièrement contenue dans A.

Dans notre modèle, un temps de service est généré pour tous les contacts, y compris ceux qui abandonnent. La variable de contrôle $A/\bar{\mu}$ somme tous ces temps de service. Si nous utilisions plutôt, comme variable de contrôle, la somme des durées de service pour les contacts servis seulement, l'espérance de cette variable serait inconnue, car elle dépendrait du nombre espéré d'abandons qui n'est pas connu en général.

Nous pourrions penser qu'utiliser $A/\bar{\mu}$ apporte une plus grande réduction de la variance que A puisque cette variable contient les temps de service de tous les contacts en plus du nombre d'arrivées. Dans cet exemple, $A/\bar{\mu}$ est à peine plus efficace que A et est même moins efficace dans le cas de $G(s_0)$. Ceci pourrait s'expliquer par le fait que $1/\bar{\mu}$

Tableau 3.IV – Résultats de l'application de variables de contrôle sur l'exemple de centre de contacts avec B aléatoire

$\bar{G}_n(s_0)$	VC	$\hat{\beta}^*$	S_n^2	Facteur	$\bar{L}_n$	$\hat{\beta}^*$	S_n^2	Facteur
1 418	—	—	77 418	1,00	60,9	—	9 622	1,00
	A	0,384	36 230	2,14		0,155	2 872	3,35
	B	636	36 515	2,12		258	2 908	3,30
	V	0,140	76 917	1,01		0,00974	9 620	1,00
	$1/\bar{\mu}$	−5,54	77 213	1,00		1,81	9 600	1,00
	$A/\bar{\mu}$	0,00378	37 207	2,08		0,00155	2 833	3,40
$\bar{W}_n/a$	VC	$\hat{\beta}^*$	S_n^2	Facteur	$\bar{N}_{\mathrm{B}n}/N$	$\hat{\beta}^*$	S_n^2	Facteur
12,7	—	—	905	1,00	54,8%	—	0,024	1,00
	A	0,0405	447	2,03		0,000290	0,00047	51,50
	B	67,2	449	2,02		0,481	0,00061	39,10
	V	0,00337	905	1,00		0,0000652	0,023	1,00
	$1/\bar{\mu}$	0,533	903	1,00		0,00493	0,024	1,01
	$A/\bar{\mu}$	0,000405	443	2,05		$2{,}9 * 10^{-6}$	0,00034	70,80

n'est pas efficace ici.

Dans le cas du taux d'occupation $\bar{N}_{\mathrm{B}n}/N$, les facteurs de réduction sont beaucoup plus grands et A réduit davantage la variance que B. Ces deux variables sont surpassées par $A/\bar{\mu}$ qui réduit ici la variance d'un facteur 71 comparativement à 52 pour A et 39 pour B. La variable N_{B} correspond ici à la somme des temps de service pour tous les contacts servis qui, s'il y a peu d'abandons, est proche de $A/\bar{\mu}$. Cela explique pourquoi $A/\bar{\mu}$ est efficace comme variable de contrôle sur le taux d'occupation.

Sachant que A et B sont à peu près aussi efficaces l'une que l'autre, nous avons tenté de fixer $B = 1$ pour voir ce qui se passe avec A. Cela revient à fixer $\alpha_0 = \infty$ de façon à ce que $\mathrm{Var}[B] = 0$. Comme le montre le tableau 3.V, qui présente ce nouveau cas, les variables de contrôle A et $A/\bar{\mu}$ sont beaucoup moins efficaces que dans le modèle original. Les facteurs de réduction de la variance ne dépassent pas 2, à l'exception du facteur pour le taux d'occupation des agents avec $A/\bar{\mu}$ comme variable de contrôle. Mais ce facteur de 43 est inférieur au facteur 71 obtenu quand B est aléatoire. La réduction est moindre en partie parce que fixer $B = 1$ supprime déjà une grande partie de la variance. Par contre, l'écart entre le facteur de réduction de la variance obtenue avec A et avec

$A/\bar{\mu}$ est plus grand avec $B = 1$ qu'avec B aléatoire puisque l'élimination de B comme source de variance augmente l'importance des temps de service dans la variance.

3.5.3 Comparaison avec l'estimation indirecte

Supposons que nous estimons le nombre moyen de contacts servis $\mathbb{E}[S]$ dans un cas où il n'y a que peu d'abandons. Sachant que $S = A - L$, nous pouvons remplacer S par l'estimateur indirect $a - L$. La variance de l'estimateur indirect est plus petite que celle de la variable aléatoire originale seulement si $\mathrm{Var}[L] < \mathrm{Var}[S]$. Conditionnellement au nombre d'arrivées A, les variables S et L sont approximativement binomiales avec paramètres A et p pour S et $1-p$ pour L, où p est la probabilité qu'un contact soit servi. Alors, S et L sont approximativement Poisson avec taux ap [79, p. 59] et $a(1-p)$, respectivement. Ainsi, l'efficacité de l'estimateur indirect varie en fonction d'une probabilité p qui est inconnue.

Si nous appliquons la variable de contrôle A sur S, nous obtenons l'estimateur $S - \beta(A-a)$ qui a été proposé dans [58, section 6.2]. Si $\beta = 1$, nous retrouvons l'estimateur indirect. Ainsi, la variable de contrôle réduit la variance aussi bien, sinon mieux, que l'estimateur indirect. Ce même raisonnement s'applique pour comparer l'estimateur indirect $a - S_{\mathrm{B}}(s_0) - L$ de $S_{\mathrm{G}}(s_0)$ ainsi que sur d'autres estimateurs semblables : la variable de contrôle A est au moins aussi efficace.

Supposons maintenant que nous estimons le niveau de service avec la fonction de

Tableau 3.V – Résultats de l'application de variables de contrôle sur l'exemple de centre de contacts avec $B = 1$

$\bar{G}_n(s_0)$	VC	$\hat{\beta}^*$	S_n^2	Facteur	$\bar{L}_n$	$\hat{\beta}^*$	S_n^2	Facteur
1 538	—	—	1 427	1,00	32,3	—	60,7	1,00
	A	0,636	757	1,88		0.089	47,6	1.27
	$A/\bar{\mu}$	0,0016	1 341	1,06		0,00080	39,5	1,53
$\bar{W}_n/a$	VC	$\hat{\beta}^*$	S_n^2	Facteur	$\bar{N}_{\mathrm{B}n}/N$	$\hat{\beta}^*$	S_n^2	Facteur
4,59	—	—	2,23	1,00	55,8%	—	0,00034	1,00
	A	0,0142	1,89	1,18		0.000315	0,00017	1,96
	$A/\bar{\mu}$	0,000138	1,59	1,40		$3,1 * 10^{-6}$	$7,9 * 10^{-6}$	42,70

plusieurs moyennes

$$\frac{\bar{S}_{\mathrm{G}n}(s_0)}{\bar{S}_n + \bar{L}_{\mathrm{B}n}(s_0)}$$

où $\bar{S}_{\mathrm{G}n}(s_0)$, $\bar{S}_n$ et $\bar{L}_{\mathrm{B}n}(s_0)$ sont des estimateurs Monte Carlo de $S_{\mathrm{G}}(s_0)$, $S(0,T)$ et $L_{\mathrm{B}}(s_0)$ respectivement. L'estimation indirecte, appliquée séparément sur le numérateur et le dénominateur, semble intéressante à première vue, mais il faut garder à l'esprit que réduire la variance sur ces deux parties du rapport séparément réduit aussi la corrélation entre ces deux parties. Si nous appliquons plutôt la variable de contrôle $\bar{A}_n$, correspondant au nombre moyen d'arrivées pour n réplications, nous obtenons

$$\frac{\bar{S}_{\mathrm{G}n}(s_0)}{\bar{S}_n + \bar{L}_{\mathrm{B}n}(s_0)} - \beta_1(\bar{A}_n - a) = \frac{\bar{A}_n - \bar{S}_{\mathrm{B}n}(s_0) - \bar{L}_n}{\bar{S}_n + \bar{L}_{\mathrm{B}n}(s_0)} - \beta_1(\bar{A}_n - a).$$

Avec $\beta_1 = 1/(\bar{S}_n + \bar{L}_{\mathrm{B}n}(s_0))$, nous retrouvons un estimateur indirect appliqué sur le numérateur. Comme le montre l'équation (3.7), le coefficient optimal pour la variable de contrôle s'ajuste en fonction de la corrélation entre la variable et les deux composantes du rapport ainsi qu'à la corrélation existante entre ces composantes tandis que l'estimation indirecte ne tient compte que de la variance au numérateur. C'est pourquoi nous pouvons nous attendre à ce que cette variable fasse mieux que l'estimation indirecte, même pour le niveau de service.

3.5.4 Mesures de dispersion des arrivées comme variable de contrôle

Comme nous l'avons vu, B et A sont les variables de contrôle les plus efficaces pour notre exemple. Par contre, la distribution des arrivées au cours de la journée pourrait avoir un plus grand impact que le nombre d'arrivées. C'est pourquoi nous avons tenté de remplacer A par une mesure de dispersion utilisant les temps d'arrivées uniformisés.

Supposons pour commencer que $B = 1$. Soit $\lambda(t)$ le taux d'arrivée au temps t et $\Lambda(t) = \int_0^t \lambda(s)ds$ le taux d'arrivée cumulatif au temps t, c'est-à-dire le nombre moyen d'arrivées se produisant dans l'intervalle $[0,t]$. Soient $T_1, T_2, \ldots, T_n$ les temps d'arrivée des contacts pour une journée donnée. Puisque ces temps sont générés par un processus

de Poisson de taux $\lambda(t)$ au temps t, les temps $\tilde{T}_j = \Lambda(T_j)$, pour $j = 1,\ldots,n$, peuvent être produits à partir d'un processus de Poisson de taux 1. La valeur maximale de $\tilde{T}_j$ est $\Lambda(T) = a$, où T est le temps de fin de la simulation et $a = \mathbb{E}[A]$ est le nombre moyen d'arrivées par jour. Soit enfin $U_{(j)} = \tilde{T}_j/a$ pour $j = 1,\ldots,n$ les temps normalisés sur $[0,1]$. Fixons $U_{(0)} = 0$ et $U_{(n+1)} = 1$.

Comme variable de contrôle, nous pouvons utiliser la moyenne M des temps inter-arrivées normalisés. Étant donné que les arrivées suivent un processus de Poisson, ces temps sont i.i.d. et suivent la loi exponentielle de moyenne $1/a$. La somme des temps normalisés est proche de 1 et doit être divisée par A, ce qui revient approximativement à utiliser $1/A$ comme variable de contrôle.

Comme première mesure de dispersion, nous pouvons utiliser la variance empirique Σ des temps inter-arrivées qui a comme espérance $1/a^2$.

Maintenant, si nous conditionnons sur $A = n$, nous savons que les $U_{(j)}$ sont uniformément distribués sur $[0,1)$ et cela nous ramène au cadre défini dans [54]. Nous pouvons alors utiliser toutes les mesures de regroupement (*clustering* en anglais) proposées dans cet article. En particulier, la somme des carrés des espacements est donnée par

$$S_{m,n} = \sum_{i=0}^{n-m+1} (n(U_{(i+m)} - U_{(i)}))^2.$$

D'après [54],

$$\mathbb{E}[S_{m,n}] = \frac{m(m+1)(n-m+2)n^2}{(n+1)(n+2)} = \frac{m(m+1)(n^3-(m-2)n^2)}{n^2+3n+2}.$$

La valeur m est un paramètre déterministe fixé par l'utilisateur.

La variable de contrôle utilisée est $S_m = S_{A,m}$ et son espérance est

$$\mathbb{E}[S_m] = \sum_{x=0}^{\infty} p(x)\,\mathbb{E}[S_{x,m}] = \sum_{x=0}^{\infty} \left(\frac{e^{-a}a^x}{x!}\right) \mathbb{E}[S_{x,m}]$$

où $p(x)$ est la fonction de masse de la loi Poisson de taux a. Cette espérance doit bien

entendu être estimée numériquement en tronquant la sommation, ce qui ajoute du biais dans l'estimateur. Les bornes de x, notées L et R, peuvent par exemple être choisies par la méthode de Fox et Glynn [30] qui permet d'avoir $\mathbb{P}[L \leq A \leq R] \geq 1 - \varepsilon$ pour n'importe quel $\varepsilon > 0$.

Si B est aléatoire, nous devons utiliser l'intégration numérique pour calculer l'espérance de M et Σ. De même, pour calculer l'espérance $\mathbb{E}[S_m]$, nous devons estimer l'intégrale

$$\mathbb{E}[S_m] = \int_0^1 \mathbb{E}[S_m \mid B = F_B^{-1}(u)]du$$

avec une méthode d'intégration numérique. Ici, $F_B^{-1}(u)$ est la fonction de répartition inverse de B.

Le tableau 3.VI montre les résultats avec ces trois nouvelles variables de contrôle appliquées à notre exemple de centre de contacts. Comme nous pouvons le voir en comparant ce tableau avec le tableau 3.V, la variable M a exactement le même effet que A et Σ est encore moins efficace que M. L'effet de S_m sur la variance des quatre mesures testées se compare lui aussi à celui de A. Pour cette expérience, nous avons testé avec $m = 50$. Avec $m = 10$ et $m = 100$, les facteurs de réduction de la variance sont à peu près les mêmes. La valeur de m a ainsi peu d'effet sur la réduction de la variance dans cet exemple.

Le tableau 3.VII montre les résultats si B est aléatoire. Dans le cas de $G(s_0)$, la variable M surpasse la variable A d'un facteur d'environ 2, mais A donne le meilleur facteur de réduction de la variance pour L. Cela pourrait s'expliquer par le fait que la corrélation linéaire entre $G(s_0)$ et $1/A$ est plus grande que celle avec A quand A suit la loi binomiale négative comme c'est le cas avec le processus d'arrivées doublement stochastique. Ceci est possible puisque $G(s_0)$ n'est pas une fonction linéaire de A : le nombre de contacts servis diminue d'abord lentement, puis de plus en plus rapidement, tandis que A augmente.

Nous avons également tenté de remplacer la somme des carrés par la somme des

Tableau 3.VI – Variance avec les mesures de dispersion des arrivées, pour $B = 1$

$\bar{G}_n(s_0)$	VC	$\hat{\beta}^*$	S_n^2	Facteur	$\bar{L}_n$	$\hat{\beta}^*$	S_n^2	Facteur
	M	$-1,75*10^6$	757	1,88		$-2,44*10^5$	47,7	1.27
	Σ	$-8,39*10^8$	983	1,45		$-7,42*10^7$	57,2	1,06
	M,Σ	$(-1,49*10^6,$ $-2.22*10^8)$	742	1,92		$(-3,09*10^5,$ $5,43*10^7)$	46,7	1,30
	S_m	0,000217	870	1,64		$3,41*10^{-5}$	46,9	1,29
$\bar{W}_n/a$	VC	$\hat{\beta}^*$	S_n^2	Facteur	$\bar{N}_{Bn}/N$	$\hat{\beta}^*$	S_n^2	Facteur
	M	$-3,91*10^4$	1,89	1,18		-868	$1,71*10^{-4}$	1,96
	Σ	$-1,15*10^7$	2,14	1,04		$-3,67*10^5$	$2,50*10^{-4}$	1,34
	M,Σ	$(-50\,452,$ $9,47*10^6)$	1,87	1,19		$(-853,$ $-12\,732)$	0,000171	1,96
	S_m	$5,59*10^{-6}$	1,86	1,20		$1,12*10^{-7}$	0,000187	1,79

Tableau 3.VII – Variance avec les mesures de dispersion des arrivées, pour $B \sim$gamma

$\bar{G}_n(s_0)$	VC	$\hat{\beta}^*$	S_n^2	Facteur	$\bar{L}_n$	$\hat{\beta}^*$	S_n^2	Facteur
	M	$-1,04*10^6$	16 363	4,73		$-236\,387$	6 461	1.49
	Σ	$-5,51*10^8$	23 234	3,33		$-9,76*10^7$	7 921	1,21
	M,Σ	$(-1,22*10^6,$ $1,08*10^8)$	16 190	4,78		$(-844\,576,$ $3,57694*10^8)$	4 558	2,11
	S_m	0,00015	36 264	2,13		$6,09*10^{-5}$	2 876	3,35
$\bar{W}_n(s_0)/a$	VC	$\hat{\beta}^*$	S_n^2	Facteur	$\bar{N}_{Bn}/N$	$\hat{\beta}^*$	S_n^2	Facteur
	M	$-56\,447$	725	1,25		-601	0,00353	6,78
	Σ	$-2,25*10^7$	815	1,11		$-290\,239$	0,0089	2,68
	M,Σ	$(-219\,253,$ $9,58*10^7)$	588	1,54		$(-1\,288,$ $404\,437)$	0,0011	21,8
	S_m	$1,59*10^{-5}$	447	2,02		$1,14*10^{-7}$	0,000477	50,2

logarithmes sans obtenir d'amélioration significative. Ces variables de contrôle ne fonctionnent pas très bien, car elles correspondent à des statistiques sur un grand nombre de contacts. Leur effet est ainsi dilué à l'intérieur de l'horizon simulé. Par contre, nous avons tenté de simuler sur des horizons plus courts, mais toujours sans obtenir d'amélioration notable.

3.5.5 Application de la combinaison avec la stratification

Nous avons vu que la variable B avait un impact important sur la performance du système si bien que nous l'avons choisie comme variable de stratification. Pour ce faire, nous avons divisé l'intervalle $[0,1)$ en m régions de longueur $1/m$ (ainsi, $p_s = 1/m$). Pour générer une observation dans la strate s, il suffit alors de générer $U_{s,j}$ uniformément sur $[(s-1)/m, s/m)$, de prendre $B = F_B^{-1}(U_{s,j})$ et de générer tous les autres nombres aléatoires de la façon habituelle.

Comme variables de contrôle, nous avons testé le nombre A d'arrivées. Nous savons que $\mathbb{E}[A] = a$ et $\mathrm{Var}[A] = a^2/\alpha_0 + a$. Étant donné que A suit la loi de Poisson conditionnellement à B, nous savons que $\mathbb{E}[A \mid U = u] = \mathrm{Var}[A \mid U = u] = aF_B^{-1}(u)$. Par contre, pour estimer $\mathbb{E}[A \mid S = s]$, nous devons évaluer l'intégrale

$$\mathbb{E}[A_{s,j}] = m\int_{(s-1)/m}^{s/m} a(u)du = am\int_{(s-1)/m}^{s/m} F_B^{-1}(u)du = am\int_{F_B^{-1}((s-1)/m)}^{F_B^{-1}(s/m)} bf_B(b)db$$

où $f_B(b)$ est la fonction de densité de B, évaluée à b.

Plutôt que simuler quelques réplications avec les différentes méthodes comme on le ferait pour les appliquer en pratique et comparer les résultats, nous avons choisi d'effectuer les expérimentations de façon plus exhaustive afin d'obtenir une meilleure précision dans nos comparaisons. D'abord, pour chacun des points $U = u_i = (i+0,5)/1\,000$, pour $i = 0,\ldots,999$, nous avons simulé $n = 10\,000$ réplications indépendantes afin d'obtenir des estimés pour $\mu(u_i)$, $\sigma^2(u_i) = \mathrm{Var}[X \mid U = u_i]$, $\mathrm{Cov}[X,C \mid U = u_i]$ et $\beta^*_{sc}(u_i)$. Puisque le nombre d'arrivées est Poisson conditionnellement à B, nous savons en théorie que

$\mathrm{Var}[A \mid U = u_i] = aF_{\mathrm{B}}^{-1}(u_i)$ si bien qu'il n'est pas nécessaire de l'estimer.

Nous avons ensuite utilisé ces mille points pour construire des splines cubiques lissées [24] estimant $\mu(u)$, $\sigma^2(u)$, $\mathrm{Cov}[X,C \mid U = u]$ et $\beta_{\mathrm{sc}}^*(u)$. Une telle spline est composée de $n+1$ polynômes cubiques dont les coefficients sont estimés à partir de points (U_i,Y_i). Le i^{e} polynôme de la spline, pour $i = 1,\ldots,n-1$, est $S_i(u)$ tandis que la spline complète est définie par

$$S(u) = S_i(u), \qquad \text{pour } u \in [U_{i-1},U_i].$$

Pour $u < U_0$ et $u > U_{n-1}$, la spline n'est pas définie avec précision, mais l'algorithme utilisé effectue une extrapolation en utilisant des polynômes linéaires notés $S_0(u)$ pour $u < U_0$ et $S_n(u)$ pour $u > U_{n-1}$. Le polynôme $S_i(u)$ est lié au polynôme $S_{i+1}(u)$ de façon à ce que $S_i(U_{i+1}) = S_{i+1}(U_{i+1})$, $S_i'(U_{i+1}) = S_{i+1}'(U_{i+1})$ et $S_i''(U_{i+1}) = S_{i+1}''(U_{i+1})$.

L'algorithme de lissage minimise la fonction

$$\rho \sum_{i=0}^{n-1} ((Y_i - S_i(U_i))w_i)^2 + (1-\rho) \int_{U_0}^{U_{n-1}} \left(S''(u)\right)^2 du$$

où $\rho \in [0,1]$ est un paramètre du lissage. Plus ρ est près de 1, plus la spline passe par les points (U_i,Y_i), mais moins elle est lisse. Si $\rho = 1$, nous avons une spline d'interpolation tandis qu'avec $\rho = 0$, nous avons approximativement une droite.

Les poids $w_i > 0$ permettent d'ajuster la contribution de chaque point à l'erreur. Plus w_i est élevé, plus le point (U_i,Y_i) correspondant est considéré important et plus la spline passe près de ce point. Nous avons utilisé $w_i = 1$ pour tous les points, mais dans le cas où les extrémités $(0,Y_0)$ et $(1,Y_0)$ sont connues, nous avons utilisé une très grande valeur de w_0 et w_{n-1} pour forcer la spline à toucher ces points.

Ces fonctions permettent de calculer, par intégration numérique, un estimé de μ ainsi que chacun des termes de la décomposition (3.18). Cela nous indique que $\mu \approx 1\,419$ dans le cas de $G(s_0)$ et $\mu \approx 60,5$ pour L.

La figure 3.1 montre la courbe $\beta^*_{sc}(u)$ dans le cas de $G(s_0)$, qui correspond au nombre d'appels ayant attendu moins de s_0, et la variable de contrôle A. Il est intéressant de remarquer que la correction optimale appliquée par la variable de contrôle est positive pour certaines valeurs de U et négative pour d'autres. Pour expliquer cela, nous devons garder à l'esprit que la constante $\beta^*_{sc}(u)$ est très fortement influencée par la corrélation entre X et C. Une petite valeur de U produit un faible facteur d'achalandage, d'où un petit nombre d'arrivées. La charge de travail des agents est alors petite si bien que peu d'appels doivent attendre. Si U est suffisamment petit mais non nul, nous avons même $X = A$ si bien que $\lim_{U \to 0} \beta^*_{sc}(U) = 1$. Dans ce cas, ajouter quelques appels (en augmentant U) fait augmenter le nombre d'appels ayant attendu moins de s_0, d'où la corrélation positive. Vers $u = 0,86$, le trafic devient par contre si élevé que l'ajout de quelques nouveaux appels augmente les temps d'attente et réduit le nombre d'appels ayant attendu moins de s_0, d'où la corrélation négative entre X et A. Si le trafic est encore plus élevé, à peu près à $u = 0,95$, la charge devient si énorme que très peu d'appels peuvent être servis sans attendre très longtemps, d'où une réduction radicale de la corrélation entre X et A. Quand $u \to 1$, le nombre d'arrivées tend vers l'infini et le nombre d'appels servis tend vers 0, d'où une corrélation nulle entre X et A. Le graphique de $\mathrm{Cov}[X, A \mid U = u]$ ressemble beaucoup à celui de $\beta^*_{sc}(u)$ si bien que nous ne le présentons pas ici.

La figure 3.2 montre comment la variance de X évolue avec l'augmentation du nombre d'arrivées. D'après ce graphique, la fonction $\sigma^2(u)$ est croissante jusqu'à environ $u = 0,86$. À ce point, les temps d'attente sont très élevés et le nombre d'appels ayant attendu moins de s_0 devient très petit. Si $u = 0$, $A = X = 0$ si bien que $\sigma^2(0) = 0$. Si $u = 1$, le nombre d'arrivées est infini et aucun appel n'attend moins de s_0 si bien que $X = 0$. Alors, $\sigma^2(1) = 0$.

La figure 3.3 montre la fonction $\mu(u)$ qui est croissante jusqu'à environ 0,86 puis devient décroissante. À ce point, le facteur d'achalandage devient suffisamment élevé pour qu'ajouter des arrivées réduise le nombre d'appels ayant attendu moins de s_0. Comme nous l'avons vu plus haut, $X = 0$ si $u = 0$ ou $u = 1$ si bien qu'il est normal

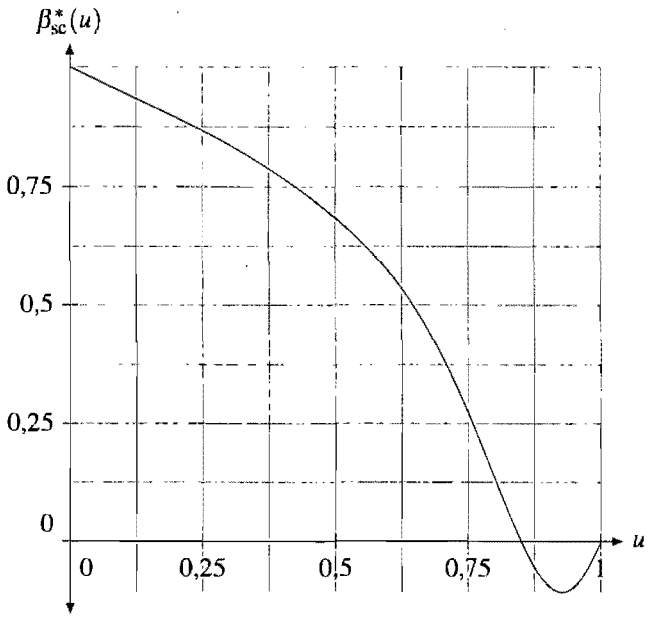

Figure 3.1 – Fonction $\beta^*_{sc}(u)$ pour le nombre d'appels attendant moins de s_0, approximée par des splines cubiques lissées sur 1 000 points

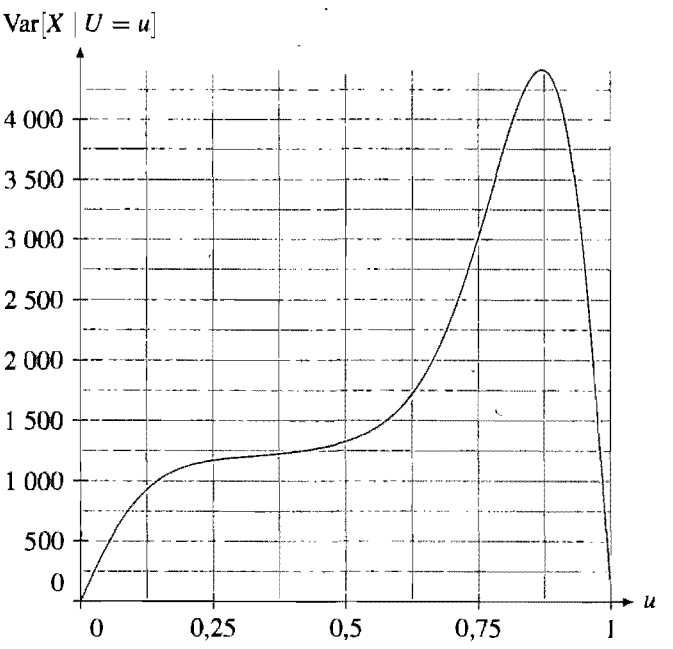

Figure 3.2 – Fonction $\sigma^2(u)$ pour le nombre d'appels attendant moins de s_0, approximée par des splines cubiques lissées sur 1 000 points

que $\mu(0) = \mu(1) = 0$.

Dans le cas du nombre d'abandons et du temps d'attente, toujours avec A comme variable de contrôle, les graphiques sont très semblables à ceux pour le nombre d'appels ayant attendu moins de s_0. Par contre, comme le montre la figure 3.4, la fonction $\beta^*_{\text{sc}}(u)$ est toujours positive. De plus, les quatre fonctions tendent vers l'infini si $u \to 1$ puisqu'avec un nombre d'arrivées infini, le nombre d'abandons et le temps d'attente tendent vers l'infini eux aussi.

Le comportement du taux d'occupation des agents ressemble quant à lui à celui du nombre d'appels ayant attendu moins de s_0, à l'exception de $\mu(u)$ qui converge vers 1 au lieu de 0 quand $u \to 1$ et $\beta^*_{\text{sc}}(u)$ qui est toujours positive. Ceci s'explique de la façon suivante. Plus u est grand, plus le nombre d'arrivées est élevé et plus les agents sont occupés. Si le nombre d'arrivées est très grand, les agents sont toujours occupés si bien qu'il est logique que $\mu(1) = 1$, $\sigma^2(u) = 0$ et $\text{Cov}[X, A \mid U = u] = 0$.

Nous avons également remarqué que modifier le trafic n'affecte pas beaucoup les courbes. Si nous augmentons A pour accroître le trafic, le point d'inflexion de $\mu(u)$ et le point où $\beta^*_{\text{sc}}(u) = 0$ pour $G(s_0)$ seront décalés vers la gauche tandis que les courbes pour L et W tendront plus rapidement vers l'infini quand $u \to 1$. D'un autre côté, si nous diminuons le trafic, les points d'inflexion seront décalés vers la droite et les courbes avec une asymptote à 1 tendront moins rapidement vers l'infini. Si le trafic est suffisamment bas, il se peut même que $\beta^*_{\text{sc}}(u)$ soit toujours positive, quelle que soit la mesure de performance.

À partir de ces résultats, nous avons décidé de comparer nos méthodes de combinaison pour $G(s_0)$ et L puisque W se comporte de façon semblable à L et N_{B}/N se comporte à peu près comme $G(s_0)$.

Nous avons comparé les méthodes de combinaison suivantes :

(1) Pas de variable de contrôle, seulement la stratification ($b_{s,j} = 0$) ;

(2) VC $C - c_s$ avec coefficient constant $b_{s,j} = \beta^*$;

(3) VC $C - c_s$ avec coefficient constant $b_{s,j} = \beta^*_{\text{scp}}$;

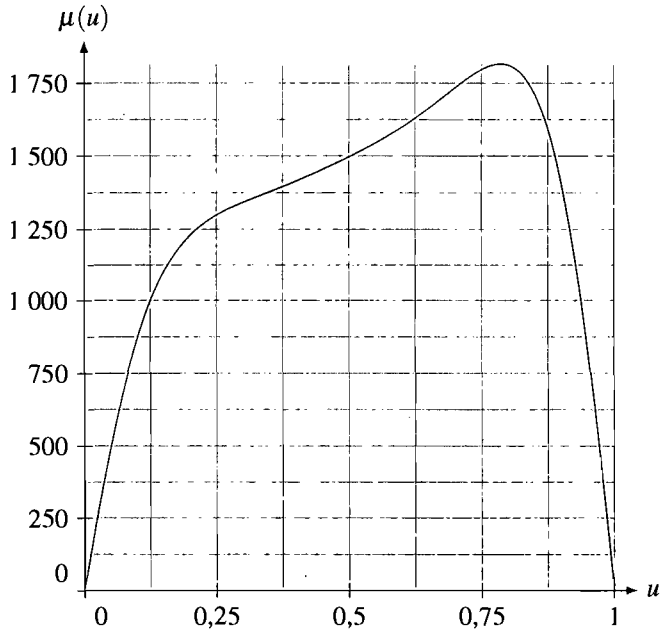

Figure 3.3 – Fonction $\mu(u)$ pour le nombre d'appels attendant moins de s_0, approximée par des splines cubiques lissées sur 1 000 points

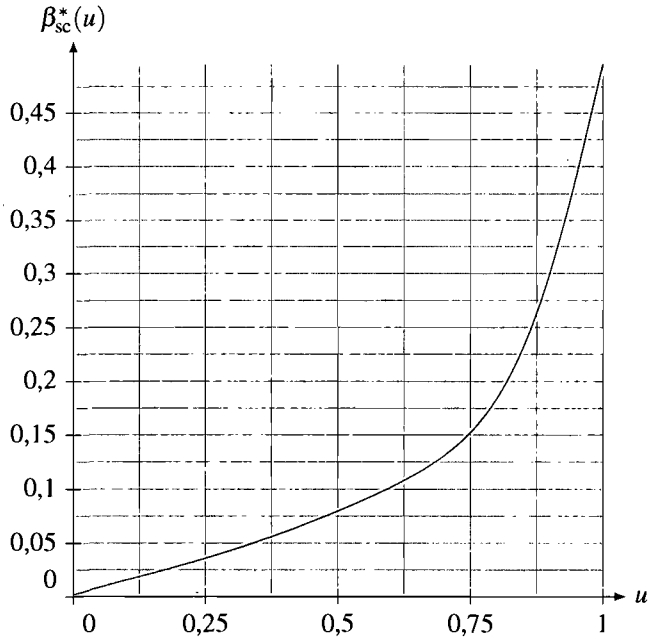

Figure 3.4 – Fonction $\beta^*_{sc}(u)$ pour le nombre d'abandons, approximée par des splines cubiques lissées sur 1 000 points

(4) VC $C - c_s$ avec coefficient $b_{s,j} = \beta^*_{\text{sc},s}$ pour chaque strate ;

(5) VC $C - c(U)$ avec coefficient $b_{s,j} = \beta^*_{\text{sc}}(U)$.

Nous avons testé avec différentes valeurs de m allant de 2 à 1 000. Nous avons également considéré les cas extrêmes où $m = 1$ et $m \to \infty$. Dans le premier cas, aucune stratification n'est employée si bien que la méthode (1) revient à la méthode Monte Carlo classique. Les méthodes (2), (3) et (4) sont équivalentes et consistent à appliquer une variable de contrôle à l'estimateur Monte Carlo.

Si $m \to \infty$, la variance avec allocation proportionnelle tend vers $\mathbb{E}[\sigma^2(u)]$ tandis que la variance avec allocation optimale tend vers $(\mathbb{E}[\sigma(u)])^2$. Comme nous l'avons discuté précédemment, les méthodes (4) et (5) sont équivalentes dans ce cas extrême. Les résultats avec $m \to \infty$ nous donnent la plus grande réduction de variance qui peut être obtenue seulement en augmentant m.

Le tableau 3.VIII regroupe les résultats pour le nombre d'appels ayant attendu moins de s_0. Chaque zone du tableau représente une des méthodes que nous avons étudiées, chaque ligne correspond à un terme de la décomposition (3.18) tandis que chaque colonne est dédiée à une valeur de m. La figure 3.5 présente quant à elle un graphique de la variance de $G(s_0)$ en fonction du logarithme en base 2 du nombre de strates, avec l'allocation optimale, pour chacune des cinq méthodes comparées.

Dans cet exemple, la stratification à elle seule réduit la variance d'un facteur allant de 2,4 (allocation proportionnelle avec deux strates) à 38,5 (allocation optimale avec un très grand nombre de strates), selon le nombre de strates et le type d'allocation utilisés. Comme on peut s'y attendre, plus m est grand, plus la variance est petite et la variance avec l'allocation proportionnelle est toujours plus grande que celle avec l'allocation optimale.

Le terme $\frac{1}{m}\sum_{s=1}^{m}(\mu_s - \mu)^2$ de la décomposition (3.18) est le même pour toutes les méthodes de stratification, car il n'est pas affecté par la variable de contrôle. Comme le nombre de valeurs différentes de μ_s augmente avec m, il est normal que la variance des moyennes des strates soit une fonction croissante de m.

Le terme $\frac{1}{m}\sum_{s=1}^{m}(\sigma_s - \bar{\sigma})^2$, quant à lui, est croissant quand m est petit pour la même raison que le terme précédent. Par contre, si m est suffisamment grand pour occasionner une réduction de variance importante dans les strates ($m \geq 20$ pour cet exemple), la variance des σ_s diminue avec m puisque plus le nombre de strates est élevé, plus la variance dans chaque strate est petite et plus la variabilité de cette variance entre les strates est faible. Ceci peut être observé pour toutes les méthodes à l'exception de (4), sur laquelle nous reviendrons plus loin.

Clairement, les méthodes (2) et (3) qui utilisent un coefficient global sont à peine plus efficaces dans cet exemple que la méthode (1) qui n'utilise aucune variable de contrôle. Avec la méthode (2), le même coefficient $\beta^* = 0,39$ est employé quel que soit m, ce qui n'est pas optimal. Par conséquent, il arrive que la variance obtenue avec la méthode (2) est supérieure à celle obtenue avec la méthode (1). Pour cet exemple, ceci se produit uniquement avec l'allocation proportionnelle, mais cette augmentation de variance pourrait aussi survenir avec l'allocation optimale dans d'autres exemples. Avec la méthode (3), qui utilise la constante optimale, la variance est toujours plus petite qu'avec les méthodes (1) et (2) dans cet exemple, même si la constante β^*_{scp} utilisée est optimale seulement pour l'allocation proportionnelle.

Nous observons également que si m est suffisamment élevé, les méthodes (2) et (3) semblent converger. Par contre, si $m \to \infty$, $\beta^*_{\text{scp}} \to \mathbb{E}[\text{Cov}[X,C \mid U]]/\mathbb{E}[\text{Var}[C \mid U]]$ qui diffère en général de $\beta^* = \text{Cov}[X,C]/\text{Var}[C]$. Dans cet exemple particulier, la constante optimale quand m est grand est proche de la constante optimale sans stratification, d'où la similitude des résultats obtenus.

La méthode (4) est clairement la meilleure dans cet exemple. Le facteur de réduction de la variance maximal dans ce cas est de 73 si nous utilisons l'allocation optimale avec un très grand nombre de strates. Avec cette méthode, la variance des σ_s entre les strates est à peu près constante par rapport à m. Ceci peut s'expliquer intuitivement par le fait que les coefficients β_s sont choisis de façon à minimiser la variance dans chaque strate, indépendamment des autres strates, ce qui réduit par le fait même la variabilité des σ_s

entre les strates.

Dans cet exemple, la méthode (5) est au moins quatre fois moins efficace pour réduire la variance que la méthode (4). Le tableau 3.IX montre que pour les petites et les grandes valeurs de U, la variance de l'espérance conditionnelle à $U_{s,j}$ domine l'espérance de la variance conditionnelle à $U_{s,j}$. La méthode (4) réduit très efficacement la composante dominante de la variance tandis que la méthode (5) la laisse inchangée. De même, pour la plupart des strates, utiliser β_s réduit pratiquement autant l'espérance de la variance conditionnelle à $U_{s,j}$ (méthode (4)) qu'employer une fonction $\beta(u)$ (méthode (5)).

Le tableau 3.X montre la variance dans le cas où X correspond au nombre d'abandons, toujours avec A comme variable de contrôle. Les résultats sont très similaires au cas de $G(s_0)$, mais ici, nous remarquons que la méthode (4) surpasse nettement les autres méthodes.

À partir de ces résultats, nous pouvons conclure que la méthode (4), qui utilise une constante pour chaque strate, est en général la plus avantageuse avec un nombre modéré de strates. Utiliser la méthode (5) peut être avantageux si nous pouvons démontrer qu'elle est plus efficace que la méthode (4) pour un exemple particulier ou si m est très grand. Dans ce second cas, il peut être plus facile d'estimer une fonction $\beta^*_{\mathrm{sc}}(u)$ qu'un très grand nombre de constantes $\beta^*_{\mathrm{sc},s}$.

Tableau 3.VIII – Termes de la décomposition de la variance de $G(s_0)$ pour différentes méthodes d'estimation et différentes valeurs de m

Méthode	m	1	2	3	10	20	50	100	500	1 000	∞
	$\frac{1}{m}\sum_{s=1}^{m}(\mu_s-\mu)^2$	0	44 610	55 448	68 600	71 803	73 868	74 514	74 928	74 986	75 096
(1)	$n\mathrm{Var}[\bar{X}_n]$	77 444	77 444	77 444	77 444	77 444	77 444	77 444	77 444	77 444	77 444
	$n\mathrm{Var}[\bar{X}_{\mathrm{stratp},n,m}]$	77 444	32 834	21 995	8 844	5 641	3 575	2 930	2 515	2 457	2 347
	$n\mathrm{Var}[\bar{X}_{\mathrm{strato},n,m}]$	—	30 933	20 476	5 778	3 537	2 516	2 247	2 071	2 046	2 010
	$\frac{1}{m}\sum_{s=1}^{m}(\sigma_s-\bar{\sigma})^2$	0	1 901	1 519	3 066	2 103	1 060	682	444	412	337
(2)	$n\mathrm{Var}[\bar{X}_n]$	35 291	77 358	83 389	82 180	79 979	78 302	77 719	77 296	77 245	77 188
	$n\mathrm{Var}[\bar{X}_{\mathrm{stratp},n,m}]$	35 291	32 748	27 941	13 580	8 177	4 433	3 205	2 368	2 259	2 092
	$n\mathrm{Var}[\bar{X}_{\mathrm{strato},n,m}]$	—	30 280	19 580	5 297	3 103	2 042	1 749	1 546	1 517	1 480
	$\frac{1}{m}\sum_{s=1}^{m}(\sigma_s-\bar{\sigma})^2$	0	2 468	8 360	8 283	5 073	2 391	1 456	822	742	612
(3)	$n\mathrm{Var}[\bar{X}_n]$	35 291	73 348	77 097	76 783	77 004	77 401	77 435	77 281	77 241	77 188
	$n\mathrm{Var}[\bar{X}_{\mathrm{stratp},n,m}]$	35 291	28 738	21 649	8 183	5 201	3 533	2 921	2 353	2 255	2 092
	$n\mathrm{Var}[\bar{X}_{\mathrm{strato},n,m}]$	—	28 734	19 971	6 460	4 102	2 722	2 150	1 633	1 562	1 479
	$\frac{1}{m}\sum_{s=1}^{m}(\sigma_s-\bar{\sigma})^2$	0	4	1 678	1 724	1 099	811	771	720	694	614
	β^*_{sc}	0,390	0,196	0,074	-0,210	-0,243	-0,108	0,057	0,299	0,342	0,392
(4)	$n\mathrm{Var}[\bar{X}_n]$	35 291	53 078	59 966	70 427	73 534	75 595	76 240	76 644	76 692	76 786
	$n\mathrm{Var}[\bar{X}_{\mathrm{stratp},n,m}]$	35 291	8 468	4 517	1 827	1 732	1 727	1 726	1 715	1 706	1 690
	$n\mathrm{Var}[\bar{X}_{\mathrm{strato},n,m}]$	—	5 470	2 606	1 180	1 124	1 098	1 090	1 074	1 069	1 060
	$\frac{1}{m}\sum_{s=1}^{m}(\sigma_s-\bar{\sigma})^2$	0	2 999	1 912	647	608	629	636	641	637	630
(5)	$n\mathrm{Var}[\bar{X}_n]$	76 786	76 786	76 786	76 786	76 786	76 786	76 786	76 786	76 786	76 786
	$n\mathrm{Var}[\bar{X}_{\mathrm{stratp},n,m}]$	76 786	32 176	21 337	8 186	4 983	2 917	2 272	1 857	1 799	1 690
	$n\mathrm{Var}[\bar{X}_{\mathrm{strato},n,m}]$	—	30 293	19 762	4 873	2 599	1 565	1 293	1 114	1 090	1 060
	$\frac{1}{m}\sum_{s=1}^{m}(\sigma_s-\bar{\sigma})^2$	0	1 883	1 576	3 313	2 384	1 352	979	744	709	630

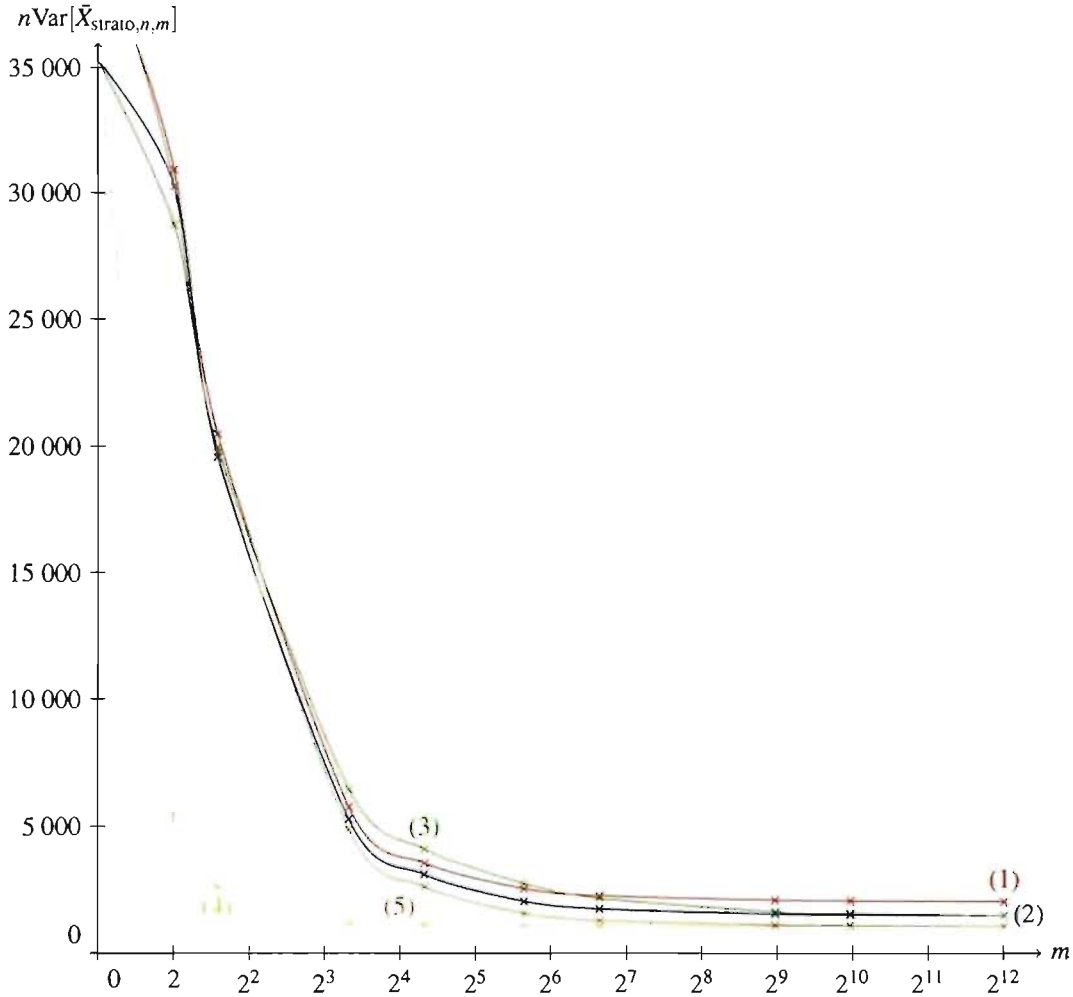

Figure 3.5 – Variance de $G(s_0)$ en fonction du nombre de strates avec allocation proportionnelle

Tableau 3.IX – Termes des décompositions (3.22) et (3.23) pour chaque strate dans le cas de $G(s_0)$ avec $m = 20$ strates

		$\mathbb{E}[\mathrm{Var}[X_{\mathrm{sc},s,j} \mid U_{s,j}]]$			$\mathrm{Var}[\mathbb{E}[X_{\mathrm{sc},s,j} \mid U_{s,j}]]$	
s	μ_s	Méthode (1)	Méthode (5)	Méthode (4)	Méthode (5)	Méthode (4)
1	766	743	21	22	12 250	369,000
2	963	915	31	31	1 314	0.021
3	1 069	998	54	54	670	0.012
4	1 149	1 060	85	85	449	0.019
5	1 218	1 101	133	133	339	0.019
6	1 278	1 144	183	183	276	0.023
7	1 334	1 186	251	251	234	0.031
8	1 385	1 228	335	335	207	0.030
9	1 433	1 281	434	434	185	0.043
10	1 479	1 339	565	566	167	0.034
11	1 523	1 429	721	721	155	0.074
12	1 565	1 530	921	921	140	0.104
13	1 605	1 684	1 175	1 176	125	0.099
14	1 643	1 889	1 496	1 497	112	0.168
15	1 677	2 199	1 934	1 936	87	0.341
16	1 707	2 653	2 515	2 517	57	0.517
17	1 728	3 367	3 330	3 334	20	0.834
18	1 732	4 547	4 528	4 537	10	1.909
19	1 692	6 647	6 373	6 393	555	8.679
20	1 427	10 009	8 706	8 750	48 508	543,000

Tableau 3.X – Termes de la décomposition de la variance de L pour différentes méthodes d'estimation et différentes valeurs de m

Méthode	m	1	2	3	10	20	50	100	500	1 000	∞
	$\frac{1}{m}\sum_{s=1}^{m}(\mu_s-\mu)^2$	0	2 458	3 743	6 552	7 538	8 330	8 662	8 965	8 996	9 007
	$n\mathrm{Var}[\bar{X}_n]$	9 192	9 192	9 192	9 192	9 192	9 192	9 192	9 192	9 192	9 192
(1)	$n\mathrm{Var}[\bar{X}_{\mathrm{stratp},n,m}]$	9 192	6 735	5 449	2 640	1 654	863	530	228	196	186
	$n\mathrm{Var}[\bar{X}_{\mathrm{strato},n,m}]$	—	3 880	2 337	579	307	172	133	104	101	99
	$\frac{1}{m}\sum_{s=1}^{m}(\sigma_s-\bar{\sigma})^2$	0	2 855	3 111	2 061	1 347	690	398	124	95	86
	$n\mathrm{Var}[\bar{X}_n]$	2 681	4 964	5 985	7 925	8 484	8 872	9 015	9 135	9 148	9 153
(2)	$n\mathrm{Var}[\bar{X}_{\mathrm{stratp},n,m}]$	2 681	2 506	2 242	1 373	946	542	353	170	152	147
	$n\mathrm{Var}[\bar{X}_{\mathrm{strato},n,m}]$	—	2 231	1 474	421	242	148	119	97	95	94
	$\frac{1}{m}\sum_{s=1}^{m}(\sigma_s-\bar{\sigma})^2$	0	275	768	952	703	394	234	73	57	53
	$n\mathrm{Var}[\bar{X}_n]$	2 681	4 667	5 483	7 331	8 074	8 709	8 958	9 134	9 148	9 153
(3)	$n\mathrm{Var}[\bar{X}_{\mathrm{stratp},n,m}]$	2 681	2 209	1 739	779	536	380	296	167	152	147
	$n\mathrm{Var}[\bar{X}_{\mathrm{strato},n,m}]$	—	2 198	1 579	586	392	257	181	104	96	94
	$\frac{1}{m}\sum_{s=1}^{m}(\sigma_s-\bar{\sigma})^2$	0	12	161	193	144	123	115	65	55	53
	β^*_{sc}	0,153	0,206	0,242	0,352	0,388	0,365	0,303	0,180	0,160	0,154
	$n\mathrm{Var}[\bar{X}_n]$	2 681	3 365	4 312	6 747	7 681	8 456	8 787	9 085	9 112	9 120
(4)	$n\mathrm{Var}[\bar{X}_{\mathrm{stratp},n,m}]$	2 681	907	569	195	143	126	125	121	116	113
	$n\mathrm{Var}[\bar{X}_{\mathrm{strato},n,m}]$	—	556	285	94	76	71	70	69	68	68
	$\frac{1}{m}\sum_{s=1}^{m}(\sigma_s-\bar{\sigma})^2$	0	351	284	101	66	55	55	52	47	45
	$n\mathrm{Var}[\bar{X}_n]$	9 120	9 120	9 120	9 120	9 120	9 120	9 120	9 120	9 120	9 120
(5)	$n\mathrm{Var}[\bar{X}_{\mathrm{stratp},n,m}]$	9 120	6 662	5 377	2 568	1 582	790	458	155	124	113
	$n\mathrm{Var}[\bar{X}_{\mathrm{strato},n,m}]$	—	3 832	2 293	537	268	137	100	72	69	68
	$\frac{1}{m}\sum_{s=1}^{m}(\sigma_s-\bar{\sigma})^2$	0	2 831	3 084	2 031	1 314	653	358	83	54	45

CHAPITRE 4

VARIABLES ALÉATOIRES COMMUNES

Les techniques de réduction de la variance comme celles du chapitre précédent travaillent sur une seule configuration. Comme nous avons vu dans l'introduction, il arrive souvent que nous souhaitions simuler un même modèle avec des paramètres différents, pour l'analyse de sensibilité et pour l'optimisation. Réduire la variance pour chaque configuration permet certes de réduire la variance pour toute l'analyse, mais nous pouvons en plus exploiter la similarité qui existe entre les configurations. Une façon de le faire est d'utiliser les variables aléatoires communes, c'est-à-dire utiliser les mêmes variables aléatoires aux mêmes endroits pour simuler toutes les configurations qui nous intéressent.

Dans ce chapitre, nous appliquons cette technique aux centres de contacts et étudions les problèmes posés. La section suivante explique la technique en général tandis que la section 4.2 se consacre à l'estimation de dérivées avec variables aléatoires communes dans le cas où on s'intéresse à la performance par rapport à un paramètre continu. La section 4.3 applique les techniques des deux premières sections sur un exemple simple de centre de contacts et étudie les problèmes de synchronisation posés. Enfin, la section 4.4 présente quelques preuves de convergence de la variance d'une différence de mesures de performance s'appliquant à un modèle spécifique de centre de contacts et à certains paramètres continus.

Une partie des analyses et des résultats de ce chapitre est publiée dans [60]. La description de la technique ainsi que les preuves de convergence s'inspirent fortement de [58].

4.1 Technique de base

Nous souhaitons parfois comparer deux ou plusieurs configurations similaires, ne différant par exemple que par un paramètre $\theta \in \Upsilon \subseteq \mathbb{R}$. Comme dans la section 3.1, nous nous intéressons à une fonction de plusieurs moyennes. Soit alors $\mathbf{X}(\theta)$ un vecteur aléatoire dont la réalisation dépend d'un paramètre $\theta \in \Upsilon$, $\boldsymbol{\mu}(\theta) = \mathbb{E}[\mathbf{X}(\theta)]$ le vecteur de mesures de performance pour une valeur de θ, et $\nu(\theta) = g(\boldsymbol{\mu}(\theta))$ le résultat de la fonction de plusieurs espérances qui nous intéresse, en fonction du paramètre θ.

Nous souhaitons estimer la différence

$$\nu(\theta_2) - \nu(\theta_1)$$

pour $\theta_1 < \theta_2$ et $\delta = \theta_2 - \theta_1$. Soit pour cela $\hat{\nu}_n(\theta, \mathbf{U}) = g(\hat{\boldsymbol{\mu}}_n(\theta, \mathbf{U}))$ un estimateur de $\nu(\theta)$, où $\hat{\boldsymbol{\mu}}_n(\theta, \mathbf{U})$ est un estimateur asymptotiquement sans biais de $\boldsymbol{\mu}(\theta)$ utilisant une séquence $\mathbf{U}$ de variables aléatoires uniformes sur l'intervalle $[0, 1)$. La différence est alors estimée par

$$\hat{\Delta}_n = \hat{\nu}_n(\theta_2, \mathbf{U}_2) - \hat{\nu}_n(\theta_1, \mathbf{U}_1)$$

et sa variance est

$$\mathrm{Var}[\hat{\Delta}_n] = \mathrm{Var}[\hat{\nu}_n(\theta_1, \mathbf{U}_1)] + \mathrm{Var}[\hat{\nu}_n(\theta_2, \mathbf{U}_2)] - 2\mathrm{Cov}[\hat{\nu}_n(\theta_1, \mathbf{U}_1), \hat{\nu}_n(\theta_2, \mathbf{U}_2)].$$

Cet estimateur est bien entendu biaisé à moins que g ne soit linéaire.

Si les deux systèmes sont simulés indépendamment, $\mathbf{U}_1$ et $\mathbf{U}_2$ sont indépendants si bien que la covariance entre les deux estimateurs est nulle et la variance devient

$$\mathrm{Var}[\hat{\Delta}_n] = \mathrm{Var}[\hat{\nu}_n(\theta_1, \mathbf{U}_1)] + \mathrm{Var}[\hat{\nu}_n(\theta_2, \mathbf{U}_2)] \approx 2\mathrm{Var}[\hat{\nu}_n(\theta_1, \mathbf{U}_1)]$$

si θ_1 est assez près de θ_2.

Pour réduire cette variance, nous pouvons utiliser les *variables aléatoires communes*

[10, 65] en fixant $\mathbf{U}_1 = \mathbf{U}_2$. Pour ce faire, les deux systèmes sont simulés avec les mêmes nombres aléatoires, en veillant le plus possible à utiliser les mêmes nombres aux mêmes endroits. Dans le cas de bon nombre de modèles, cela suffit pour induire une corrélation positive entre $\hat{v}_n(\theta_1, \mathbf{U}_1)$ et $\hat{v}_n(\theta_2, \mathbf{U}_2)$ qui réduit $\mathrm{Var}[\hat{\Delta}_n]$.

Nous pouvons bien évidemment combiner les variables aléatoires communes avec toute autre technique de réduction de la variance. Pour cela, nous pouvons soit appliquer ces techniques séparément sur $\hat{v}_n(\theta_1, \mathbf{U}_1)$ et $\hat{v}_n(\theta_2, \mathbf{U}_2)$, soit le faire sur la différence $\hat{\Delta}_n$. La première stratégie est parfaitement envisageable avec des variables aléatoires indépendantes, mais la seconde est évidemment plus judicieuse dans le cas des variables aléatoires communes étant donné qu'elle tient compte de la covariance non nulle entre les deux termes de la différence.

Dans certains cas, les variables aléatoires communes permettent même de sauver du temps de calcul. En effet, au lieu de recalculer les nombres aléatoires pour chaque configuration, il est possible de les stocker en mémoire pour simplement les réutiliser par la suite. Par contre, avec des modèles complexes utilisant beaucoup de variables aléatoires, cette technique prend trop de mémoire et du temps est gaspillé pour gérer cette mémoire. Dans de tels cas, il est plus avantageux de réinitialiser les générateurs de nombres aléatoires au début de leur séquence pour reproduire les nombres pour les différentes configurations testées.

Nous nous concentrons ici sur le cas unidimensionnel où $\mathbf{X} = (X)$, $g(\mathbf{X}) = X$ et $\boldsymbol{\mu} = (\mu)$, pour lequel on veut estimer $\mu(\theta_2) - \mu(\theta_1)$ par $\Delta = f(\theta_2, \mathbf{U}) - f(\theta_1, \mathbf{U})$.

4.2 Estimation d'une dérivée

Nous nous intéressons également à des différences lorsque δ est très petit, par exemple pour estimer une dérivée par la méthode des différences finies

$$\frac{\hat{v}_n(\theta + \delta, \mathbf{U}_2) - \hat{v}_n(\theta, \mathbf{U}_1)}{\delta}.$$

Sans variables aléatoires communes, la variance d'un tel estimateur est approximativement $2\text{Var}[\hat{v}_n(\theta, \mathbf{U})]/\delta^2 = \mathcal{O}(1/\delta^2)$ et tend vers l'infini si $\delta \to 0$.

Par contre, si nous utilisons les variables aléatoires communes, nous obtenons souvent un estimateur utilisable dont la variance est bornée. Parfois, il n'est pas possible de borner la variance, mais nous pouvons obtenir une convergence dans $\mathcal{O}(1/\delta)$, ce qui est mieux que celle dans $\mathcal{O}(1/\delta^2)$ obtenue avec les variables aléatoires indépendantes.

Pour démontrer cette convergence, il existe des théorèmes qui s'appliquent pour $\mathbf{X} = (X)$ et qui sont présentés dans [58]. D'abord, supposons que $f(\theta, \mathbf{U})$ est continue sur Υ et dérivable par rapport à θ sur $D(\mathbf{U}) \subseteq \Upsilon$, où $\Upsilon \backslash D(\mathbf{U})$ est un ensemble dénombrable. Si nous pouvons trouver une variable aléatoire $\Gamma(\mathbf{U})$ indépendante de θ telle que

$$\sup_{\theta \in D(\mathbf{U})} |f'(\theta, \mathbf{U})| \leq \Gamma(\mathbf{U})$$

et $\mathbb{E}[\Gamma(\mathbf{U})^2] < \infty$, nous avons $\text{Var}[\Delta] = \mathcal{O}(\delta^2)$ si bien que $\text{Var}[\Delta/\delta] = \mathcal{O}(1)$. Ceci correspond au corollaire 6.6 de [58] qui est aussi énoncé dans [65].

Sans cette condition, démontrer la borne devient plus complexe. Selon le théorème 6.5 présenté dans [58], il faut d'abord borner la probabilité que Δ^2 excède une variable aléatoire Γ qui ne dépend pas de θ. Ensuite, l'inégalité de Holder permet d'aboutir à une borne si certains moments d'ordre supérieur à 2 de Δ sont bornés. Une borne peut aussi être démontrée lorsque Δ est borné avec probabilité 1. Plutôt qu'énoncer ce théorème, nous en développons des variantes simplifiées et mieux adaptées à notre modèle dans la section 4.4.1.

4.3 Application aux centres de contacts

Revenons au modèle de centre de contacts utilisé à la section 3.5.1. Supposons que nous multiplions le temps de service moyen des contacts par $1 - \delta$ pour une petite valeur de δ et comparons avec le cas où $\delta = 0$. Cela équivaut à multiplier le paramètre d'échelle du temps de service exponentiel par $1/(1 - \delta)$. Dans ce contexte, nous estimons la va-

riation du nombre moyen $\mathbb{E}[G(s_0)]$ de contacts ayant attendu moins de s_0, du nombre moyen d'abandons $\mathbb{E}[L]$, du temps d'attente moyen $\mathbb{E}[W]/\mathbb{E}[A]$ et du taux d'occupation $\mathbb{E}[N_B]/\mathbb{E}[N]$, par rapport à δ.

Nous distinguons dans ce modèle plusieurs types de variables aléatoires : la variable servant à générer le facteur d'achalandage B, les variables générant les temps inter-arrivées, celles engendrant les temps de service et celles produisant les temps de patience. Pour chacun de ces types de variables, nous utilisons une séquence différente de variables aléatoires afin d'obtenir une bonne synchronisation entre les deux systèmes. Nous utilisons aussi l'inversion pour générer toutes les variables aléatoires et initialisons les séquences de variables aléatoires aux mêmes germes dans chacune des deux configurations comparées.

Un problème survient lorsque nous cherchons à utiliser les mêmes nombres aléatoires aux mêmes endroits dans ce modèle, car le changement de durées de service peut faire en sorte qu'un client abandonnant avec $\delta = 0$ n'abandonne pas avec $\delta > 0$. Dans un tel cas, un temps de service est généré dans la première configuration et non dans la seconde. Dans ce contexte, nous disposons de deux options pour générer les temps de service :

(a) Générer un temps de service pour chaque nouveau contact ; ce temps de service sera inutilisé pour les contacts ayant abandonné,

(b) Générer un temps de service seulement pour les contacts qui n'abandonnent pas.

La première stratégie nous assure que les mêmes contacts reçoivent les mêmes temps de service dans les deux configurations. La seconde option nous assure que si un temps de service très long provoque un engorgement et des abandons dans une première configuration, ce temps de service sera aussi utilisé dans la seconde configuration. Nous disposons de deux options similaires pour les temps de patience :

(c) Générer un temps de patience pour chaque nouveau contact ; ce temps de patience sera inutilisé pour les contacts servis immédiatement,

(d) Générer un temps de patience seulement pour les contacts qui doivent attendre

en file.

À priori, il n'existe aucun moyen de choisir la méthode de synchronisation pour un modèle donné ; il faut procéder par expérimentation.

Nous avons effectué de telles expérimentations pour notre modèle afin de comparer les différentes méthodes. Nous avons également testé le cas sans synchronisation, qui utilise une seule séquence de variables aléatoires. Le tableau 4.I présente les résultats pour $n = 10\ 000$ simulations avec $\delta = 0,1$, $\delta = 0,01$ et $\delta = 0,001$. Chaque colonne correspond à une valeur de δ. La valeur $\bar{\Delta}_n$ donne la moyenne de la différence sur les n simulations tandis que $\mathrm{Var}[\Delta_i] = n\mathrm{Var}[\bar{\Delta}_n]$ donne la variance de la différence pour une réplication particulière. Chaque rangée du tableau représente une des combinaisons d'options de synchronisation présentée ci-haut. La partie supérieure du tableau donne les résultats avec variables aléatoires indépendantes (VAI) tandis que la partie inférieure donne ceux avec les variables aléatoires communes (VAC).

Nous remarquons d'abord que sans variables aléatoires communes, la variance est indépendante de δ et la méthode de synchronisation n'a aucun impact significatif. Les différences de valeurs sont uniquement dues au bruit dans la simulation. Par contre, avec les variables aléatoires communes, nous obtenons une nette amélioration puisque la variance semble dans $\mathcal{O}(\delta)$. Dans la section 4.4, nous démontrons qu'elle est effectivement dans $\mathcal{O}(\delta^{1-\varepsilon})$ pour tout $\varepsilon > 0$, ce qui est très près de $\mathcal{O}(\delta)$.

Étonnamment, la méthode $b + d$ sans synchronisation donne de bons résultats ici. L'explication la plus probable est l'importance du facteur d'achalandage B qui demeure le même dans les deux configurations, même sans synchronisation. Pour illustrer cela, simulons un modèle avec $B = 1$ (taux d'arrivée déterministe). Comme le montrent les résultats du tableau 4.II, la différence de variance entre les cas avec plusieurs séquences de nombres aléatoires et celui avec une seule séquence sont beaucoup plus grandes que si le taux d'arrivée était aléatoire. Mais la méthode fonctionne assez bien malgré tout. Cela pourrait s'expliquer par le fait que la plupart des variables aléatoires correspondent à des durées. Qu'un temps long soit une durée de service ou une durée de patience,

Tableau 4.I – Impact d'un changement du temps de service moyen sur le nombre $G(s_0)$ de contacts ayant attendu moins de s_0, avec variables aléatoires communes et $n = 10\,000$

Méthode	$\delta = 0,1$		$\delta = 0,01$		$\delta = 0,001$	
	$\bar{\Delta}_n$	$\widehat{\mathrm{Var}}[\Delta_i]$	$\bar{\Delta}_n$	$\widehat{\mathrm{Var}}[\Delta_i]$	$\bar{\Delta}_n$	$\widehat{\mathrm{Var}}[\Delta_i]$
VAI (a + c)	76,93	192 680	7,48	160 486	-0,135	157 745
VAI (a + d)	76,57	192 374	7,00	160 168	-0,560	157 592
VAI (b + c)	76,96	192 034	7,16	159 605	-0,216	156 794
VAI (b + d)	77,14	192 495	7,34	159 970	-0,215	157 136
VAC pas de sync., (b + d)	76,08	11 036	6,71	2 417	0,203	1 979
VAC (a + c)	78,75	9 307	8,48	122	0,862	5
VAC (a + d)	78,78	9 502	8,51	389	0,786	197
VAC (b + c)	78,73	9 714	8,14	237	0,821	21
VAC (b + d)	78,90	9 646	8,42	312	0,819	81

il engendrera dans les deux cas un engorgement du système. Ainsi, la synchronisation ne se brise de façon importante que lorsqu'un temps inter-arrivée devient un temps de service ou de patience, ou inversement.

Dans ce modèle, si nous utilisons la méthode $a+c$, nous pouvons simuler avec une seule séquence de variables aléatoires. En effet, après avoir généré le facteur B, nous générons un temps inter-arrivée, un temps de patience et un temps de service pour chaque contact, dans cet ordre. Puisqu'aucun autre nombre aléatoire n'est nécessaire, nous pouvons utiliser une seule séquence sans perdre la synchronisation.

Ici, la méthode de synchronisation $a+c$, qui génère des temps de service et de patience pour tous les contacts, donne la plus petite variance parmi les méthodes testées. Cette méthode est suivie de près par $b+d$. Le choix de la stratégie de synchronisation a un impact important pour δ petit, mais avec δ grand, son impact est négligeable. Il faut garder à l'esprit que ces observations ne s'appliquent qu'à cet exemple. En général, la meilleure méthode de synchronisation pourrait différer d'une situation à l'autre. Il est donc bon d'expérimenter et de comparer les méthodes.

Nous avons également expérimenté avec le nombre moyen d'abandons ; les résultats sont présentés au tableau 4.III. Nous observons un comportement de la variance

Tableau 4.II – Impact d'un changement du temps de service moyen sur le nombre $G(s_0)$ de contacts ayant attendu moins de s_0, avec variables aléatoires communes, $n = 10\,000$ et $B = 1$

Méthode	$\delta = 0,1$		$\delta = 0,01$		$\delta = 0,001$	
	$\bar{\Delta}_n$	$\widehat{\mathrm{Var}}[\Delta_i]$	$\bar{\Delta}_n$	$\widehat{\mathrm{Var}}[\Delta_i]$	$\bar{\Delta}_n$	$\widehat{\mathrm{Var}}[\Delta_i]$
VAI (a + c)	47.6	2 921	6,00	2 917	0.90	2 927
VAI (a + d)	47.7	2 895	5,00	2 944	0.70	2 972
VAI (b + c)	47.6	2 921	6,00	2 917	0.90	2 927
VAI (b + d)	47.7	2 895	5,00	2 944	0.70	2 972
VAC, pas de sync. (b + d)	46.9	1 048	5.52	749	0.45	331
VAC (a + c)	46.8	184	5.36	11	0.56	1
VAC (a + d)	47.0	233	5.46	82	0.54	21
VAC (b + c)	46.8	184	5.36	11	0.56	1
VAC (b + d)	47.0	233	5.46	82	0.54	21

identique à celui pour $G(s_0)$. Par contre, avec le temps d'attente moyen, pour lequel les résultats sont présentés au tableau 4.IV, la variance semble dans $\mathcal{O}(\delta^2)$.

Pour réduire davantage la variance, nous pouvons combiner les variables aléatoires communes avec d'autres techniques, comme les variables de contrôle et la stratification. Pour cela, nous pouvons appliquer la même méthode qu'au chapitre précédent, en remplaçant la variable X par une différence, estimée sans ou avec variables aléatoires communes. Comme variables de stratification et de contrôle, nous utilisons B et A comme au chapitre précédent. Nous avons été tenté d'utiliser la différence du nombre d'arrivées entre les configurations, mais cette différence vaut 0 avec les variables aléatoires communes.

Nous avons expérimenté avec la méthode (4) qui utilise un coefficient différent pour chaque strate, avec variables aléatoires indépendantes et communes. Dans le cas des variables indépendantes, la variable de contrôle est le nombre moyen d'arrivées pour les deux configurations testées. Nous avons utilisé $m = 100$ strates et simulé 50 réplications pilotes par strate pour estimer β_s^*, σ_s^2 et $\sigma_{\mathrm{sc},s}^2$, sans et avec les variables aléatoires communes. Ensuite, nous avons simulé avec 10 000 réplications pour obtenir un estimé de la

Tableau 4.III – Impact d'un changement du temps de service moyen sur le nombre moyen d'abandons, avec variables aléatoires communes pour $n = 10\,000$

Méthode	$\delta = 0,1$		$\delta = 0,01$		$\delta = 0,001$	
	$\bar{\Delta}_n$	$\widehat{\mathrm{Var}}[\Delta_i]$	$\bar{\Delta}_n$	$\widehat{\mathrm{Var}}[\Delta_i]$	$\bar{\Delta}_n$	$\widehat{\mathrm{Var}}[\Delta_i]$
VAI (a + c)	-20,40	15 764	-3,528	19 477	-1,714	19 947
VAI (a + d)	-20,28	15 807	-3,650	19 545	-1,800	20 012
VAI (b + c)	-20,31	15 736	-3,630	19 546	-1,756	20 025
VAI (b + d)	-20,39	15 744	-3,673	19 602	-1,864	20 063
VAC pas de sync, (b + d)	-18,18	917	-1,648	280	-0,018	237
VAC (a + c)	-18,84	755	-2,065	10	-0,214	0,4
VAC (a + d)	-18,93	774	-2,014	31	-0,144	14,2
VAC (b + c)	-18,89	772	-2,071	18	-0,200	1,4
VAC (b + d)	-18,89	768	-2,119	18	-0,248	3,2

Tableau 4.IV – Impact d'un changement du temps de service moyen sur le temps d'attente moyen, avec variables aléatoires communes et $n = 10\,000$

Méthode	$\delta = 0.1$		$\delta = 0.01$		$\delta = 0.001$	
	$\bar{\Delta}_n$	$\widehat{\mathrm{Var}}[\Delta_i]$	$\bar{\Delta}_n$	$\widehat{\mathrm{Var}}[\Delta_i]$	$\bar{\Delta}_n$	$\widehat{\mathrm{Var}}[\Delta_i]$
VAI (a + c)	-5,837	1 511	-1,244	1 950	-0,703	2 014
VAI (a + d)	-5,769	1 502	-1,234	1 942	-0,709	2 006
VAI (b + c)	-5,860	1 497	-1,215	1 956	-0,674	2 017
VAI (b + d)	-5,834	1 498	-1,180	1 947	-0,646	2 013
VAC pas de sync, (b + d)	-4,987	151	-0,445	43,8	0,017	42,6
VAC (a + c)	-5,337	146	-0,621	2,0	-0,064	0,1
VAC (a + d)	-5,308	145	-0,595	6,7	-0,027	3,2
VAC (b + c)	-5,334	142	-0,611	2,3	-0,058	0,1
VAC (b + d)	-5,328	143	-0,624	2,8	-0,061	0,4

variation de $G(s_0)$.

Le tableau 4.V présente les résultats de cette expérience. Chaque groupe de deux colonnes donne les résultats pour une valeur de δ, sans et avec la variable de contrôle. Chaque ligne fournit quant à elle un terme de la décomposition (3.18). La partie supérieure du tableau concerne les variables aléatoires indépendantes tandis que la partie inférieure donne les résultats avec les variables aléatoires communes.

Ces résultats montrent que la stratification apporte une réduction de variance intéressante combinée aux variables aléatoires communes. Par contre, la variable de contrôle n'a que peu d'effet. Ceci s'explique par le fait que $G(s_0)$ ne varie pas beaucoup d'une configuration à l'autre lorsque δ est petit, d'où une différence petite faiblement corrélée avec A.

Nous remarquons aussi qu'avec les variables aléatoires indépendantes ou les variables aléatoires communes pour un petit δ, la variable de contrôle augmente la variance au lieu de la diminuer. Ceci est causé par une mauvaise estimation des constantes β_s^*. Avec les variables aléatoires indépendantes, la variance est si élevée que les constantes β_s^* sont difficiles à estimer. Fixer un δ petit contribue à rendre cette estimation encore plus difficile, même avec les variables aléatoires communes.

Pour toutes les expériences avec variables aléatoires indépendantes, le même facteur d'achalandage B est utilisé pour chaque configuration étant donné que c'est la variable de stratification. Par conséquent, la variance pour le cas sans stratification avec variables aléatoires indépendantes rapportée dans le tableau est inférieure à la variance que nous aurions obtenue avec Monte Carlo classique, où B serait différent pour les deux configurations. Cela montre encore une fois que B contribue beaucoup à la variance.

4.4 Preuves de convergence pour un modèle de centre de contacts

Dans la section précédente, nous avons observé empiriquement que la variance de la différence semblait dans $\mathcal{O}(\delta)$ pour $G(s_0)$ et L et dans $\mathcal{O}(\delta^2)$ pour W/a. Dans cette

Tableau 4.V – Impact d'un changement du temps de service moyen sur le nombre $G(s_0)$ de contacts ayant attendu moins de s_0, avec variables aléatoires communes, $n = 10\,000$, 100 strates et variable de contrôle

		$\delta = 0,1$		$\delta = 0,01$		$\delta = 0,001$	
		Pas de VC	VC	Pas de VC	VC	Pas de VC	VC
VAI	$n\mathrm{Var}[\bar{\Delta}_n]$	12 930	12 944	4 823	4 942	4 781	4 895
	$n\mathrm{Var}[\bar{\Delta}_{\mathrm{sp},n}]$	4 482	4 517	4 722	4 843	4 745	4 859
	$n\mathrm{Var}[\bar{\Delta}_{\mathrm{so},n}]$	3 860	3 898	4 015	4 104	4 030	4 110
	$\frac{1}{m}\sum_{s=1}^{m}(\sigma_s - \bar{\sigma})^2$	622	619	707	740	715	749
	$\frac{1}{m}\sum_{s=1}^{m}(\mu_s - \mu)^2$	8 448	8 427	101	99	36	36
VAC	$\mathrm{Var}[\bar{\Delta}_n]$	9 071	8 986	119	118	5,1	5.1
	$\mathrm{Var}[\bar{\Delta}_{\mathrm{sp},n}]$	460	388	33	32	4,1	4,2
	$\mathrm{Var}[\bar{\Delta}_{\mathrm{so},n}]$	265	227	18	18	1,9	2,0
	$\frac{1}{m}\sum_{s=1}^{m}(\sigma_s - \bar{\sigma})^2$	196	161	14	14	2,2	2,2
	$\frac{1}{m}\sum_{s=1}^{m}(\mu_s - \mu)^2$	8 611	8 598	86	86	1,0	0,9

section, nous analysons la convergence de façon théorique et montrons que la variance est dans $\mathcal{O}(\delta^{1-\varepsilon})$ pour tout $\varepsilon > 0$ dans le cas des deux premières mesures de performance et dans $\mathcal{O}(\delta^2)$ dans le cas de la troisième. Les preuves de convergence de cette section s'appliquent à un modèle de centre de contacts dans lequel $K = I = 1$, semblable à celui utilisé dans la section précédente mais un peu plus général.

Avant d'exposer les preuves, décrivons ce modèle plus en détails. Au temps t, les contacts arrivent selon un processus de Poisson de taux $B\lambda(t)$ où B est un facteur d'achalandage aléatoire tel que $\mathbb{E}[B] = 1$ et $\mathbb{E}[B^q] < \infty$ pour $q \geq 1$. La fonction de densité de B est $f_{\mathrm{B}}(x)$. Nous supposons également que $\lambda(t) \leq \bar{\lambda} < \infty$ pour tous t. Nous considérons que le centre ouvre au temps 0 et ferme au temps T. Soit A le nombre d'arrivées dans une journée et soit $a = \mathbb{E}[A] = \int_0^T \lambda(t)dt$ le nombre espéré d'arrivées.

Le centre emploie $N(t)$ agents au temps t. Si le nombre d'agents diminue, les contacts en cours de service sont terminés et les agents ne commencent pas de nouveaux services tant que le nombre d'agents occupés n'est pas inférieur à $N(t)$.

Un contact j arrivé au temps T_j est prêt à attendre P_j unités de temps et son service nécessitera S_j unités de temps. Nous supposons que $\mathbb{E}[S_j] \leq \bar{\mu} < \infty$. Si le nombre

d'agents occupés est inférieur à $N(t)$, le contact est alors servi immédiatement. Dans le cas contraire, il attend dans une file FIFO jusqu'à ce qu'un agent se libère ou que son temps de patience P_j soit épuisé. Dans ce dernier cas, le contact a abandonné et est perdu.

Les temps de patience et de service sont indépendants les uns des autres mais peuvent dépendre du temps d'arrivée. Soit $f_{\mathrm{P},t}(x)$ et $f_{\mathrm{S},t}(x)$ la densité du temps de patience et du temps de service, respectivement, pour un contact arrivé au temps t. Soit également $F_{\mathrm{P},t}(x)$ la fonction de répartition correspondante pour le temps de patience. Nous supposons que $f_{\mathrm{P},t}(x) \leq \kappa < \infty$ pour tous $t, x \in \mathbb{R}$.

Dans les preuves suivantes, nous traitons le cas où δ est un paramètre d'échelle ou de localisation des durées de service. Dans le cas du paramètre d'échelle, nous considérons que chaque temps de service S_j est multiplié par $1-\delta$ avec $0 \leq \delta < 1$. D'un autre côté, avec le paramètre de localisation, $\delta \geq 0$ est retranché de S_j et la différence est arrondie à 0 si elle est négative ; le temps de service devient alors $\max(0, S_j - \delta)$. Ces types de paramètres permettent d'étudier l'impact d'un changement du temps de service moyen et les preuves ne dépendent pas de la loi de probabilité particulière du temps de service.

4.4.1 Lemmes de base

Les lemmes suivants sont utilisés dans plusieurs des preuves de convergence qui suivent et sont des versions simplifiées du théorème 6.5 de [58].

Lemme 1 (Différence à moments finis) *Dans notre modèle, si $|\Delta| \leq A$ avec probabilité 1 pour toute valeur de δ, alors pour tous $q > 1$ tel que $\mathbb{E}[B^{2q}] < \infty$, il existe une constante $K_0 < \infty$ telle que*

$$K_{\mathrm{z}}(\delta, q) \stackrel{\mathrm{def}}{=} \sup_{\theta_1, \theta_2 \in \Upsilon} \mathbb{E}[\Delta^{2q}] \leq K_0.$$

Ceci est vrai pour tous $q > 1$ si tous les moments de B sont finis.

Preuve. Pour un facteur d'achalandage fixé et pour toute valeur de δ, nous avons

$$\mathbb{E}[\Delta^{2q} \mid B = b] \leq \mathbb{E}[A^{2q} \mid B = b] \leq (ab)^{2q},$$

car si $B = b$, A suit la loi de Poisson de taux ab et le k^{e} moment d'une variable de Poisson de taux λ est borné par λ^k. Alors,

$$\begin{aligned}\mathbb{E}[\Delta^{2q}] &= \int_0^\infty \mathbb{E}[\Delta^{2q} \mid B = b] f_{\text{B}}(b) db \\ &\leq \int_0^\infty (ab)^{2q} f_{\text{B}}(b) db \\ &= a^{2q}\mathbb{E}[B^{2q}] = K_0 < \infty\end{aligned}$$

si $\mathbb{E}[B^{2q}] < \infty$. □

Lemme 2 (Borne sur la variance de la différence) *Sous les hypothèses du lemme 1, si* $\mathbb{P}[\Delta \neq 0] \leq K_1 \delta = \mathcal{O}(\delta)$, *alors* $\text{Var}[\Delta] = \mathcal{O}(\delta^{1-\varepsilon})$ *pour* $\varepsilon = 1/q$.

Preuve. En appliquant l'inégalité de Holder avec $1/q = 1 - 1/p$, nous obtenons

$$\begin{aligned}\text{Var}[\Delta] &\leq \mathbb{E}[\Delta^2] \\ &= \mathbb{E}[\Delta^2 \mathbb{I}[\Delta^2 = 0]] + \mathbb{E}[\Delta^2 \mathbb{I}[\Delta^2 > 0]] \\ &\leq (\mathbb{E}[\Delta^{2q}])^{1/q} \mathbb{E}[\mathbb{I}^p[\Delta^2 > 0]]^{1/p} \\ &\leq K_{\text{z}}(\delta, q)^{1/q} \mathbb{P}[\Delta^2 > 0]^{1/p}.\end{aligned}$$

Par le lemme 1,

$$\begin{aligned}\text{Var}[\Delta] &\leq (K_0)^{1/q} \mathbb{P}[\Delta \neq 0]^{1-1/q} \\ &\leq (K_0)^{1/q} (K_1 \delta)^{1-\varepsilon} \\ &= \mathcal{O}(\delta^{1-\varepsilon}). \qquad \square\end{aligned}$$

Lemme 3 *Sous les hypothèses des lemmes 1 et 2, si* $\mathbb{P}[\Delta \neq 0 \mid B = b] = \mathcal{O}(\delta b^k)$ *pour k tel que* $\mathbb{E}[B^k] < \infty$, *alors* $\mathbb{P}[\Delta \neq 0] = \mathcal{O}(\delta)$ *si bien que* $\text{Var}[\Delta] = \mathcal{O}(\delta^{1-\varepsilon})$ *pour* $\varepsilon = 1/q$.

Preuve.

$$\begin{aligned}
\mathbb{P}[\Delta \neq 0] &= \int_0^\infty \mathbb{P}[\Delta \neq 0 \mid B = b] f_{\mathrm{B}}(b) db \\
&\leq K_2 \int_0^\infty \delta b^k f_{\mathrm{B}}(b) db \\
&= K_2 \mathbb{E}[B^k] \delta \\
&= \mathcal{O}(\delta). \qquad \square
\end{aligned}$$

Grâce à ce dernier lemme, il suffit, lorsque $|\Delta| \leq A$ avec probabilité 1, de montrer que $\mathbb{P}[\Delta \neq 0 \mid B = b] = \mathcal{O}(\delta b^k)$ pour montrer que $\mathrm{Var}[\Delta] = \mathcal{O}(\delta^{1-\varepsilon})$. Si tous les moments de B sont finis, cela s'applique pour n'importe quel $\varepsilon > 0$. Nous utiliserons cela dans les deux sous-sections suivantes.

4.4.2 Différence du nombre de contacts avec un temps d'attente inférieur à s_0

Soient S_j et $S_j(\delta)$ le temps de service du j^{e} contact dans la première et la seconde configuration, respectivement. Soient S_{G} et $S_{\mathrm{G}}(\delta)$ le nombre de contacts ayant attendu moins de s_0 dans les configurations 1 et 2 respectivement. Nous avons alors $\Delta_{\mathrm{G}} = S_{\mathrm{G}}(\delta) - S_{\mathrm{G}}$. La fonction Δ_{G} est constante par morceaux et donc discontinue. Comme $S_{\mathrm{G}} \leq A$ et $S_{\mathrm{G}}(\delta) \leq A$ avec probabilité 1, nous avons $|\Delta_{\mathrm{G}}| \leq A$ avec probabilité 1.

Soit $V_j = T_j + \min(P_j, s_0)$ le *temps virtuel seuil* du contact j, avec T_j le temps d'arrivée, P_j le temps de patience et s_0 le temps d'attente acceptable. Soit $V.$ le temps virtuel seuil pour un contact quelconque. Soit $S_{\mathrm{S},j}$ le temps de début de service du contact j dans la configuration 1 et $S_{\mathrm{S},j}(\delta)$ le temps de début de service dans la configuration 2. Si le contact j abandonne dans les configurations 1 ou 2, $S_{\mathrm{S},j}$ et $S_{\mathrm{S},j}(\delta)$ correspondent respectivement aux temps auxquels le service aurait débuté si le contact j n'avait pas abandonné. Soit $D_j = |S_{\mathrm{S},j} - S_{\mathrm{S},j}(\delta)|$ la différence entre les temps de début de service entre les deux configurations. Cette différence ne peut pas dépasser la somme des dif-

férences de durées de service de tous les contacts précédant j. Alors, pour tout j, nous avons

$$D_j \leq \sum_{\ell=1}^{j-1} (S_\ell - S_\ell(\delta)) = \delta \sum_{\ell=1}^{j-1} S_\ell. \tag{4.1}$$

si δ est un paramètre d'échelle de la loi des temps de service.

Habituellement, $S_{S,j}(\delta) \leq S_{S,j}$, mais pour certains contacts, il se peut que $S_{S,j}(\delta) > S_{S,j}$. Cela peut se produire si un contact $i < j$ qui abandonnait dans la configuration 1 et qui n'abandonne pas dans la configuration 2 a un très long temps de service. Ce temps de service peut occasionner un délai pour le contact j dans la configuration 2, d'où $S_{S,j}(\delta) > S_{S,j}$.

Un contact a une bonne qualité de service s'il est servi avant son temps de seuil, c'est-à-dire si $S_{S,j} \leq V_j$. Autrement, il a une mauvaise qualité de service. Avec la réduction du temps de service moyen, la qualité du service d'un contact j passe de mauvaise à bonne si et seulement si

$$S_{S,j}(\delta) \leq V_j < S_{S,j}$$

et de bonne à mauvaise si et seulement si

$$S_{S,j} \leq V_j < S_{S,j}(\delta).$$

En général, la qualité du service d'un contact change de statut si et seulement si

$$V_j \in \mathcal{I}_j \qquad \text{où} \qquad \mathcal{I}_j = [\min(S_{S,j}, S_{S,j}(\delta)), \max(S_{S,j}, S_{S,j}(\delta))). \tag{4.2}$$

Pour examiner le comportement de $V.$ conditionnel à $B = b$, prenons un intervalle $[t, t+\varepsilon]$ pour $\varepsilon > 0$ petit. Cet intervalle contient un des V_j si un contact arrive dans l'intervalle $[t - s_0, t - s_0 + \varepsilon]$ et n'abandonne pas avant s_0 unités de temps ou si un contact arrive au temps x où $t - s_0 \leq x \leq t + \varepsilon$ et abandonne durant l'intervalle $[t, t+\varepsilon]$. La

probabilité de ces deux événements est bornée par

$$\int_{t-s_0}^{t-s_0+\varepsilon}(1-F_{\mathrm{P},x}(s_0))\,b\lambda(x)dx+\int_{t-s_0}^{t+\varepsilon}\int_0^{\varepsilon}f_{\mathrm{P},x}(t-x+y)dy\,b\lambda(x)dx+o(\varepsilon) \quad (4.3)$$

$$\begin{aligned}
&\leq b\bar{\lambda}\int_{t-s_0}^{t-s_0+\varepsilon}dx+b\kappa\bar{\lambda}\int_{t-s_0}^{t+\varepsilon}\int_0^{\varepsilon}dydx+o(\varepsilon)\\
&= b\bar{\lambda}\,\varepsilon+b\kappa\bar{\lambda}\,\varepsilon\int_{t-s_0}^{t+\varepsilon}dx+o(\varepsilon)\\
&= b\bar{\lambda}\,\varepsilon+b\kappa\bar{\lambda}\,\varepsilon\,s_0+o(\varepsilon)\\
&= b\bar{\lambda}\,\varepsilon\,(1+\kappa s_0)+o(\varepsilon). \qquad (4.4)
\end{aligned}$$

Rappelons que κ correspond ici à la borne sur la densité du temps de patience.

La probabilité que $\mathcal{I}_j$ contienne V_j pour un contact j particulier est bornée par la probabilité que $\mathcal{I}_j$ contienne n'importe quel $V_\cdot$. Pour borner cette probabilité, nous partitionnons $\mathcal{I}_j$ en sous-intervalles de longueur ε, bornons la probabilité pour chaque sous-intervalle et sommons toutes ces bornes. Si $\varepsilon \to 0$, la somme correspond à une intégrale de Rieman par rapport au temps t. La borne pour tout sous-intervalle est donnée par l'expression (4.4) et ne dépend pas de t si bien qu'il est facile d'intégrer sur tout l'intervalle pour obtenir

$$\begin{aligned}
&\mathbb{P}[V_j\in\mathcal{I}_j\mid B=b,S_{\mathrm{S},j}(\delta),S_{\mathrm{S},j}]\\
\leq\ &\mathbb{P}[V_\cdot\in\mathcal{I}_j\mid B=b,S_{\mathrm{S},j}(\delta),S_{\mathrm{S},j}]\\
\leq\ &\int_{\mathcal{I}_j}b\bar{\lambda}\,(1+\kappa s_0)dt\\
=\ &b\bar{\lambda}\,(1+\kappa s_0)D_j.
\end{aligned}$$

Alors, $(b\bar{\lambda})(1+\kappa s_0)D_j$ est une borne sur la probabilité que (4.2) soit vraie.

Si le temps de patience est stationnaire, c'est-à-dire $f_{\mathrm{P},t}(x)=f_{\mathrm{P}}(x)\,\forall t\geq 0$, et si $f_{\mathrm{P}}(x)$ est une fonction continue, la borne donnée par l'expression (4.3) devient

$$\int_{t-s_0}^{t-s_0+\varepsilon}(1-F_{\mathrm{P}}(s_0))\,b\lambda(x)dx+\int_{t-s_0}^{t+\varepsilon}\int_0^{\varepsilon}f_{\mathrm{P}}(t-x+y)dy\,b\lambda(x)dx+o(\varepsilon)$$

$$\begin{aligned} &\leq (1-F_{\mathrm{P}}(s_0))b\bar{\lambda}\,\varepsilon + b\bar{\lambda}\int_{t-s_0}^{t+\varepsilon}\int_0^{\varepsilon} f_{\mathrm{P}}(t-x+y)dydx + o(\varepsilon) \\ &= (1-F_{\mathrm{P}}(s_0))b\bar{\lambda}\,\varepsilon + b\bar{\lambda}\int_{t-s_0}^{t+\varepsilon} \varepsilon f_{\mathrm{P}}(t-x)dx + o(\varepsilon) \end{aligned}$$

étant donné que si ε est suffisamment petit, nous pouvons supposer que la densité du temps de patience ne varie pas dans un intervalle de taille ε. Alors, l'expression devient

$$\begin{aligned} & b\bar{\lambda}\,\varepsilon\int_{s_0}^{\infty} f_{\mathrm{P}}(x)dx + b\bar{\lambda}\,\varepsilon\int_0^{s_0} f_{\mathrm{P}}(x)dx + o(\varepsilon) \\ = \;& b\bar{\lambda}\,\varepsilon + o(\varepsilon). \end{aligned}$$

La densité du temps de patience, $f_{\mathrm{P}}(x)$, n'a pas à être bornée, contrairement au cas non stationnaire. La probabilité que (4.2) soit vraie est alors bornée par $b\bar{\lambda}D_j$.

Soit J^* l'indice du premier contact qui change de statut d'une configuration à l'autre. Si aucun changement de statut ne se produit, nous avons $J^* = \infty$. Pour que $\Delta_{\mathrm{G}} \neq 0$, il faut au moins que $J^* < \infty$. Par contre, même si $J^* < \infty$, nous pouvons avoir $\Delta_{\mathrm{G}} = 0$ si le nombre de contacts dont la qualité de service passe de bonne à mauvaise est égal au nombre de contacts pour lesquels elle passe de mauvaise à bonne.

En combinant les bornes trouvées pour $\mathbb{P}[V_j \in \mathcal{I}_j \mid B = b, S_{\mathrm{S},j}(\delta), S_{\mathrm{S},j}]$ et D_j, nous obtenons

$$\begin{aligned} & \mathbb{P}[\Delta_{\mathrm{G}} \neq 0 \mid B = b] \\ \leq\; & \mathbb{P}[J^* < \infty \mid B = b] \\ =\; & \sum_{j=1}^{\infty} \mathbb{E}[\mathbb{I}[j \leq A]\mathbb{P}[J^* = j \mid S_{\mathrm{S},j}, S_{\mathrm{S},j}(\delta), B = b] \mid B = b] \\ \leq\; & \sum_{j=1}^{\infty} \mathbb{E}\left[\mathbb{I}[j \leq A]b\bar{\lambda}\,(1+\kappa s_0)D_j \mid B = b\right] \\ \leq\; & b\bar{\lambda}\,(1+\kappa s_0)\sum_{j=1}^{\infty}\mathbb{E}\left[\mathbb{I}[j \leq A]\delta\sum_{\ell=1}^{j-1} S_\ell \mid B = b\right]. \end{aligned}$$

Puisque les temps de service sont indépendants du processus d'arrivées,

$$\begin{aligned}
& \mathbb{P}[\Delta_G \neq 0 \mid B = b] \\
\leq\ & b\delta\bar{\lambda}\,(1+\kappa s_0)\sum_{j=1}^{\infty}\mathbb{P}[j \leq A \mid B = b]\mathbb{E}\left[\sum_{\ell=1}^{j-1} S_\ell\right] \\
=\ & b\delta\bar{\lambda}\,(1+\kappa s_0)\sum_{j=1}^{\infty}\mathbb{P}[j \leq A \mid B = b]\sum_{\ell=1}^{j-1}\mathbb{E}[S_\ell] \\
\leq\ & b\delta\bar{\mu}\bar{\lambda}\,(1+\kappa s_0)\sum_{j=1}^{\infty}\mathbb{P}[j \leq A \mid B = b](j-1) \\
=\ & b\delta\bar{\mu}\bar{\lambda}\,(1+\kappa s_0)\sum_{a=0}^{\infty}\mathbb{P}[A = a \mid B = b]\sum_{j=1}^{a}(j-1) \\
=\ & b\delta\bar{\mu}\bar{\lambda}\,(1+\kappa s_0)\sum_{a=0}^{\infty}\mathbb{P}[A = a \mid B = b]a(a-1)/2 \\
\leq\ & b\delta\bar{\mu}\bar{\lambda}\,(1+\kappa s_0)\mathbb{E}[A^2 \mid B = b] \\
=\ & a^2b^3\delta\bar{\mu}\bar{\lambda}\,(1+\kappa s_0) \\
=\ & \mathcal{O}(\delta b^3).
\end{aligned}$$

On peut aussi démontrer que $\mathbb{P}[\Delta_G \neq 0 \mid B = b] = \mathcal{O}(\delta b^3)$ si δ est un paramètre de localisation du temps de service, avec $S_j(\delta) = \max(0, S_j - \delta) \geq S_j - \delta$. Dans ce cas,

$$D_j \leq \sum_{\ell=1}^{j-1} S_\ell - S_\ell(\delta) = \delta(j-1)$$

si bien que $\mathbb{E}[D_j] \leq \delta(j-1)$ et la preuve précédente tient en remplaçant $\bar{\mu}$ par 1.

Puisque $|\Delta_G| \leq A$ avec probabilité 1, cela montre, par le lemme 3, que $\mathrm{Var}[\Delta_G] = \mathcal{O}(\delta^{1-\varepsilon})$.

4.4.3 Différence du nombre d'abandons

Pour le nombre d'abandons, nous démontrons, en utilisant le lemme 3, que la variance de la différence est dans $\mathcal{O}(\delta^{1-\varepsilon})$ pour n'importe quel $\varepsilon > 0$. Soient L et $L(\delta)$ le

nombre d'abandons dans les configurations 1 et 2, respectivement. Alors, $\Delta_L = L(\delta) - L$. Étant donné que le nombre d'abandons ne peut jamais excéder le nombre d'arrivées, nous avons $|\Delta_L| < A$ avec probabilité 1.

Au temps $T_j + P_j$, le contact j abandonne s'il n'est pas servi. De façon analogue à la sous-section précédente, un changement de statut se produit si

$$T_j + P_j \in \mathfrak{I}_j. \tag{4.5}$$

Pour que $\Delta_L \neq 0$, au moins un contact arrivé doit changer de statut d'une configuration à l'autre.

Soit un intervalle $[t, t+\varepsilon]$ avec ε petit. Un contact abandonne durant l'intervalle $[t, t+\varepsilon]$ s'il arrive à un temps x où $0 \leq x < t+\varepsilon$ et a un temps de patience entre $t-x$ et $t+\varepsilon-x$. La probabilité de cet événement est bornée par

$$\begin{aligned}
& \int_0^{t+\varepsilon} \left(\int_0^{\varepsilon} f_{P,x}(t-x+y)dy \right) b\lambda(x)dx + o(\varepsilon) \\
\leq\ & b\kappa\bar{\lambda} \int_0^{t+\varepsilon} \left(\int_0^{\varepsilon} dy \right) dx + o(\varepsilon) \\
=\ & b\kappa\bar{\lambda} \int_0^{t+\varepsilon} \varepsilon dx + o(\varepsilon) \\
=\ & b\kappa t\bar{\lambda}\,\varepsilon + o(\varepsilon) \\
\leq\ & b\kappa T\bar{\lambda}\,\varepsilon + o(\varepsilon).
\end{aligned}$$

Nous utilisons la borne κ sur la densité du temps de patience ainsi que le fait que $t \leq T$ pour tous temps t de la simulation. En appliquant le même raisonnement qu'à la sous-section précédente, nous avons que la probabilité que (4.5) soit vraie est bornée par $b\kappa T\bar{\lambda}D_j$.

Si $f_{P,t}(x) = f_P(x)\,\forall t$ et $f_P(x)$ est continue, la probabilité peut être bornée par $b\bar{\lambda}D_j$ puisque dans ce cas, l'intégrale se fait sur une densité. Ainsi, la densité $f_P(x)$ n'a pas à être bornée comme dans le cas non stationnaire.

Soit J^* l'indice du premier contact qui change de statut d'une configuration à l'autre. Si aucun contact ne change de statut avec la réduction du temps de service, $J^* = \infty$. Alors,

$$\begin{aligned} & \mathbb{P}[J^* = j \mid S_{\mathrm{S},j}, S_{\mathrm{S},j}(\delta), B = b] \\ \leq\ & \mathbb{P}[T_j + P_j \in \mathcal{I}_j \mid S_{\mathrm{S},j}, S_{\mathrm{S},j}(\delta), B = b] \\ \leq\ & \bar{\lambda} b \kappa T D_j. \end{aligned}$$

Comme dans la sous-section précédente, nous obtenons alors $\mathbb{P}[\Delta_{\mathrm{L}} \neq 0 \mid B = b] \leq \mathbb{P}[J^* < \infty \mid B = b] = \mathcal{O}(\delta b^3)$, avec δ un paramètre d'échelle ou de localisation. La même preuve s'applique même si Δ_{L} représente une différence de nombre d'abandons.

Puisque $|\Delta_{\mathrm{L}}| \leq A$ avec probabilité 1, cela montre, par le lemme 3, que $\mathrm{Var}[\Delta_{\mathrm{L}}] = \mathcal{O}(\delta^{1-\epsilon})$.

4.4.4 Différence du temps d'attente moyen

Soient W_j et $W_j(\delta)$ le temps d'attente observé du contact j dans la première et la seconde configuration, respectivement. Ce temps observé est différent du temps d'attente conditionnel à un service, donné par $S_{\mathrm{S},j} - T_j$ pour la première configuration et $S_{\mathrm{S},j}(\delta) - T_j$ pour la seconde configuration. Nous avons alors $\Delta_{\mathrm{W}} = \sum_{j=1}^{A} (W_j(\delta) - W_j)$ la somme des temps d'attente pour les contacts servis et ayant abandonné. La mesure de performance estimée est $\mathbb{E}[\Delta_{\mathrm{W}}]/a$, mais a ne varie pas d'une configuration à l'autre si nous utilisons les variables aléatoires communes et ne changeons pas le taux d'arrivée.

Nous avons, avec probabilité 1,

$$\begin{aligned} |\Delta_{\mathrm{W}}| &= \left| \sum_{j=1}^{A} (W_j(\delta) - W_j) \right| \\ &\leq \sum_{j=1}^{A} |W_j(\delta) - W_j| \end{aligned}$$

$$
\begin{aligned}
&= \sum_{j=1}^{A} |\min(P_j, S_{\mathrm{S},j}(\delta) - T_j) - \min(P_j, S_{\mathrm{S},j} - T_j)| \\
&= \sum_{j=1}^{A} \xi_j
\end{aligned}
$$

où $\xi_j = |\min(P_j, S_{\mathrm{S},j}(\delta) - T_j) - \min(P_j, S_{\mathrm{S},j} - T_j)|$.

Si le contact j est servi dans le cas des deux configurations, le j^{e} terme de la somme précédente devient $\xi_j = |S_{\mathrm{S},j}(\delta) - S_{\mathrm{S},j}| = D_j$. Si le contact j abandonne dans les deux configurations, le j^{e} terme devient $\xi_j = 0 \leq D_j$. Si le contact j abandonne dans la configuration 1 mais est servi dans la configuration 2, nous avons $P_j < S_{\mathrm{S},j} - T_j$, $S_{\mathrm{S},j}(\delta) - T_j \leq P_j$ et $\xi_j = |S_{\mathrm{S},j}(\delta) - T_j - P_j| = |P_j - S_{\mathrm{S},j}(\delta) + T_j| < |S_{\mathrm{S},j} - T_j - S_{\mathrm{S},j}(\delta) + T_j| = D_j$. Enfin, si le contact j abandonne dans la configuration 2 mais est servi dans la configuration 1, nous avons $P_j < S_{\mathrm{S},j}(\delta) - T_j$, $S_{\mathrm{S},j} - T_j \leq P_j$ et $\xi_j = |P_j - S_{\mathrm{S},j} - T_j| < D_j$. Ainsi, dans tous les cas, le j^{e} terme de la somme est borné par D_j. Cela nous donne

$$|\Delta_{\mathrm{W}}| \leq \sum_{j=1}^{A} D_j.$$

Si δ est un paramètre d'échelle, pour tout contact j, nous avons $S_j(\delta) = (1-\delta)S_j$ et

$$
\begin{aligned}
|\Delta_{\mathrm{W}}| &\leq \delta \sum_{j=1}^{A} \sum_{\ell=1}^{j-1} S_\ell \\
&= \delta \sum_{j=1}^{A-1} (A-j) S_j \\
&\leq \delta A \sum_{j=1}^{A-1} S_j.
\end{aligned}
$$

Alors,

$$\mathrm{Var}[\Delta_{\mathrm{W}}] \leq \mathbb{E}[\Delta_{\mathrm{W}}^2]$$

$$\leq \delta^2 \mathbb{E}\left[A^2 \sum_{j_1=1}^{A-1} \sum_{j_2=1}^{A-1} S_{j_1} S_{j_2}\right]$$
$$= \delta^2 \int_0^\infty \left(\sum_{a=0}^{\infty} \mathbb{P}[A=a \mid B=b] \mathbb{E}\left[A^2 \sum_{j_1=1}^{A-1} \sum_{j_2=1}^{A-1} S_{j_1} S_{j_2} \mid A=a, B=b\right]\right) f_B(b) db.$$

Puisque les temps de service sont indépendants entre eux et ne dépendent pas du processus d'arrivées,

$$\begin{aligned}
\mathrm{Var}[\Delta_W] &\leq \delta^2 \int_0^\infty \left(\sum_{a=0}^{\infty} \mathbb{P}[A=a \mid B=b] a^2 \sum_{j_1=1}^{a-1} \sum_{j_2=1}^{a-1} \mathbb{E}[S_{j_1}]\mathbb{E}[S_{j_2}]\right) f_B(b) db \\
&\leq \delta^2 \bar{\mu}^2 \int_0^\infty \left(\sum_{a=0}^{\infty} \mathbb{P}[A=a \mid B=b] a^2 (a-1)(a-1)\right) f_B(b) db \\
&\leq \delta^2 \bar{\mu}^2 \int_0^\infty \left(\sum_{a=0}^{\infty} \mathbb{P}[A=a \mid B=b] a^4\right) f_B(b) db \\
&= \delta^2 \bar{\mu}^2 \int_0^\infty \mathbb{E}[A^4 \mid B=b] f_B(b) db \\
&\leq \delta^2 \bar{\mu}^2 \int_0^\infty a^4 b^4 f_B(b) db \\
&= \delta^2 \bar{\mu}^2 a^4 \mathbb{E}[B^4] \\
&< \infty
\end{aligned}$$

puisque tous les moments de B sont finis. Alors, $\mathrm{Var}[\Delta_W] = \mathcal{O}(\delta^2)$.

Si δ est un paramètre de localisation, $S_j(\delta) = \max(0, S_j - \delta)$ et $|\Delta_W| \leq \sum_{j=1}^{A} \delta(j-1) \leq \delta A^2$. Alors, $\mathrm{Var}[\Delta_W] \leq \mathbb{E}[\Delta_W^2] \leq \delta^2 \mathbb{E}[A^4] = \mathcal{O}(\delta^2)$ puisque $\mathbb{E}[A^4] < \infty$ étant donné que tous les moments de B et d'une variable de Poisson sont bornés.

CHAPITRE 5

AUGMENTATION DE LA VITESSE DE SIMULATION À L'AIDE D'UNE CHAÎNE DE MARKOV EN TEMPS CONTINU ET DE L'UNIFORMISATION

Comme nous l'avons vu précédemment, utiliser la simulation pour effectuer de l'optimisation ou même de l'analyse de sensibilité lorsqu'il y a plusieurs scénarios à tester est souvent trop long en raison des milliers de réplications que cela nécessite. Nous pouvons dans certains cas améliorer les choses en employant des techniques de réduction de la variance, mais les adaptations demandées par ces techniques et leur efficacité dépendent de la structure du modèle. Par exemple, dans le chapitre 3, notre application de la stratification dépendait du processus d'arrivées utilisé. Avec un autre processus, il faudrait trouver une nouvelle variable de stratification.

Ce problème d'efficacité est habituellement contourné en utilisant des formules d'approximation très rapides à calculer pour restreindre l'espace des solutions pendant les premières étapes de l'optimisation et en employant la simulation seulement en dernier lieu pour raffiner la solution obtenue. Mais puisque les approximations sont trop grossières, il en résulte fréquemment des solutions sous-optimales, voire non réalisables. Dans ce dernier cas, une recherche locale est souvent utilisée pour réparer la solution qui devient réalisable tout en demeurant sous-optimale en général.

Pour traiter ce problème, nous proposons un compromis entre les approximations grossières et la simulation détaillée par événements discrets. Cela consiste à modéliser le centre de contacts par une chaîne de Markov en temps continu (CMTC) uniformisée, à simuler la chaîne de Markov en temps discret (CMTD) imbriquée correspondante et à calculer des espérances conditionnelles aux états visités et au nombre de transitions. Cette méthode, appelée *conversion en temps discret*, a été proposée initialement par [31].

En théorie, il existe des méthodes exactes permettant, pour n'importe quelle CMTC, d'obtenir la probabilité à long terme ou à un temps quelconque de se trouver dans un

état donné. Le calcul de probabilités à long terme est discuté dans tous les livres traitant des chaînes de Markov, comme [79]. Le calcul de probabilités à un temps quelconque est plus complexe en général, mais des formules simples existent si la CMTC est uniformisée [40, 42]. Malheureusement, ces méthodes ne conviennent que si l'espace d'états de la CMTC est petit.

Dans notre cas, la simulation est nécessaire en raison de l'espace d'états qui est grand et complexe ainsi que de la possibilité que les paramètres du système évoluent dans le temps. Par contre, utiliser la conversion en temps discret a pour avantage d'éviter d'avoir à générer des variables aléatoires exponentielles et de gérer une liste d'événements.

Les centres de contacts pour lesquels il est intéressant d'effectuer de l'optimisation reçoivent un grand volume de contacts par jour et emploient beaucoup d'agents. Par contre, plusieurs de ces centres peuvent être modélisés avec un nombre modéré de types de contacts et de groupes d'agents. Notre technique de simulation est, pour de tels cas, plus rapide que la simulation détaillée par événements discrets tout en donnant des résultats à peu près aussi réalistes. En effet, notre CMTC impose moins d'hypothèses simplificatrices que les modèles utilisés par les approximations courantes. Dans les cas où les hypothèses de notre CMTC ne sont pas toutes satisfaites, la dernière phase de l'optimisation peut utiliser un modèle plus détaillé et la simulation par événements discrets pour raffiner la solution trouvée. Bien entendu, des techniques de réduction de la variance, comme celles abordées dans les chapitres précédents, peuvent être appliquées, autant sur le modèle simplifié de CMTC que sur le modèle détaillé simulé par événements.

La première section de ce chapitre expose la méthode de conversion en temps discret pour estimer l'espérance d'une somme de coûts par transition dans un contexte général. Nous y comparons aussi différentes variantes de la méthode et examinons les problèmes posés par la variation des paramètres dans le temps.

Dans la seconde section, nous appliquons cette méthode aux centres de contacts avec plusieurs types de contacts et groupes d'agents, d'abord pour une seule période, puis pour plusieurs périodes. Nous discutons aussi des difficultés qui surviennent lorsque

nous voulons implanter la méthode efficacement pour ce modèle et comment appliquer efficacement les variables aléatoires communes dans ce contexte. En particulier, nous examinons comment la recherche indexée est utilisée pour accélérer la génération des transitions. Nous développons aussi une technique de partitionnement de l'ensemble d'états permettant de bénéficier d'un taux de transition adaptatif, ce qui permet de réduire le nombre de transitions fictives dans les cas où leur taux est trop élevé.

La troisième section discute des problèmes posés par le calcul de statistiques comme le niveau de service et le temps d'excès moyen, qui correspondent respectivement à la proportion des contacts répondus après un temps d'attente inférieur à un seuil et au temps moyen que chaque contact passe en file après avoir attendu un temps minimal acceptable. L'estimation de ces mesures de performance n'est pas triviale, car elles dépendent des temps d'attente qui ne sont pas générés par notre méthode. Nous traitons ce problème en calculant des probabilités et des espérances conditionnelles au nombre de transitions passées par le client dans la file et au nombre total de transitions pendant toute la simulation.

La quatrième section présente des résultats numériques qui montrent que notre méthode de simulation est plus efficace que la simulation par événements discrets dans les cas où le nombre de types de contacts et de groupes d'agents est modéré. La dernière section analyse la synchronisation des variables aléatoires communes sur un exemple numérique utilisant le simulateur simplifié développé dans ce chapitre et montre qu'il est mieux de simuler chaque configuration avec le même taux de transition, au pire en partitionnant l'ensemble d'états en un petit nombre de sous-ensembles.

L'application de la conversion en temps discret aux centres de contacts, le traitement des problèmes qu'elle pose ainsi que des estimateurs pour le niveau de service et le temps d'excès moyen sont nouveaux et représentent la contribution scientifique principale de ce chapitre, dont une partie est publiée dans [13]. Ce dernier article décrit le modèle de simulation que nous présentons ici et l'applique sur un problème d'optimisation du nombre d'agents dans un centre de contacts, sur une seule période. Le présent chapitre

se concentre quant à lui sur la simulation.

5.1 Simulation de chaînes de Markov en temps continu en général

Avant d'expliquer notre méthode de simulation, nous revenons à la définition d'une chaîne de Markov et présentons la notation mathématique utilisée. Nous définissons ensuite des fonctions de coûts représentant les quantités à estimer avec la méthode et rappelons la technique d'uniformisation. Puis nous exposons et comparons différentes variantes de la méthode de conversion en temps discret. Enfin, nous discutons des problèmes qui surviennent lorsque les paramètres peuvent varier dans le temps.

5.1.1 Chaîne de Markov en temps continu sur horizon fini

Rappelons maintenant la définition d'une CMTC, qui est donnée dans [58, 79]. Soit $\{X(t), t \geq 0\}$ un processus stochastique en temps continu avec un espace d'états $\mathcal{S}$ dénombrable. Un tel processus est dit Markovien si pour tout temps t, $X(t)$ contient suffisamment d'informations pour générer $\{X(s), s > t\}$. En d'autres mots, si X est une variable aléatoire pouvant être écrite comme une fonction de $\{X(s), s > t\}$, la loi de probabilité de X conditionnelle à $\{X(s), s \leq t\}$ est la même que la loi de X conditionnelle à $X(t)$. Le processus Markovien $\{X(t), t \geq 0\}$ est une chaîne de Markov en temps continu (CMTC) stationnaire si, en plus, pour $s \geq 0$,

$$\mathbb{P}[X(s+t) = j \mid \{X(u), 0 \leq u < s\}, X(s) = i] = \mathbb{P}[X(s+t) = j \mid X(s) = i] \stackrel{\text{def}}{=} p_{i,j}(t)$$

ne dépend pas de s.

Supposons que

$$q_{i,j} = \lim_{t \to 0} \frac{p_{i,j}(t)}{t}$$

existe et appelons cette limite le *taux de transition* de l'état i vers l'état j. Le taux de

transition hors de l'état i est quant à lui

$$q_i = \sum_{j \in \mathcal{S} \setminus \{i\}} q_{i,j}.$$

Le *générateur infinitésimal* d'une telle CMTC est une matrice $\mathbf{Q}$ de dimensions $|\mathcal{S}| \times |\mathcal{S}|$ dont chaque élément est $q_{i,j}$ pour $i \neq j$ et $q_{i,i} = -q_i$. Dans cette thèse, nous n'allons considérer que des CMTC pour lesquels les taux de transition existent.

Pour une telle CMTC, il est possible de démontrer que le temps de séjour dans tout état $i \in \mathcal{S}$ est exponentiel avec taux q_i et une transition de l'état i vers l'état j a lieu après ce séjour avec probabilité $p_{i,j} = q_{i,j}/q_i$, indépendamment du temps passé dans l'état i.

Pour toute CMTC, la *chaîne de Markov en temps discret (CMTD) imbriquée* correspondante est définie comme le processus stochastique $\{X_n, n \geq 0\}$ avec $X_n = X(\tau_n)$. Ici, $\tau_0 = 0$ et τ_n, pour $n \geq 1$, est le temps auquel la n^{e} transition de $\{X(t), t \geq 0\}$ se produit. Les probabilités de transition de la CMTD imbriquée sont $\mathbb{P}[X_{n+1} = j \mid \{X_0, \ldots, X_{n-1}\}, X_n = i] = \mathbb{P}[X_{n+1} = j \mid X_n = i] = p_{i,j}$ et sa *matrice de transitions*, notée $\mathbf{P}$, est formée par ces $p_{i,j}$.

Lorsque l'horizon est fini mais aléatoire, nous considérons le cas où T est le temps auquel la CMTC atteint pour la première fois n'importe quel état parmi un sous-ensemble $\mathcal{B} \subset \mathcal{S}$. Dans ce cas, soit $B(\mathcal{B}) = \min(n : X_n \in \mathcal{B}, X_j \notin \mathcal{B}, j = 0, \ldots, n-1)$ le nombre de transitions nécessaires pour atteindre $\mathcal{B}$. Le temps d'arrêt $T = \tau_{B(\mathcal{B})}$ et le nombre de transitions $M(T) = B(\mathcal{B})$ à considérer sont aléatoires.

Pour restreindre notre CMTC sur l'horizon déterministe $[0, T]$, définissons $M(t) = \max(n : \tau_n \leq t)$ comme le nombre de transitions nécessaires pour atteindre le temps t et considérons seulement les $M(T)$ premières transitions. Le temps d'arrêt T est alors déterministe, mais le nombre de transitions à simuler est aléatoire.

5.1.2 Fonctions de coûts

Nous désirons estimer une espérance $\mathbb{E}[C]$ où C représente un certain coût évalué sur l'intervalle de temps $[0,T]$. Pour certains modèles, nous pouvons exprimer C comme proposé dans [31] :

$$\int_0^T f(X(t))dt, \tag{5.1}$$

où $f : \mathcal{S} \to \mathbb{R}$ est une fonction associant un coût à chaque état. Cela correspond au coût total accumulé par le processus $\{X(t), 0 \leq t \leq T\}$. Le processus $\{X(t), t \geq 0\}$ est constant par morceaux si bien que nous pouvons récrire (5.1) comme

$$\sum_{n=0}^{M(T)-1} (\tau_{n+1} - \tau_n) f(X_n) + (T - \tau_{M(T)}) f(X_{M(T)}). \tag{5.2}$$

Si T est le temps (aléatoire) auquel la CMTC atteint un sous-ensemble d'états $\mathcal{B}$, $\tau_{M(T)} = \tau_{B(\mathcal{B})} = T$ si bien que le dernier terme de l'expression précédente est 0.

Dans d'autres modèles, C est une somme de coûts imposés par transition. Soit pour cela $g : (\mathcal{S}, \mathcal{S}) \to \mathbb{R}$ une fonction associant un coût à chaque transition. Le coût moyen est alors défini comme l'espérance de

$$\sum_{n=0}^{M(T)-1} g(X_n, X_{n+1}). \tag{5.3}$$

Les deux expressions précédentes ainsi que d'autres fonctions de coûts peuvent être écrites comme la fonction générale

$$C = \sum_{n=0}^{M(T)} C_n \tag{5.4}$$

où C_n dépend de toute l'histoire de la CMTC pendant l'intervalle $[0, \tau_n]$. Cela inclut la séquence des $M(\tau_n)$ premiers états visités et les temps $\tau_0, \ldots, \tau_n$.

5.1.3 Uniformisation

L'uniformisation [46] consiste à utiliser le même taux de transition q pour tous les états de la CMTC. Pour ce faire, nous supposons que notre CMTC est *uniformisable*, c'est-à-dire qu'il existe un taux de transition maximal

$$q \stackrel{\text{def}}{=} \sup_{i \in \mathcal{S}} q_i < \infty.$$

Nous augmentons alors le taux de transition q_i à q pour tous les états $i \in \mathcal{S}$ en permettant des transitions fictives qui ne provoquent aucun changement d'état. La probabilité d'une transition fictive depuis l'état i est $P_{i,i}(q) = 1 - q_i/q$ tandis que les autres probabilités de transition deviennent $P_{i,j}(q) = q_i p_{i,j}/q$. Soit $\{Y_n, n \geq 0\}$ la CMTD utilisant ces probabilités de transition $P_{i,j}(q)$. La matrice de transitions de $\{Y_n, n \geq 0\}$ peut être écrite comme $\mathbf{P}(q) = \mathbf{Q}/q + \mathbf{I}$, où $\mathbf{Q}$ est le générateur infinitésimal de $\{X(t), t \geq 0\}$ et $\mathbf{I}$ est la matrice identité.

Soit $\{N(t), t \geq 0\}$ un processus de Poisson avec taux q indépendant de $\{Y_n, n \geq 0\}$ et avec des instants de saut $0 = T_0 < T_1 < \cdots$. En utilisant ce processus, nous pouvons établir un lien entre la CMTD uniformisée et la CMTC originale : $X(t) = Y_{N(t)}$ et $Y_n = X(T_n)$. La combinaison de la CMTD avec matrice de transitions $\mathbf{P}(q)$ et du processus de Poisson $\{N(t), t \geq 0\}$ définit complètement la CMTC uniformisée.

De façon semblable à $B(\mathcal{B})$ au début de cette section, définissons $J(\mathcal{B}) = \min(n : Y_n \in \mathcal{B}, Y_j \notin \mathcal{B}, j = 0, \ldots, n-1)$ comme le nombre de transitions (incluant les transitions fictives) nécessaires pour atteindre $\mathcal{B}$ avec la CMTC uniformisée et $T = U_{J(\mathcal{B})}$ comme le temps de fin de l'horizon aléatoire.

Avec une CMTC générale, chaque temps inter-transition $\tau_{n+1} - \tau_n$ suit la loi exponentielle avec taux q_{X_n}. Pour toute transition n, le temps τ_n suit donc une loi de probabilité correspondant à une somme de n valeurs exponentielles avec des taux différents. Lorsque $\{X(t), t \geq 0\}$ est généré en utilisant $\{Y_n, n \geq 0\}$ et $N(t)$, le temps T_n de la n^{e} transition, qui peut être fictive, suit la loi Erlang avec paramètres n et q. De plus, si T

est déterministe, le nombre de transitions $N(T)$ suit la loi de Poisson avec taux qT si la CMTC est uniformisée.

La fonction de coûts C peut être récrite pour la CMTC uniformisée en remplaçant X_n par Y_n, τ_n par T_n et $M(T)$ par $N(T)$ dans les expressions de la sous-section précédente. En particulier, pour un coût accumulé par rapport au temps, la fonction C sera

$$\sum_{n=0}^{N(T)-1} (T_{n+1} - T_n) f(Y_n) + (T - T_{N(T)}) f(Y_{N(T)}). \tag{5.5}$$

Si un coût est imposé pour chaque transition, y compris les transitions fictives, la fonction C devient

$$\sum_{n=0}^{N(T)-1} g(Y_n, Y_{n+1}). \tag{5.6}$$

Enfin, la fonction de coûts générale devient

$$\sum_{n=0}^{N(T)} C_n \tag{5.7}$$

où C_n peut dépendre de la séquence des $N(T_n)$ premiers états visités et des instants $T_0, \ldots, T_n$. Dans le reste de ce chapitre, sauf indication contraire, nous considérons le coût C sur la CMTC uniformisée.

5.1.4 Conversion en temps discret pour la simulation

Il existe plusieurs façons de calculer ou d'estimer $\mathbb{E}[C]$. Dans le cas où C peut être exprimé comme un coût accumulé dans le temps (5.1), $\mathbb{E}[C]$ peut être estimée en utilisant les probabilités $\pi_j(t) = \sum_{i \in \mathcal{S}} p_{i,j}(t)\pi_i(0)$ à partir desquelles nous pouvons obtenir $\mathbb{E}[f(X(t))]$ pour n'importe quel temps t. Nous supposons ici que $\pi_i(0) = \mathbb{P}[X(0) = i]$ est connue pour tous $i \in \mathcal{S}$. À l'aide de l'intégration numérique, nous pouvons alors calculer $\int_0^T \mathbb{E}[f(X(t))]dt$, qui est équivalente à l'espérance de (5.1). Calculer les probabilités $\pi_j(t)$ de façon exacte pour une CMTC en général demande de résoudre des équations

différentielles [79], ce qui est difficile et coûteux en temps de calcul.

Dans le cas d'une CMTC uniformisée, il existe une méthode exacte pour calculer $\pi_j(t)$ pour toute valeur de $j \in \mathcal{S}$ et de t [40, 42]. Cette méthode consiste essentiellement à calculer $\pi_j^{(n)} = \mathbb{P}[Y_n = j]$ pour toutes les valeurs de n et à estimer la somme pondérée

$$\pi_j(t) = \sum_{n=0}^{\infty} \mathbb{P}[N(t) = n]\pi_j^{(n)} = \sum_{n=0}^{\infty} \pi_j^{(n)} \frac{e^{-qt}(qt)^n}{n!}.$$

Enfin, les probabilités à long terme π_j, c'est-à-dire $\pi_j(t)$ si t est grand, peuvent être calculées en résolvant le système d'équations $\boldsymbol{\pi}\mathbf{P} = \mathbf{P}$, où $\boldsymbol{\pi}$ est un vecteur de taille $|\mathcal{S}|$ regroupant les valeurs de π_j. Toutes ces méthodes exactes ne conviennent que si l'espace d'états de la CMTC est petit.

Une méthode simple et générale pour estimer $\mathbb{E}[C]$ sur l'horizon $[0,T]$ lorsque l'espace d'états est grand consiste à simuler des transitions jusqu'à atteindre le temps T pour ensuite générer C à partir de la réalisation de la CMTC obtenue. L'état X_0 est i avec probabilité $\mathbb{P}[X_0 = i]$ fixée, pour $i \in \mathcal{S}$. Pour chaque étape $n = 0, \ldots, M(T) - 1$, si la CMTD imbriquée se trouve dans l'état $X_n = i$, une variable aléatoire exponentielle $\tau_{n+1} - \tau_n$ de taux q_i est générée puis l'état devient $X_{n+1} = j$ avec probabilité $p_{i,j}$. Nous pouvons certes nous passer d'une liste d'événements pour cette simulation, ce qui diminue le temps de calcul par rapport à une simulation détaillée par événements discrets, mais il faut malgré tout générer des variables exponentielles. L'uniformisation n'apporte aucun gain si nous utilisons cette méthode.

Nous pouvons éviter de générer des variables aléatoires exponentielles en utilisant la conversion en temps discret. Cette méthode consiste à ne générer que la CMTD imbriquée uniformisée pour ensuite calculer l'espérance du coût conditionnelle à la séquence des états visités et au nombre de transitions simulées.

Définissons $\mathcal{G}_m$ comme la σ-algèbre engendrée par $Y_0, \ldots, Y_m$. Notre fonction de

coûts peut être écrite comme l'espérance de

$$\mathbb{E}[C \mid \mathcal{G}_{N(T)}] = \sum_{n=0}^{N(T)} \mathbb{E}[C_n \mid \mathcal{G}_{N(T)}].$$

Examinons maintenant comment calculer $\mathbb{E}[C \mid \mathcal{G}_{N(T)}]$ pour une réalisation donnée de $\mathcal{G}_{N(T)}$, et comment générer $\mathcal{G}_{N(T)}$.

5.1.5 Calcul de $\mathbb{E}[C \mid \mathcal{G}_{N(T)}]$

Le calcul de $\mathbb{E}[C \mid \mathcal{G}_{N(T)}]$ lorsque $\mathcal{G}_{N(T)}$ est connue dépend de la fonction de coûts utilisée. Le cas d'un coût accumulé de façon continue dans le temps a déjà été traité dans [31] pour le cas où T est le temps où la CMTC atteint $\mathcal{B}$ et pour celui où T est déterministe. Pour le premier cas, nous avons

$$\mathbb{E}[C \mid \mathcal{G}_{N(T)}, N(T) = J(\mathcal{B})] = \sum_{n=0}^{N(T)-1} \frac{f(Y_n)}{q}.$$

Pour le second cas, l'espérance conditionnelle devient

$$\mathbb{E}[C \mid \mathcal{G}_m, N(T) = m] = T \sum_{n=0}^{N(T)} \frac{f(Y_n)}{N(T)+1}.$$

Cette dernière espérance conditionnelle est calculée dans l'article [31] qui propose même une expression (plutôt complexe) pour le cas sans uniformisation.

Calculer $\mathbb{E}[C \mid \mathcal{G}_{N(T)}]$ si C est une somme de coûts imposés par transition ne cause aucune difficulté puisque C est $\mathcal{G}_{N(T)}$-mesurable dans ce cas. Par contre, en général, il faut développer une formule pour $\mathbb{E}[C_n \mid \mathcal{G}_{N(T)}]$ adaptée aux coûts C_n particuliers rencontrés dans nos modèles. Nous verrons des exemples de cela dans les sections 5.3.1 à 5.3.3.

5.1.6 Génération de $\mathcal{G}_{N(T)}$

Pour générer $N(T)$ et $\mathcal{G}_{N(T)}$ dans le cas de l'horizon aléatoire que nous considérons, il suffit de générer des transitions jusqu'à ce que $Y_n \in \mathcal{B}$. Dans cette section, nous nous concentrons sur le cas plus complexe d'un horizon déterministe, dans lequel le temps d'arrêt n'a rien à voir avec l'état de la CMTC. Les méthodes présentées ici s'inspirent fortement de [31].

La méthode la plus simple pour générer $\mathcal{G}_{N(T)}$ lorsque l'horizon est déterministe consiste à générer $N(T)$ depuis la loi de Poisson avec moyenne qT pour ensuite générer la CMTD imbriquée $Y_0, \ldots, Y_{N(T)}$. Cela nous donne l'estimateur $C_{\text{SIM},1} = \mathbb{E}[C \mid \mathcal{G}_{N(T)}]$.

Nous pouvons aussi générer la CMTD $Y_0, Y_1, \ldots$ pour ensuite intégrer sur les différentes valeurs possibles de $N(T)$, en pondérant avec la fonction de masse de la loi de Poisson. Sachant que $\mathcal{G}_1 \subset \cdots \subset \mathcal{G}_\infty$, nous pouvons construire l'estimateur

$$C_{\text{SIM},2} = \mathbb{E}[C \mid \mathcal{G}_\infty] = \sum_{k=0}^{\infty} \frac{(qT)^k e^{-qT}}{k!} \mathbb{E}[C \mid \mathcal{G}_k, N(T) = k].$$

Étant donné que nous intégrons par rapport au nombre de transitions tout en conservant la même séquence $Y_0, Y_1, \ldots$, $\text{Var}[C_{\text{SIM},2}] \leq \text{Var}[C_{\text{SIM},1}]$. En pratique, il nous faut bien entendu tronquer la somme $C_{\text{SIM},2}$ pour la calculer en temps fini, ce qui nous donne

$$\tilde{C}_{\text{SIM},2} = \mathbb{E}[C \mid \mathcal{G}_R] = \sum_{k=L}^{R} \frac{(qT)^k e^{-qT}}{k!} \mathbb{E}[C \mid \mathcal{G}_k, N(T) = k].$$

Les bornes L et R sont ajustées de telle sorte que $\mathbb{P}[L \leq N(T) \leq R] \geq 1 - \varepsilon$ avec ε très petit.

Si qT est grand, $N(T)$ suit approximativement la loi normale si bien que nous pouvons utiliser une approximation normale pour fixer des valeurs pour L et R. Nous pouvons également utiliser les bornes proposées dans [30] pour trouver les seuils de façon

plus exacte. Ce dernier article montre également que

$$\left| \sum_{k=0}^{\infty} \mathbb{P}[N(T)=k]w_k - \sum_{k=L}^{R} \mathbb{P}[N(T)=k]w_k \right| \leq 2\varepsilon\tilde{w}$$

où w_k est une pondération bornée par une constante $\tilde{w} < \infty$. Par contre, si $w_k = \mathbb{E}[C \mid \mathcal{S}_k, N(T)=k]$, il se peut que $w_k \to \infty$ si $k \to \infty$.

Supposons maintenant que $|\mathbb{E}[C_n \mid \mathcal{S}_k, N(T)=k]| \leq \tilde{C} < \infty$. Dans le cas des fonctions de coûts qui nous intéressent, cela impose que $f(i) \leq \tilde{f} < \infty$ et $g(i,j) \leq \tilde{g} < \infty$. Alors,

$$\begin{aligned}
& |C_{\mathrm{SIM},2} - \tilde{C}_{\mathrm{SIM},2}| \\
= & \left| \sum_{k=0}^{L-1} \frac{(qT)^k e^{-qT}}{k!} \mathbb{E}[C \mid \mathcal{S}_k, N(T)=k] + \sum_{k=R+1}^{\infty} \frac{(qT)^k e^{-qT}}{k!} \mathbb{E}[C \mid \mathcal{S}_k, N(T)=k] \right| \\
\leq & \sum_{k=0}^{L-1} \frac{(qT)^k e^{-qT}}{k!} \sum_{n=0}^{k} |\mathbb{E}[C_n \mid \mathcal{S}_k, N(T)=k]| + \sum_{k=R+1}^{\infty} \frac{(qT)^k e^{-qT}}{k!} \sum_{n=0}^{k} |\mathbb{E}[C_n \mid \mathcal{S}_k, N(T)=k]| \\
\leq & \tilde{C} \left(\sum_{k=0}^{L-1} \frac{(qT)^k e^{-qT}}{k!} (k+1) + \sum_{k=R+1}^{\infty} \frac{(qT)^k e^{-qT}}{k!} (k+1) \right) \\
= & \tilde{C}(\mathbb{E}[N(T)+1 \mid N(T) < L]\mathbb{P}[N(T) < L] + \mathbb{E}[N(T)+1 \mid N(T) > R]\mathbb{P}[N(T) > R]) \\
= & \tilde{C}(\mathbb{E}[N(T)+1] - \mathbb{E}[N(T)+1 \mid L \leq N(T) \leq R]\mathbb{P}[L \leq N(T) \leq R]) \\
\leq & \tilde{C}(\mathbb{E}[N(T)+1] - \mathbb{E}[N(T)+1 \mid L \leq N(T) \leq R](1-\varepsilon)) \\
\approx & \tilde{C}(qT+1)\varepsilon
\end{aligned}$$

en supposant que $\mathbb{E}[N(T) \mid L \leq N(T) \leq R] \approx \mathbb{E}[N(T)]$. D'après cette borne, plus qT est élevé, plus la valeur de ε à utiliser doit être petite pour une même erreur d'approximation. Par contre, en pratique, l'erreur est beaucoup plus petite puisque $C_n = 0$ pour plusieurs transitions (par exemple, pour les transitions fictives) si bien qu'en général, $\mathbb{E}[C \mid \mathcal{S}_k, N(T)=k] \ll \tilde{C}(qT+1)\varepsilon$.

Nous pouvons aussi obtenir des réalisations de $\mathcal{S}_{N(T)}$ en générant un (petit) nombre de valeurs aléatoires $N(T,1), \ldots, N(T,m)$ depuis la loi de Poisson avec taux qT et la chaîne $Y_0, \ldots, Y_{N^*(T)}$ où $N^*(T) = \max(N(T,1), \ldots, N(T,m))$. Nous pouvons alors esti-

mer C par

$$C_{\mathrm{SIM},3} = \mathbb{E}[C \mid \mathcal{G}_{N^*(T)}, N(T,1), \ldots, N(T,m)] = \frac{1}{m}\sum_{j=1}^{m} \mathbb{E}[C \mid \mathcal{G}_{N(T)}, N(T) = N(T,j)].$$

Cet estimateur est un cas intermédiaire entre $C_{\mathrm{SIM},1}$ et $C_{\mathrm{SIM},2}$ si bien que nous pouvons nous attendre à ce que $\mathrm{Var}[C_{\mathrm{SIM},2}] \leq \mathrm{Var}[C_{\mathrm{SIM},3}] \leq \mathrm{Var}[C_{\mathrm{SIM},1}]$. Nous pouvons aussi générer $N(T,j)$ par stratification. Pour cela, nous générons V_j uniformément sur l'intervalle $[(j-1)/m, j/m)$ et l'utilisons pour générer $N(T,j)$ par inversion. L'estimateur de C demeure encore $C_{\mathrm{SIM},3}$.

5.1.7 Variance dans un modèle simple

Si le coût est accumulé dans le temps ou imposé pour chaque transition, la fonction de coûts tient compte de toute l'histoire de de la CMTC. Le nombre de transitions exact n'a alors pas un impact aussi important que la séquence des états visités. Par conséquent, intégrer sur ce nombre dans de tels cas ne devrait pas permettre une réduction importante de la variance.

Par contre, il est facile de construire un cas où $C_{\mathrm{SIM},2}$ réduit beaucoup la variance par rapport à $C_{\mathrm{SIM},1}$. Supposons pour cela que $C_n = 0$ pour $n = 0, \ldots, N(T)-1$ et $C_n = n$ pour $n = N(T)$. Le coût dépend alors directement de $N(T)$ si bien qu'intégrer par rapport à ce nombre réduit la variance à 0.

Construisons maintenant un exemple semblable à un centre de contacts, où $C_{\mathrm{SIM},2}$ ne réduit pas beaucoup la variance par rapport à $C_{\mathrm{SIM},1}$. Dans ce modèle (très simple), des événements se produisent selon un processus de Poisson stationnaire avec taux λ. Chaque événement a un coût 1 avec probabilité p et un coût 0 avec probabilité $1-p$. Chaque événement pourrait correspondre à un contact tandis que p donnerait la probabilité que le contact abandonne.

Un tel modèle peut être représenté par une CMTC à deux états pour laquelle, pour $i = 0, 1$, $q_i = \lambda$, $p_{i,1} = p$ et $p_{i,0} = 1-p$. Fixons également $g(i,j) = j$ et considérons C

comme une somme de coûts imposés par transition.

Si nous fixons $N(T) = m$, le nombre d'événements avec coût 1 suit la loi binomiale avec paramètres m et p. Si $N(T)$ suit la loi de Poisson, le nombre d'événements suit aussi la loi de Poisson avec taux $\lambda T p$ [79, p. 59]. Ainsi, $\mathbb{E}[C_{\text{SIM},1}] = \text{Var}[C_{\text{SIM},1}] = \lambda T p$.

Dans le cas de $C_{\text{SIM},2}$, nous avons

$$\begin{aligned} C_{\text{SIM},2} &= \sum_{k=0}^{\infty} \frac{(\lambda T)^k e^{-\lambda T}}{k!} \mathbb{E}[C \mid \mathcal{G}_k, N(T) = k] \\ &= \sum_{k=0}^{\infty} \frac{(\lambda T)^k e^{-\lambda T}}{k!} \sum_{n=0}^{k-1} \mathbb{E}[C_n \mid \mathcal{G}_k, N(T) = k]. \end{aligned}$$

Dans cet exemple, si le nombre de transitions est connu mais la séquence des états ne l'est pas, l'espérance conditionnelle de C_n ne dépend que de l'état après la n^{e} transition, qui est une variable aléatoire de Bernouilli de paramètre p indépendante des états précédents. Nous allons nommer cette variable B_{n+1} pour pouvoir écrire

$$C_{\text{SIM},2} = \sum_{k=0}^{\infty} \frac{(\lambda T)^k e^{-\lambda T}}{k!} \sum_{n=1}^{k} B_n.$$

Sachant que $\mathbb{E}[B_n] = p$, nous avons $\mathbb{E}[C_{\text{SIM},2}] = \lambda T p$. Nous avons aussi

$$\begin{aligned} C_{\text{SIM},2} &= \sum_{n=1}^{\infty} B_n \sum_{k=n}^{\infty} \frac{(\lambda T)^k e^{-\lambda T}}{k!} \\ &= \sum_{n=1}^{\infty} B_n \mathbb{P}[N(T) \geq n]. \end{aligned}$$

Puisque les B_n sont indépendants, nous avons

$$\begin{aligned} \text{Var}[C_{\text{SIM},2}] &= \sum_{n=1}^{\infty} \text{Var}[B_n] \mathbb{P}^2[N(T) \geq n] \\ &= p(1-p) \sum_{n=1}^{\infty} \mathbb{P}^2[N(T) \geq n] \qquad (5.8) \\ &\leq p(1-p) \sum_{n=1}^{\infty} \mathbb{P}[N(T) \geq n] \end{aligned}$$

$$
\begin{aligned}
&= p(1-p)\mathbb{E}[N(T)] \\
&= \lambda T p(1-p) \\
&< \mathrm{Var}[C_{\mathrm{SIM},1}].
\end{aligned}
$$

Si p correspond à une probabilité d'abandon, nous pouvons nous attendre à ce que p soit petit et $1-p \approx 1$. Dans ce cas, $\mathrm{Var}[C_{\mathrm{SIM},1}]$ n'est pas beaucoup plus grande que la borne que nous venons de montrer pour $\mathrm{Var}[C_{\mathrm{SIM},2}]$.

En pratique, nous ne pouvons pas calculer $C_{\mathrm{SIM},2}$ exactement ; nous devons l'estimer avec $\tilde{C}_{\mathrm{SIM},2}$ qui tronque les valeurs possibles de $N(T)$. Puisque $\tilde{C}_{\mathrm{SIM},2} \approx C_{\mathrm{SIM},2}$, nous supposons que $\mathrm{Var}[\tilde{C}_{\mathrm{SIM},2}] \approx \mathrm{Var}[C_{\mathrm{SIM},2}]$.

Si λT est grand, $N(T)$ est approximativement normal et nous pouvons trouver une valeur de r relativement petite pour laquelle la probabilité que $N(T)$ prend des valeurs hors de l'intervalle $[\lambda T - r\sqrt{\lambda T}, \lambda T + r\sqrt{\lambda T}]$ est petite. Ainsi, $\mathbb{P}[N(T) \geq n] \approx 1$ pour $n < \lambda T - r\sqrt{\lambda T}$ et $\mathbb{P}[N(T) \geq n] \approx 0$ pour $n > \lambda T + r\sqrt{\lambda T}$. Dans ces deux cas $\mathbb{P}[N(T) \geq n] \approx \mathbb{P}^2[N(T) \geq n]$. Ainsi, $\mathbb{P}[N(T) \geq n]$ est loin de $\mathbb{P}^2[N(T) \geq n]$ seulement pour environ $2r\sqrt{\lambda T}$ valeurs de n si bien que nous pouvons penser que $\mathrm{Var}[\tilde{C}_{\mathrm{SIM},2}] \approx \lambda T p(1-p)$.

Prenons par exemple un nombre moyen d'arrivées de $\lambda T = 25\,000$ et une probabilité d'abandon de $p = 0{,}03$. Nous avons alors $\mathrm{Var}[C_{\mathrm{SIM},1}] = 750$ et $\mathrm{Var}[C_{\mathrm{SIM},2}] \leq 727{,}5$. Nous pouvons rechercher une valeur L telle que $\Phi((L-\lambda T)/\sqrt{\lambda T}) \leq \varepsilon_1$ et $\Phi((R-\lambda T)/\sqrt{\lambda T}) \geq 1-\varepsilon_2$, où $\Phi(\cdot)$ est la fonction de répartition de la loi normale standard. Avec $\varepsilon_1 = \varepsilon_2 = 10^{-6}/2$, nous obtenons $L = 24\,226$ et $R = 25\,774$. Avec ces paramètres, $\mathbb{P}[N(T) \geq n] \approx 1$ pour $n < L$, $\mathbb{P}[N(T) \geq n] \approx 0$ pour $n > R$ et est loin de 1 ou 0 pour seulement 1 548 valeurs. Nous pouvons utiliser cela pour calculer une approximation de la variance :

$$
\begin{aligned}
\mathrm{Var}[C_{\mathrm{SIM},2}] &= p(1-p)\sum_{n=1}^{\infty}\mathbb{P}^2[N(T) \geq n] \\
&\approx p(1-p)\left(L - 1 + \sum_{n=L}^{R}\mathbb{P}^2[N(T) \geq n]\right)
\end{aligned}
$$

$$\approx 724{,}9,$$

ce qui est près de notre borne de 727,5.

La figure 5.1 montre la progression de $\mathrm{Var}[C_{\mathrm{SIM},1}]$ et de $\mathrm{Var}[C_{\mathrm{SIM},2}]$ en fonction de p, pour $\lambda T = 25\,000$. Pour ces deux valeurs de λT, $\mathrm{Var}[C_{\mathrm{SIM},1}]$ est très près de la borne sur $\mathrm{Var}[C_{\mathrm{SIM},2}]$ pour de petites valeurs de p.

Pour que l'approximation de $\mathrm{Var}[C_{\mathrm{SIM},2}]$ par la borne $\lambda T p(1-p)$ soit valide quel que soit le nombre moyen d'arrivées, il faut que la somme $\sum_{n=1}^{\infty} \mathbb{P}^2[N(T) \geq n]$ soit une fonction à peu près linéaire de λT. Comme le montre la figure 5.2, le graphique de la somme de carrés par rapport à λT correspond à une droite. Nous pouvons ainsi supposer que la borne $\lambda T p(1-p)$ est une bonne approximation de $\mathrm{Var}[C_{\mathrm{SIM},2}]$.

Examinons maintenant le cas de

$$C_{\mathrm{SIM},3} = \frac{1}{m}\sum_{j=1}^{m} \mathbb{E}[C \mid \mathcal{G}_{N(T,j)}, N(T) = N(T,j)] = \frac{1}{m}\sum_{j=1}^{m}\sum_{n=1}^{N(T,j)} B_n = \frac{1}{m}\sum_{j=1}^{m} \tilde{B}_{N(T,j)}$$

où $\tilde{B}_x = \sum_{n=1}^{x} B_n$. La variance peut être écrite comme

$$\begin{aligned}
\mathrm{Var}[C_{\mathrm{SIM},3}] &= \frac{1}{m^2}\sum_{j=1}^{m} \mathrm{Var}[\tilde{B}_{N(T,j)}] + \frac{2}{m^2}\sum_{j_1=1}^{m-1}\sum_{j_2=j_1+1}^{m} \mathrm{Cov}[\tilde{B}_{N(T,j_1)}, \tilde{B}_{N(T,j_2)}] \\
&= \frac{\lambda T p}{m} + \frac{2}{m^2}\sum_{j_1=1}^{m-1}\sum_{j_2=j_1+1}^{m} \mathrm{Cov}[\tilde{B}_{N(T,j_1)}, \tilde{B}_{N(T,j_2)}].
\end{aligned}$$

Le lemme suivant est nécessaire pour analyser les covariances.

Lemme 4 *Supposons que X et Y sont des variables aléatoires discrètes correspondant à des entiers positifs, indépendantes entre elles et indépendantes des B_n. Alors,*

$$\mathrm{Cov}[\tilde{B}_X, \tilde{B}_Y] = \mathbb{E}[\min(X,Y)]\mathrm{Var}[B_n].$$

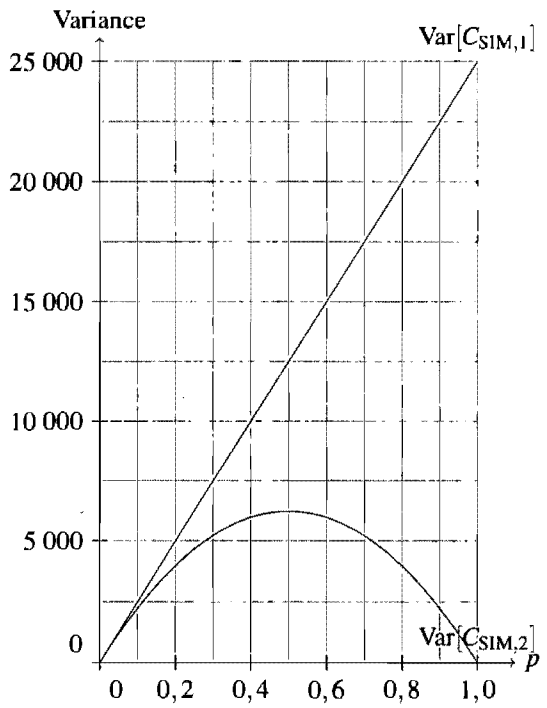

Figure 5.1 – Variance des estimateurs $C_{SIM,1}$ et $C_{SIM,2}$ en fonction de p

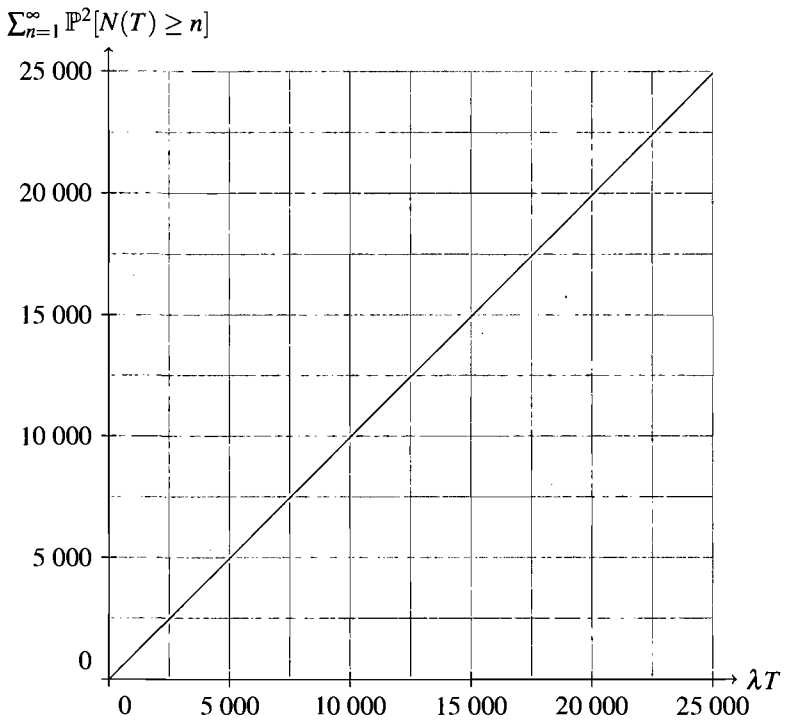

Figure 5.2 – Comparaison de λT avec $\sum_{n=1}^{\infty} \mathbb{P}^2[N(T) \geq n]$

Preuve. Nous utiliserons l'égalité

$$\mathrm{Cov}[A,B] = \mathbb{E}[\mathrm{Cov}[A,B \mid \mathcal{G}]] + \mathrm{Cov}[\mathbb{E}[A \mid \mathcal{G}], \mathbb{E}[B \mid \mathcal{G}]]$$

qui est vraie pour toute paire de variables aléatoires A, B et tout ensemble de variables aléatoires $\mathcal{G}$. D'abord,

$$\begin{aligned}\mathrm{Cov}[\tilde{B}_X, \tilde{B}_Y \mid X,Y] &= \mathrm{Cov}[\tilde{B}_{\min(X,Y)}, \tilde{B}_{\min(X,Y)} + \tilde{B}_{\max(X,Y)} - \tilde{B}_{\min(X,Y)} \mid X,Y] \\ &= \mathrm{Var}[\tilde{B}_{\min(X,Y)} \mid X,Y] + \mathrm{Cov}[\tilde{B}_{\min(X,Y)}, \tilde{B}_{\max(X,Y)} - \tilde{B}_{\min(X,Y)} \mid X,Y].\end{aligned}$$

La variable $\tilde{B}_{\min(X,Y)} = \sum_{n=1}^{\min(X,Y)} B_n$ est indépendante de $\tilde{B}_{\max(X,Y)} - \tilde{B}_{\min(X,Y)} = \sum_{n=\min(X,Y)+1}^{\max(X,Y)} B_n$ puisque les B_n sont indépendants. Ainsi, le deuxième terme de l'équation précédente vaut 0. Alors,

$$\mathbb{E}[\mathrm{Cov}[\tilde{B}_X, \tilde{B}_Y \mid X,Y]] = \mathbb{E}[\mathrm{Var}[\tilde{B}_{\min(X,Y)} \mid X,Y]] = \mathbb{E}[\min(X,Y)]\mathrm{Var}[B_n].$$

De plus,

$$\mathrm{Cov}[\mathbb{E}[\tilde{B}_X \mid X,Y], \mathbb{E}[\tilde{B}_Y \mid X,Y]] = \mathrm{Cov}[X\mathbb{E}[B_n], Y\mathbb{E}[B_n]] = (\mathbb{E}[B_n])^2\mathrm{Cov}[X,Y] = 0.$$

Alors,

$$\mathrm{Cov}[\tilde{B}_X, \tilde{B}_Y] = \mathbb{E}[\min(X,Y)]\mathrm{Var}[B_n]. \qquad \square$$

Fixons $X = N(T, j_1)$ et $Y = N(T, j_2)$ pour $j_1 \neq j_2$. Alors,

$$\mathrm{Cov}[\tilde{B}_{N(T,j_1)}, \tilde{B}_{N(T,j_2)}] = p(1-p)\mathbb{E}[\min(N(T,j_1), N(T,j_2))].$$

Essayons maintenant de trouver la valeur de $\mathbb{E}[\min(N(T,j_1), N(T,j_2))]$. Pour cela,

$$\mathbb{P}[\min(N(T,j_1), N(T,j_2)) \geq x] = \mathbb{P}[N(T,j_1) \geq x, N(T,j_2) \geq x]$$

$$= \mathbb{P}^2[N(T) \geq x]$$

étant donné que $N(T, j_1)$ et $N(T, j_2)$ sont indépendants. Puisque

$$\mathbb{E}[\min(N(T, j_1), N(T, j_2))] = \sum_{k=0}^{\infty} \mathbb{P}^2[N(T) \geq k],$$

nous pouvons utiliser le fait que $\mathbb{P}^2[N(T) \geq k] \leq \mathbb{P}[N(T) \geq k]$ pour établir que

$$\mathbb{E}[\min(N(T, j_1), N(T, j_2))] \leq \lambda T.$$

La même analyse qu'avec $\mathrm{Var}[C_{\mathrm{SIM},2}]$ s'applique ici pour permettre de supposer que $\mathbb{E}[\min(N(T, j_1), N(T, j_2))] \approx \lambda T$.

Alors,

$$\begin{aligned}
\mathrm{Var}[C_{\mathrm{SIM},3}] &\leq \frac{\lambda T p}{m} + \frac{2}{m^2} \sum_{j_1=1}^{m-1} \sum_{j_2=j_1+1}^{m} \lambda T p(1-p) \\
&= \frac{p}{m}(\lambda T + \lambda T(1-p)(m-1)) \\
&= \frac{\lambda T p}{m}(1 + (1-p)(m-1)) \\
&= \frac{\lambda T p}{m}(m - pm + p) \\
&= \lambda T p(1 - p + p/m).
\end{aligned}$$

Avec $m = 1$, $\mathrm{Var}[C_{\mathrm{SIM},3}] = \lambda T p = \mathrm{Var}[C_{\mathrm{SIM},1}]$. Plus m augmente, plus la borne sur $\mathrm{Var}[C_{\mathrm{SIM},3}]$ diminue. Avec $m \to \infty$, nous retrouvons la borne $\lambda T p(1-p)$ sur $\mathrm{Var}[C_{\mathrm{SIM},2}]$.

5.1.8 Variation des paramètres dans le temps

Jusqu'à présent, nous avons considéré un système stationnaire sur un horizon fini, avec un temps d'arrêt soit déterministe, soit correspondant au moment aléatoire où la CMTC atteint un sous-ensemble d'états. Nous pouvons lever cette restriction tout sim-

plement en permettant aux paramètres de la CMTC de dépendre du temps. Pour cela, soit

$$q_{i,j}(t) = \lim_{\delta \to 0} \frac{\mathbb{P}[X(t+\delta) = j \mid X(t) = i]}{\delta}$$

le taux de transition de i vers j au temps t. Soit également $q_i(t) = \sum_{j \in \mathcal{S} \setminus \{i\}} q_{i,j}(t)$ le taux de transition depuis l'état i au temps t. En général, il est difficile de générer le temps de séjour dans chaque état, qui n'est plus exponentiel puisque le taux de transition $q_i(t)$ varie de façon continue dans le temps. Pour résoudre ce problème, il suffit d'appliquer l'uniformisation en utilisant le taux de transition maximal

$$q = \sup_{i \in \mathcal{S}, 0 \leq t \leq T} q_i(t)$$

sous l'hypothèse que $q < \infty$. Le temps entre les transitions est alors de nouveau exponentiel, mais les probabilités de transition changent avec t. Puisque le temps t n'est pas généré par la conversion en temps discret, il faudrait, pour appliquer cette méthode, intégrer numériquement sur t à chaque étape pour obtenir les probabilités de transition. En général, cela rend la méthode beaucoup trop coûteuse lorsque $|\mathcal{S}|$ est grand.

Pour remédier à cela, nous supposons que les paramètres du système peuvent varier dans le temps selon une fonction constante par morceaux. En particulier, dans les centres de contacts que nous considérons, l'horizon simulé est habituellement divisé en intervalles de temps pendant lesquels les paramètres sont constants.

Nous pouvons alors simuler la CMTC avec le taux de transition q en utilisant la conversion en temps discret. Supposons pour cela que tous les paramètres sont fixés pour $t \in [t_{p-1}, t_p)$, avec $p = 0, \ldots, P+1$, $0 = t_{-1} \leq t_0 < t_1 < \cdots < t_P \leq t_{P+1} = T$ une séquence de temps. Nous devons, pour chaque intervalle $[t_{p-1}, t_p)$, générer le nombre de transitions $N(t_p) - N(t_{p-1})$ pour ensuite simuler la CMTD imbriquée avec probabilités $P_{i,j} = q_{i,j}(t_{p-1})/q$ et $P_{i,i} = 1 - q_i(t_{p-1})/q$, pour $i, j \in \mathcal{S}$.

Nous pouvons générer $N(t_p) - N(t_{p-1})$ indépendamment pour $p = 0, \ldots, P+1$, depuis la loi de Poisson avec taux $(t_p - t_{p-1})q$ ou encore générer $N(T)$ puis le nombre

de transitions pour chaque intervalle conditionnellement à $N(T)$ sachant que si $N(T)$ est connu, le vecteur $\big(N(t_p)-N(t_{p-1})\big)_{p=0}^{P+1}$ suit la loi multinomiale avec paramètres $N(T),(t_0-0)/T,(t_1-t_0)/T,\ldots,(t_P-t_{P-1})/T,(T-t_P)/T$. La seconde méthode permet de subdiviser l'intervalle $[0,T]$ de plusieurs façons différentes, pour traiter le cas où nous avons plusieurs paramètres variant avec le temps.

Toutefois, dans certains intervalles, il se peut que le taux de transition $\sup_{i\in\mathcal{S}} q_i(t_{p-1})$ soit beaucoup plus petit que q. Beaucoup de transitions fictives seront alors simulées inutilement. Pour traiter ce problème, il suffit de simuler une CMTC différente pour chacun des intervalles en fixant l'état initial de la CMTC pour l'intervalle p à l'état final de la période $p-1$. Le type d'horizon peut même varier d'une chaîne à l'autre. Par exemple, dans un centre de contacts, la chaîne pour la période de fermeture $[t_P,T]$ est simulée sur horizon aléatoire, jusqu'à ce que le système soit vide, tandis que l'horizon est déterministe pour toute autre période.

5.2 Application aux centres de contacts

Dans cette section, nous présentons un modèle de CMTC pour simuler un centre de contacts sur une seule période. Il est par contre facile de généraliser ce modèle à plusieurs périodes. Contrairement aux modèles de CMTC proposés dans [26] qui concernent un centre d'appels mixte avec seulement deux types d'appels et un ou deux groupes d'agents et qui sont résolus numériquement, notre modèle permet en théorie un nombre arbitraire de types de contacts et de groupes d'agents. Mais il ne prend en charge que les appels entrants.

Dans ce modèle, nous supposons que les contacts de type k arrivent selon un processus de Poisson avec un taux λ_k indépendamment des contacts des autres types. Les temps de service des contacts de type k servis par des agents du groupe i sont i.i.d. exponentiels avec moyenne $1/\mu_{k,i}$. Chacun des N_i agents du groupe i est formé pour servir un sous-ensemble $S_i \subseteq \{1,\ldots,K\}$ des types de contacts. Si $k \notin S_i$, nous pouvons fixer $\mu_{k,i}=0$.

Un contact de type k ne pouvant pas être servi immédiatement lors de son arrivée abandonne sans attendre en file avec probabilité ρ_k. S'il attend en file, il abandonne si son temps d'attente excède son temps de patience. Les temps de patience pour les contacts de type k sont i.i.d. exponentiels avec moyenne $1/\nu_k$. Les contacts qui abandonnent sont perdus ; le modèle ne considère pas les rappels. La file k du modèle, pour $k = 1, \ldots, K$, ne contient que des contacts de type k et est de type premier arrivé premier servi.

La politique de routage suivante est utilisée pour choisir un groupe d'agents lors d'une arrivée. Pour cela, chacun des types de contact possède une liste $I_{k,1}, I_{k,2}, \ldots, I_{k,l_k}$ d'ensembles de groupes d'agents, ordonnée par préférence. Ces ensembles forment une partition de $\{i = 1, \ldots, I : k \in S_i\}$, l'ensemble des groupes d'agents qui peuvent servir les contacts de type k. Si un agent est disponible dans un des groupes de $I_{k,1}$, le nouveau contact est acheminé vers le groupe de $I_{k,1}$ ayant le plus d'agents libres. Si plusieurs groupes $i \in I_{k,1}$ contiennent le même nombre d'agents libres, nous prenons le groupe avec le plus grand nombre d'agents libres et la plus petite valeur de i. Sinon, les ensembles de groupes $I_{k,2}$, $I_{k,3}$, etc. sont testés de façon similaire à $I_{k,1}$ dans l'ordre donné par la liste afin de choisir le premier sous-ensemble de groupes qui contient au moins un agent libre. Avec cette politique, les contacts *débordent* d'un ensemble de groupes d'agents vers l'autre.

Pour choisir un contact en file lors d'une fin de service, chaque groupe d'agents dispose d'une liste $K_{i,1}, K_{i,2}, \ldots, K_{i,m_i}$ d'ensembles de types de contacts, ordonnée par préférence. Ces sous-ensembles forment une partition de S_i. D'abord, si une file de $K_{i,1}$ comprend au moins un contact, le service débute pour le contact des files de $K_{i,1}$ ayant passé le plus grand nombre de transitions en attente. Si plusieurs files $k \in K_{i,1}$ comportent un contact ayant passé le même nombre de transitions en attente, nous prenons la file avec le plus petit indice k. Dans le cas contraire, les ensembles $K_{i,2}$, $K_{i,3}$, etc. sont testés de façon semblable à $K_{i,1}$ et dans l'ordre donné par la liste, dans le but de trouver le premier sous-ensemble contenant une file avec au moins un contact.

Les politiques précédentes s'inspirent de celles utilisées couramment dans les centres

de contacts réels. D'autres politiques de routage tenant compte de l'état du système pourraient également être implantées sans modifier le modèle.

Examinons maintenant comment construire une CMTC pour ce modèle, ainsi que les problèmes posés par l'implantation efficace d'un simulateur utilisant la conversion en temps discret pour cette CMTC. Nous abordons l'estimation des mesures de performance qui nous intéressent dans la section suivante.

5.2.1 Construction de la CMTC

Définissons tout d'abord l'espace d'états avant de traiter des différents types de transitions. Il est nécessaire de connaître à chaque instant le nombre de contacts en service auprès des agents de chaque groupe afin d'empêcher le routage de contacts vers des groupes d'agents qui ne comportent aucun membre disponible. Il faut aussi conserver le nombre de contacts de chaque type en attente afin de permettre au routeur d'appliquer une sélection de type de contact pour un agent devenant libre. De plus, si le temps de service ne dépend pas uniquement du groupe d'agents de destination ou s'il est nécessaire de connaître à tout moment la répartition des contacts en service en fonction de leur type, nous devons conserver le nombre de contacts de chaque type k en service auprès de chaque groupe d'agents i.

L'état du système est alors représenté par un vecteur contenant les valeurs de Q_k, le nombre de contacts de type k en attente, et de $S_{k,i}$, le nombre de contacts de type k en service par un agent de groupe i. Évidemment, $Q_k \geq 0$ et $S_{k,i} \geq 0$. Habituellement, $\sum_{k=1}^{K} S_{k,i} \leq N_i$, c'est-à-dire que le nombre d'agents occupés dans le groupe i est inférieur ou égal au nombre d'agents dans ce groupe. Mais il peut arriver que cette condition soit violée si le nombre d'agents varie dans le temps, comme nous le verrons dans la section 5.2.4. Il est commun de trouver des centres de contacts avec une douzaine de types de contacts, une douzaine de groupes d'agents avec au total une centaine d'agents et pour lesquels la file d'attente peut contenir plusieurs centaines de contacts. Si nous représentons un tel centre avec notre modèle, l'espace d'états est très grand et comporte

plusieurs dimensions.

Le tableau 5.I résume les $K(I+3)$ types possibles d'événements pour ce système. Chaque ligne donne le type de l'événement, le taux auquel l'événement survient en fonction de l'état du système et l'effet usuel de l'événement sur le modèle. Plus précisément, lorsqu'un contact de type k entre dans le système, un groupe d'agents i' contenant au moins un agent libre lui est associé en utilisant la politique de routage. Si un tel groupe i' est trouvé, $S_{k,i'}$ est augmenté d'une unité et un nouveau service débute. Si aucun agent n'est disponible pour servir le nouveau contact, selon le type de l'événement, soit le contact abandonne immédiatement et aucun changement d'état ne survient, soit il entre dans la file d'attente et Q_k augmente d'une unité.

Lorsqu'un agent du groupe i termine le service d'un contact de type k, $S_{k,i}$ diminue d'une unité puis si $\sum_{k=1}^{K} S_{k,i} < N_i$, une file d'attente k' contenant au moins un contact est choisie par la politique de routage. Si une telle file existe, $Q_{k'}$ diminue d'une unité et $S_{k',i}$ augmente d'une unité. Enfin, lorsqu'un contact de type k abandonne, la taille de la file Q_k diminue d'une unité.

Nous devons borner la capacité de la file à $H < \infty$ pour que le taux de transition maximal soit fini. Dans le cas où $\sum_{k=1}^{K} Q_k = H$, les arrivées sont interdites jusqu'à ce que la taille de la file diminue. Quand un événement correspondant à une arrivée survient au moment où la file est pleine, aucun changement d'état ne se produit. Pour approximer un modèle avec capacité de file illimitée, il faut trouver, par des expériences pilotes, une petite valeur de H qui donne une très petite probabilité qu'un contact soit bloqué. Évidemment, plus H est élevée, plus le taux de transition et le temps de simulation seront élevés.

Le taux de transition maximal est donné par la somme des taux maximaux d'arrivée, de fin de service et d'abandon. Ainsi,

$$q = \lambda + \sum_{i=1}^{I} N_i \mu_i + H\nu$$

Tableau 5.I – Types d'événements possibles et effet habituel sur l'état

Type	Taux	Effet habituel
Pour $k = 1, \ldots, K$,		
Arrivée de type k avec service immédiat	λ_k	Augmente le nombre d'agents occupés.
Arrivée de type k avec abandon immédiat	$\rho_k \lambda_k$	Aucun effet sur l'état.
Arrivée de type k, sans abandon immédiat	$(1-\rho_k)\lambda_k$	Augmente la taille de la file.
Abandon de type k	$Q_k \nu_k$	Diminue la taille de la file.
Pour $k = 1, \ldots, K$ et $i = 1, \ldots, I$,		
Fin d'un service de type k par un agent du groupe i	$S_{k,i}\mu_{k,i}$	Diminue le nombre d'agents occupés ou la taille de la file.

avec

$$\begin{aligned} \lambda &= \sum_{k=1}^{K} \lambda_k, \\ \mu_i &= \max_{k \in S_i} \mu_{k,i}, \\ \nu &= \max_{k \in \{1,\ldots,K\}} \nu_k. \end{aligned}$$

Il arrive parfois que nous souhaitions examiner l'effet d'un changement de paramètre sur la performance du système, pour effectuer une analyse de sensibilité ou dans le but d'optimiser un paramètre. Si la technique des variables aléatoires communes est utilisée pour réduire la variance sur les différences obtenues, garder le même taux de transition pour chaque configuration favorise la synchronisation entre les différentes configurations testées. De plus, si les paramètres varient dans le temps et si nous subdivisons l'horizon en périodes, nous verrons à la section 5.3.5 que faire varier le taux de transition d'une période à l'autre pose des problèmes pour l'estimation de certaines mesures de performance. Il est ainsi plus facile de conserver le même taux de transition pendant l'horizon

entier. C'est pourquoi nous utilisons parfois un taux de transition maximal

$$\tilde{q} = \tilde{\lambda} + \sum_{i=1}^{I} \tilde{N}_i \tilde{\mu}_i + \tilde{H}\tilde{\nu} > q$$

où $\tilde{\lambda} \geq \lambda$, $\tilde{N}_i \geq N_i$, etc. sont des bornes supérieures sur les différents paramètres du modèle.

5.2.2 Simulation avec la recherche indexée

Pour simuler la CMTC précédente, nous devons générer les transitions, ce qui revient à sélectionner de façon aléatoire le prochain état j avec probabilité $p_{i,j}$. Cela équivaut à générer une variable aléatoire selon une loi discrète. La méthode la plus simple pour cela est l'inversion par recherche linéaire [53]. Avec cette méthode, pour générer une réalisation d'une variable discrète X dont la fonction de répartition est $F(x)$, nous générons U depuis la loi uniforme sur l'intervalle $[0,1)$ pour ensuite rechercher de façon linéaire la plus petite valeur de x pour laquelle $F(x) \geq U$, et utilisons $X = x$. Mais cette recherche est coûteuse pour générer un type de transition parmi $1 + K(I+3)$ types différents (incluant une possible transition fictive), si K ou I sont grands. La plupart des méthodes pour accélérer la génération précalculent des tables en considérant que les paramètres de la loi discrète sont fixés. Ici, les paramètres dépendent de l'état initial qui varie dans le temps. Si $|\mathcal{S}|$ est grand, précalculer et stocker une table pour chaque état est inacceptable.

Appliquée de façon naïve, la recherche linéaire peut nuire à la synchronisation lorsque nous utilisons les variables aléatoires communes. Pour illustrer cela, prenons $K = I = 1$ et supposons que nous faisons varier le nombre N_1 d'agents dans le centre de contacts entre 0 et $\tilde{N}_1$. Si nous associons l'intervalle $[0, \lambda_1/\tilde{q})$ aux arrivées, $[\lambda_1/\tilde{q}, (\lambda_1 + N_1\mu_1)/\tilde{q})$ aux fins de service et $[(\lambda_1 + N_1\mu_1)/\tilde{q}, (\lambda_1 + N_1\mu_1 + Q_1\nu)/\tilde{q})$ aux abandons, l'intervalle associé aux abandons varie clairement en fonction de N_1 si bien que changer le nombre d'agents affecte le type de certains événements : des fins de service peuvent devenir des abandons ou vice versa. En général, pour favoriser la synchronisation, le type de l'évé-

nement correspondant à chaque valeur $U \in [0,1)$ doit dépendre le moins possible des paramètres du modèle.

Dans le cas où il n'y a qu'un seul type de contact et un seul groupe d'agents, le petit nombre de types d'événements rend acceptable l'utilisation de la recherche linéaire. Par contre, avec plusieurs types de contacts et groupes d'agents, nous utilisons la recherche indexée pour accélérer la génération des transitions. Examinons plus en détails ces deux choix d'implantations.

5.2.2.1 Simulation avec un seul type de contact et groupe d'agents

Pour $K = I = 1$, la recherche linéaire est la méthode la plus efficace pour générer les événements puisqu'il n'y a que peu de types d'événements. Plus précisément, le sous-intervalle $[0, \tilde{\lambda}/\tilde{q})$ est réservé aux arrivées, le sous-intervalle $[\tilde{\lambda}/\tilde{q}, \tilde{\lambda}/\tilde{q} + \tilde{N}_1\tilde{\mu}_1/\tilde{q})$ est dédié aux fins de service et l'intervalle $[1 - \tilde{H}\tilde{\nu}/\tilde{q}, 1)$ est associé aux abandons. Par contre, même si l'uniforme U tombe dans l'intervalle associé à un type d'événement, en fonction de la valeur de U, de l'état du système et de la valeur de ses paramètres, une transition fictive peut survenir au lieu d'un événement. Le tableau 5.II donne les sous-intervalles $[U_1, U_2)$ de $[0,1)$ associés à chacun des trois types d'événements ; tout autre sous-intervalle produit une transition fictive. Chaque ligne du tableau donne un sous-intervalle et la description d'un événement.

Ainsi, pour générer une transition dans le cas où $K = I = 1$, il suffit de générer une variable aléatoire uniforme U sur l'intervalle $[0,1)$ et d'effectuer un test pour chacun des quatre intervalles donné dans le tableau. Pour les fins de service et les abandons,

Tableau 5.II – Sous-intervalles de $[0,1)$ associés à des événements dans le cas où $K = I = 1$

U_1	U_2	Événement
0	$\rho\lambda/\tilde{q}$	Arrivée avec abandon immédiat possible
$\rho\lambda/\tilde{q}$	$\lambda/\tilde{q}$	Arrivée sans abandon immédiat
$(\tilde{\lambda} + (s(U)-1)\tilde{\mu}_1)/\tilde{q}$	$U_1 + \mu_1/\tilde{q}$	Fin de service, pour $s(U) \in \{1,\ldots,S_{1,1}\}$
$(\tilde{q} - \tilde{H}\tilde{\nu} + (\ell(U)-1)\tilde{\nu})/\tilde{q}$	$U_1 + \nu/\tilde{q}$	Abandon, avec $\ell(U) \in \{1,\ldots,Q_1\}$

nous devons déterminer les fonctions $s(U)$ et $\ell(U)$. Pour une fin de service, l'intervalle $[\tilde{\lambda}/\tilde{q}, (\tilde{\lambda} + \tilde{N}_1\tilde{\mu}_1)/\tilde{q})$ peut être subdivisé en $\tilde{N}_1$ sous-intervalles, un pour chaque agent. Nous pouvons déterminer à quel agent

$$s(U) = \left\lfloor \frac{U\tilde{q} - \tilde{\lambda}}{\tilde{\mu}_1} \right\rfloor + 1$$

correspond l'uniforme U puis générer une fin de service si $s(U) \in \{1, \ldots, S_{1,1}\}$ et U est dans l'intervalle donné par l'avant-dernière ligne du tableau 5.II.

Pour un abandon, selon un argument semblable, il suffit de tester si $\ell(U) \in \{1, \ldots, Q_1\}$ pour

$$\ell(U) = \left\lfloor \frac{U\tilde{q} - \tilde{\lambda} - \tilde{N}_1\tilde{\mu}_1}{\tilde{\nu}} \right\rfloor + 1.$$

Cette dernière valeur donne la position, dans la file, du contact ayant abandonné.

La figure 5.3 présente un exemple de partition de l'intervalle $[0, 1)$ pour un modèle avec un taux d'arrivée $\lambda = 0,5342$, un taux de fin de service $\mu = 0,01$, un taux d'abandon $\nu = 0,001$, $N_1 = 52$ agents et une capacité de file d'attente de $H = 80$. Dans cet exemple, tous les agents sont occupés et la file d'attente contient 40 clients. La zone orange est consacrée aux arrivées avec abandon immédiat si aucun agent n'est disponible tandis que la zone jaune est dédiée aux autres arrivées. La zone verte, qui représente les fins de service, est subdivisée en 52 sous-intervalles, un pour chaque agent du modèle. La zone rouge, qui représente les abandons, n'est pas subdivisée sur le diagramme, car les sous-intervalles qui en résulteraient seraient trop petits avec le taux d'abandon utilisé. La zone blanche représente enfin les transitions fictives.

5.2.2.2 Simulation avec plusieurs types de contacts ou groupes d'agents

Pour les cas où $K > 1$ ou $I > 1$, nous partitionnons $[0, 1)$ de façon semblable au cas où $K = I = 1$, mais nous utilisons la recherche indexée pour générer le plus possible d'informations, puis recourons à des méthodes moins efficaces lorsque nécessaire. Pour

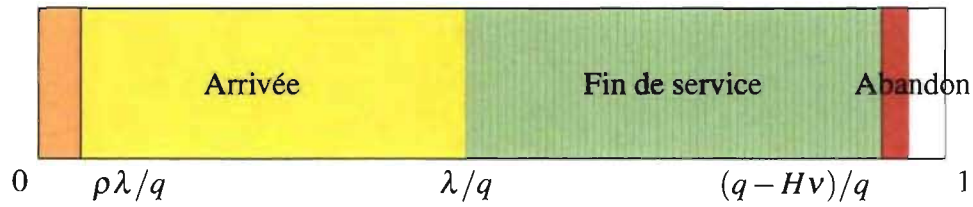

Figure 5.3 – Exemple de partition de l'intervalle $[0,1)$ dans un cas où $K = I = 1$

cela, nous divisons l'intervalle $[0,1)$ en 2^a sous-intervalles de taille identique, où a est un entier positif. À chaque sous-intervalle, nous associons ensuite une arrivée de type k, une fin de service par un agent du groupe i ou un abandon potentiels. Un sous-intervalle peut également être subdivisé à nouveau en 2^b parties égales, où b est aussi un entier positif, et un événement est associé à chaque partie.

La figure 5.4 présente un exemple d'un tel découpage, pour un modèle avec deux types de contacts et deux groupes d'agents et pour lequel tous les agents sont occupés et la file est pleine. La dernière ligne montre le découpage de l'intervalle $[0,1)$ en fonction du type d'événement. Ce découpage ressemble à celle pour l'exemple avec un type de contact et un groupe d'agents de la figure 5.3, mais le sous-intervalle $[0,\tilde{\lambda}/\tilde{q})$ dédié aux arrivées est séparé en deux : une partie $[w_1,w_2)$ pour les arrivées du premier type et l'autre, $[w_2,w_3)$, pour celles du deuxième type. De même, l'intervalle pour les fins de service est séparé de façon à avoir des parties pour chaque groupe d'agents, nommées respectivement $[v_1,v_2)$ et $[v_2,v_3)$. Nous reviendrons plus tard sur le sens général de w_k et v_i.

Les deux premières lignes de la figure montrent quant à elles comment l'index de recherche sépare cet intervalle. D'abord, l'intervalle est divisé en huit parties égales comme le montre la ligne centrale de la figure. En particulier, nous pouvons voir que le premier sous-intervalle est entièrement dédié aux arrivées de type 1 tandis que le troisième est consacré aux arrivées de type 2. Mais le second sous-intervalle, qui englobe des valeurs de U pour des arrivées des deux types, doit être subdivisé à nouveau comme

le montre la ligne supérieure de la figure.

Pour générer un événement, nous devons en premier lieu générer un indice uniformément parmi l'ensemble $\{1,\ldots,2^a\}$. Sachant que l'indice peut être encodé en utilisant a bits, il suffit de générer a bits aléatoires uniformes et de prendre l'indice correspondant à la représentation décimale de ces bits. Si le sous-intervalle généré doit être séparé encore, nous générons b bits aléatoires et prenons l'indice correspondant.

Plus les intervalles sont petits, plus l'événement correspondant à la transition pourra être généré souvent avec a bits et un seul accès à un tableau, mais plus le tableau précalculé prendra de la mémoire. Avec des intervalles trop grands, il faudra générer plusieurs indices et faire plusieurs accès à des tableaux pour aboutir à un événement représentant une transition, d'où un temps de génération plus élevé. En pratique, par contre, étant donné que les accès à des tableaux sont très rapides, le nombre d'accès à effectuer pour atteindre un sous-intervalle correspondant à un événement à exécuter ne fait pas une grosse différence.

Avec les générateurs que nous avons utilisés, il est plus facile d'obtenir des groupes de bits plutôt que des bits individuels. Pour cette raison, un groupe de $\tilde{a}$ bits aléatoires est toujours généré pour chaque transition même si seule une fraction de ces bits est nécessaire.

Nous avons fixé

$$a = \min(8, \min(x : q_* > 2^{-x})) \text{ où } q_* = \min_{k\in\{1,\ldots,K\},i\in\{1,\ldots,I\}} (\lambda_k/q, \mu_i/q, \nu_k/q)$$

est le plus petit taux de transition de la CMTC. Cette formule fait en sorte que nous recherchons la plus grande valeur de a engendrant des sous-intervalles plus petits que q_*, jusqu'à un maximum de $a = 8$. Bien que seuls huit bits soient utilisés au maximum pour sélectionner le premier découpage de $[0, 1)$, le nombre maximal de bits utilisés pour générer un événement est fixé à $\tilde{a} = 31$, la taille d'un entier usuel en Java.

Pour chaque transition, nous générons $\tilde{a}$ bits aléatoires et exécutons des actions

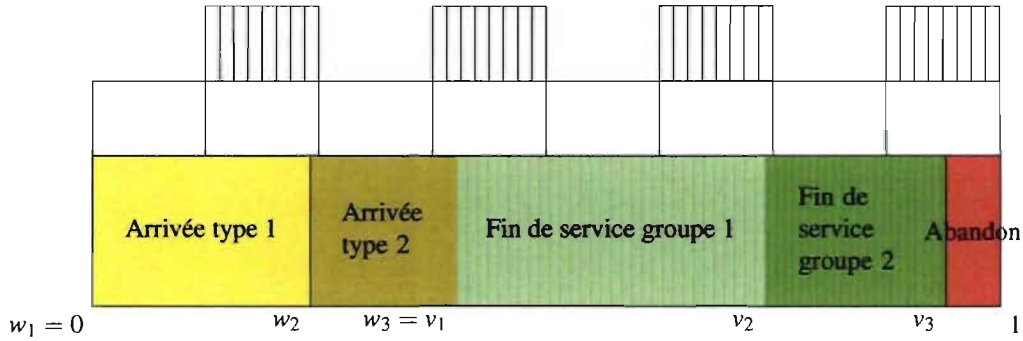

Figure 5.4 – Exemple d'index de recherche pour un modèle à deux types de contacts et deux groupes d'agents

dépendant du type de l'événement correspondant au sous-intervalle sélectionné. Pour prendre une décision, l'événement ainsi choisi peut utiliser diverses informations internes comme le sous-intervalle $[U_1, U_2)$ de $[0, 1)$ qui lui correspond, l'état de la CMTD imbriquée, ainsi que les bits aléatoires non utilisés pour choisir l'intervalle.

La recherche indexée permet de générer certaines informations en temps constant, par exemple le type d'une nouvelle arrivée et le groupe d'un agent terminant un service. La recherche linéaire est par contre nécessaire pour choisir le type d'un contact lors d'une fin de service ou d'un abandon ou pour décider de générer une transition fictive, car la valeur de k dans ces cas dépend de $S_{k,i}$ ou Q_k qui varient dans le temps. Des méthodes d'inversion plus sophistiquées comme la recherche binaire imposent un temps de calcul additionnel si bien qu'elles ne sont efficaces qu'avec un nombre très élevé de types de contacts. Des méthodes de rejet permettant de trouver k rapidement sont également disponibles [29, 32, 74], mais nous préférons utiliser l'inversion pour une meilleure synchronisation lorsque les variables aléatoires communes sont utilisées. Examinons plus en détails comment chacun des types d'événement est traité.

Dans le cas d'une arrivée, le type du contact k est lié au sous-intervalle $[U_1, U_2)$ généré de façon aléatoire. Ce sous-intervalle aléatoire est inclus dans l'intervalle déterministe associé aux arrivées potentielles de type k, à savoir $[w_k, w_{k+1})$ où $w_k = \sum_{j=1}^{k-1} \tilde{\lambda}_j / \tilde{q}$

pour $k = 1, \ldots, K+1$. En utilisant des bits aléatoires additionnels, nous pouvons générer U uniformément sur $[U_1, U_2)$. Si $U - w_k < \rho_k \lambda_k / \tilde{q}$, une arrivée survient avec abandon immédiat en cas de non-disponibilité des agents. Si $\rho_k \lambda_k / \tilde{q} \leq U - w_k < \lambda_k / \tilde{q}$, une arrivée sans aucune possibilité d'abandon immédiat se produit. Enfin, si $U \geq \lambda_k / \tilde{q}$, une transition fictive se produit. Ce dernier cas ne survient que si $\lambda_k < \tilde{\lambda}_k$, par exemple si le taux d'arrivée varie dans le temps.

Un exemple de partition de l'intervalle $[w_k, w_{k+1})$ en fonction de ces trois cas est présenté sur la figure 5.5. Les zones orange, jaune et blanche représentent respectivement le sous-intervalle pour l'arrivée avec possibilité d'abandon immédiat, l'arrivée sans abandon immédiat et la transition fictive.

Si le sous-intervalle $[U_1, U_2)$ correspond à une fin de service par un agent du groupe i, nous savons que ce sous-intervalle est inclus dans $[v_i, v_{i+1})$ où $v_i = \tilde{\lambda} / \tilde{q} + \sum_{j=1}^{i-1} \tilde{\mu}_j \tilde{N}_j / \tilde{q}$ pour $i = 0, \ldots, I+1$. Le sous-intervalle de $[v_i, v_{i+1})$ associé aux contacts de type k est quant à lui noté $[V_{k,i}, V_{k+1,i})$, avec $V_{k,i} = v_i + \sum_{j=1}^{k-1} S_{j,i} \tilde{\mu}_{j,i} / \tilde{q}$. Contrairement aux v_i qui ne dépendent que des paramètres du modèle et qui sont déterministes, les $V_{k,i}$ sont aléatoires puisqu'ils dépendent de l'état de la CMTC. En utilisant des bits aléatoires additionnels, nous pouvons générer une valeur U uniformément sur $[U_1, U_2)$. Nous recherchons ensuite le plus petit indice k tel que $U < V_{k+1,i}$ et générons une transition fictive si $U \geq V_{K+1,i}$.

Si nous utilisons la CMTC pour examiner l'effet d'un changement du taux de service ou si ce taux de service peut varier dans le temps, nous aurons $\mu_{k,i} < \tilde{\mu}_{k,i}$ pour certains k

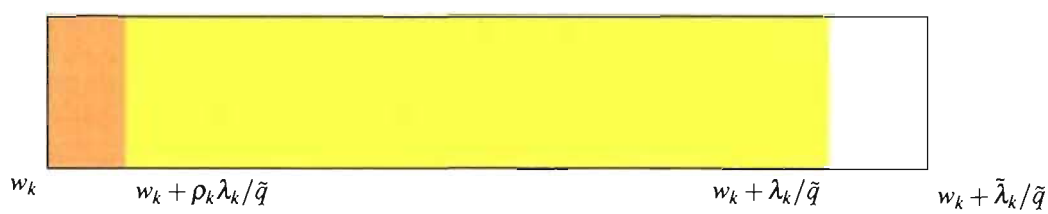

Figure 5.5 – Exemple de partition de l'intervalle $[w_k, w_{k+1})$ associé à une arrivée de type k

et certains i. Nous devons alors effectuer des tests additionnels pour déterminer si chaque transition est fictive ou pas. Pour cela, l'intervalle $[V_{k,i}, V_{k+1,i})$ peut être subdivisé en $S_{k,i}$ sous-intervalles de longueur $\tilde{\mu}_{k,i}$ représentant les agents occupés. De façon analogue au cas avec $K = I = 1$, une fin de service ne survient que si $0 \leq U - V_{k,i} - (s_{k,i}(U) - 1)\tilde{\mu}_{k,i}/\tilde{q} < \mu_{k,i}/\tilde{q}$ et $s_{k,i}(U) \in \{1, \ldots, S_{k,i}\}$, où $s_{k,i}(U) = \lfloor (U - V_{k,i})\tilde{q}/\tilde{\mu}_{k,i} \rfloor + 1$ est le numéro de l'agent sélectionné par l'uniforme U.

La figure 5.6 présente un exemple de partition de l'intervalle $[v_i, v_{i+1})$ pour un modèle à trois types de contacts. Chaque petit rectangle de la figure représente un intervalle associé à un des vingt-deux agents du modèle. Les zones vertes représentent des portions de l'intervalle correspondant à une fin de service tandis que les zones blanches dénotent des portions associées aux transitions fictives. Chaque type de contact est représenté par une teinte différente de vert. Le premier type de contact, dans l'intervalle $[V_{1,i}, V_{2,i})$, utilise sept des vingt-deux agents disponibles, le second type dans l'intervalle $[V_{2,i}, V_{3,i})$ mobilise sept agents tandis que le troisième type en occupe cinq. Les deux rectangles blancs à la fin de la ligne sur la figure représentent deux agents libres. Pour les deux premiers types de contacts, nous avons $\mu_{k,i} = \tilde{\mu}_{k,i}$, c'est-à-dire que le taux de service est au maximum permis par les paramètres du modèle. Alors, si U tombe dans un rectangle correspondant à un agent servant un des deux premiers types de contacts, une fin de service a lieu. Ceci est représenté sur la figure par le fait que les rectangles sont complètement peints en vert. Par contre, pour le troisième type, nous avons $\mu_{k,i} < \tilde{\mu}_{k,i}$ si bien qu'il ne suffit pas que U tombe dans le rectangle correspondant à l'agent donné pour engendrer une fin de service ; l'uniforme doit tomber dans la bonne partie du rectangle. Sur la figure, les derniers rectangles sont alors partiellement colorés en vert.

Nous utilisons une technique semblable pour sélectionner le type de contact lors d'un abandon. Dans ce cas, l'intervalle lié aux abandons est $[1 - H\tilde{\nu}/\tilde{q}, 1)$ tandis que celui associé aux abandons de type k est $[Z_k, Z_{k+1})$ avec $Z_k = 1 - H\tilde{\nu}/\tilde{q} + \sum_{j=1}^{k-1} Q_j \tilde{\nu}_j/\tilde{q}$. Nous cherchons alors le plus petit indice k tel que $U < Z_{k+1}$. Une transition fictive survient si $U \geq Z_{K+1}$. De plus, si $\nu_k < \tilde{\nu}_k$, nous devons calculer $\ell_k(U) = \lfloor (U - Z_k)\tilde{q}/\tilde{\nu}_k \rfloor + 1$ et

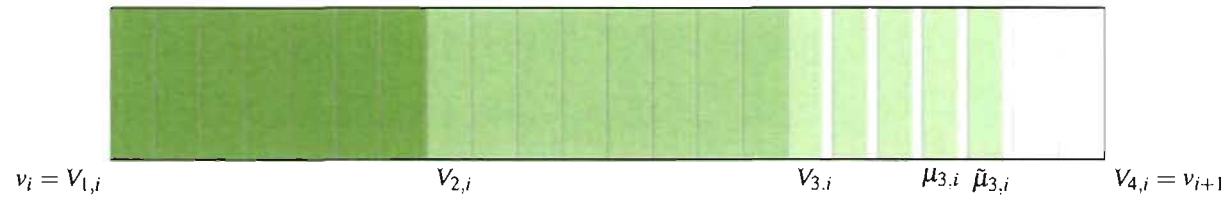

Figure 5.6 – Exemple de partition de l'intervalle $[v_i, v_{i+1})$ associé à une fin de service d'un agent du groupe i

générer un abandon seulement si $\ell_k(U) \in \{1, \ldots, Q_k\}$ et $0 \leq U - Z_k - (\ell_k(U) - 1)\tilde{\nu}_k/\tilde{q} < \nu_k/\tilde{q}$. La valeur de $\ell_k(U)$ donne le numéro du contact de type k en attente qui est susceptible d'abandonner.

Dans certains cas, il est possible de simplifier la génération des arrivées, des fins de service et des abandons. Pour les arrivées, si ρ_k et λ_k ne varient pas dans le temps ou d'une configuration à l'autre, le type de contact k ainsi qu'un indicateur d'abandon immédiat en cas de non-disponibilité d'agents peuvent être liés à l'indice généré de façon aléatoire. Les valeurs de U_1 et U_2 ne sont alors pas nécessaires.

Dans le cas d'une fin de service potentielle par un agent du groupe i, si le temps de service ne dépend pas du type de contact et les valeurs $S_{k,i}$ ne sont pas nécessaires pour la simulation ou le calcul statistique, la recherche linéaire peut être évitée puisque nous n'avons pas besoin de la valeur de k. Le choix entre une fin de service ou une transition fictive se fait alors de façon semblable au cas où $K = I = 1$. De façon semblable, si $K = 1$, il n'est pas nécessaire de générer un type de contact lors d'un abandon. Le test devient alors semblable à celui dans le cas où $K = I = 1$.

5.2.3 Réduction des transitions fictives

Nous avons remarqué que dans certains cas, le taux de transition fictive est élevé, ce qui dégrade la performance du simulateur étant donné que plusieurs nombres aléatoires qui prennent du temps à générer sont en quelque sorte gaspillés. Les transitions fictives se produisent surtout lorsqu'un taux de transition $\tilde{q} > q$ est employé pour simuler avec

plusieurs valeurs de paramètres. Même si $\tilde{q} = q$, ces transitions surviennent souvent si une capacité de file élevée a été utilisée pour réduire la probabilité de contacts bloqués. De plus, si le nombre de groupes d'agents est modéré, il peut arriver que certains groupes soient peu utilisés, ce qui occasionne aussi des transitions fictives.

Dans [31], Fox et Glynn proposent de considérer le nombre de transitions fictives successives précédant une véritable transition depuis l'état $i \in \mathcal{S}$ comme une variable aléatoire géométrique avec paramètre $q_i/\tilde{q}$. Ce nombre généré, il nous faut générer une transition conditionnellement au fait qu'elle n'est pas fictive, ce qui exige un index de recherche pour chaque état de la chaîne.

Nous avons opté pour le compromis suivant employant un petit nombre d'index de recherche. Nous pouvons partitionner $\mathcal{S}$ en R sous-ensembles et déterminer le taux de transition maximal $q(r)$ pour chaque sous-ensemble $r = 1, \ldots, R$. Si l'état du système est dans le sous-ensemble r, nous générons un certain nombre de transitions fictives successives selon une loi géométrique avec paramètre $q(r)/\tilde{q}$. Ensuite, l'état suivant est obtenu en utilisant une chaîne de Markov en temps discret avec taux de transition maximal $q(r)$ et son propre index de recherche. Cette étape peut bien entendu générer une nouvelle transition fictive.

Évidemment, pour réduire le temps de génération des transitions fictives, il faut que pour chaque état i d'un même sous-ensemble r, q_i soit semblable à $q(r)$. Le nombre R de sous-ensembles doit être petit afin d'éviter d'encombrer la mémoire avec des index de recherche. Il est aussi souhaitable que beaucoup de transitions soient générées dans un mode d'opération r tel que $q(r)$ est petit puisque cela permet de générer beaucoup de transitions fictives successives. La sélection ou la mise à jour de r lors de chaque transition doit également être rapide afin d'éviter que cette opération n'affecte la performance du simulateur au point d'annuler les bénéfices d'un taux de transition adaptatif.

Si nous utilisons la recherche linéaire pour sélectionner les événements dans le cas où $K = I = 1$, nous devons générer le nombre de transitions fictives successives précédant chaque transition depuis la loi géométrique en utilisant l'inversion. Par contre, la

fonction de répartition inverse d'une variable aléatoire géométrique de paramètre p est $F^{-1}(u) = \ln(1-u)/\ln(1-p)$. Le calcul de ces logarithmes accroît malheureusement le temps de calcul. Par contre, si nous utilisons la recherche indexée même si $K = I = 1$, nous obtenons un temps d'exécution légèrement plus élevé qu'avec la recherche linéaire, mais partitionner l'état permet de gagner un peu de temps. La recherche indexée est utilisée pour générer à la fois le nombre de transitions fictives et l'événement de la transition principale.

Il existe un grand nombre de façons de partitionner $\mathcal{S}$ et il ne semble pas facile en général de trouver une partition réduisant le temps de génération des transitions fictives. De plus, si nous comparons plusieurs configurations avec les variables aléatoires communes, utiliser des découpages différents pour chaque configuration risque d'affecter la synchronisation. Il vaut donc mieux trouver une partition convenant à toutes les configurations à tester.

Une façon de partitionner $\mathcal{S}$ est d'établir des seuils sur la taille de la file d'attente et le nombre d'agents. Examinons maintenant comment organiser les seuils pour qu'il soit facile de tenir à jour le sous-ensemble r quel que soit l'état.

5.2.3.1 Seuils sur la taille de la file d'attente

Une première heuristique consiste à partitionner en fonction de la taille de la file d'attente uniquement. Soit pour cela $0 \leq \eta_1 < \cdots < \eta_R = \tilde{H}$ une suite croissante de seuils sur la taille de la file. À chaque valeur de $r = 1, \ldots, R$ correspond un mode d'opération avec sa propre valeur η_r, un taux de transition $q(r) = \tilde{q} - (\tilde{H} - \eta_r)\tilde{\nu} \leq \tilde{q}$ et un index de recherche. Le mode d'opération courant est déterminé par le plus petit entier r tel que $\sum_{k=1}^{K} Q_k \leq \eta_r$, c'est-à-dire que la taille totale de la file est inférieure ou égale au seuil η_r. Cette heuristique est intéressante si $\tilde{\nu}$ est grand ou encore si la file est souvent petite tandis qu'une capacité élevée est utilisée pour réduire la probabilité des contacts bloqués.

La façon la plus simple de fixer les seuils est de les répartir également entre 0 et $\tilde{H}$.

Nous avons alors $\eta_r = \lfloor \tilde{H}r/R \rfloor$. Si nous savons que la taille de la file est la plupart du temps petite, nous pouvons aussi placer davantage de seuils petits et ne mettre qu'un seuil élevé. Par exemple, si la capacité de la file était fixée à 1 000, nous pourrions avoir un seuil à 20, un à 100, puis un à 1 000.

Pour améliorer ce partitionnement, nous pouvons établir des seuils sur le nombre d'agents en plus de la taille de la file. Si $I = 1$ et N_1 ne varie pas dans le temps, nous pouvons fixer des seuils par rapport au nombre de clients dans le système au lieu du nombre dans la file. Par contre, si $I > 1$, nous devons partitionner $\mathcal{S}$ en fixant des seuils pour chaque groupe d'agents en plus de la file d'attente. Nous examinons cela dans la sous-section suivante.

5.2.3.2 Suite non ordonnée de vecteurs de seuils

Partitionner $\mathcal{S}$ en utilisant des seuils sur la taille de la file et plusieurs groupes d'agents est plus compliqué et souvent inutile, car dans les modèles typiques, le taux d'occupation des agents est assez élevé, par exemple 80% ou même 90%. Dans ce cas, peu de fins de service fictives se produisent. Si le nombre de groupes d'agents était modéré et si le taux d'occupation différait beaucoup entre ces groupes, le partitionnement de $\mathcal{S}$ avec des seuils sur le nombre d'agents dans les groupes avec taux d'occupation faible pourrait être envisageable. Un autre cas auquel nous avons pensé est la simulation d'un horizon à plusieurs périodes avec le même taux de transition $\tilde{q}$ pour chaque période. Dans les périodes où le nombre d'agents N_i est beaucoup plus petit que le maximum $\tilde{N}_i$, l'utilisation de seuils sur le nombre d'agents serait envisageable.

L'idée pour effectuer le partitionnement consiste à établir un certain nombre R de vecteurs de seuils dans $\mathbb{N}^{I+1}$. Les I premières composantes d'un tel vecteur donnent des seuils sur le nombre d'agents occupés dans chacun des I groupes tandis que la dernière composante donne un seuil sur la taille de la file. Pour chaque vecteur, nous calculons un taux de transition et construisons un index de recherche.

Tout état $i \in \mathcal{S}$ peut être projeté dans la boîte rectangulaire regroupant les R vecteurs

de seuils en calculant le nombre total d'agents occupés dans chaque groupe à partir des valeurs de $S_{k,i}$ et la taille totale de la file à partir des Q_k. Il est ainsi important, pour que tous les états soient inclus dans le partitionnement, que le vecteur $(\tilde{N}_1, \ldots, \tilde{N}_I, \tilde{H})$ fasse partie des R vecteurs de seuils.

La boîte est partitionnée en un certain nombre de sous-boîtes rectangulaires contenant chacune au plus un vecteur de seuils placé dans le coin supérieur. Les sous-boîtes doivent être construites de façon à ce qu'il soit facile de maintenir à jour la sous-boîte courante. Pour chacune de ces sous-boîtes, nous pouvons déterminer le vecteur de seuils le plus près dont toutes les composantes sont supérieures ou égales aux points de la sous-boîte. Ce vecteur correspond au mode courant d'opération r.

La figure 5.7 illustre cette idée pour un cas où $I = 1$. L'espace d'états est projeté sur un rectangle de dimensions $\tilde{N}_1$ par $\tilde{H}$ séparé en un certain nombre de sous-rectangles. Chaque vecteur de seuils est représenté par un point d'une couleur différente et la couleur de chaque sous-rectangle correspond à celle d'un des vecteurs.

5.2.4 Problèmes posés par la variation des paramètres dans le temps

Jusqu'à présent, nous avons supposé que les paramètres du centre de contacts sont fixes pour tout l'horizon. Nous levons maintenant cette contrainte en divisant l'horizon en périodes pendant lesquelles tous les paramètres sont fixes. Comme discuté dans la section 5.1.8, nous construisons une CMTC pour chaque période et simulons ces chaînes bout à bout. Toutes ces chaînes sont simulées sur un horizon déterministe, à l'exception de celle représentant la période de fermeture qui est simulée sur horizon aléatoire, jusqu'à atteindre l'état correspondant au système vide. Mais la possibilité que les paramètres varient dans le temps a un impact à la fois sur la structure du modèle ainsi que sur la façon de calculer certaines mesures de performance. Souvent, les modifications imposées rendent l'état de la CMTC ou les calculs beaucoup trop complexes si bien que nous recourons à des approximations.

Nous avons construit la CMTC pour prendre en compte la possibilité que le nombre

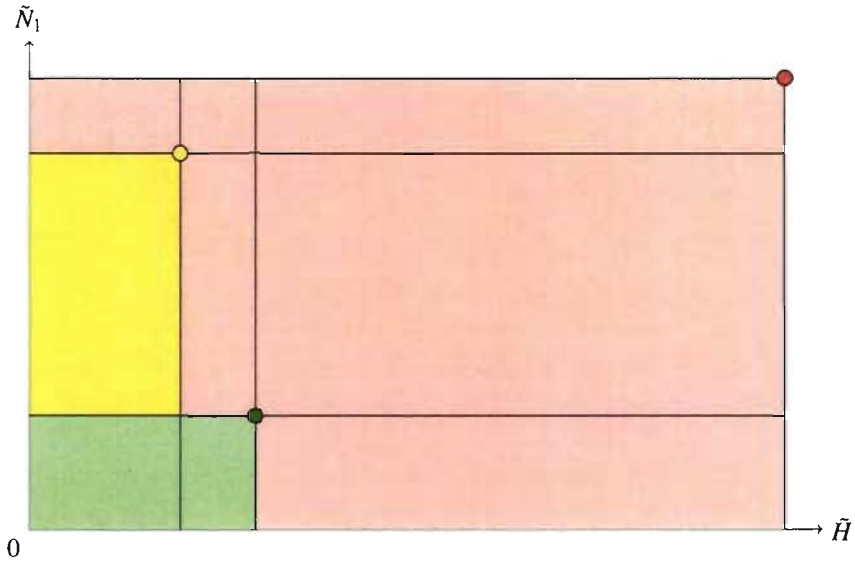

Figure 5.7 – Exemple de partitionnement de $\mathcal{S}$ en deux dimensions, avec trois vecteurs de seuils

d'agents occupés soit supérieur au nombre d'agents présents dans les groupes. Soit $N_{i,p}$ le nombre d'agents du groupe i pendant la période p. Si, pour une période p, $N_{i,p-1} > N_{i,p}$, il se peut qu'au début de la période p, $\sum_{k=1}^{K} S_{k,i} > N_{i,p}$. Lorsque cela se produit, le groupe d'agents n'accepte aucun nouveau contact jusqu'à ce que $\sum_{k=1}^{K} S_{k,i} < N_{i,p}$. Cette possibilité doit être prise en compte lors du calcul des taux de transition si ce taux diffère d'une période à l'autre, car le taux de fin de service maximal pour la période p doit prendre en compte $\max_{j=1,\ldots,p} N_{i,j} \geq N_{i,p}$ agents dans le groupe i au lieu de $N_{i,p}$. Outre ce changement du taux de transition, il faut tester le nombre d'agents occupés après chaque fin de service pour ne retirer un contact de la file que si ce nombre est inférieur au nombre d'agents dans la période courante.

Modéliser le fait que le temps de patience moyen peut dépendre de la période d'arrivée est par contre plus difficile. Cela impose de remplacer Q_k par $\sum_{p=1}^{P} Q_{k,p}$ où $Q_{k,p}$ est le nombre de contacts arrivés pendant la période p encore dans la file d'attente. Le taux d'abandon devient alors $\sum_{k=1}^{K} \sum_{p=1}^{P} Q_{k,p} \nu_{k,p}$ où $\nu_{k,p}$ est le taux d'abandon d'un contact de type k arrivé pendant la période p. Si le temps de service moyen peut lui

aussi dépendre de la période d'arrivée, nous devons remplacer $S_{k,i}$ par $\sum_{p=1}^{P} S_{k,i,p}$ où $S_{k,i,p}$ est le nombre de contacts de type k encore en service par des agents du groupe i, arrivés pendant la période p. Le taux de fin de service en fonction de l'état devient alors $\sum_{k=1}^{K}\sum_{i=1}^{I}\sum_{p=1}^{P} S_{k,i,p}\mu_{k,i,p}$ où $\mu_{k,i,p}$ est le taux de fin de service pour les contacts de type k servis par des agents du groupe i et arrivés pendant la période p. En plus de la dimension plus élevée de l'état, cette extension du modèle impose des recherches linéaires dans $\mathcal{O}(KP)$ plutôt que $\mathcal{O}(K)$ lors de la sélection du type de contact ayant terminé un service ou ayant abandonné. Dans ces deux cas, il faut en effet choisir une période d'arrivée en plus d'un type de contact.

Plutôt que complexifier l'état de cette façon, nous avons choisi de supposer qu'au début d'une période p, le temps de service restant de tous les contacts de type k en cours de service par des agents du groupe i est multiplié par $\mu_{k,p-1}/\mu_{k,p}$. De façon semblable, au début de la période p, tous les temps de patience restants des contacts de type k en attente sont multipliés par $\nu_{k,p-1}/\nu_{k,p}$. Ainsi, il n'est pas nécessaire de compter les contacts en service et en attente séparément pour chaque période et nous pouvons utiliser le modèle que nous avons développé précédemment pour le cas à une seule période. Appelons cette hypothèse H.1.

Si $\mu_{k,p} < \mu_{k,p-1}$ pour une période p, H.1 allonge le temps de service pour certains contacts, d'où une charge de travail accrue pour les agents pendant la période p. Cela peut occasionner des abandons et baisser le niveau de service par rapport au modèle original sans l'hypothèse. Le niveau de service peut aussi augmenter en raison des abandons qui réduisent la charge de travail des agents. L'effet inverse se produit dans le cas où $\mu_{k,p} > \mu_{k,p-1}$. Par contre, cette approximation n'aura un impact important que si la moyenne du temps de service ou de patience varie beaucoup entre les périodes et si les périodes sont courtes.

Par exemple, prenons un centre de contacts avec $K = I = 1$ et deux périodes principales d'une demi-heure. En moyenne, il y a 100 arrivées par période. Le temps de patience moyen est 1000s tandis que le temps de service moyen est 600s pour la pre-

mière période et 900s pour la seconde. Le nombre d'agents est 31 dans la première période et 50 dans la seconde. Avec ces paramètres, plusieurs services débutent dans la première période mais se terminent dans la seconde. La partie supérieure du tableau 5.III présente le taux d'occupation et le taux d'abandon pour cet exemple, estimés par 10 000 réplications indépendantes à l'aide du simulateur simplifié utilisant l'hypothèse H.1 et du simulateur par événements discrets (SED), pour les deux périodes du modèle et pour tout l'horizon.

Ces résultats montrent clairement que, pour cet exemple, H.1 ajoute du biais dans les estimateurs calculés. L'allongement des temps de service augmente suffisamment le taux d'abandon pour occasionner ici une baisse du niveau de service. Si le temps de service moyen pendant la seconde période est 700s au lieu de 900s, comme le montre la partie inférieure du tableau, l'écart entre les deux estimateurs est moins grand, mais le biais n'en demeure pas moins non négligeable.

En pratique, nous pouvons utiliser H.1 puisque les temps de service et de patience moyens diffèrent très peu entre les périodes. De plus, il faut garder à l'esprit que ce modèle de CMTC est principalement utilisé pour les premières phases de l'optimisation. Dans de tels cas, le biais est souvent acceptable puisque la simulation par événements discrets est utilisée en dernier lieu pour raffiner la solution.

Tableau 5.III – Impact de H.1 sur le taux d'occupation et le taux d'abandon, pour un modèle simple

		Taux d'occupation		Taux d'abandon	
		H.1	SED	H.1	SED
$1/\mu_{1,2} = 900$	Période 1	70,9%	70,8%	1,79	1,76
	Période 2	83,3%	76,7%	1,41	0,49
	Global	78,6%	74,4%	3,20	2,25
$1/\mu_{1,2} = 700$	Période 1	70,8%	70,8%	1,76	1,77
	Période 2	83,8%	80,9%	1,16	0,72
	Global	78,4%	76,7%	2,92	2,49

5.3 Calcul des statistiques avec le modèle de chaîne de Markov

5.3.1 Estimation du niveau de service

Si nous avons une seule période de longue durée, nous pouvons voir le niveau de service comme $F_W(s)$ où s est le temps d'attente acceptable et $F_W(\cdot)$ est la fonction de répartition du temps d'attente à long terme. Cette fonction $F_W(s)$ peut être calculée en conditionnant sur le type de contact et l'état de la chaîne de la façon suivante.

$$F_W(s) = \sum_{k=1}^{K} \frac{\lambda_k}{\lambda} \left(\sum_{x \in \mathcal{S}} \mathbb{P}[W_k(x) \leq s \mid X(0) = x] \pi_x \right) \tag{5.9}$$

où $\pi_x = \lim_{t \to \infty} \mathbb{P}[X(t) = x]$ est la probabilité stationnaire pour $x \in \mathcal{S}$ et $W_k(x)$ est le temps d'attente à long terme d'un client de type k arrivé au moment où l'état est x. Avec deux types de contacts dont un seul peut attendre en file et un ou deux groupes d'agents comme dans l'article [26], nous pouvons calculer les probabilités limites π_x de façon analytique. Pour la probabilité conditionnelle, il existe une forme fermée si la loi du temps de service et de patience ne dépend pas de k et de i. Dans le cas contraire, comme proposé dans [42], nous pouvons faire une copie de la CMTC originale, interdire les arrivées de type k pour la copie, l'initialiser à l'état x et évaluer, de façon analytique ou par simulation, la probabilité que la file de type k soit vide au temps s.

Toutefois, dès que $K > 1$ et $I > 1$, le nombre de termes dans (5.9) devient vite très grand et les méthodes analytiques pour calculer les probabilités sont trop coûteuses. Des calculs ou des approximations doivent alors être faits de façon numérique. De plus, la méthode précédente ne convient que pour un horizon infini. Nous devons alors trouver une autre façon d'estimer le niveau de service.

Le niveau de service dépend du nombre moyen de contacts ayant attendu moins de s, noté $\mathbb{E}[S_G(s,0,T)]$ où

$$S_G(s,0,T) = \sum_{n=1}^{N(T)} \mathbb{I}[W_n \leq s] \mathbb{I}[\text{Début de service à la transition } n].$$

Ici, n est un numéro de transition tandis que W_n est le temps d'attente associé à la transition n. Si cette transition ne concerne pas un contact abandonnant ou débutant son service, nous fixons $W_n = \infty$. Cela correspond à un exemple de coût de type (5.7), avec $C_0 = 0$ et

$$C_n = \mathbb{I}[W_n \leq s]\mathbb{I}[\text{Début de service à la transition } n]$$

pour $n > 0$. Nous pouvons de façon semblable obtenir $L_{\mathrm{G}}(s,0,T)$, le nombre de contacts ayant abandonné après avoir attendu moins de s. Nous pouvons aussi trouver $S_{\mathrm{G},k}(s,0,T)$ ou $L_{\mathrm{G},k}(s,0,T)$ pour n'importe quel type de contact.

Si nous connaissons $\mathcal{G}_{N(T)}$, nous savons quelles transitions correspondent à des débuts de service, mais nous ne connaissons pas les temps d'attente W_n. L'espérance conditionnelle du coût pour chaque transition $n > 0$ est alors

$$\mathbb{E}[C_n \mid \mathcal{G}_{N(T)}] = \mathbb{I}[\text{Début de service à la transition } n]\mathbb{P}[W_n \leq s \mid \mathcal{G}_{N(T)}].$$

Pour générer W_n, nous devons générer tous les temps de séjour exponentiels, ce qui revient à simuler la CMTC sans la conversion en temps discret. Par contre, sans les temps de séjour, nous pouvons, pour chaque contact, générer le nombre δ_n de transitions passées dans la file d'attente. Pour cela, nous devons tenir à jour une file d'attente pour chaque type de contact. Le numéro de la transition courante est ajouté à la fin de la file k lorsqu'une arrivée de type k avec mise en attente se produit. Lors d'une fin de service, le premier élément de la file k' est retiré si une valeur k' a pu être sélectionnée par le routeur. Dans le cas d'un abandon de type k, un élément de la file k est choisi de façon aléatoire et uniforme pour être retiré. Nous supposons pour la suite que l'état des files d'attente est inclus dans l'état de la CMTC. Ainsi, la σ-algèbre $\mathcal{G}_{N(T)}$ contient toute l'information nécessaire pour calculer δ_n, pour $n = 1,\ldots,N(T)$.

L'implantation de la structure de données représentant la file d'attente est bien sûr déterminante pour l'efficacité du programme de simulation. Nous avons opté pour un tampon circulaire implanté sous la forme d'un tableau et d'une valeur numérique don-

nant l'indice du premier élément valide du tableau. Ainsi, le cas fréquent où le premier élément est supprimé est traité en temps constant. L'ajout d'un élément à la fin du tableau se fait également en temps constant, mais la suppression d'un élément quelconque s'effectue en temps linéaire. Nous pourrions obtenir des performances semblables avec une liste chaînée, mais la liste prendrait plus de mémoire et la suppression d'un élément quelconque se ferait toujours en temps linéaire.

Examinons maintenant comment calculer $\mathbb{P}[W_n \leq s \mid \mathcal{G}_{N(T)}]$. Lorsque T est aléatoire, le taux de transition est uniformisé à q et W_n suit la loi Erlang (ou gamma) avec paramètre de forme δ_n et paramètre d'échelle q (qui a une moyenne δ_n/q). Il suffit alors d'évaluer la fonction de répartition gamma à la valeur s pour obtenir la probabilité conditionnelle $\mathbb{P}[W_n \leq s \mid \delta_n]$. Pour sauver du temps, nous pouvons précalculer cette probabilité conditionnelle pour différentes valeurs de δ_n et conserver le résultat pour toute nouvelle valeur de δ_n. Étant donné que le résultat ne dépend pas du nombre de transitions, nous pouvons réutiliser les valeurs déjà calculées pour toutes les futures réplications de la simulation.

Par contre, avec T déterministe, le nombre de transitions est fixé à $N(T)$ et les instants des transitions, $T_1, \ldots, T_{N(T)}$, sont répartis uniformément sur $[0, T]$. Si $\delta_n \leq N(T)$, l'événement $W_n \leq s$ se produit si et seulement si plus de δ_n transitions se produisent pendant l'intervalle $[T_n - s, T_n]$ ou encore pendant $[0, s]$ puisque les temps sont uniformément distribués. Le temps W_n correspond alors à la statistique d'ordre δ_n d'un échantillon de $N(T)$ variables aléatoires uniformes. Voir [23] pour les résultats standards sur de telles statistiques. Alors, $\mathbb{P}[W_n \leq s \mid \delta_n, N(T)] = \mathbb{P}[B \geq \delta_n]$ où B est une variable binomiale de paramètres $n = N(T)$ et $p = s/T$. Dans ce cas, l'espérance dépend du nombre de transitions, ce qui nous oblige à la recalculer à chaque réplication. Ceci rend particulièrement coûteux l'estimateur $C_{\mathrm{SIM},2}$ abordé à la section 5.1.6 puisqu'il faut conserver toutes les valeurs de δ_n et recalculer la somme de probabilités conditionnelles pour chaque valeur de $N(T)$ testée.

Habituellement, s est petit par rapport à T si bien que s/T est petit. Si le nombre de transitions $N(T)$ est élevé, B suit approximativement la loi de Poisson avec taux

$N(T)s/T$. Si T est grand, $N(T)/T \approx q$. De plus, si $Y \sim \text{Poisson}(qs)$ et $X \sim \text{Erlang}(\delta_n, q)$, $\mathbb{P}[Y \geq \delta_n] = \mathbb{P}[X \leq s]$. Ainsi, si T est grand et s est petit, nous avons $\mathbb{P}[W_n \leq s \mid \delta_n, N(T)] \approx \mathbb{P}[W_n \leq s \mid \delta_n]$.

Nous avons comparé les deux probabilités conditionnelles sur un exemple où $q = 0,115$, $s = 20$ et $T \in \{46\,800, 1800\}$. Pour chaque valeur de T, nous avons calculé les probabilités conditionnelles pour un nombre de transitions $N(T,j) = \lfloor qT + j\sqrt{qT} \rfloor$ où $j \in \{-2,0,2\}$, avec $\delta_n \in \{1,3,5\}$. Le tableau 5.IV présente les probabilités conditionnelles obtenues avec la méthode Erlang adaptée à l'horizon aléatoire pour lequel le temps d'arrêt est celui de l'atteinte d'un sous-ensemble d'états, et la méthode binomiale pour l'horizon déterministe. La première probabilité conditionnelle est considérée ici comme une approximation de la seconde. Chaque groupe de deux colonnes fournit les résultats pour une valeur de δ_n tandis que chaque ligne correspond à un scénario dans lequel T et $N(T)$ sont fixés. Comme nous pouvons le voir, l'approximation proposée est bonne pour un horizon long mais se dégrade si T diminue. Les résultats montrés ici ne concernent que de petites valeurs de δ_n. Si δ_n est grand, la probabilité conditionnelle est près de 0 et l'approximation est bonne peu importe T.

Tableau 5.IV – Comparaison de $\mathbb{P}[W_n \leq s \mid \delta_n]$ calculée avec la méthode Erlang et de $\mathbb{P}[W_n \leq s \mid \delta_n, N(T)]$ calculée avec la méthode binomiale

		$\delta_n = 1$		$\delta_n = 3$		$\delta_n = 5$	
T	$N(T)$	Erlang	Binomial	Erlang	Binomial	Erlang	Binomial
46 800	5 235	0,900	0,893	0,404	0,387	0,084	0,077
46 800	5 382	0,900	0,900	0,404	0,404	0,084	0,084
46 800	5 528	0,900	0,906	0,404	0,420	0,084	0,091
1 800	178	0,900	0,863	0,404	0,317	0,084	0,050
1 800	207	0,900	0,901	0,404	0,404	0,084	0,083
1 800	235	0,900	0,928	0,404	0,485	0,084	0,123

5.3.2 Estimation du temps d'attente moyen

Nous pouvons certes estimer le temps d'attente moyen global en employant la loi de Little $q(0,T) = \lambda w(0,T)$. L'espérance $q(0,T) = \mathbb{E}[Q(0,T)]$ correspond à la taille moyenne de la file qui peut facilement être estimée comme nous le verrons dans la sous-section 5.3.4. Par contre, cette loi ne fonctionne pas si le taux d'arrivée varie dans le temps.

Pour estimer le temps d'attente moyen, $w(0,T) = \mathbb{E}[W(0,T)]/\mathbb{E}[A(0,T)]$ de façon générale, nous avons besoin d'estimer l'espérance $\mathbb{E}[W(0,T)]$. Ceci est possible avec un estimateur de type (5.7) avec $C_0 = 0$ et $C_n = \mathbb{I}[\text{Début de service ou abandon à la transition } n]W_n$ pour $n > 0$. Encore une fois, nous pouvons ajuster C_n pour estimer l'espérance de la somme des temps d'attente pour les contacts servis ou ayant abandonné, ou encore celle de la somme des temps d'attente des contacts d'un certain type.

Examinons maintenant comment calculer $\mathbb{E}[W_n \mid \mathcal{G}_{N(T)}]$. Si T est aléatoire, $\mathbb{E}[W_n \mid \mathcal{G}_{N(T)}] = \mathbb{E}[W_n \mid \delta_n] = \delta_n/q$ puisque le temps d'attente suit une loi Erlang comme nous l'avons vu dans la sous-section précédente. Si T est déterministe, W_n/T correspond à la statistique d'ordre δ_n d'un échantillon de $N(T)$ uniformes sur $[0,1)$. La loi de probabilité de cette statistique est bêta avec paramètres δ_n et $N(T)+1-\delta_n$, d'où $\mathbb{E}[W_n/T \mid \mathcal{G}_{N(T)}] = \mathbb{E}[W_n/T \mid \delta_n, N(T)] = \delta_n/(N(T)+1)$. Ainsi, $\mathbb{E}[W_n \mid \delta_n, N(T)] = T\delta_n/(N(T)+1)$.

Puisque $N(T)$ suit la loi de Poisson, nous avons $N(T)/T \approx q$ si T est suffisamment grand. Alors, $T\delta_n/(N(T)+1) = \delta_n/(N(T)/T+1/T) \approx \delta_n/q$. Ainsi, si T est grand, nous pouvons approximer le temps d'attente par δ_n/q même si l'horizon est déterministe.

5.3.3 Estimation du temps d'excès moyen

Nous pouvons estimer le temps d'excès moyen $\mathbb{E}[\sum_{j=1}^{A(t_1,t_2)} \max(W_j - s, 0)]/\mathbb{E}[A(0,T)]$ à l'aide d'une expression de type (5.7) semblable à celle de la section précédente, à l'exception que W_n est remplacé par $\max(W_n - s, 0) = (W_n - s)^+$. Pour utiliser la conversion en temps discret, il nous faut calculer $\mathbb{E}[(W_n - s)^+ \mid \mathcal{G}_{N(T)}] = \mathbb{P}[W_n > s \mid \mathcal{G}_{N(T)}](\mathbb{E}[W_n \mid$

$\mathcal{G}_{N(T)}, W_n > s] - s)$. Nous pouvons facilement calculer $\mathbb{P}[W_n > s \mid \mathcal{G}_{N(T)}]$ à partir de $\mathbb{P}[W_n \leq s \mid \mathcal{G}_{N(T)}]$ pour lequel le cas a déjà été traité dans la sous-section 5.3.1. Par contre, $\mathbb{E}[W_n \mid \mathcal{G}_{N(T)}, W_n > s]$ correspond, selon le type d'horizon, à l'espérance d'une variable aléatoire Erlang ou bêta tronquée sur $[s, \infty)$ ou $[s, T)$ pour laquelle aucune forme fermée ne semble exister. Nous estimons cette espérance en effectuant une intégration numérique seulement sur l'intervalle $[0, s)$ plutôt que l'intervalle $[s, T]$ qui est beaucoup plus grand. Soit $f_\gamma(\delta_n, q, x)$ la fonction de densité gamma avec paramètres δ_n et q, évaluée à x, et soit $F_\gamma(\delta_n, q, x)$ la fonction de répartition correspondante. Nous pouvons écrire l'expression du temps d'attente conditionnel comme

$$\begin{aligned}
\mathbb{E}[W_n \mid W_n > s, \delta_n] &= \int_s^\infty \frac{x f_\gamma(\delta_n, q, x)}{1 - F_\gamma(\delta_n, q, s)} dx \\
(1 - F_\gamma(\delta_n, q, s))\mathbb{E}[W_n \mid W_n > s, \delta_n] &= \int_s^\infty x f_\gamma(\delta_n, q, x) dx \\
&= \int_0^\infty x f_\gamma(\delta_n, q, x) dx - \int_0^s x f_\gamma(\delta_n, q, x) dx \\
&= \mathbb{E}[W_n \mid \delta_n] - \int_0^s x f_\gamma(\delta_n, q, x) dx.
\end{aligned}$$

D'où

$$\mathbb{E}[W_n \mid W_n > s, \delta_n] = \frac{\delta_n / q - \int_0^s x f_\gamma(\delta_n, q, x) dx}{1 - F_\gamma(\delta_n, q, s)}. \tag{5.10}$$

Dans le cas d'un horizon déterministe, nous pouvons appliquer un raisonnement similaire pour obtenir $\mathbb{E}[W_n/T \mid \delta_n, N(T), W_n/T > s/T]$ et multiplier cette espérance par T. Cela nous donne

$$\mathbb{E}[W_n \mid W_n > s, \delta_n, N(T)] = T \frac{\delta_n / (N(T) + 1) - \int_0^{s/T} x f_\beta(\delta_n, N(T) + 1 - \delta_n, x) dx}{1 - F_\beta(\delta_n, N(T) + 1 - \delta_n, s/T)} \tag{5.11}$$

où $f_\beta(\delta_n, N(T) + 1 - \delta_n, x)$ et $F_\beta(\delta_n, N(T) + 1 - \delta_n, x)$ correspondent respectivement à la fonction de densité et à la fonction de répartition de la loi bêta avec paramètres δ_n et $N(T) + 1 - \delta_n$.

Si s est petit et δ_n est grand, nous pouvons approximer l'intégrale dans les équations

(5.10) et (5.11) par 0 puisque la probabilité que le temps d'attente soit supérieur à s est dans ce cas près de 1. Appelons cette approximation A1. Dans ces mêmes conditions et pour les mêmes raisons, nous pouvons aussi approximer l'espérance conditionnelle par $\mathbb{E}[W_n \mid \mathcal{G}_{N(T)}]$. Appelons cette seconde approximation A2.

Le tableau 5.V donne l'espérance conditionnelle du temps d'attente pour différentes valeurs des paramètres. Les paramètres sont les mêmes que pour l'exemple du tableau 5.IV de la sous-section précédente. La première ligne du tableau donne les résultats avec la méthode Erlang convenant pour un horizon aléatoire avec un temps d'arrêt correspondant au temps d'atteinte d'un sous-ensemble d'états, et pour laquelle $N(T)$ n'a aucun impact. Les autres lignes donnent les résultats avec bêta pour l'horizon déterministe. Chaque triplet de colonnes du tableau donne respectivement la valeur de $\mathbb{E}[W_n \mid W_n > s, \mathcal{G}_{N(T)}]$ calculée avec l'intégration numérique (I), ainsi que les approximations A1 et A2. Chaque ligne correspond à un scénario avec une valeur de T et de $N(T)$. L'intégration numérique a été effectuée avec la méthode de Simpson en subdivisant le domaine d'intégration en 10 000 sous-intervalles.

Dans le tableau, nous pouvons voir que, quelle que soit la méthode d'estimation, A1 et A2 approximent bien $\mathbb{E}[W_n \mid \mathcal{G}_{N(T)}, W_n > s]$ quand δ_n est grand. L'approximation A1 fait un peu mieux que A2 pour un δ_n modéré. Bien entendu, il est normal que les deux approximations soient très mauvaises avec un δ_n petit tel que $\mathbb{E}[W_n \mid \mathcal{G}_{N(T)}] < s$.

De façon semblable au nombre de contacts ayant attendu moins que s et au temps d'attente, l'approximation Erlang sur un horizon déterministe est mauvaise si T ou δ_n sont petits.

En résumé, si δ_n est grand, nous recommandons l'utilisation de A1 qui n'est pas beaucoup plus difficile à calculer que A2 tout en étant plus précise. Par contre, si δ_n est petit, nous recommandons l'intégration numérique avec un nombre modéré d'intervalles, par exemple 1 000. Pour décider si δ_n est grand, nous recommandons d'utiliser $\mathbb{P}[W_n \leq s \mid \mathcal{G}_{N(T)}]$. Si cette probabilité conditionnelle est près de 0, nous pouvons considérer δ_n comme grand et utiliser les approximations.

Tableau 5.V – Comparaison des différentes méthodes pour estimer $\mathbb{E}[W_n \mid W_n > s, \mathcal{G}_{N(T)}]$

		$\delta_n = 1$			$\delta_n = 5$			$\delta_n = 10$		
T	$N(T)$	I	A1	A2	I	A1	A2	I	A1	A2
Erlang		28,7	86,7	8,7	46,0	47,5	43,5	87,0	87,0	87,0
46 800	5 235	28,9	83,8	8,9	47,1	48,4	44,7	89,4	89,4	89,4
46 800	5 382	28,7	86,8	8,7	46,0	47,4	43,5	87,0	87,0	86,9
46 800	5 528	28,5	89,9	8,5	45,0	46,6	42,3	84,7	84,7	84,6
1 800	178	29,9	73,5	10,1	52,1	52,9	50,3	100,6	100,6	100,6
1 800	207	28,6	87,4	8,7	45,8	47,2	43,3	86,5	86,5	86,5
1 800	235	27,5	105,4	7,6	41,3	43,5	38,1	76,3	76,3	76,3

5.3.4 Estimation d'autres mesures de performance

La taille moyenne de la file est estimée en utilisant une expression de type (5.5), avec une fonction $f(\cdot)$ donnant la taille de la file $\sum_{k=1}^{K} Q_k$. Nous pouvons aussi estimer la taille de la file k de façon semblable.

Nous pouvons aussi estimer le taux d'occupation des agents par $o(0,T) = \mathbb{E}[N_{\mathrm{B}}(0,T)] / \mathbb{E}[N_{\mathrm{B}}(0,T) + N_{\mathrm{F}}(0,T)]$. Les variables aléatoires $N_{\mathrm{B}}(0,T)$ et $N_{\mathrm{F}}(0,T)$ correspondent aussi à la forme (5.5) avec une fonction $f(\cdot)$ donnant le nombre d'agents occupés $N_{\mathrm{B}}(t) = \sum_{i=1}^{I} \sum_{k=1}^{K} S_{k,i}$ et libres $N_{\mathrm{F}}(t) = \max(N_{\mathrm{B}}(t), \sum_{i=1}^{I} N_i)$. Rappelons que $S_{k,i}$ fait partie de l'état de la CMTC qui varie en fonction du temps t. La variable $N_{\mathrm{B}}(0,T) + N_{\mathrm{F}}(0,T)$ n'est aléatoire que lorsque le nombre d'agents varie d'une période à l'autre, dans le cas d'une simulation sur des périodes multiples.

Plusieurs autres mesures de performance telles que le nombre moyen de contacts servis, d'abandons, de contacts qui doivent attendre en file, etc., peuvent être estimées simplement en comptant le nombre de transitions d'un certain type ; ce sont des estimateurs de type (5.6).

5.3.5 Mesures de performance pour les contacts arrivés et servis pendant des périodes différentes

Lorsque le simulateur prend en charge des périodes multiples, un problème survient lors de l'estimation de mesures de performance en raison des contacts qui arrivent pendant une période et sont servis ou abandonnent pendant la suivante. Habituellement, nous comptons tous les événements concernant un contact dans une même période, celle de son arrivée par exemple. Mais lorsqu'une fin de service se produit, le nombre de transitions passé dans le système par le contact concerné est inconnu. Pour connaître ce temps de séjour, il faudrait conserver le numéro de transition des débuts de service et associer, à chacun de ces numéros de transition, le nombre de transitions passées en attente. Cela augmenterait évidemment la consommation de mémoire par le programme et en diminuerait l'efficacité. En fin de compte, cela reviendrait à avoir un objet distinct pour chaque contact, comme dans le simulateur à événements discrets. Nous nous contentons donc de toujours compter les événements concernant un contact pour la période pendant laquelle il termine son service ou abandonne.

Nous avons rencontré des difficultés additionnelles avec les mesures dépendant du temps d'attente. Supposons par exemple qu'un contact a passé d_1 transitions pendant la période $p-1$ et d_2 transitions pendant la période p. Nous supposons que le taux de transition pour ces deux périodes est q_1 et q_2 et que le nombre de transitions pendant les périodes est n_1 et n_2. Soit W_1 et W_2 le temps (inconnu) passé par le contact pendant les périodes $p-1$ et p, respectivement. Pour estimer le temps d'attente, il suffit d'utiliser $\mathbb{E}[W_1+W_2 \mid d_1,d_2,n_1,n_2] = \mathbb{E}[W_1 \mid d_1,n_1]+\mathbb{E}[W_2 \mid d_2,n_2]$. Par contre, les choses se compliquent pour les autres mesures de performance dépendant du niveau de service.

Prenons en particulier $\mathbb{P}[W_1+W_2 \leq s \mid d_1,d_2,n_1,n_2]$ utilisée pour estimer le niveau de service. Conditionnellement au fait que le contact a attendu un temps $W_1 = w_1 \leq s$ pendant la période $p-1$, $\mathbb{P}[W_1+W_2 \leq s \mid d_1,d_2,n_1,n_2,W_1=w_1] = \mathbb{P}[W_2 \leq s-w_1 \mid d_2,n_2]$. Cette probabilité est calculée comme à la section 5.3.1 traitant le niveau de service. Nous

pouvons intégrer le temps d'attente w_1 pour obtenir

$$\begin{aligned} & \mathbb{P}[W_1 + W_2 \leq s \mid d_1, d_2, n_1, n_2] \\ = & \frac{1}{t_{p-1} - t_{p-2}} \int_0^s \mathbb{P}[W_2 \leq s - w_1 \mid d_2, n_2] f_\beta(d_1, n_1 + 1 - d_1, w_1/(t_{p-1} - t_{p-2})) dw_1. \end{aligned}$$

Nous utilisons la densité $f_\beta(a,b,w)$ étant donné que la première période considérée est toujours une période principale, simulée sur un horizon déterministe. La nature de la seconde période affecte quant à elle la probabilité pour W_2 : la méthode binomiale est utilisée si la période est principale (horizon déterministe) tandis que la méthode Erlang est mise en œuvre pour la période de fermeture (horizon aléatoire).

Nous pouvons certes évaluer cette probabilité par intégration numérique, mais cela imposera un temps de calcul important puisqu'il faut le faire pour chaque contact arrivé pendant une période et servi ou abandonnant pendant la suivante.

Nous pouvons alléger le coût de cette intégration numérique en considérant que le domaine d'intégration $[0,s]$ est très petit. Il n'est alors pas nécessaire d'utiliser un très grand nombre de sous-intervalles avec la méthode Simpson.

Pour éviter cette intégration numérique, nous pouvons approximer la probabilité en prenant le même taux de transition pour les deux périodes considérées. Soit

$$q^* = \begin{cases} q_1 & \text{si } d_1 > d_2, \\ q_2 & \text{sinon,} \end{cases}$$

le taux de transition dominant, c'est-à-dire celui utilisé pour générer la majorité des transitions pour le contact. Si q_1 ne diffère pas trop de q_2, nous pouvons utiliser ce taux pour évaluer la probabilité que le temps d'attente soit inférieur à s étant donné que le contact concerné a attendu $d_1 + d_2$ transitions et que $n_1 + n_2$ transitions ont été simulées pendant un intervalle de temps $[t_{p-2}, t_p]$. Le calcul suppose un horizon déterministe, à moins que p corresponde à la période de fermeture.

Par contre, si q_1 diffère beaucoup de q_2, il est plus judicieux de corriger la valeur

de d_1 ou d_2 pour compenser le changement du taux de transition correspondant à la période dans laquelle le contact a le moins attendu. Supposons pour le moment que $d_1 > d_2$. Calculons maintenant $d_2' = d_2 q_1/q_2$ et $n_2' = n_2 q_1/q_2$. Nous pouvons maintenant calculer la probabilité que le temps d'attente soit inférieur à s en conditionnant sur $d_1 + \lfloor d_2' \rfloor$ transitions en file d'attente et $n_1 + \lfloor n_2' \rfloor$ transitions au total, pendant l'intervalle de temps $[t_{p-2}, t_p]$. Nous pouvons ensuite recalculer la probabilité, cette fois avec $d_1 + \lceil d_2' \rceil$ transitions d'attente et $n_1 + \lceil n_2' \rceil$ transitions au total pour ensuite interpoler linéairement entre ces deux probabilités par rapport à d_2'. Une technique similaire s'applique pour corriger d_1 et n_1 dans le cas où $d_1 \leq d_2$.

Nous avons testé ces deux approximations pour un exemple avec $q_1 = 0,115$, $q_2 = xq_1$, des périodes de durée 1 800 et $s = 20$. Le tableau 5.VI présente les résultats, avec une ligne par combinaison de (n_1, n_2, d_1, d_2). Le premier groupe de colonnes présente les résultats avec l'intégration numérique par la méthode Simpson avec 10 000 sous-intervalles, la technique la plus précise dont nous disposons pour estimer la probabilité conditionnelle. Le second groupe donne le résultat avec la même méthode, mais avec seulement 20 sous-intervalles. La paire de colonnes A1 donne les résultats avec la première approximation se contentant de supposer que le taux de transition est uniforme pour les deux périodes, sans ajuster le nombre de transitions passées en file d'attente. La paire de colonnes A2 donne les résultats avec l'approximation employant l'interpolation linéaire. Les colonnes P et F donnent respectivement les résultats lorsque la seconde période est principale et celle de fermeture.

Ce tableau montre qu'aucune de ces trois approximations n'est meilleure que les autres dans tous les cas. L'approximation qui fonctionne le mieux varie en fonction des paramètres. Parfois même, l'intégration numérique avec 20 sous-intervalles fait moins bien que A1 ou A2.

En pratique, il vaut donc mieux appliquer l'intégration numérique avec davantage que 20 sous-intervalles, par exemple 100 ou 1 000. Évidemment, cela impose un coût de calcul important.

Tableau 5.VI – Comparaison des approximations pour $\mathbb{P}[W_1 + W_2 \leq s \mid d_1, d_2, n_1, n_2]$

					Int. 10 000		Int. 20		A1		A2	
x	n_1	n_2	d_1	d_2	P	F	P	F	P	F	P	F
1,10	178	197	1	1	0,652	0,617	0,622	0,588	0,719	0,617	0,693	0,614
1,10	178	197	3	1	0,161	0,148	0,161	0,148	0,201	0,158	0,219	0,157
1,10	178	197	5	3	0,002	0,001	0,002	0,001	0,003	0,001	0,004	0,002
1,10	178	227	1	3	0,216	0,215	0,201	0,200	0,249	0,190	0,235	0,197
1,10	178	227	3	3	0,026	0,026	0,026	0,026	0,044	0,027	0,035	0,025
1,10	178	227	5	5	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000
1,10	178	257	1	5	0,037	0,054	0,034	0,049	0,044	0,036	0,041	0,040
1,10	178	257	3	1	0,161	0,172	0,161	0,172	0,201	0,224	0,219	0,216
1,10	178	257	5	3	0,002	0,002	0,002	0,002	0,003	0,003	0,004	0,004
1,10	207	197	1	1	0,695	0,658	0,659	0,624	0,719	0,657	0,693	0,656
1,10	207	197	3	5	0,004	0,002	0,004	0,002	0,005	0,002	0,004	0,002
1,10	207	197	5	3	0,003	0,002	0,003	0,002	0,003	0,002	0,004	0,003
1,10	207	227	1	3	0,236	0,235	0,218	0,217	0,249	0,223	0,235	0,233
1,10	207	227	3	5	0,004	0,004	0,004	0,004	0,005	0,003	0,004	0,003
1,10	207	227	5	1	0,031	0,031	0,031	0,031	0,030	0,036	0,035	0,034
1,10	207	257	1	3	0,236	0,286	0,218	0,265	0,249	0,259	0,235	0,269
1,10	207	257	3	5	0,004	0,006	0,004	0,006	0,005	0,005	0,004	0,005
1,10	207	257	5	1	0,031	0,034	0,031	0,034	0,030	0,047	0,035	0,044
1,10	235	197	1	1	0,728	0,690	0,688	0,651	0,719	0,692	0,693	0,694
1,10	235	197	3	3	0,047	0,035	0,047	0,035	0,044	0,035	0,035	0,035
1,10	235	197	5	5	0,000	0,000	0,000	0,000	0,000	0,000	0,000	0,000
1,10	235	227	1	5	0,045	0,044	0,040	0,039	0,044	0,046	0,041	0,052
1,10	235	227	3	1	0,262	0,262	0,262	0,262	0,201	0,257	0,219	0,252
1,10	235	227	5	3	0,005	0,005	0,005	0,005	0,003	0,005	0,004	0,006
1,10	235	257	1	5	0,045	0,066	0,040	0,059	0,044	0,059	0,041	0,066
1,10	235	257	3	3	0,047	0,059	0,047	0,059	0,044	0,059	0,035	0,057
1,10	235	257	5	1	0,048	0,052	0,048	0,052	0,030	0,059	0,035	0,056
1,50	207	275	1	1	0,764	0,740	0,727	0,703	0,859	0,748	0,764	0,734
1,50	207	275	3	5	0,011	0,007	0,011	0,007	0,025	0,006	0,006	0,004
1,50	207	275	5	3	0,006	0,005	0,006	0,005	0,003	0,006	0,009	0,007
1,50	207	310	1	3	0,373	0,373	0,347	0,347	0,453	0,324	0,359	0,358
1,50	207	310	3	1	0,248	0,248	0,248	0,248	0,201	0,324	0,268	0,268
1,50	207	310	5	3	0,006	0,006	0,006	0,006	0,003	0,009	0,009	0,009

Une autre possibilité est de faire en sorte que q soit le même pour toutes les périodes, ce qui rend A1 exacte. Si beaucoup de contacts arrivent pendant une période et sont servis pendant une autre, les transitions fictives occasionnées par une valeur de q fixe pourraient bien imposer un coût moins élevé que de multiples intégrations numériques.

5.4 Expérimentations numériques

Nous avons testé notre méthode de simulation sur des exemples de diverses tailles et complexités, comportant une seule période. Les résultats obtenus devraient se généraliser sans problème à un cas avec plusieurs périodes. Avant de présenter nos résultats de simulation, nous décrivons ces exemples. Nous avons choisi des exemples avec beaucoup d'agents, pour lesquels l'usage de la CMTC est intéressant.

5.4.1 Exemples testés

Exemple 1. Pour $K = I = 1$, nous avons utilisé un exemple semblable à celui de la section 3.5.1 mais avec davantage d'agents. Puisque nous souhaitons simuler 13 périodes d'une heure et que le temps est en secondes, nous avons $T = 13 * 3\ 600 = 46\ 800$. Nous souhaitons générer 25 000 contacts pendant l'intervalle $[0,T]$ si bien que $\lambda = 25\ 000/46\ 800 \approx 0,534188$. Nous avons fixé la durée moyenne des services à $1/\mu_1 =$ 100s, le nombre d'agents à $N_1 = 58$, la capacité de la file à $H = 80$ et il n'y a pas d'abandon. La capacité de la file a été maintenue à 80 pour les trois exemples suivants afin que ces exemples soient simulés à peu près avec le même taux de transition.

Nous avons testé une variante de cet exemple avec abandons, que nous appelons 1a. Pour cela, nous avons fixé la probabilité d'abandon immédiat à $\rho_1 = 1/10$, le taux d'abandon à $\nu_1 = 1/1\ 000$, le nombre d'agents à $N_1 = 52$ et la capacité de la file à $H = 80$ et conservé les mêmes valeurs pour les autres paramètres. Nous avons utilisé moins d'agents, car avec 58 agents comme dans la variante sans abandon, le niveau de service grimpait à 92%.

Exemple 2. Nous avons repris l'exemple 1a et augmenté le nombre de types de contacts à 3, avec un seul groupe d'agents. Le taux d'arrivée global est $\lambda = 0,534188$ comme dans l'exemple précédent. Le taux d'arrivée par type de contact est $\lambda_1 = \lambda/4$ et $\lambda_2 = \lambda_3 = 3\lambda/8$. Le temps de service moyen pour ces trois types de contacts est respectivement $1/\mu_1 = 100$s, $1/\mu_2 = 110$s et $1/\mu_3 = 120$s tandis que les paramètres pour les abandons sont $\rho_k = 0,1$ et $1/\nu_k = 1\,000$s pour $k = 1,2,3$. Un agent devenant libre sert un contact de type 2 en priorité, se tourne vers les contacts de type 3 si aucun contact de type 2 n'est en attente et sert un contact de type 1 si aucun contact de types 2 ou 3 n'est en attente. Nous avons fixé le nombre d'agents à $N_1 = 52$.

Nous avons testé une variante de cet exemple pour laquelle $\mu_1 = \mu_2 = \mu_3 = 1/100$ étant donné que des temps de service moyens égaux permettent d'éviter la sélection d'un type de contact lors d'une fin de service.

Exemple 3. Pour construire cet exemple, nous avons augmenté le nombre de groupes d'agents à trois dans l'exemple 1a. Les nouveaux contacts sont servis en priorité par des agents du groupe 1. Si aucun agent du groupe 1 n'est disponible, les contacts débordent vers ceux du groupe 2, puis vers ceux du groupe 3 si aucun agent du groupe 2 n'est disponible. Pour cet exemple, nous avons fixé $N_1 = N_3 = 20$ et $N_2 = 12$.

Le temps de service ne dépend pas du groupe d'agents si bien que les agents ne sont pas différenciés. Séparer les agents en trois groupes est équivalent ici à n'avoir qu'un seul groupe. Nous avons construit cet exemple seulement pour examiner l'effet de plusieurs groupes d'agents sur la performance du simulateur.

Exemple 4. Pour cet exemple, nous avons $K = I = 3$. Comme dans l'exemple 2, $\lambda_1 = \lambda/4$, $\lambda_2 = \lambda_3 = 3\lambda/8$ et $\lambda = 0,534188$. Les temps de patience et de service moyens sont aussi fixés comme dans l'exemple 2. Les groupes d'agents, quant à eux, sont séparés comme dans l'exemple 3. Pour le routage, la liste de groupes d'agents pour le type de contact k est $\{k\}$, $\{k+1\}$, ..., $\{K\}$, $\{1\}$, ..., $\{k-1\}$. La liste de files d'attente pour un

agent du groupe i est quant à elle $\{i\}, \{i-1\}, \ldots, \{1\}, \{I\}, \ldots, \{i+1\}$.

Nous avons également testé une variante pour laquelle $\mu_k = 1/100$ pour $k = 1,2,3$.

Exemple 5. Pour cet exemple, utilisé dans [13], nous avons $K = 3$ et $I = 6$. Le taux d'arrivée total étant λ, les taux d'arrivée pour chaque type de contact sont $\lambda_1 = \lambda_2 = 4\lambda/11$ et $\lambda_3 = 3\lambda/11$. Nous souhaitons générer 14 300 arrivées en 13 heures si bien que $\lambda = 14\,300/46\,800 \approx 0,30556$. Les paramètres pour les abandons sont $\rho_k = 1/100$ et $\nu_k = 1/1\,000$ pour $k = 1,2,3$. Pour ce qui est des durées de service, nous avons $\mu_{1,1} = \mu_{2,2} = \mu_{3,3} = 11/3\,600$ et $\mu_{1,4} = \mu_{1,5} = \mu_{2,5} = \mu_{2,6} = \mu_{3,6} = \mu_{3,4} = 1/360$. Le vecteur d'affectation des agents est $(36,35,27,3,5,4)$. Les paramètres du routage sont donnés dans le tableau 5.VII tandis que nous avons fixé $H = 100$.

Si nous simulons chacun de ces exemples avec 1 000 réplications indépendantes et estimons le niveau de service (2.1), le nombre moyen d'abandons, le temps d'attente moyen et le taux d'occupation, nous obtenons les résultats du tableau 5.VIII. Chaque ligne correspond à un exemple tandis que chaque colonne donne une statistique.

5.4.2 Comparaison de $C_{\mathrm{SIM},1}$, $\tilde{C}_{\mathrm{SIM},2}$ et de la simulation par événements discrets

Nous avons d'abord comparé la conversion en temps discret avec la simulation par événements en ce qui a trait à la variance obtenue pour différentes mesures de performance et au temps d'exécution. Dans le cas de la conversion en temps discret, nous avons

Tableau 5.VII – Paramètres du routage pour l'exemple 5

Type de contact	Groupes d'agents
1	1 4 5
2	2 5 6
3	3 6 4

Groupe d'agents	Types de contacts
1	1
2	2
3	3
4	1 3
5	2 1
6	3 2

Tableau 5.VIII – Estimés des mesures de performance principales pour les exemples testés, avec 1 000 réplications indépendantes

Exemple	Niveau de service	Nombre moyen d'abandons	Temps d'attente moyen (s.)	Taux d'occupation
1	83%	—	2,88	84%
1a	81%	1 709	9,48	95%
2	71%	3 321	41,60	99%
2, $\mu_k = \mu$	88%	1 713	9,46	95%
3	80%	1 713	9,48	95%
4	55%	3 249	38,90	99%
4, $\mu_{k,i} = \mu_i$	85%	1 714	9,46	95%
5	81%	222	12,40	90%

comparé les estimateurs $C_{\text{SIM},1}$ et $\tilde{C}_{\text{SIM},2}$ abordés à la section 5.1.6. Pour ce faire, nous avons simulé 1 000 réplications avec les trois méthodes pour l'exemple 1 avec abandons et pour l'exemple 5. Pour le cas de $\tilde{C}_{\text{SIM},2}$, nous avons choisi $\varepsilon = 10^{-8}$. Les résultats de ces simulations sont présentés dans le tableau 5.IX. Les lignes de ce tableau correspondent à des estimés de mesures de performance, à l'exception de la dernière qui donne des temps d'exécution. Pour chacun des deux exemples, quatre colonnes sont utilisées dans le tableau pour donner respectivement la moyenne obtenue par simulation, la variance avec $C_{\text{SIM},1}$, celle avec $\tilde{C}_{\text{SIM},2}$, ainsi que celle avec la simulation par événements discrets (SED).

Comme nous pouvions nous en attendre d'après l'analyse effectuée dans la sous-section 5.1.7, intégrer sur le nombre de transitions $N(T)$ ne réduit pas beaucoup la va-

Tableau 5.IX – Comparaison de la variance obtenue avec les différentes méthodes de simulation

	Exemple 1a				Exemple 5			
		Variance				Variance		
Mesure	Moyenne	$C_{\text{SIM},1}$	$\tilde{C}_{\text{SIM},2}$	SED	Moyenne	$C_{\text{SIM},1}$	$\tilde{C}_{\text{SIM},2}$	SED
$\mathbb{E}[S]$	23 290	12 132	2 385	12 567	14 078	10 644	4 263	10 280
$\mathbb{E}[L]$	1 699	15 320	15 266	13 532	224	1 708	1 702	1 862
$\mathbb{E}[N_{\text{B}}]$	49,6	0,058	0,058	0,054	98,6	0,901	0,901	0,995
$\mathbb{E}[Q]$	5,0	0,666	0,666	0,578	3,8	0,572	0,570	0,619
Temps CPU (en s.)		8	8	76		9	9	58

riance sur les exemples que nous avons testés, pour la plupart des mesures de performance. Le temps d'exécution est semblable pour les deux estimateurs. Nous employons donc $C_{\mathrm{SIM},1}$ dans les tests suivants.

Mais $\tilde{C}_{\mathrm{SIM},2}$ offre une réduction de variance intéressante par rapport à $C_{\mathrm{SIM},1}$ pour le nombre de contacts servis. Dans ces exemples, presque tous les contacts sont servis. Si nous revenons au modèle simple de la section 5.1.7, la probabilité p de l'événement, qui correspond ici à une fin de service, est près de 1 et non pas près de 0 étant donné qu'il y a peu d'abandons. La borne sur $\mathrm{Var}[C_{\mathrm{SIM},2}]$ n'est alors plus approximativement égale à $\mathrm{Var}[C_{\mathrm{SIM},1}]$.

Pour réduire la variance sur le nombre de contacts servis sans appliquer $\tilde{C}_{\mathrm{SIM},2}$ dans les cas où $\mathrm{Var}[L] < \mathrm{Var}[S]$ (comme dans celui de l'exemple 5), nous pouvons remplacer S par $\mathbb{E}[A] - L$ ou encore utiliser $A - \mathbb{E}[A]$ comme variable de contrôle. Comme nous l'avons vu à la section 3.5.3, la seconde méthode est aussi bonne, sinon meilleure, que la première. Mais ces estimateurs sont sans biais seulement si $\mathbb{E}[A]$ est connue. Si nous simulons plusieurs périodes et comptons les événements concernant un contact dans sa période de sortie plutôt que celle de son arrivée, comme nous le faisons pour éviter d'avoir à créer un objet par contact, le nombre moyen de contacts observés dans une période p donnée n'est pas un estimateur sans biais du nombre espéré d'arrivées pendant cette période. Utiliser la méthode indirecte ou la variable de contrôle ajoute alors du biais aux estimateurs par période.

Avec le simulateur par événements discrets, le temps d'exécution est beaucoup plus élevé et la variance est très proche de celle obtenue avec le simulateur utilisant la CMTC uniformisée. Ceci est normal puisque les mesures testées ici ne dépendent pas du temps d'attente par rapport auquel nous intégrons en calculant les espérance conditionnelles.

5.4.3 Comparaison des temps d'exécution

Pour chaque exemple, nous avons simulé 1 000 réplications indépendantes avec la conversion en temps discret sans taux de transition adaptatif, sans et avec estimation du

niveau de service. Nous avons également exécuté le simulateur par événements sur ce même modèle. Toutes les simulations ont été effectuées en utilisant un générateur de nombres aléatoires de type LFSR113. Ce générateur produit directement des bits qui peuvent être rapidement convertis en indices entiers auxquels correspondent des événements dans le cas où la recherche indexée est utilisée.

Pour cette expérience, le simulateur utilisant la conversion en temps discret, aussi appelé *simulateur simplifié* dans le reste de ce chapitre, considère un modèle moins réaliste dans lequel au temps T, tous les services en cours sont arrêtés et tous les contacts en file, perdus. D'un autre côté, rendu au temps T, le simulateur par événements, aussi appelé *simulateur détaillé* dans le reste de ce chapitre, termine tous les services et vide toutes les files avant d'arrêter la simulation. Cette différence peut ajouter du biais, mais ce biais nous semble négligeable étant donné que nous avons pris un horizon très long et que la taille de la file demeure relativement stationnaire au cours de cet horizon. De cette façon, le nombre de contacts qui ne sont pas traités avec le modèle simplifié et qui le sont avec le modèle détaillé est petit par rapport au nombre total de contacts traités. Pour que le simulateur utilisant la conversion en temps discret vide la file à la fin de l'horizon, il faut simuler une période de fermeture après la période principale, comme nous l'avons expliqué dans les sections 5.1.8 et 5.2.4.

Le tableau 5.X présente les résultats de simulation avec les différents exemples. Chaque ligne du tableau représente un exemple tandis que les colonnes donnent dans l'ordre le nombre moyen de transitions $\mathbb{E}[N(T)]$, un estimé de la proportion de transitions fictives ainsi que les temps d'exécution sans estimation du niveau de service (NS), avec estimation du niveau de service et avec le simulateur détaillé. Les dernières colonnes fournissent les facteurs de gain de performance du simulateur simplifié sans et avec estimation du niveau de service par rapport au simulateur détaillé.

Nous remarquons que l'estimation du niveau de service ralentit l'exécution d'un facteur allant de 1,2 à 1,6 dépendant de l'exemple testé. Le simulateur simplifié est entre 4 et 10 fois plus rapide que le simulateur détaillé. Dans ces deux cas, le facteur dépend de

Tableau 5.X – Temps d'exécution pour les différents exemples testés

Exemple	qT	%. trans. fictives	Temps de calcul (en s.) Sans NS	Avec NS	SED	Facteurs $\frac{\text{SED}}{\text{Sans NS}}$	$\frac{\text{SED}}{\text{Avec NS}}$
1	52 144	4,2%	6	7	60	10,0	8,0
1a	53 080	8,7%	5	8	76	15,0	10,0
1a, MRG32k3a	53 080	8,7%	8	10	73	8,9	7,0
2, $\mu_k = \mu$	53 080	8,7%	9	15	75	8,2	5,0
2	53 080	10,3%	11	18	83	7,3	4,5
3	53 080	8,7%	9	12	76	8,5	6,3
4, $\mu_{k,i} = \mu_i$	53 080	8,7%	10	17	79	7,8	4,7
4	53 080	10,2%	12	20	89	7,5	4,5
4, MRG32k3a	53 080	10,2%	15	23	92	6,1	4,0
5	34 554	17,6%	6	9	58	10,5	6,5

la complexité du modèle traité.

Examinons maintenant plus en profondeur les facteurs qui affectent la performance du simulateur. Si nous augmentons le nombre de types de contacts sans toucher au nombre de groupes d'agents, nous remarquons une forte augmentation du temps d'exécution par rapport au cas où $K = I = 1$. Ceci s'explique par la nécessité d'effectuer des recherches linéaires pour sélectionner des types de contacts lors des fins de service et des abandons. En fixant $\mu_k = \mu$, nous pouvons éviter la sélection d'un contact lors d'une fin de service mais pas celle lors d'un abandon. C'est pourquoi le temps d'exécution de l'exemple 2 avec $\mu_k = \mu$ est supérieur à celui de l'exemple 1a.

Le temps d'exécution de l'exemple 3 est plus grand que celui de l'exemple 1 avec abandons mais plus petit que celui de l'exemple 2. Dans ce cas, la recherche linéaire n'est nécessaire que pour déterminer le groupe d'agents des nouveaux contacts.

Évidemment, le temps d'exécution de l'exemple 4 est plus élevé que celui des exemples précédents. Par contre, même avec l'estimation du niveau de service sur cet exemple de taille modérée, le simulateur simplifié prend 4,5 fois moins de temps que son équivalent détaillé.

Le temps d'exécution de l'exemple 5 est quant à lui plus petit que celui de l'exemple 4 étant donné que pour cet exemple, le taux de transition est plus petit.

Le temps d'exécution est passé à générer des nombres aléatoires, à exécuter les événements lors des transitions et à calculer des statistiques. Si nous remplaçons le générateur LFSR113 par un générateur MRG32k3a qui est plus lent (voir [55] pour plus d'informations à propos de ces générateurs), nous observons en effet un accroissement du temps d'exécution. Dans l'exemple 1 avec abandons, utiliser le MRG augmente le temps d'exécution d'un facteur 1,25 tandis qu'avec l'exemple 4, le temps n'augmente que d'un facteur 1,15. Ainsi, plus le modèle est complexe, plus la génération des variables aléatoires est dominée par d'autres tâches.

Le calcul des statistiques impose lui aussi un certain coût de calcul, car il nécessite plusieurs accès à des tableaux en mémoire. La seule façon de réduire ce coût serait de réduire le nombre de mesures de performance estimées.

5.4.4 Impact de la taille et de la complexité du modèle

Pour examiner comment le programme réagit au changement de la taille du modèle, nous avons tenté d'augmenter le taux d'arrivée dans les exemples 1a et 5. Pour chaque scénario, nous avons ajusté le nombre d'agents afin d'obtenir un niveau de service d'environ 80%. Dans le cas de l'exemple 5, nous avons augmenté le taux global λ tout en gardant la même proportion des arrivées entre les types de contacts. Pour chaque taux d'arrivée, nous avons testé les exemples avec 1 000 réplication indépendantes. Le tableau 5.XI présente les résultats de ces simulations. La partie supérieure du tableau concerne l'exemple 1a tandis que la partie inférieure présente les résultats pour l'exemple 5. Chaque ligne correspond à un scénario tandis que les colonnes donnent le nombre moyen d'arrivées pendant l'horizon $[0, T]$, le nombre d'agents et la capacité de la file d'attente, le nombre moyen de transitions et les temps d'exécution avec le simulateur simplifié et le programme détaillé, respectivement. La dernière colonne donne le rapport entre les deux temps de simulation. Nous remarquons ici que le gain en performance augmente avec le taux d'arrivée. Cet écart de performance s'explique par le fait que plus le taux d'arrivée est élevé, plus le simulateur détaillé doit créer d'objets pour

représenter les contacts tandis que le simulateur simplifié ne crée pas davantage d'objets si le taux d'arrivée augmente. De plus, le gain est moins élevé pour l'exemple 5 que pour l'exemple 1 a étant donné les recherches linéaires nécessaires pour que le simulateur simplifié traite l'exemple 5.

Nous avons aussi, dans l'exemple 4, augmenté K et I à 6 puis à 9 en dupliquant les trois types de contacts et les trois groupes d'agents une et deux fois, respectivement. Pour chaque type de contact, le taux d'arrivée de l'exemple 4 original a ensuite été divisé par deux ou trois selon la variante afin de tester avec le même taux global d'arrivée. De même, le nombre d'agents dans chaque groupe a été divisé par deux ou par trois selon la variante. Pour les deux variantes, la capacité de la file a été fixée à 80. Nous avons simulé chaque variante avec 1 000 réplications indépendantes. Le tableau 5.XII présente les résultats de cette expérimentation, chaque ligne correspondant à un scénario et chaque colonne donnant la valeur de K, un temps d'exécution ou un facteur de gain comme dans le tableau 5.XI. Comme nous pouvons nous en attendre, la différence de performance entre les deux simulateurs diminue avec l'accroissement du nombre de types de contacts.

Ainsi, utiliser le simulateur simplifié n'est pas très intéressant dans un cas où K ou I sont très grands, par exemple plus de 50. Une solution possible est d'approximer le modèle complexe en agrégeant des types de contacts et des groupes d'agents. Ceci pourrait être acceptable, car le modèle simplifié n'est habituellement utilisé que pendant les premières étapes de l'optimisation et un modèle plus détaillé est employé pour raffiner les solutions obtenues.

5.4.5 Test avec un taux de transition adaptatif

Nous avons testé l'heuristique développée à la section 5.2.3 sur l'exemple 5, pour différentes capacités de file H et nombres R de sous-ensembles de $\mathcal{S}$. Pour créer les partitions de $\mathcal{S}$, nous avons utilisé des seuils sur la taille de la file d'attente répartis également entre 0 et H.

Le tableau 5.XIII montre le résultat de ces tests. Pour chaque combinaison testée de

Tableau 5.XI – Temps d'exécution en fonction du taux d'arrivée

	λT	N	H	qT	Temps CMTC (en s.)	Temps SED (en s.)	Facteur
Ex. 1a	1 660	5	30	5 404	1	6	4,6
	25 000	52	80	57 292	11	89	8,2
	50 000	100	130	104 756	18	188	10,0
	75 000	148	170	152 220	25	299	12,0
Ex. 5	14 300	110	100	34 554	9	58	6,5
	25 000	176	100	55 420	19	115	6,1
	50 000	342	120	105 380	37	298	8,1
	75 000	510	150	154 092	53	431	8,1

Tableau 5.XII – Temps d'exécution en fonction du nombre de types de contacts

K	Temps CMTC (en s.)	Temps SED (en s.)	Facteur
3	20	89	4,5
6	24	94	3,9
9	30	98	3,2

H et de R, nous avons simulé 1 000 réplications indépendantes. Avec $R = 1$, le temps d'exécution augmente beaucoup avec H. Plus R est grand, plus le taux de croissance du temps par rapport à H est petit. Par contre, plus R est élevé, plus le gain supplémentaire résultant de l'augmentation de R est petit.

Ces résultats montrent que le taux de transition adaptatif avec un petit nombre de sous-ensembles de $\mathcal{S}$ permet de fixer une capacité de file élevée afin de réduire la probabilité qu'un contact soit bloqué. La majeure partie du temps d'exécution est alors passée avec un taux de transition $q(r)$ petit tandis qu'un taux plus élevé est employé dans les cas (rares) où la taille de la file devient élevée.

5.5 Étude de la synchronisation des variables aléatoires communes pour le simulateur simplifié

Avec la simulation par événements discrets, nous utilisons une séquence de variables aléatoires distincte pour chaque type d'événement, ce qui favorise une bonne synchronisation entre les configurations comparées avec variables aléatoires communes ; les mêmes nombres sont utilisés à peu près aux mêmes endroits. Par contre, avec le modèle de CMTC, une séquence de nombres aléatoires est toujours utilisée pour la sélection du type d'événement si bien que des problèmes de synchronisation peuvent apparaître.

Comme nous l'avons vu dans la section 5.2, si nous souhaitons évaluer l'effet d'un

Tableau 5.XIII – Temps d'exécution (en secondes) en fonction de la capacité de la file d'attente et du nombre de sous-ensembles de $\mathcal{S}$

$H \backslash R$	1	2	5	10
100	9,24	8,63	8,27	8,32
300	10,64	9,60	9,32	9,40
500	12,35	10,85	9,91	9,67
700	14,70	12,50	10,74	10,35
900	15,95	13,14	11,50	11,32
1 000	16,72	14,42	11,80	11,29

paramètre sur la performance du centre de contacts, il vaut mieux simuler chaque configuration avec le même taux de transition pour obtenir une bonne synchronisation. Par contre, cela nous oblige parfois à utiliser un taux de transition plus élevé qui peut augmenter le temps d'exécution. Utiliser un taux de transition adaptatif comme nous l'avons vu à la sous-section 5.2.3 pourrait alléger ce problème, mais aussi diminuer la synchronisation.

Dans cette section, nous présentons d'abord des résultats numériques comparant diverses méthodes de gestion du taux de transition sur un exemple. Pour mieux comprendre les résultats obtenus, nous analysons ensuite le comportement du modèle lorsque seul le nombre de transitions change d'une configuration à l'autre. Nous examinons ensuite ce qui se passe avec la synchronisation lorsque le taux de transition est changé en fixant une borne différente sur le nombre d'agents. Nous proposons des méthodes pour restaurer la synchronisation, puis examinons le cas où le taux de transition est adaptatif.

Nous en concluons que la meilleure stratégie consiste à utiliser un taux de transition identique pour chaque configuration ou au pire un taux adaptatif avec une petit nombre de sous-ensembles de $\mathcal{S}$.

5.5.1 Exemple numérique

Pour étudier l'impact du taux de transition sur l'efficacité des variables aléatoires communes, nous avons pris l'exemple 1a de la section 5.4.1 avec un seul type de contact, un seul groupe d'agents et une moyenne de 25 000 contacts durant l'horizon de longueur 46 800, pour un nombre d'agents allant de 50 à 55. Nous avons estimé la variation de la performance en fonction du nombre d'agents, par rapport au cas avec 52 agents, sans et avec les variables aléatoires communes, pour différents modes de gestion du taux de transition. Le cas avec 52 agents est pris comme base ici, car il donne 80% de niveau de service.

Pour chaque mode de gestion testé, nous avons simulé 1 000 réplications indépendantes avec 52 agents. Ensuite, pour $N_1 = 50, \ldots, 55$, nous avons simulé 1 000 réplica-

tions avec les mêmes variables aléatoires que la simulation de base, puis 1 000 autres réplications avec des variables aléatoires indépendantes. Nous avons ensuite calculé la moyenne et la variance empirique de la différence pour $N_1 = 50, \ldots, 55$.

Comme modes de gestion, nous avons testé le taux de transition q dépendant du nombre d'agents ainsi qu'un taux uniformisé, avec $\tilde{N}_1 = 60$, avec un partitionnement de l'ensemble d'états $\mathcal{S}$ en différents nombres de morceaux R. Le partitionnement a été effectué avec la méthode de la sous-section 5.2.3 et ne change pas en fonction du nombre d'agents testé N_1.

Le tableau 5.XIV présente les résultats de cette analyse de sensibilité dans le cas du nombre de contacts ayant attendu moins de vingt secondes avant d'être servis ou d'abandonner. Chaque ligne donne les résultats pour un nombre d'agents particulier tandis que, comme à la section 4.3, chaque paire de colonnes fournit la variation estimée $\bar{\Delta}_n$ du nombre de contacts ayant attendu moins de vingt secondes par rapport au cas de base avec 52 agents ainsi que la variance empirique de cette variation. Le premier groupe de colonnes correspond au cas où le taux de transition maximal dépend du nombre d'agents. Les autres paires donnent les résultats pour les cas où le taux de transition maximal est fixé en utilisant 60 comme borne sur le nombre d'agents et $\mathcal{S}$ est partitionné en $R = 1, 2, 5, 10$ sous-ensembles. La partie supérieure du tableau contient les résultats avec variables aléatoires indépendantes tandis que la partie inférieure donne les résultats avec les variables aléatoires communes.

Avec les variables aléatoires indépendantes, la variance de la différence ne change pas de façon significative avec la méthode de gestion du taux de transition. Par contre, elle diminue si le nombre d'agents est plus élevé. En effet, plus le nombre d'agents dans la seconde configuration est élevé, plus la variance de la probabilité (aléatoire) qu'un contact attende moins de vingt secondes est petite, ce qui réduit la variance de $G(s)$. La réduction de la variance pour la seconde configuration réduit également la variance de la différence estimée avec les variables aléatoires indépendantes.

Avec les variables aléatoires communes, la variance augmente quand le nombre

Tableau 5.XIV – Impact d'un changement du nombre d'agents sur le nombre moyen de contacts ayant attendu moins de vingt secondes

		q dépend de N_1		q uniformisé							
				$R=1$		$R=2$		$R=5$		$R=10$	
	N_1	$\bar{\Delta}_n$	$\widehat{\text{Var}}[\Delta_i]$	$\bar{\Delta}_n$	$\widehat{\text{Var}}[\Delta_i]$	$\bar{\Delta}_n$	$\widehat{\text{Var}}[\Delta_i]$	$\bar{\Delta}_n$	$\widehat{\text{Var}}[\Delta_i]$	$\bar{\Delta}_n$	$\widehat{\text{Var}}[\Delta_i]$
VAI	50	-3 357	1 499 959	-3 342	1 465 081	-3 343	1 464 156	-3 340	1 452 514	-3 337	1 469 114
	51	-1 523	1 234 826	-1 515	1 203 824	-1 515	1 203 888	-1 512	1 192 911	-1 504	1 197 006
	53	1 162	828 743	1 160	820 189	1 160	819 914	1 164	813 693	1 170	818 025
	54	2 069	722 476	2 063	706 988	2 063	706 397	2 065	701 073	2 074	707 224
	55	2 752	636 980	2 754	632 004	2 754	631 466	2 756	626 874	2 765	633 628
VAC	50	-3 302	266 134	-3 308	207 605	-3 308	207 587	-3 311	209 859	-3 311	215 246
	51	-1 482	78 936	-1 487	60 873	-1 487	60 895	-1 488	61 757	-1 486	62 563
	53	1 167	56 135	1 164	46 949	1 164	46 863	1 164	46 254	1 169	47 583
	54	2 070	139 696	2 064	126 070	2 064	126 020	2 067	124 830	2 073	124 842
	55	2 745	224 214	2 747	215 483	2 747	215 461	2 753	212 449	2 760	211 493

d'agents s'éloigne de 52, car la corrélation entre les deux configurations comparées diminue. Peu importe la méthode de gestion du taux de transition, la variance de la différence estimée avec les variables aléatoires communes est toujours inférieure à celle avec les variables indépendantes. Mais comme nous nous attendions, utiliser un taux de transition identique pour chaque configuration réduit davantage la variance que faire varier le taux en fonction du nombre d'agents. Partitionner l'ensemble d'états pour obtenir un taux de transition adaptatif réduit la plupart du temps la variance par rapport à un taux uniformisé avec un seul sous-ensemble de $\mathcal{S}$. Mais dans certains cas, la variance augmente un peu si R est assez élevé, sans dépasser celle obtenue avec un taux de transition dépendant du nombre d'agents. Cette augmentation de la variance est sans doute causée par une réduction de la synchronisation tandis que l'analyse de cette section nous aidera à comprendre les raisons de la diminution. Le tableau 5.XV montre des résultats similaires pour le nombre moyen d'abandons.

5.5.2 Impact du changement du nombre de transitions sur la synchronisation

Supposons que les paramètres du système sont fixes et simulons avec deux taux de transition différents : q et $\tilde{q} > q$. La première variable aléatoire générée est $N(T)$,

Tableau 5.XV – Impact d'un changement du nombre d'agents sur le nombre moyen d'abandons

				q uniformisé							
		q dépend de N_1		$R=1$		$R=2$		$R=5$		$R=10$	
	N_1	$\bar{\Delta}_n$	$\widehat{\text{Var}}[\Delta_i]$	$\bar{\Delta}_n$	$\widehat{\text{Var}}[\Delta_i]$	$\bar{\Delta}_n$	$\widehat{\text{Var}}[\Delta_i]$	$\bar{\Delta}_n$	$\widehat{\text{Var}}[\Delta_i]$	$\bar{\Delta}_n$	$\widehat{\text{Var}}[\Delta_i]$
VAI	50	501	32 775	500	32 642	500	32 666	499	32 088	498	32 506
	51	239	29 715	238	30 062	238	30 071	237	29 600	236	29 751
	53	-220	25 094	-220	25 813	-220	25 839	-221	25 463	-222	25 604
	54	-419	23 486	-417	23 976	-417	24 001	-418	23 553	-419	23 607
	55	-597	22 191	-597	22 429	-597	22 448	-598	22 077	-599	22 071
VAC	50	498	1 516	501	846	501	847	501	909	501	925
	51	239	575	239	297	239	297	239	322	238	350
	53	-218	530	-217	256	-217	258	-217	269	-218	295
	54	-416	1 190	-414	656	-414	657	-414	662	-415	678
	55	-595	1 924	-594	1 173	-594	1 175	-594	1 147	-595	1 177

le nombre de transitions à simuler. Nous devons ensuite générer un nombre aléatoire pour chacune des transitions. Si nous générons $N(T)$ par inversion à partir du même uniforme mais en utilisant deux taux de transition différents, la valeur de $N(T)$ augmente avec le taux de transition. Il faudra ainsi davantage de nombres aléatoires pour simuler les transitions pour la configuration avec le taux plus élevé. Soit $N(T,1)$ le nombre de transitions de moyenne qT et $N(T,2)$ le nombre avec moyenne $\tilde{q}T$.

Lorsque nous fixons le taux de transition à $\tilde{q}$, les $N(T,1)$ premiers nombres aléatoires uniformes générés pour simuler les transitions sont identiques aux nombres générés si nous simulions avec taux q en utilisant la même séquence d'uniformes. Les $D = N(T,2) - N(T,1)$ nombres restants pour simuler avec taux $\tilde{q}$ sont par contre complètement indépendants et n'apparaissent pas du tout lorsque nous utilisons le taux q. Dans ce cas, plus D est élevé, plus la variance d'une variable aléatoire dépendant de la séquence des uniformes générés avec taux de transition $\tilde{q}$ est élevée par rapport à la variance de cette même variable aléatoire générée avec un taux de transition q. De même, la corrélation entre les deux configurations diminue quand D augmente. Ainsi, le changement du nombre de transitions généré explique déjà en partie pourquoi la variance de la différence est plus élevée si nous comparons deux configurations avec un taux de

transition différent que si nous les comparons avec le même taux.

5.5.3 Impact sur la synchronisation du changement de la borne sur le nombre d'agents

Avec un taux de transition maximal fixe, peu importe le nombre d'agents du modèle, le nombre d'arrivées sera identique si nous employons les mêmes nombres aléatoires pour simuler chaque configuration. Le nombre d'événements correspondant à des abandons ou à des fins de service est également conservé. Selon le nombre d'agents, certains abandons ou fins de service peuvent devenir des transitions fictives, mais un abandon ne peut pas devenir une fin de service ou vice versa. La synchronisation des nombres aléatoires est ici très semblable à celle obtenue dans le simulateur à événements discrets.

Supposons qu'aucun paramètre ne change entre les configurations, à l'exception de la borne sur le nombre d'agents qui est fixée à N_1 avec le taux de transition q et $\tilde{N}_1 > N_1$ pour le taux $\tilde{q}$. L'analyse que nous faisons ici s'applique aussi si μ est borné par $\tilde{\mu}$ et des analyses très semblables seraient possibles pour le cas où $\tilde{\lambda} > \lambda$ ou $\tilde{H}\tilde{\nu} > H\nu$.

Prenons maintenant une transition particulière, dont le type est généré par inversion à partir d'un uniforme U. La figure 5.8 montre la répartition des types d'événements dans le cas de l'exemple 1a, avec $N_1 = 52$ et $\tilde{N}_1 = 100$. Nous avons choisi une valeur élevée pour $\tilde{N}_1$ afin de bien mettre en évidence les différences de la répartition des événements. En pratique, le nombre d'agents maximal est plus petit, car même avec la meilleure synchronisation possible, la trajectoire de la CMTC avec 52 agents sera très différente de celle avec 100 agents.

L'axe vertical de la figure représente l'intervalle $[0, 1)$ qui est subdivisé en fonction des probabilités des événements. La zone blanche de la figure représente un sous-intervalle auquel seulement des transitions fictives sont associées, peu importe l'état du système si le nombre d'agents est N_1.

Cette figure illustre le fait que selon la valeur de U, le type de l'événement généré par inversion peut changer en fonction du taux de transition. En examinant la figure,

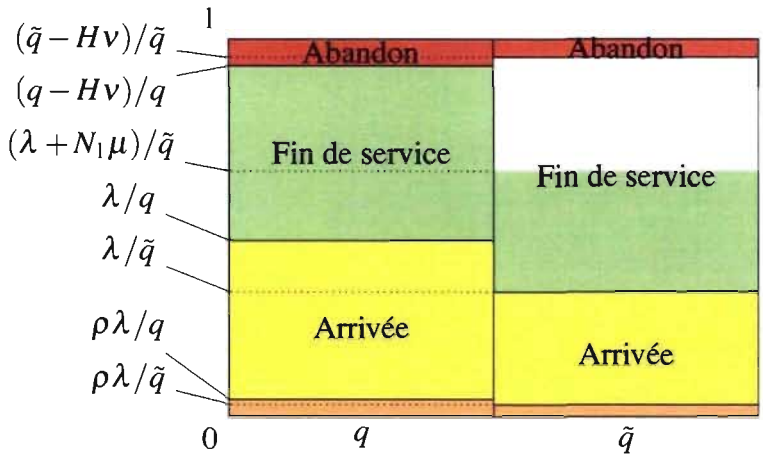

Figure 5.8 – Répartition des types d'événements pour deux taux de transition maximaux différents

nous pouvons dresser la liste des sous-intervalles de $[0,1)$ pour lesquels des problèmes de synchronisation peuvent survenir. Le tableau 5.XVI résume les différents types de désynchronisations qu'il est possible d'observer dans une telle situation. Chaque ligne représente un type de désynchronisation tandis que les colonnes donnent l'intervalle qui est associé au problème de synchronisation, les conditions sous lesquelles il survient et sa nature, c'est-à-dire le type de transition avec taux q par rapport au nouveau type avec taux $\tilde{q}$.

Les trois premières lignes du tableau décrivent des désynchronisations qui font en

Tableau 5.XVI – Désynchronisations possibles si q passe à $\tilde{q}$ quand N_1 passe à $\tilde{N}_1$

Intervalle	Condition	Événement avec q	Événement avec $\tilde{q}$
$\left[\frac{\lambda}{\tilde{q}}, \frac{\lambda}{q}\right[$	$U > \frac{\lambda+S_{1,1}\mu}{\tilde{q}}$	Arrivée	Transition fictive
$\left[\frac{\lambda}{\tilde{q}}, \frac{\lambda}{q}\right[$	$U \leq \frac{\lambda+S_{1,1}\mu}{\tilde{q}}$	Arrivée	Fin de service
$\left[\frac{\tilde{q}-H\nu}{\tilde{q}}, 1\right]$	$\frac{\tilde{q}(U-1)+H\nu}{\tilde{q}\nu} \leq Q_1 < \frac{q(U-1)+H\nu}{q\nu}$	Transition fictive	Abandon
$\left[\frac{\lambda+N_1\mu}{\tilde{q}}, \frac{\lambda+\tilde{N}_1\mu}{\tilde{q}}\right[$	$U \leq \frac{\lambda+S_{1,1}\mu}{q}$	Fin de service	Transition fictive
$\left[\frac{\rho\lambda}{\tilde{q}}, \frac{\rho\lambda}{q}\right[$	$S_{1,1} = N_1$	Abandon immédiat	Mise en attente
$\left[\frac{q-H\nu}{q}, \frac{\tilde{q}-H\nu}{\tilde{q}}\right[$	$Q_1 \geq \frac{q(U-1)+H\nu}{\nu q}$	Abandon	Transition fictive

sorte qu'après la transition concernée, il y a moins de contacts dans le système avec taux $\tilde{q}$ que dans celui avec taux q. En particulier, si $\lambda/\tilde{q} \leq U < \lambda/q$, une arrivée se produit dans la configuration avec taux q tandis qu'a lieu une fin de service ou une transition fictive avec taux $\tilde{q}$. Pour avoir la fin de service avec taux $\tilde{q}$, il faut que $S_{1,1}$ soit suffisamment grand pour que $\tilde{q}U - \tilde{q}\lambda \leq S_{1,1}\mu$. De même, une transition fictive peut se transformer en abandon si la taille de la file est suffisamment grande pour provoquer l'abandon dans la configuration avec taux $\tilde{q}$ mais suffisamment petite pour provoquer la transition fictive dans le cas où le taux est q. Pour choisir entre un abandon ou une transition fictive avec taux q, nous devons calculer $U-(q-H\nu)/q = ((U-1)q+H\nu)/q$ et diviser ce nombre par ν. Cela donne la taille minimale de la file pour que l'abandon se produise. Une expression semblable permet d'obtenir la taille minimale pour qu'un abandon ait lieu avec taux $\tilde{q}$. Évidemment, Q_1 doit dépasser la borne pour $\tilde{q}$ mais non pour q.

Les trois dernières lignes du tableau concernent des désynchronisations qui font en sorte qu'il y a plus de contacts avec $\tilde{q}$ qu'avec q après la transition concernée. En particulier, une fin de service peut ne pas avoir lieu et devenir une transition fictive, ce qui augmente le temps de service d'un contact et par le fait même le nombre de contacts dans le système après la transition. Le nombre d'agents occupés $S_{1,1}$ doit bien sûr être assez grand pour provoquer la fin de service dans le cas où le taux est fixé à q. Il faut donc avoir la condition $U \leq (\lambda + S_{1,1}\mu)/q$. Si tous les agents sont occupés, un contact peut également joindre la file d'attente au lieu d'abandonner immédiatement. Un abandon qui se produisait avec le taux de transition fixé à q peut enfin devenir une transition fictive avec taux $\tilde{q}$, ce qui fait qu'un contact passe plus de transitions dans le système avec le taux $\tilde{q}$. Bien entendu, pour que cela se produise, la taille de la file doit être suffisamment grande pour que l'abandon se produise avec taux q.

Si $\tilde{N}_1 \gg N_1$, il peut arriver que $\lambda/q > (\lambda + N_1\mu)/\tilde{q}$ et que toutes les fins de service dans la configuration avec taux q deviennent des transitions fictives avec taux $\tilde{q}$. Nous n'examinons pas ce cas en détails, car lorsqu'il survient, q est si différent de $\tilde{q}$ que

la synchronisation est très faible entre les deux configurations en raison du nombre de transitions $N(T)$ beaucoup plus élevé avec $\tilde{q}$.

Il faut garder à l'esprit qu'une désynchronisation isolée n'a qu'un effet local sur l'évolution du modèle : un contact de moins ou de plus en service ou en attente dans la configuration avec taux $\tilde{q}$ que dans celle avec taux q. Si le nombre d'agents dans le modèle est grand et si le taux d'abandon est petit, l'impact de cette différence est petit. Ainsi, à une étape n, les deux systèmes comparés sont susceptibles de se trouver dans un état très semblable et ainsi demeurer relativement synchronisés. Par contre, il faut pour cela que le nombre de désynchronisations soit petit.

La probabilité qu'une arrivée ou un abandon soient perdus en raison du changement de taux de transition est proportionnelle à $1/q - 1/\tilde{q}$. La probabilité qu'une fin de service devienne une transition fictive est quant à elle inférieure ou égale à $(\tilde{N}_1 - N_1)\mu/\tilde{q}$. Si la valeur de ces expressions est petite pour les paramètres du modèle, nous pouvons considérer que l'impact du changement de taux de transition sur la synchronisation est minime. Ces expressions indiquent en particulier que le taux de transition et le nombre d'agents ne doivent pas trop différer entre les configurations comparées.

Le tableau 5.XVII regroupe les probabilités des différents événements pour l'exemple 1a, avec trois taux de transition différents, obtenus en fixant $\tilde{N}_1 = 52, 55, 60$. D'après les deux dernières lignes du tableau, la probabilité qu'un événement change de type lorsque la borne passe de 52 à 55 ou 60 est assez petite ; la séquence des états visités ne devrait pas trop changer entre ces deux configurations.

Examinons maintenant, pour cet exemple, la variation du nombre de contacts dans

Tableau 5.XVII – Probabilité des différents événements pour l'exemple 1a avec 25 000 contacts en moyenne pendant un horizon $T = 46\,800$

$\tilde{N}_1$	52	55	60
$\tilde{q}$	$q = 1,134$	1,164	1,214
$1/q - 1/\tilde{q}$	0	0,02	0,06
$\mu(\tilde{N}_1 - N_1)/\tilde{q}$	0,03 0		0,07

le système par rapport au nombre de transitions simulées. Pour cela, fixons $T = 300$ et simulons un nombre aléatoire de transitions pour $\tilde{N}_1 = 52, 55, 60$. Pour chacune des trois simulations, nous réutilisons les mêmes nombres aléatoires et conservons le nombre Z_n de contacts dans le système à chaque transition n. La figure 5.9 présente la variation du nombre de contacts dans le système en fonction du numéro de la transition. La courbe rouge correspond à la configuration avec borne $\tilde{N}_1 = 52$, la courbe bleue représente celle avec $\tilde{N}_1 = 55$ agents tandis que la courbe verte montre la trajectoire avec $\tilde{N}_1 = 60$ agents. Les trois courbes ne s'arrêtent pas au même endroit, car nous avons simulé un nombre de transitions différent pour chaque configuration.

Comme nous pouvons le voir, la trajectoire du nombre de contacts est assez semblable dans les cas où la borne sur le nombre d'agents est 52 et 55. Dans le second cas, le nombre de contacts est légèrement inférieur si la file est petite et supérieur quand la file est assez grande. Ces observations confirment l'analyse précédente des désynchronisations. La courbe verte diffère bien entendu plus fortement des deux autres courbes. Malgré tout, ces courbes sont assez près les unes des autres.

Pour examiner l'effet du changement d'échelle causé par la variation de $N(T)$ lorsque le taux de transition passe de q à $\tilde{q}$, nous avons tracé de nouvelles courbes en remplaçant n par le temps discret normalisé $t = Tn/N(T)$. Soit $Z(t)$ le nombre de contacts dans le système au temps normalisé t. La figure 5.10 présente les courbes tracées en fonction de ce temps normalisé.

Encore une fois, les courbes restent assez près les unes des autres mais suivent des trajectoires assez différentes. Ces courbes montrent bien que la synchronisation devient mauvaise dès que $N(T)$ n'est plus le même d'une configuration à l'autre. Évidemment, si nous simulons avec un nombre différent d'agents pour chaque configuration ou sur un horizon plus long, la désynchronisation sera encore plus importante.

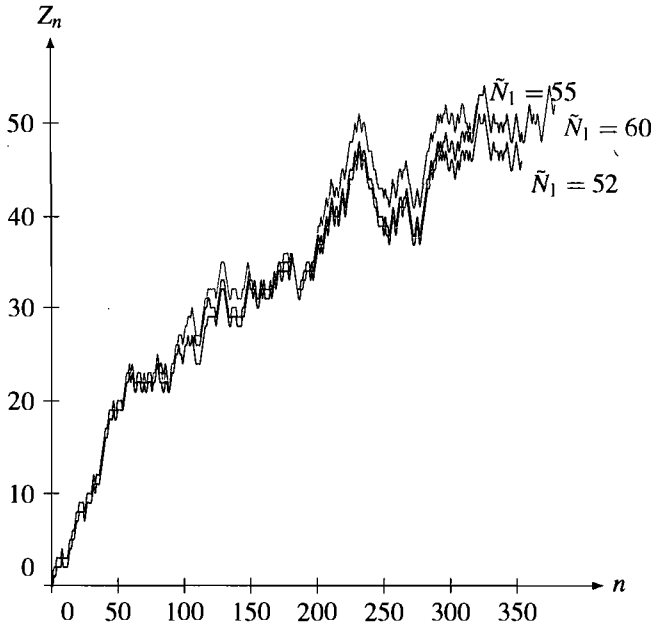

Figure 5.9 – Progression du nombre Z_n de contacts dans le système en fonction du numéro n de transition, pour trois taux de transition différents

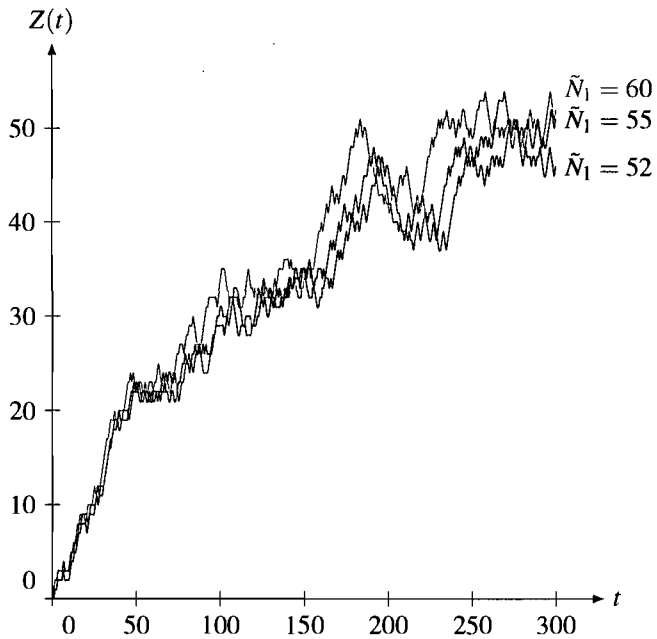

Figure 5.10 – Progression du nombre $Z(t)$ de contacts dans le système en fonction du temps discret normalisé $t = Tn/N(T)$, pour trois taux de transition différents

5.5.4 Restauration de la synchronisation

La séquence de nombres aléatoires utilisée par le simulateur est divisée en un nombre très grand de sous-séquences [66]. Au début de la simulation, le générateur est initialisé au début de sa séquence et un saut à la sous-séquence suivante est effectué à la fin de chaque réplication. De cette façon, peu importe les événements qui se produisent pendant une réplication, la synchronisation se trouve restaurée au début de la réplication suivante.

Pour améliorer la synchronisation entre des configurations avec taux de transition différents, il faudrait sauter à la sous-séquence suivante en m points où l'état est identique d'une configuration à l'autre. Mais cette heuristique ne va que tenter de rétablir la synchronisation qui risque de se briser à nouveau après quelques transitions si le taux de transition diffère beaucoup entre les configurations. Le gain apporté par les m sauts du générateur de nombres aléatoires est ainsi incertain.

Nous pourrions imaginer employer une séquence de nombres différente pour générer les instants de chaque type d'événements. Sans conversion en temps discret, les événements se produisent selon un processus de Poisson de taux q qu'il est possible de décomposer en trois processus générant des arrivées, des fins de service potentielles et des abandons. Nous pourrions alors générer des temps d'occurrence d'événements avec ces trois processus séparément, avec une séquence de nombres aléatoires distincte pour chaque processus. Par contre, avec cette stratégie, nous perdrions les bénéfices de la conversion en temps discret et reviendrions à une simulation avec liste d'événements.

Pour appliquer cette idée en temps discret, nous pourrions utiliser le fait que le nombre de transitions séparant deux arrivées suit une loi géométrique de paramètre λ/q. Nous pourrions ainsi générer tous les instants correspondant à des arrivées jusqu'à ce qu'une valeur plus grande que $N(T)$ soit produite. Nous pourrions faire de même avec les instants de fins de service et d'abandons potentiels, mais nous ne connaissons aucune façon de combiner les trois processus stochastiques si, par exemple, une arrivée a été générée au même moment qu'une fin de service. Dans tous les cas, la méthode prendrait

beaucoup de mémoire et reviendrait à avoir une liste d'événements.

Supposons maintenant que, si nous sommes dans l'état $i \in \mathcal{S}$, nous générons un nombre géométrique Z de transitions fictives successives avec paramètres $q_i/\tilde{q}$ en utilisant une valeur aléatoire avant de générer la transition principale avec taux q_i en employant une deuxième valeur aléatoire. Clairement, seul Z varie en fonction de $\tilde{q}$ tandis que l'état j résultant de la transition principale est le même si des valeurs aléatoires identiques sont tirées. Avec cette méthode, seul le nombre de transitions fictives séparant les transitions principales est affecté par le taux de transition ; la séquence des états visités est la même si les paramètres du modèle ne sont pas changés.

Par contre, dès qu'un paramètre du modèle est altéré, la séquence des états visités est affectée, les taux $q_i/\tilde{q}$ changent et cela a un impact sur le nombre de transitions fictives successives séparant les transitions principales. La synchronisation peut donc être perdue tout comme avec les méthodes précédentes. Pour favoriser la synchronisation, il faudrait au moins que les paramètres utilisés pour générer les Z soient semblables d'une configuration à l'autre.

Pour cela, nous pouvons réutiliser l'idée de la sous-section 5.2.3. Notre implantation génère Z et j indépendamment, car elle utilise un nombre fixe de bits aléatoires pour Z et les autres bits pour j. Si le mode d'opération est r, les transitions principales sont générées avec taux $q(r)$, peu importe le taux de transition global $\tilde{q}$. Ainsi, peu importe le taux $\tilde{q}$, la synchronisation est préservée si r est le même d'une configuration à l'autre. Nous illustrons cela dans la sous-section suivante.

5.5.5 Impact sur la synchronisation d'un taux de transition adaptatif

Prenons encore une fois l'exemple 1a avec 25 000 contacts en moyenne sur $T = 46\,800$, avec $\tilde{N}_1 = 100$. Nous avons utilisé une valeur élevée de $\tilde{N}_1$ afin que les courbes suivantes illustrent mieux la perte de synchronisation.

Supposons d'abord que nous séparons $\mathcal{S}$ en $R = 2$ sous-ensembles de façons à ce que $r = 1$ si et seulement si $S_{1,1} \leq 52$. Ainsi, si nous simulons avec n'importe quel nombre

d'agents inférieur ou égal à 52, nous avons toujours $r = 1$ si bien que la synchronisation est préservée. En fonction du nombre d'agents, des abandons ou des fins de service peuvent devenir des transitions fictives ou vice versa, mais tout autre changement de type d'événement est impossible. En particulier, une arrivée ne peut pas devenir une fin de service comme c'était le cas lorsque nous simulions avec des taux de transition différents. Évidemment, la séquence des événements produits diffère du cas où $R = 1$ en raison des transitions fictives successives générées.

Supposons maintenant que nous simulons avec 53 agents et comparons les résultats avec 52 agents. Tant que le nombre de contacts dans le système est inférieur ou égal à 52, le mode d'opération reste $r = 1$ et nous utilisons le taux de transition $q(1)$; la synchronisation par rapport au cas à 52 agents est préservée. Par contre, dès que les 53 agents sont occupés, r passe à 2 et nous simulons avec un taux de transition différent du cas à 52 agents ; les effets sur la synchronisation risquent de ressembler à ce qui se passe avec un taux de transition dépendant du nombre d'agents, mais la perturbation sera moins grande étant donné le temps passé avec $r = 1$ dans les deux configurations.

Nous avons testé cela avec l'exemple 1a, pour un horizon $T = 300$. La figure 5.11 montre la progression du nombre de contacts dans le système en fonction du numéro de transition, pour un nombre d'agents de 51, 52 et 53. La courbe bleue, pour le cas à 52 agents, est très semblable à la courbe rouge pour le cas à 51 agents. Cela nous mène à penser que la synchronisation entre les deux configurations est bonne. Par contre, la courbe verte pour le cas à 53 agents diffère fortement des deux autres à partir du moment où $Z_n > 52$, ce qui trahit une mauvaise synchronisation avec le cas à 52 agents.

Si nous augmentons le nombre R de sous-ensembles de $\mathcal{S}$, nous pouvons améliorer la synchronisation, car le taux de transition $q(r)$ a plus de chances d'être similaire entre les configurations testées. Il est recommandé de fixer les seuils de façon à ce que le plus grand nombre possible de configurations soit simulé avec un taux inférieur ou égal à une même valeur $q(r)$. Par exemple, si nous simulons avec 51, 52 et 53 agents, il est envisageable de fixer un seuil à 52 agents pour que deux configurations sur trois soient

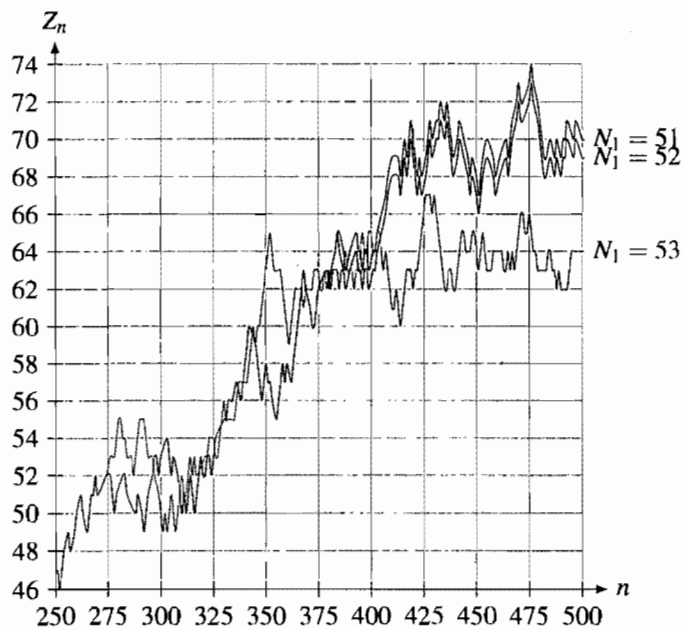

Figure 5.11 – Nombre Z_n de contacts dans le système en fonction du numéro n de transition, pour différents nombre d'agents N_1, avec borne supérieure $\tilde{N}_1 = 100$ et $R = 2$ sous-ensembles de S

simulées avec un seuil de 52 agents et qu'une seule configuration soit simulée avec un seuil de 100 agents. Le troisième seuil est fixé de façon à simuler la configuration à 53 agents avec un taux de transition moins élevé que celui obtenu avec la borne de 100 agents. Une valeur possible pour ce seuil est 60. Ainsi, pour notre illustration, nous supposons que $R = 3$ avec $r = 1$ si $S_{1,1} \leq 52$ et $r = 2$ si $52 < S_{1,1} \leq 60$. Alors, lorsque 53 agents sont occupés, le taux de transition pour 60 agents est utilisé au lieu du taux plus élevé pour 100 agents, ce qui améliore la synchronisation. Nous avons testé l'exemple 1a avec ces nouveaux seuils, pour 51, 52 et 53 agents. Comme le montre la figure 5.12, les courbes pour ces trois nombres d'agents se ressemblent un peu plus que celles sur la figure 5.11. Cela confirme que nous avons amélioré légèrement la synchronisation en augmentant le nombre de sous-ensembles de $\mathcal{S}$ et explique pourquoi la variance d'une différence diminue si R augmente.

À la lumière de ces résultats, nous pouvons penser que plus R est grand, plus la synchronisation est bonne, mais cette hypothèse se trouve contredite par les résultats des tableaux 5.XIV et 5.XV. Changer le nombre d'agents affecte inévitablement la trajectoire de la CMTC. Plus les sous-ensembles de $\mathcal{S}$ sont grands, moins la perturbation de $X(t)$ risque de changer le mode d'opération r au temps t, d'où une meilleure synchronisation. Avec des sous-ensembles de $\mathcal{S}$ petits, qui sont créés si R est grand, il y a de bonnes chances pour que r diffère en fonction du nombre d'agents, d'où une réduction de la synchronisation. Pour cette même raison, il est recommandé d'utiliser des seuils espacés les uns des autres, par exemple 52 et 60, et non pas 52 et 53.

À la lumière de ces résultats, la meilleure stratégie de simulation consiste à fixer une borne sur le nombre d'agents à simuler afin de tester chaque configuration avec le même taux de transition. Si la borne est très élevée, il est possible d'accélérer un peu la simulation en partitionnant $\mathcal{S}$ à l'aide d'un petit nombre de seuils sur le nombre d'agents ou sur la taille de la file.

Si une telle borne est inconnue, par exemple dans le cas d'un problème d'optimisation, nous pouvons fixer une borne initiale $\tilde{N}_1$ avant de commencer les expérimentations.

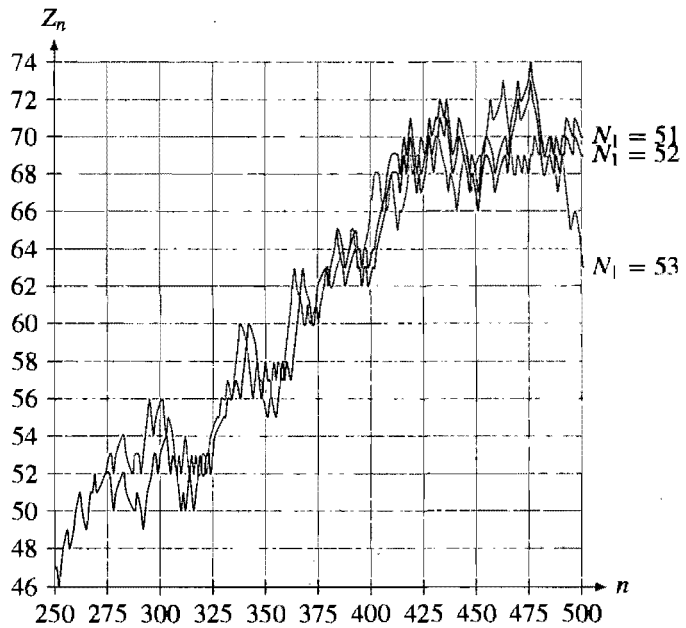

Figure 5.12 – Nombre Z_n de contacts dans le système en fonction du numéro n de transition, pour différents nombre d'agents N_1, avec borne supérieure $\tilde{N}_1 = 100$ et $R = 3$ sous-ensembles de S

S'il est nécessaire d'augmenter la borne, disons à $\tilde{N}_2$, de nouvelles configurations sont simulées avec un taux de transition plus élevé que les premières. Évidemment, la borne ne doit pas changer trop souvent, sinon la synchronisation entre les nouvelles configurations et les plus anciennes sera mauvaise.

CHAPITRE 6

SCISSION ET RECOMBINAISON POUR L'ÉVALUATION DE LA PERFORMANCE EN FONCTION DE PARAMÈTRES DU MODÈLE

Comme nous l'avons expliqué précédemment, l'analyse de sensibilité et l'optimisation demandent d'évaluer la performance d'un modèle pour plusieurs configurations différentes. La simulation est nécessaire pour tenir compte de toute la complexité, mais simuler toutes les configurations requises demande souvent beaucoup de temps.

Nous avons étudié dans les chapitres précédents des méthodes permettant de simuler des configurations plus efficacement. En particulier, nous avons exploré des techniques de réduction de la variance s'appliquant sur différents modèles de centres de contacts et examiné comment la vitesse de simulation peut être augmentée en utilisant une chaîne de Markov en temps continu uniformisée et la conversion en temps discret. Par contre, même si chaque configuration d'un modèle est relativement rapide à simuler, le grand nombre de configurations à tester fait souvent en sorte que l'expérimentation demeure très coûteuse. Ce problème survient particulièrement souvent dans le cas de l'optimisation.

Pour réduire ce coût, outre simplifier le modèle jusqu'à en arriver à une approximation grossière de la réalité, nous pouvons tenter d'exploiter la similitude existante entre les configurations. Par exemple, si nous simulons un centre de contacts avec N agents, le comportement du même système avec $N + 1$ agents sera identique jusqu'à ce qu'une arrivée survienne à un moment où N agents sont occupés.

Dans ce chapitre, nous tentons d'exploiter cette similitude en utilisant une méthode de scission et de recombinaison (*split and merge* en anglais). Cette méthode consiste à scinder une réplication mère en deux ou plusieurs copies chaque fois que les paramètres testés ont un impact sur l'évolution du modèle et simuler toutes les copies en parallèle. Chaque copie peut à son tour se séparer en deux ou plusieurs copies et ainsi de suite.

Dans le pire cas, cela produit un arbre dont la taille augmente de façon exponentielle avec le nombre de points de décision, mais le nombre total de copies ne peut pas excéder le nombre total de configurations testées et nous allons nous concentrer sur les cas courants où le taux de croissance de l'arbre est plus petit. Il est également possible de réduire la taille de l'arbre en recombinant des copies pour lesquelles l'état du modèle est identique.

L'Ecuyer et Vázquez-Abad [68] proposent une méthode de ce genre permettant d'estimer une fonction continue $f(\theta)$ pour toutes les valeurs de $\theta \in [a,b]$. Nous adaptons cette méthode pour la rendre applicable à une version légèrement simplifiée du modèle de CMTC proposé dans le chapitre précédent. Nous discutons aussi de son applicabilité à un modèle plus général et plus détaillé.

La méthode que nous adaptons ici partage certains principes avec la technique de scission classique présentée par exemple dans [62]. Dans la méthode classique, des copies parallèles sont créées lorsque la variance anticipée est grande. Lorsque la variance est plus petite, certaines des copies, souvent choisies au hasard, sont détruites. Avec la technique classique, chaque copie simule avec exactement les mêmes paramètres du modèle mais des nombres aléatoires différents tandis qu'avec la méthode de scission et de recombinaison que nous présentons dans ce chapitre, chaque copie traite une configuration distincte du modèle et les copies parallèles sont simulées avec des variables aléatoires communes.

La première section de ce chapitre expose la méthode de scission et de recombinaison dans un contexte général et discute des problèmes qu'elle pose. Par contre, la méthode présentée dans cette section n'est pas efficace si elle n'est pas adaptée à une application particulière. Dans la section 6.2, nous présentons un exemple d'une telle adaptation pour les centres de contacts. En particulier, nous y examinons comment la scission interagit avec le routage des nouveaux contacts, d'abord pour le cas où le nombre d'agents dans un seul groupe peut varier, puis pour celui où tout le vecteur d'agents peut changer. Dans le dernier cas, le nombre de combinaisons de configurations est en général trop grand si bien que nous nous concentrons sur les configurations nécessaires pour estimer les

sous-gradients, qui sont couramment utilisés en optimisation pour orienter la recherche de solutions [3, 6, 17]. Nous examinons aussi comment appliquer la méthode lorsque d'autres paramètres du centre de contacts comme le taux d'arrivée, les durées moyennes de service, etc. peuvent varier. Enfin, dans la section 6.3, nous présentons des exemples numériques montrant que la méthode permet dans certains cas de réduire le temps de simulation d'un facteur pouvant aller jusqu'à 7 par rapport à la simulation séparée des configurations. Nous verrons que la méthode fonctionne bien pour estimer des sous-gradients lorsque les paramètres du modèle peuvent varier dans le temps. L'application de la méthode aux centres de contacts représente la contribution principale de ce chapitre.

6.1 Méthode générale de scission et de recombinaison

L'idée de la méthode de L'Ecuyer et Vázquez-Abad est de supposer que l'évolution du modèle ne dépend d'un paramètre θ qu'à travers des variables binaires $\eta_n = \mathbb{I}[Z_n \leq \theta]$ où Z_n est une variable aléatoire ne dépendant pas de η_j pour $j \geq n$. À chaque instant où une variable Z_n est générée pour une copie donnée de la simulation, la copie peut être séparée en deux : une copie pour laquelle $Z_n \leq \theta$ et une autre pour laquelle $Z_n > \theta$. Après n points de décision, il existe au plus 2^n copies en parallèle, mais en pratique, ce nombre est beaucoup plus petit. L'article [68] donne plusieurs exemples pour lesquels la méthode s'applique, en particulier des modèles simples de files d'attente.

De notre côté, nous nous intéressons à une fonction dépendant d'un vecteur de paramètres et souhaitons évaluer cette fonction pour tous les vecteurs d'une boîte rectangulaire. Le vecteur peut par exemple correspondre au nombre d'agents dans chaque groupe d'un centre de contacts. De plus, les variables de décision ne sont pas toujours binaires, par exemple si nous devons choisir un groupe d'agents à associer à un nouveau contact. Nous généralisons ainsi la méthode de scission et de recombinaison pour traiter ces aspects.

Pour cela, nous énonçons d'abord les hypothèses sur le modèle et la notation ma-

thématique, qui ressemblent à celles de [68]. Nous définissons ensuite un concept de trajectoires et de copies parallèles utilisé pour élaborer un algorithme de scission et de recombinaison général. Rappelons que la méthode décrite dans cette section doit être adaptée à une application particulière pour être efficace.

6.1.1 Hypothèses sur le modèle

Soit $\boldsymbol{\theta} \in \mathbb{R}^d$ un vecteur de paramètres du modèle à faire varier, par exemple le nombre d'agents dans chaque groupe d'un centre de contacts, la durée moyenne des services pour différents types de contacts, etc. Une *configuration* du modèle est caractérisée par une valeur particulière de $\boldsymbol{\theta}$. Nous nous intéressons à une fonction $f(\boldsymbol{\theta})$ pour tous $\boldsymbol{\theta} \in \boldsymbol{\Upsilon} \subseteq \mathbb{R}^d$. Le sous-ensemble de configurations $\boldsymbol{\Upsilon}$ correspond souvent à une boîte rectangulaire.

Supposons maintenant que la simulation calcule une fonction $f(\boldsymbol{\theta}, \mathbf{U})$ où le vecteur $\mathbf{U}$ représente la séquence de nombres aléatoires utilisés pour la simulation et ne dépend pas de $\boldsymbol{\theta}$. Pour estimer par simulation la fonction

$$f(\boldsymbol{\theta}) = \int_{[0,1)^s} f(\boldsymbol{\theta}, \mathbf{U})\, d\mathbf{U},$$

nous générons une suite de points $\mathbf{U}_1, \ldots, \mathbf{U}_n$ indépendants et calculons la moyenne des $f(\boldsymbol{\theta}, \mathbf{U}_i)$. Cela revient à la méthode Monte Carlo classique, sauf que nous voulons estimer la fonction pour plusieurs valeurs de $\boldsymbol{\theta}$ en même temps. Examinons maintenant comment cette fonction $f(\boldsymbol{\theta}, \mathbf{U})$ dépend de $\boldsymbol{\theta}$ dans notre modèle.

Pendant la simulation, une suite de vecteurs aléatoires $\mathbf{Z}_1, \mathbf{Z}_2, \ldots$ est générée à des temps $T_1, T_2, \ldots$ possiblement aléatoires. Le vecteur $\mathbf{Z}_n \in \mathbb{R}^q$, qui est généré au temps T_n, permet de calculer une variable de décision $\eta_n = g(\mathbf{Z}_n, \boldsymbol{\theta})$, où $g : (\mathbb{R}^q, \boldsymbol{\Upsilon}) \to \mathbb{N}$ est une fonction qui peut dépendre de $\mathbf{Z}_n$ et de $\boldsymbol{\theta}$. Un exemple simple de fonction de décision couramment utilisée dans le cas unidimensionnel où $\mathbf{Z}_n = (Z_n)$ et $\boldsymbol{\theta} = (\theta)$ est $g(\mathbf{Z}_n, \boldsymbol{\theta}) = \mathbb{I}[Z_n \leq \theta]$.

Nous supposons que T_n et $\mathbf{Z}_n$ sont des fonctions de $\mathbf{U}$ et de $\eta_1, \ldots, \eta_{n-1}$ seulement,

c'est-à-dire qu'elles ne dépendent de $\boldsymbol{\theta}$ qu'à travers les variables de décision précédentes. Supposons également que $f(\boldsymbol{\theta}, \mathbf{U}) = f(\eta_1, \ldots, \eta_\tau, \mathbf{U})$ où $\tau = \max(n : T_n \leq T) < \infty$ est le nombre total (aléatoire) de points de décision. Ici, $T < \infty$ est un temps d'arrêt qui peut être déterministe ou aléatoire. La fonction $f(\boldsymbol{\theta}, \mathbf{U})$ ne dépend donc de $\boldsymbol{\theta}$ qu'à travers un nombre fini de variables de décision.

Nous pouvons bien entendu répéter tout le processus de simulation un certain nombre de fois indépendamment et calculer, pour chaque valeur de $\boldsymbol{\theta} \in \boldsymbol{\Upsilon}$, une moyenne sur toutes les répétitions ainsi qu'un intervalle de confiance. Nous appellerons ici *macro-réplication* ces répétitions de l'expérience pour ne pas confondre avec les copies parallèles simulées pendant chaque macro-réplication et sur lesquelles nous revenons dans la sous-section suivante.

6.1.2 Simulation de trajectoires multiples

Si nous simulons le modèle précédent avec le même vecteur $\mathbf{U}$ mais en utilisant plusieurs valeurs différentes de $\boldsymbol{\theta}$, nous obtenons un certain nombre de trajectoires semblables mais qui divergent sur certains intervalles de temps. Lorsque $\mathbf{U}$ est fixé, la trajectoire ne dépend que des variables de décision η_n. À n'importe quel instant t, nous pouvons dresser la liste de ces trajectoires et associer à chaque trajectoire j un sous-ensemble $\boldsymbol{\Upsilon}_j(t)$ de configurations pour lesquelles l'évolution du modèle sur l'intervalle de temps $[0,t]$ est identique. L'ensemble $\boldsymbol{\Upsilon}_j(t)$ contient toujours au moins une valeur de $\boldsymbol{\theta}$. En tout temps, $\{\boldsymbol{\Upsilon}_1(t), \ldots\}$ forme une partition de $\boldsymbol{\Upsilon}$.

Chaque trajectoire j est caractérisée par un sous-ensemble $\boldsymbol{\Upsilon}_j(t)$, un processus stochastique représentant l'évolution du système ainsi qu'une séquence de temps et de variables de décision qui sont mis à jour au cours de la simulation. Nous supposons que l'état au temps t peut être reconstitué à partir de ces informations. En pratique, pour sauver de la mémoire, seules des fonctions dépendant de ces informations sont conservées et mises à jour pendant la simulation.

Les trajectoires sont gérées et mises à jour par des copies parallèles. Une copie r est

ainsi caractérisée par un ensemble de trajectoires filles dont l'état $X_r(t) \in \mathcal{S}$ est identique au temps courant t. Ici, $\mathcal{S}$ est l'ensemble des états possibles du système. L'évolution passée du modèle varie bien entendu d'une trajectoire à l'autre. En tout temps, chaque copie doit contenir au moins une trajectoire et chaque trajectoire doit être gérée par une et une seule copie. De plus, tant qu'il n'y a pas de recombinaisons, chaque copie gère une seule trajectoire.

La figure 6.1 présente un exemple d'évolution de l'état d'un système par rapport au temps, avec des trajectoires et des copies multiples. Chaque courbe de la figure correspond à une trajectoire identifiée par une couleur. À tout instant t, les copies parallèles correspondent aux courbes disjointes sur la figure. Du texte sur chaque segment de courbe indique les trajectoires associées à chaque copie pendant ce segment.

Pour cet exemple, nous avons trois configurations si bien que $\boldsymbol{\Upsilon} = \{\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \boldsymbol{\theta}_3\}$. Avant le premier point de décision au temps T_1, il n'y a qu'une seule copie gérant une trajectoire associée à tout l'ensemble $\boldsymbol{\Upsilon}$. Rendu au temps T_1, l'évolution du système dépend de $\boldsymbol{\theta}$ si bien que la courbe rouge se sépare en deux trajectoires recevant des sous-ensembles différents de $\boldsymbol{\Upsilon}$. Un peu plus tard, au temps M_1, l'état du système devient identique pour les deux copies si bien que les deux copies se recombinent et nous avons une seule copie gérant deux trajectoires. La copie se sépare à nouveau au temps T_2, mais les trajectoires ne sont que déplacées vers les copies filles, pas clonées comme au temps T_1. Par contre, au temps T_3, une trajectoire se voit clonée si bien que la courbe verte se sépare elle aussi et nous avons trois copies, avec une trajectoire par copie. Plus tard, les trois copies se recombinent en une seule si bien qu'à la fin de la simulation, nous avons une copie gérant les trois trajectoires.

6.1.3 Algorithme de scission

Examinons maintenant comment s'opère une scission en général. Initialement, il y a une seule copie gérant une trajectoire à laquelle $\boldsymbol{\Upsilon}$ est associé. Chacune des copies courantes est simulée en parallèle jusqu'à ce qu'une copie r produise une valeur de $\mathbf{Z}_n$

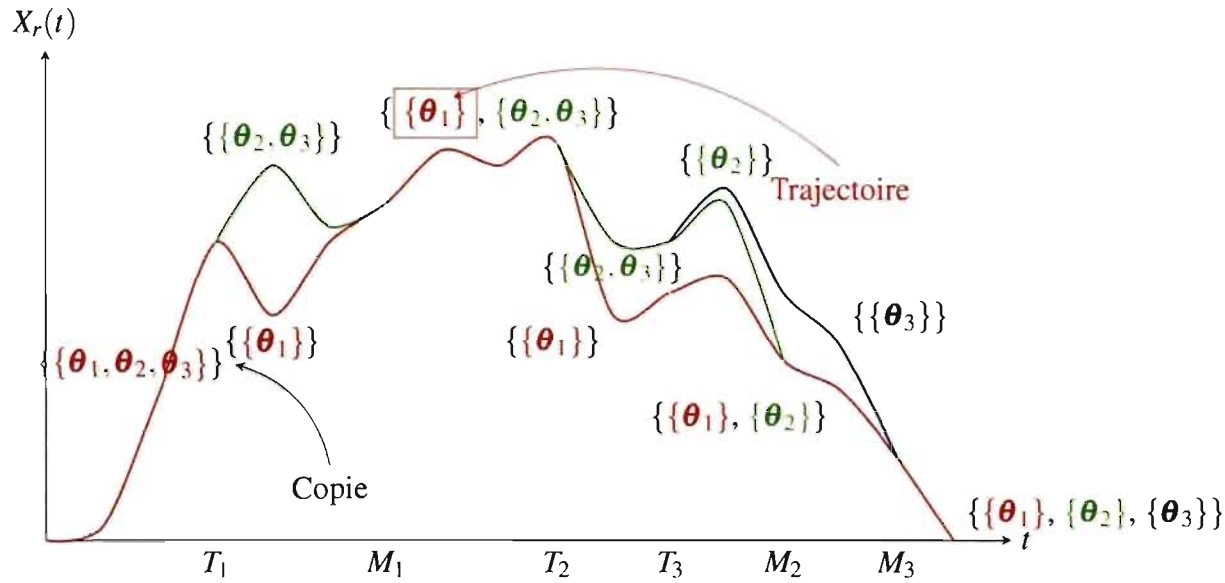

Figure 6.1 – Exemple de simulation avec des trajectoires parallèles

au temps T_n.

L'évolution du système après le temps T_n dépend de la valeur de la variable de décision η_n qui est une fonction de $\boldsymbol{\theta}$ et de $\mathbf{Z}_n$. La copie r ayant généré $\mathbf{Z}_n$ peut ainsi produire autant de copies filles qu'il y a de valeurs possibles pour $g(\mathbf{Z}_n, \boldsymbol{\theta})$. Le nombre maximal de copies qui peuvent alors être produites par une scission ne peut pas excéder la cardinalité ℓ de l'image de g.

Maintenant, soit $\boldsymbol{\Upsilon}_g(\mathbf{Z}_n, \eta) = \{\boldsymbol{\theta} \in \boldsymbol{\Upsilon} : g(\mathbf{Z}_n, \boldsymbol{\theta}) = \eta\}$ l'ensemble des valeurs de $\boldsymbol{\theta}$ pour lesquelles nous obtenons une même décision $\eta_n = \eta$ si $\mathbf{Z}_n$ est fixé. Par exemple, avec la fonction $\mathbb{I}[Z_n \leq \theta]$ pour le cas unidimensionnel, les ensembles $\boldsymbol{\Upsilon}_g(\mathbf{Z}_n, 0) = \boldsymbol{\Upsilon} \cap (-\infty, Z_n)$ et $\boldsymbol{\Upsilon}_g(\mathbf{Z}_n, 1) = \boldsymbol{\Upsilon} \cap [Z_n, \infty)$ correspondent à l'intersection de $\boldsymbol{\Upsilon}$ (qui est unidimensionnel dans ce cas) avec un intervalle ayant Z_n comme extrémité.

Au temps T_n, nous devons remplacer la copie r par au plus ℓ copies et distribuer les trajectoires gérées originellement par r dans ces copies parallèles. Plus précisément, pour toutes les trajectoires j gérées par r, nous devons répertorier toutes les valeurs possibles de η_n pouvant être obtenue avec $g(\mathbf{Z}_n, \cdot)$. Pour chacune de ces valeurs, nous

pouvons calculer l'intersection $\mathcal{I} = \Upsilon_j(T_n) \cap \Upsilon_g(\mathbf{Z}_n, \eta)$ qui donne l'ensemble des valeurs de $\boldsymbol{\theta}$ appartenant à la trajectoire j et engendrant une décision $\eta_n = \eta$ au temps T_n. Bien entendu, la fonction $g(\mathbf{Z}_n, \boldsymbol{\theta})$, la détermination des valeurs qu'elle peut prendre pour une valeur de $\mathbf{Z}_n$ donnée ainsi que le calcul efficace des intersections $\mathcal{I}$ dépendent de l'application à laquelle la méthode est adaptée.

Si $\mathcal{I}$ est vide, aucune configuration associée à la trajectoire j ne permet de générer $\eta_n = \eta$ si bien qu'aucune copie de la trajectoire ne va dans la copie associée à η. D'un autre côté, si $\mathcal{I}$ correspond à tout l'ensemble $\Upsilon_j(T_n)$ des configurations pour la trajectoire j, la trajectoire j est déplacée vers la copie pour laquelle $\eta_n = \eta$. Pour tout autre cas intermédiaire, la trajectoire j est dupliquée autant de fois qu'il y a de sous-ensembles non vides $\Upsilon_j(T_n) \cap \Upsilon_g(\mathbf{Z}_n, \eta)$ et les clones obtenus sont placés dans les copies filles appropriées.

De cette façon, chaque clone de la trajectoire originale se voit associé un sous-ensemble différent de $\Upsilon_j(T_n)$. Toute copie ne comportant aucune trajectoire associée à la fin de ce processus se voit éliminée. Chaque copie restante évolue ensuite indépendamment et peut se scinder à son tour.

La figure 6.2 illustre le processus précédent par un exemple avec trois trajectoires. La copie mère est représentée par le rectangle de gauche tandis que les copies filles correspondent à deux rectangles à droite. Chaque trajectoire est représentée par un sous-ensemble de Υ et affichée sur une ligne des rectangles représentant les copies. Nous voyons sur la figure que la première trajectoire, colorée en rouge, ainsi que la troisième colorée en bleue, se déplacent vers des copies filles. Par contre, la seconde trajectoire, colorée en vert, se voit dupliquée afin de se trouver dans chaque copie fille.

Une copie parallèle est terminée quand le temps d'arrêt est atteint pour toutes ses trajectoires filles. La simulation se termine lorsque plus aucune copie parallèle n'est en cours.

Si Υ est un ensemble fini, le nombre maximal de trajectoires et donc de copies parallèles est clairement $|\Upsilon|$. Le nombre de trajectoires au n^e point de décision est borné par

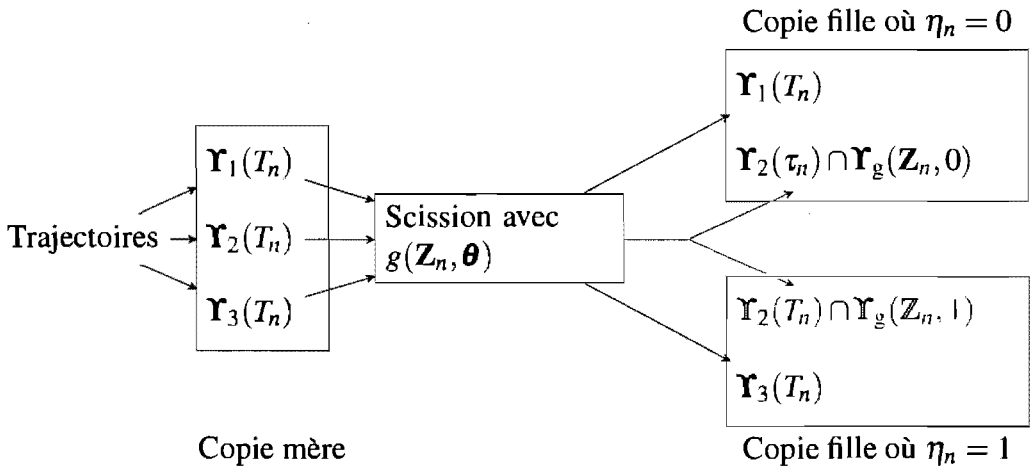

Figure 6.2 – Diagramme illustrant un exemple du processus de scission

ℓ^{n-1} étant donné que la trajectoire dépend de $n-1$ variables de décision pouvant prendre chacune au plus ℓ valeurs. Par contre, il arrive souvent que le nombre de scissions possibles à la décision n soit constant. Dans ce cas, la taille de l'arbre de simulation croît au pire de façon linéaire avec n.

6.1.4 Prise en charge de la recombinaison de copies

Si, à un temps t, l'état du système est le même pour deux copies parallèles r_1 et r_2, une recombinaison peut se produire, c'est-à-dire que toutes les trajectoires associées à r_2 sont déplacées vers r_1 et la copie r_2 est éliminée. Permettre de telles recombinaisons de copies demande d'effectuer des tests régulièrement pendant la simulation. Lors d'un tel test, il faut idéalement que toutes les copies soient synchronisées, c'est-à-dire que l'horloge soit au même temps t pour chacune des copies. Sans cette synchronisation, il faut vérifier que le temps courant est le même avant de recombiner deux copies.

La solution la plus simple pour que les copies soient toujours synchronisées au moment des tests consiste à faire la vérification à des temps fixes, à intervalles réguliers. Par contre, il arrive souvent que rien ne se produise entre les vérifications si bien que les tests sont effectuées inutilement. Pour éviter cela, nous pouvons tester pour une re-

combinaison seulement après certains événements stratégiques de la simulation. Dans un modèle de centre de contacts, par exemple, le test peut survenir après chaque $x \in \mathbb{N}$ fins de service ; souvent, $x = 1$.

La technique naïve pour déterminer quelles copies recombiner à un temps t consiste à énumérer chaque paire $(X_{r_1}(t), X_{r_2}(t))$ d'états, à rejeter celles pour lesquelles r_1 ou r_2 n'existent plus et à recombiner celles pour lesquelles $X_{r_1}(t) = X_{r_2}(t)$. Cette technique n'est acceptable que si le nombre de copies est très petit puisqu'elle impose un temps d'exécution quadratique en fonction du nombre de copies.

En pratique, les paires à vérifier doivent être choisies plus judicieusement. Par exemple, à chaque copie peut être associée une boîte rectangulaire regroupant toutes les valeurs de $\boldsymbol{\theta}$ des trajectoires associées. Une test pour la recombinaison n'a alors lieu que pour les copies dont les boîtes rectangulaires sont adjacentes. L'organisation des boîtes rectangulaires pour que les tests d'adjacence soient faciles dépend de l'application particulière à laquelle est adaptée la méthode.

6.2 Application à un centre de contacts

Appliquons maintenant la technique décrite précédemment pour estimer la performance d'un centre de contacts en fonction de différents paramètres. Pour cela, nous employons le modèle de la section 5.2. Ce modèle consiste en une CMTC uniformisable que nous simulons en utilisant la méthode de conversion en temps discret, qui consiste à simuler la CMTD imbriquée seulement et calculer des espérances conditionnelles à la séquence des états visités.

Une copie de la CMTC est associée à chaque copie parallèle en cours de simulation. Pour simplifier la notation, nous supposons qu'un vecteur $\mathbf{Z}_n$ est généré à chaque étape de la CMTD imbriquée et utilisé pour calculer $\eta_n = g(\mathbf{Z}_n, \boldsymbol{\theta})$. Le temps est mesuré en nombres de transitions et $T_n = n$ pour $n = 1, \ldots$. En général, il est possible de générer $\mathbf{Z}_n$ ou de définir g de telle sorte que $g(\mathbf{Z}_n, \cdot) = x$ peu importe $\boldsymbol{\theta}$ lorsqu'aucune décision n'est

prise à l'étape n. Par exemple, dans le cas unidimensionnel, si $g(\mathbf{Z}_n, \boldsymbol{\theta}) = \mathbb{I}[Z_n \leq \theta]$, nous pouvons fixer $Z_n = \infty$ lorsqu'une décision impliquant θ n'est pas nécessaire. De cette façon, n correspond au numéro de la transition comme dans le chapitre précédent.

De plus, comme l'explique [80], si nous faisons avancer toutes les copies de la CMTD imbriquée d'une transition à chaque étape, toutes les copies parallèles sont synchronisées dans le temps. Cela simplifie légèrement les tests d'égalité d'états lors des recombinaisons.

Nous avons essayé d'utiliser un modèle plus général et la simulation par événements discrets, mais le clonage et la comparaison de l'état dans notre implantation du simulateur devenaient si complexes que la méthode était inefficace. Par contre, en simplifiant suffisamment l'état et les événements, nous pourrions sans doute appliquer la méthode à un modèle simulé par événements discrets.

Nous traitons d'abord des paramètres continus tels que les taux d'arrivée, de fin de service et d'abandon. Ensuite, nous examinons le cas où le nombre d'agents change dans un seul groupe, puis considérons celui où il peut varier dans plusieurs groupes. Malheureusement, dans ce dernier cas, le nombre de configurations à tester est habituellement gigantesque. Nous examinons alors comment la méthode appliquée pour estimer un sous-gradient réduit substantiellement le nombre de configurations. Nous discutons enfin des problèmes qui surviennent lorsque le vecteur de paramètres $\boldsymbol{\theta}$ peut varier dans le temps.

Les premières sous-sections concernent le cas unidimensionnel de la méthode, pour lequel $\boldsymbol{\theta} = (\theta)$ et $\mathbf{Z}_n = (Z_n)$. Nous allons dans ces cas utiliser les scalaires au lieu des vecteurs.

6.2.1 Variation d'un seul paramètre continu

Dans le cas où un seul paramètre varie de façon continue, nous avons une application directe de la méthode décrite dans [68] à condition que le sous-intervalle de $[0, 1)$ associé à l'événement concerné par le paramètre ne change pas en fonction du paramètre. Nous

avons traité ce problème dans la section 5.2 afin de favoriser une bonne synchronisation avec les variables aléatoires communes.

Supposons d'abord que le taux d'arrivée des contacts d'un type l est $\theta \in [a,b]$. Pour appliquer la scission, nous générons les arrivées de type l avec le taux b le plus élevé et filtrons les événements en fonction de θ. Cela ressemble à la méthode de découpage permettant de générer un processus de Poisson avec un taux variant de façon continue dans le temps. Plus précisément, nous uniformisons la CMTC en utilisant $\tilde{q} = \sum_{k=1}^{K} \tilde{\lambda}_k + \sum_{i=1}^{I} \mu_i N_i + H\nu$ où $\tilde{\lambda}_k = \lambda_k$ pour $k \neq l$ est le taux d'arrivée des contacts de type k, $\tilde{\lambda}_l = b$, μ_i est le taux maximal de fin de service pour les agents du groupe i, N_i est le nombre d'agents dans le groupe i, ν est le taux d'abandon maximal et H est la taille maximale de la file. À chaque étape n, nous générons U_n depuis la loi uniforme. Si $U_n < \sum_{k=1}^{l-1} \tilde{\lambda}_k/\tilde{q}$ ou $U_n \geq \sum_{k=1}^{l} \tilde{\lambda}_k/\tilde{q}$, l'événement produit ne correspond pas à une arrivée de type l si bien qu'aucune décision dépendant de θ n'est prise. Dans le cas contraire, $Z_n = \tilde{q}U_n - \sum_{k=1}^{l-1} \tilde{\lambda}_k$ est uniformément distribué sur $[0, \tilde{\lambda}_l)$. Si $Z_n \leq a$, une arrivée de type l survient tandis que si $Z_n > b$, nous observons une transition fictive.

Si $Z_n \in [a,b]$, l'événement qui se produit dépend de θ si bien qu'une scission a lieu. À chaque décision binaire de ce type, l'intervalle $[a,b]$ correspondant à la trajectoire concernée se voit scindé en deux : un sous-intervalle $[a,Z_n)$ pour lequel l'arrivée n'a pas lieu et un autre sous-intervalle $[Z_n,b]$ pour lequel l'arrivée a lieu. Ceci revient à appliquer la méthode exposée précédemment en utilisant $g(\mathbf{Z}_n, \boldsymbol{\theta}) = \mathbb{I}[Z_n \leq \theta]$. Étant donné que les sous-intervalles associés aux trajectoires forment une partition de $[a,b]$, Z_n ne fait partie que d'un seul sous-intervalle si bien qu'une seule trajectoire se voit clonée. Dans le pire cas, le nombre de copies parallèles croît donc de façon linéaire avec le nombre d'arrivées.

Parfois, nous souhaitons faire varier le taux d'arrivée global sans changer la distribution du type k conditionnelle à une arrivée. Nous pouvons alors définir θ comme un multiplicateur du taux d'arrivée et fixer $\tilde{q} = b\sum_{k=1}^{K} \lambda_k + \sum_{i=1}^{I} \mu_i N_i + H\nu$. Si $b\sum_{k=1}^{l-1} \lambda_k/\tilde{q} \leq U_n < b\sum_{k=1}^{l} \lambda_k/\tilde{q}$, l'événement correspond à une arrivée potentielle de type l. La va-

riable de décision $Z_n = (\tilde{q}U_n - b\sum_{k=1}^{l-1}\lambda_k)/\lambda_l$, uniformément distribuée sur $[0,b)$, permet alors de déterminer si nous avons affaire à une arrivée ou à une transition fictive.

Nous pouvons appliquer ces mêmes idées pour d'autres paramètres continus comme $\mu_{k,i}$, ν_k, etc. Il suffit que la variable de décision binaire permette de choisir entre un événement et une transition fictive.

6.2.2 Variation d'un seul paramètre agissant sur le nombre d'agents

Supposons que nous souhaitons évaluer la performance du centre de contacts lorsque le nombre d'agents dans le groupe i est fixé à θ, pour $\theta \in \{a,\ldots,b\}$, avec $0 \leq a \leq b < \infty$. Nous simulons toutes les configurations avec le taux de transition $\tilde{q} = \sum_{k=1}^{K}\lambda_k + \sum_{i=1}^{I}\mu_i\tilde{N}_i + H\nu$, où $\tilde{N}_l = N_l$ pour $l \neq i$ et $\tilde{N}_i = b$. Ce cas est plus complexe que les précédents, car θ permet de choisir entre deux types d'événements réels : le début du service par un agent du groupe i ou le routage du contact ailleurs dans le système (vers un agent d'un autre groupe ou vers la file d'attente). Comme dans la sous-section précédente, il est important que le sous-intervalle de $[0,1)$ associé à un certain type d'événement ne change pas avec θ.

Comme nous l'avons vu dans la section 5.2, lorsqu'un contact de type k arrive, une liste d'ensembles de groupes d'agents, $I_{k,1}, I_{k,2}, \ldots$, est parcourue, et un agent libre du premier sous-ensemble de groupes de cette liste est sélectionné. Si le sous-ensemble ainsi choisi comprend plusieurs groupes, le groupe contenant le plus grand nombre d'agents libres est choisi.

Pour qu'il y ait une décision impliquant θ, il faut au moins qu'une arrivée de type $k \in S_i$ se produise à un moment où aucun contact de type k n'est en attente. Il faut aussi qu'aucun agent ne soit disponible dans les groupes précédant i dans la liste de routage pour les contacts de type k.

Si le comportement dépend de θ, le contact peut être acheminé vers un agent du groupe i, vers un agent d'un autre groupe que i ou encore vers la file d'attente. Par contre, la destination du contact non servi par un agent du groupe i ne dépend que des

variables de décision observées dans le passé, pas du nombre exact d'agents occupés dans le groupe i. Nous pouvons alors utiliser $g(\mathbf{Z}_n, \boldsymbol{\theta}) = \mathbb{I}[Z_n \leq \theta]$ en fixant $Z_n = B_i + F_i$ où $B_i = \sum_{k=1}^{K} S_{k,i}$ est le nombre d'agents occupés dans le groupe i et F_i est le nombre minimal d'agents libres nécessaires dans le groupe i pour que le nouveau contact y soit envoyé. Les deux quantités B_i et F_i dépendent de l'état de la CMTC et donc du numéro de transition n.

Déterminons maintenant la valeur de F_i. Si un contact de type k peut être envoyé vers un agent du groupe i, il existe un indice j tel que $i \in I_{k,j}$. Si $I_{k,j} = \{i\}$, $F_i = 1$ puisqu'aucun autre groupe de même niveau de préférence ne compétitionne avec i si bien qu'il suffit qu'un seul agent soit libre pour accepter le contact. En général,

$$F_i = \max(1, \max_{l<i, l \in I_{k,j}} (N_l - B_l + 1), \max_{l>i, l \in I_{k,j}} (N_l - B_l)).$$

Cela correspond au maximum entre 1 et le nombre d'agents disponibles dans tous les groupes de $I_{k,j}$ autres que i. Nous devons distinguer les groupes d'indice inférieur et supérieur à i dans la formule étant donné que si le groupe i contient le même nombre d'agents libres qu'un groupe $l < i$ de même niveau de préférence, le groupe l sera choisi au lieu du groupe i. Toutes ces informations font partie de l'état Y_n et des paramètres du modèle. Ainsi, Z_n peut être calculé à partir de l'état et ne dépend de θ qu'à travers les variables de décision précédentes.

La variable de décision η_n prend la valeur 1 si le groupe i accepte le nouveau contact et 0 si tel n'est pas le cas. Dans le premier cas, le contact est servi immédiatement par un agent du groupe i tandis que dans le second, le contact peut aller dans la file d'attente ou être servi par un agent d'un autre groupe, selon les paramètres de routage et la disponibilité des agents.

Aucune scission n'a lieu tant que le nombre d'agents occupés dans le groupe i est inférieur à a. À partir du moment où $a < Z_n \leq b$, la trajectoire courante est clonée : une copie considère l'intervalle $[a, Z_n)$ et l'autre, l'intervalle $[Z_n, b]$. Dans la première trajec-

toire, le nombre d'agents est insuffisant pour servir le nouveau contact immédiatement ($\eta_n = 0$) tandis que le nombre est suffisant ($\eta_n = 1$) dans la seconde trajectoire.

Comme dans la sous-section précédente, chaque fois qu'une valeur de Z_n est générée, une seule trajectoire est clonée s'il en existe plusieurs. Pour toutes les autres trajectoires, nous avons $\theta < Z_n$ ou $\theta > Z_n$ pour toutes les configurations. Dans le pire cas, le nombre de trajectoires croît donc de façon linéaire par rapport au nombre d'arrivées jusqu'à atteindre $|\Upsilon| = b - a + 1$. En pratique, le taux de croissance est plus petit puisque seule une fraction des arrivées correspondent à des points de scission.

Étant donné qu'un seul événement se produit à la fois et n'ajoute ou ne retranche 1 qu'à au plus une valeur de $S_{k,i}$ à la fois, Z_n varie par incréments de 1 d'un point de décision à l'autre. Alors, l'ensemble de valeurs de θ associé à toute trajectoire résultant d'un clonage a une cardinalité de 1 si bien que la trajectoire ne peut pas être clonée à nouveau. Ainsi, à tout instant, il n'existe qu'une seule trajectoire qui peut être clonée ; toutes les autres se déplacent simplement entre les copies parallèles. De plus, le nombre maximal de trajectoires ne peut pas excéder $b - a + 1$, le nombre maximal de configurations possibles avec ce modèle.

6.2.3 Problèmes posés par les recombinaisons

Lorsqu'une fin de service ou un abandon surviennent, il se peut que l'état du système dans une trajectoire donnée corresponde à l'état dans une autre trajectoire simulant avec une valeur différente de θ. Pour les deux applications précédentes, un sous-intervalle de $[a, b]$ est associé à chaque trajectoire et à chaque copie parallèle. Nous ne considérons pour la recombinaison que des paires de copies pour lesquelles l'intervalle associé est adjacent. Étant donné que les intervalles ne se chevauchent pas, ils peuvent être stockés en ordre croissant de façon à ce que le nombre de tests à effectuer corresponde au nombre de copies parallèles.

Mais comparer l'état exige ici de tester toutes les valeurs de $S_{k,i}$ et de Q_k, ce qui se fait dans $\mathcal{O}(KI)$. La comparaison peut donc être très coûteuse s'il y a beaucoup de

types de contacts ou de groupes d'agents. Pour réduire ce coût, nous n'effectuons une comparaison dans $\mathcal{O}(KI)$ qu'après avoir testé l'égalité d'une fonction de hachage $h(Y_n)$. Ce test préliminaire est très rapide si la fonction peut être calculée en temps constant pour n'importe quel état et peut éviter la comparaison exhaustive de l'état dans certains cas.

La fonction utilisée est $h(Y_n) = \sum_{i=1}^{I} B_i w_i + \left(\sum_{k=1}^{K} Q_k\right) w_{I+1}$, où $w_i = \prod_{j=1}^{i-1} \tilde{N}_j$ pour $i = 1, \ldots, I+1$. À condition de calculer et maintenir à jour les pondérations w_i, les B_i et la taille de file $\sum_{k=1}^{K} Q_k$, il est possible d'effectuer des mises à jour de h en temps constant après chaque transition.

6.2.4 Variation de plusieurs paramètres agissant sur le nombre d'agents

Examinons maintenant comment évaluer la performance du système en fonction du nombre d'agents dans chacun des I groupes. Pour ce cas plus complexe, nous simplifions la politique de routage en supposant que $I_{k,j} = \{i_{k,j}\}$ pour tous k, j, c'est-à-dire que pour tout nouveau contact de type k, une liste fixe de groupes d'agents est parcourue et le contact est envoyé vers le premier groupe de la liste comportant des agents libres.

Appelons $\boldsymbol{\theta} \in \mathbb{N}^I$ le vecteur d'agents et supposons que nous nous intéressons à un sous-ensemble $\boldsymbol{\Upsilon} \subset \mathbb{N}^I$ correspondant à une boîte rectangulaire. Nous supposons que pour toute trajectoire j et pour tout temps t, le sous-ensemble $\boldsymbol{\Upsilon}_j(t)$ de configurations associées est aussi une boîte rectangulaire $\{(\theta_1, \ldots, \theta_I) : \theta_i \in \{a_{j,i}(t), \ldots, b_{j,i}(t)\} \forall i = 1, \ldots, I\}$. Pour toute trajectoire j, nous pouvons ainsi représenter l'ensemble des configurations simplement par deux vecteurs de taille I, $(a_{j,1}(t), \ldots, a_{j,I}(t))$ et $(b_{j,1}(t), \ldots, b_{j,I}(t))$, correspondant aux deux coins opposés de la boîte.

Lors d'une arrivée de type k où $Q_k = 0$, nous devons choisir un groupe d'agents i pour le nouveau contact ou encore décider de le placer dans la file d'attente. Pour prendre cette décision, nous avons besoin de I variables, à savoir le nombre d'agents occupés B_i dans chacun des groupes $i = 1, \ldots, I$; nous avons alors $\mathbf{Z}_n = (B_1, \ldots, B_I)$. La fonction $g(\mathbf{Z}_n, \boldsymbol{\theta})$ retourne un nombre entre 1 et $I+1$ représentant la destination du contact : une valeur

entre 1 et I représente un groupe d'agents tandis que la valeur $I+1$ est réservée pour la file d'attente.

Nous pouvons, pour n'importe quelle copie, construire une boîte englobant toutes les boîtes $\Upsilon_j(t)$ des trajectoires. Supposons que pour la copie r, $\theta_i \in \{a_i, \ldots, b_i\}$ pour $i = 1, \ldots, I$. Si, pour tous $i = 1, \ldots, I$, $B_i + 1 \leq a_i$ ou $B_i + 1 > b_i$, l'événement suivant est le même peu importe la valeur de $\boldsymbol{\theta}$ à l'intérieur de la boîte locale à r ; le clonage et le déplacement de trajectoires ne surviennent pas.

Dans le cas contraire, nous devons appliquer la méthode de scission ébauchée dans la section précédente. Pour chaque trajectoire j gérée par la copie r ayant généré l'arrivée de type k, les valeurs possibles de η_n sont répertoriées en même temps que la politique de routage est appliquée, en utilisant la procédure suivante.

1. Pour tout groupe d'agents i de la liste $i_{k,1}, i_{k,2}, \ldots$,

   (a) Si $B_i + 1 > b_{j,i}(T_n)$, nous passons au groupe d'agents suivant dans la liste. Cette condition signifie que pour la trajectoire j à l'étape n, il n'existe aucune configuration $\boldsymbol{\theta} \in \Upsilon_j(T_n)$ pour laquelle le contact peut être envoyé vers un agent libre du groupe i.

   (b) Si $B_i + 1 \leq a_{j,i}(T_n)$, la trajectoire j devient j_i dans laquelle le contact est envoyé à un agent du groupe i, et est ajoutée à la copie pour laquelle $\eta_n = i$. Nous arrêtons ensuite la procédure puisque dans ce cas, pour la trajectoire j à l'étape n, le contact sera toujours envoyé vers un agent du groupe i peu importe la configuration $\boldsymbol{\theta} \in \Upsilon_j(T_n)$.

   (c) Sinon, la trajectoire j est scindée : un clone j_i associé à une copie pour laquelle $\eta_n = i$, c'est-à-dire pour laquelle le contact est envoyé vers un agent du groupe i, et l'original j qui poursuit la procédure de routage pour trouver une autre destination pour le contact. La trajectoire j_i est ajoutée à la copie pour laquelle $\eta_n = i$.

2. Si la procédure n'a pas été interrompue plus tôt, la trajectoire j devient j_{I+1} dans

laquelle le contact va en file d'attente. Cette trajectoire j_{I+1} est ajoutée à la copie pour laquelle $\eta_n = I + 1$.

La figure 6.3 présente un exemple d'application de la procédure précédente pour lequel le nombre d'agents dans le premier groupe, $i_{k,1}$, est insuffisant quelle que soit la valeur de $\boldsymbol{\theta} \in \boldsymbol{\Upsilon}_j(T_n)$ tandis qu'il y a suffisamment d'agents dans les deux autres groupes visités pour certaines configurations seulement.

Examinons maintenant comment l'ensemble de configurations $\boldsymbol{\Upsilon}_j(T_n)$ est partitionné à chaque répétition de l'étape 1c de la procédure précédente. Pour la trajectoire j_i, qui est créée pour le cas où le nouveau contact est servi dès son arrivée par un agent du groupe i, nous devons retenir les valeurs de $\boldsymbol{\theta} \in \boldsymbol{\Upsilon}_j(T_n)$ pour lesquelles il y a suffisamment d'agents dans le groupe i pour servir un nouveau contact à l'étape n. Ainsi, le sous-ensemble $\boldsymbol{\Upsilon}_{j_i}(T_n)$ est construit de façon à ce que $\theta_i \in \{B_i + 1, \ldots, b_{j,i}(T_n)\}$ et $\theta_l \in \{a_{j,l}(T_n), \ldots, b_{j,l}(T_n)\}$ pour $l \neq i$. Les configurations restantes dans la trajectoire j correspondent toutes à des cas où il n'y a pas suffisamment d'agents dans le groupe i pour servir le nouveau contact à l'étape n. Ainsi, pour la trajectoire j qui visite les autres groupes dans la liste de routage, le sous-ensemble de vecteurs devient $\boldsymbol{\Upsilon}_j(T_n) - \boldsymbol{\Upsilon}_{j_i}(T_n)$, avec en particulier $\theta_i \in \{a_{j,i}(T_n), \ldots, B_i\}$.

La figure 6.4 montre un exemple de séparation d'une boîte rectangulaire $\boldsymbol{\Upsilon}$ dans le cas bidimensionnel où $\boldsymbol{\theta} \in \mathbb{N}^2$ (deux groupes d'agents), sous la forme d'un arbre dont chaque nœud correspond à un sous-ensemble $\boldsymbol{\Upsilon}_j(t)$, représenté par une boîte rectangulaire avec un minimum et un maximum d'agents pour chaque groupe. Chaque lien indique une scission par rapport à un groupe d'agents ou la file d'attente. Chaque niveau de l'arbre représente un point de décision. La figure n'illustre qu'une partie de l'arbre de scission, dont chaque feuille représenterait une des configurations testées.

Nous supposons pour cet exemple que tout nouveau contact est envoyé vers un agent libre du groupe 1 ou vers un agent libre du groupe 2 si tous les membres du groupe 1 sont occupés. Le nombre d'agents dans cet exemple peut varier entre 20 et 23 pour le premier groupe et entre 35 et 38 pour le second. Cela nous donne un total de seize configurations.

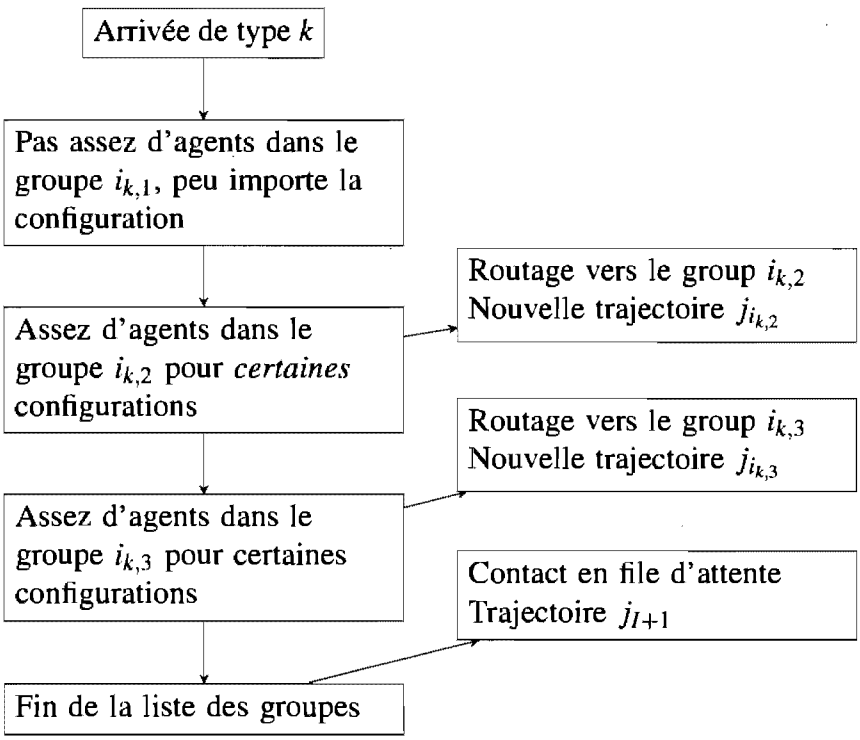

Figure 6.3 – Exemple d'application de la scission lors de l'arrivée d'un contact

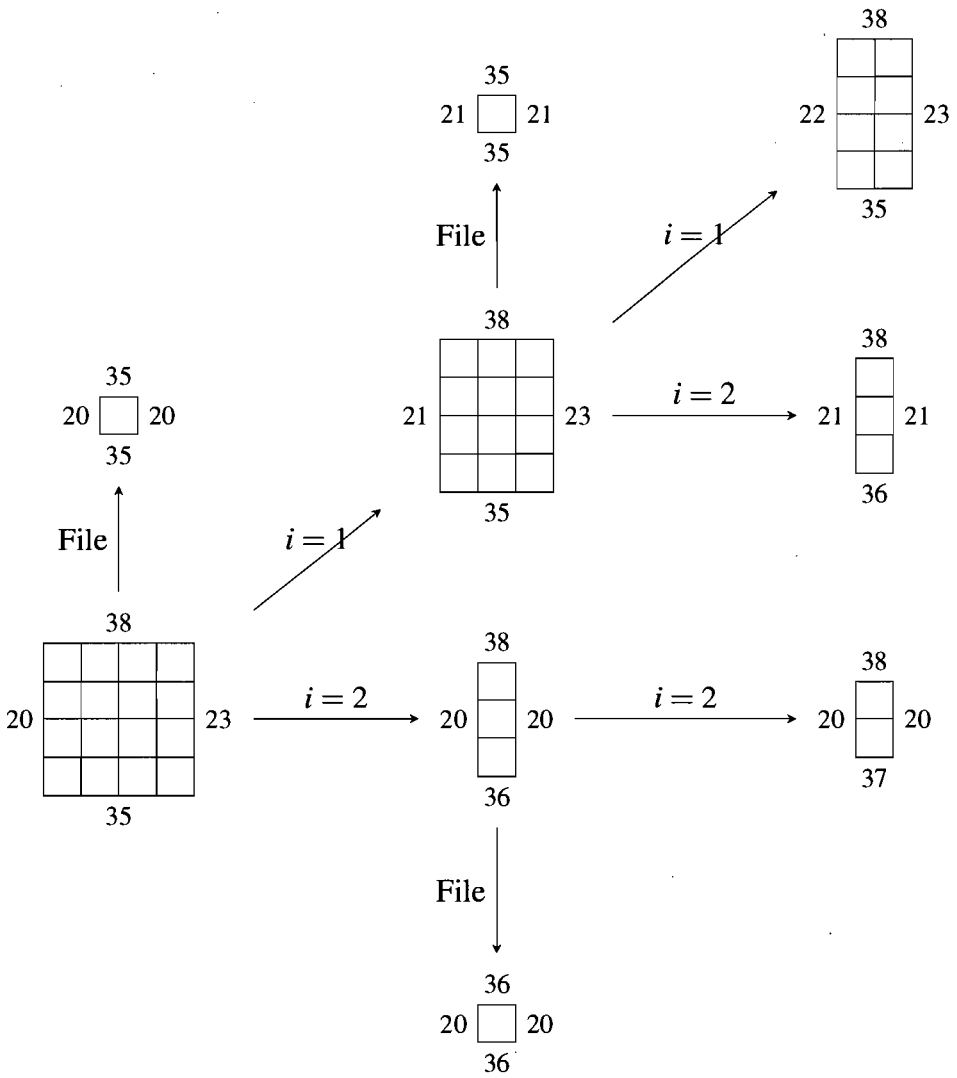

Figure 6.4 – Exemple d'arbre de scission dans un cas bidimensionnel

Supposons qu'un nouveau contact arrive à un moment où le nombre d'agents occupés dans les groupes est $B_1 = 20$ et $B_2 = 35$. Alors, la méthode crée trois sous-boîtes différentes : une pour laquelle il y a toujours assez d'agents dans le groupe 1 pour servir le contact, une pour laquelle le nombre d'agents est insuffisant dans le groupe 1 mais suffisant dans le groupe 2 et une pour laquelle le contact va en file puisqu'aucun des deux groupes ne contient d'agents libres. Ceci est illustré sur la figure par le fait que le nœud racine, en bas à gauche, possède trois enfants.

À chaque sous-ensemble de configurations est associé une trajectoire qui peut être scindée à nouveau lors d'une arrivée subséquente. Ceci est représenté sur la figure par le fait que les enfants du nœud racine possèdent à leur tour des enfants.

Supposons maintenant qu'il existe un certain indice j tel que $|I_{k,j}| > 1$. Alors, pour que le contact soit envoyé vers un agent du groupe $i \in I_{k,j}$, il faut, en plus des conditions précédentes, que $\theta_l - B_l$ soit inférieur à $\theta_i - B_i$ pour $l \in I_{k,r} \backslash i$. Si nous appliquons cette dernière contrainte pour former le sous-ensemble de configurations $\Upsilon_{j_i}(T_n)$ d'une trajectoire fille j_i, le sous-ensemble résultant ne correspond plus à une boîte rectangulaire si bien que sa représentation en mémoire est plus complexe. C'est pourquoi nous nous sommes concentrés sur les cas où $|I_{k,j}| = 1$.

6.2.5 Estimation de sous-gradients

La technique de simulation précédente permet en théorie de traiter simultanément $(b_1 - a_1 + 1) \cdots (b_I - a_I + 1)$ configurations. Mais si I est élevé, le nombre de configurations peut être énorme. Les algorithmes d'optimisation n'explorent pas toutes ces configurations, se concentrant plutôt sur les plus intéressantes d'entre elles. La méthode risque donc de simuler plusieurs configurations et d'être très coûteuse inutilement. Un autre problème est qu'il n'existe pas de façon évidente d'ordonner les boîtes rectangulaires associées aux copies afin de décider rapidement quelles boîtes sont adjacentes et candidates pour le test d'égalité d'états.

Plusieurs méthodes d'optimisation utilisent des estimés du sous-gradient pour orien-

ter la recherche de solutions. Nous pouvons en particulier estimer le sous-gradient par différences finies. Dans ce cas, la coordonnée i de l'estimateur est

$$q_i(\boldsymbol{\theta}) = \frac{f(\boldsymbol{\theta} + d_i \mathbf{e}_i) - f(\boldsymbol{\theta})}{d_i}$$

où d_i est une constante souvent égale à 1 et $\mathbf{e}_i$ est un vecteur contenant 1 pour la composante i et 0 pour toute autre composante.

Habituellement, cette estimation demande $I+1$ simulations avec des paramètres légèrement différents : une simulation dite de base destinée à estimer $f(\boldsymbol{\theta})$ et une simulation pour chaque groupe d'agents i, dans le but d'estimer $f(\boldsymbol{\theta} + d_i \mathbf{e}_i)$. Parfois, ajouter un agent dans le groupe i a peu d'effet sur la performance du système si bien que $f(\boldsymbol{\theta} + \mathbf{e}_i)$ est très près de $f(\boldsymbol{\theta})$. Pour obtenir un estimé du sous-gradient non nul, il faut alors refaire l'expérimentation avec une valeur $d_i > 1$. Il peut donc être intéressant dans certains cas de simuler simultanément pour plusieurs valeurs de d_i afin de trouver la bonne valeur plus rapidement. Ainsi, nous fixons $\boldsymbol{\Upsilon} = \cup_{i=1}^{I} \cup_{s=0}^{d_i} (\boldsymbol{\theta} + s\mathbf{e}_i)$. Par contre, nous nous intéressons habituellement davantage aux configurations où $s = 0$ ou $s = d_i$.

Nous pourrions certes utiliser la méthode précédente telle quelle pour faire ces estimations, en fixant $b_i = a_i + d_i$. Par contre, cela simulerait $\prod_{i=1}^{I}(d_i + 1)$ configurations tandis que nous n'avons besoin d'en simuler que $1 + \sum_{i=1}^{I} d_i$. Il est heureusement facile d'adapter la méthode précédente pour estimer des sous-gradients seulement. Il suffit pour cela de changer comment les ensembles $\boldsymbol{\Upsilon}_{j_i}(T_n)$ sont créés pour les trajectoires filles j_i.

La trajectoire j_i, créée à partir de la trajectoire de base, correspond aux configurations $\boldsymbol{\theta} + s\mathbf{e}_i$, pour $s = 1, \ldots, d_i$. Il est ainsi logique que pour cette trajectoire, le nombre d'agents dans tout autre groupe que i soit fixé. Alors, $\boldsymbol{\Upsilon}_{j_i}(T_n)$ est construit de façon à ce que $\theta_i \in \{B_i + 1, \ldots, b_i\}$ et $\theta_l = a_l$ pour $l \neq i$. De cette façon, il ne reste à la trajectoire fille qu'un groupe d'agents pour lequel θ_i peut changer. La trajectoire peut se scinder à nouveau jusqu'à atteindre le nombre maximal d'agents b_i dans ce groupe.

La figure 6.5 présente une partie de l'arbre des configurations possibles lorsque nous

estimons un sous-gradient dans le cas de l'exemple sur la figure 6.4. Contrairement au cas général, après la première scission, chaque configuration peut être représentée par un indice de groupe d'agents i et un intervalle.

Avec cette simplification, les sous-ensembles de configurations associés aux trajectoires peuvent être organisés de façon à pouvoir déterminer rapidement les sous-ensembles adjacents, ce qui favorise des tests rapides pour la recombinaison. Il suffit en effet, pour chaque groupe i, de maintenir une liste triée d'intervalles et de ne recombiner que les intervalles adjacents. Lors d'un test pour la recombinaison, nous parcourons ces I listes afin d'obtenir les paires de copies parallèles à comparer entre elles.

Plus précisément, une copie concerne le sous-gradient par rapport à un groupe d'agents i si elle inclut des trajectoires associées aux configurations $\boldsymbol{\theta} + s_1\mathbf{e}_i, \ldots, \boldsymbol{\theta} + s_2\mathbf{e}_i$, pour $1 \leq s_1 < s_2 \leq d_i$. Une copie de ce type, caractérisée par l'indice du groupe i et un intervalle $[s_1, s_2]$, ne peut être recombinée qu'avec la configuration de base ou une copie avec le même indice i et un sous-intervalle $[s_1, s_2]$ adjacent.

Si $d_i = 1$ pour $i = 1, \ldots, I$, à tout moment, une copie parallèle simule la configuration de base $\boldsymbol{\theta}$ tandis que pour chaque groupe d'agents i, au plus une copie simule la configuration $\boldsymbol{\theta} + \mathbf{e}_i$. La copie simulant $\boldsymbol{\theta} + \mathbf{e}_i$ ne peut, en toute logique, se recombiner qu'avec la copie simulant la configuration de base. Ainsi, quand $d_i = 1$ pour $i = 1, \ldots, I$, à chaque fois qu'une recombinaison a lieu, il n'y a que I paires de configurations à tester pour l'égalité de l'état.

6.2.6 Gestion de périodes multiples

Supposons maintenant que les paramètres peuvent varier d'une période à l'autre pour un horizon composé de $P+2$ périodes. Alors, le vecteur d'agents devient une matrice $\boldsymbol{\theta}$ de taille $I \times P$ dont chaque élément $\theta_{i,p}$ donne le nombre d'agents du groupe i pendant la période principale p. Pendant la période préliminaire, il n'y a aucun agent tandis que le nombre d'agents pendant la période de fermeture correspond au nombre pendant la dernière période principale. Comme précédemment, nous estimons une fonction $f(\boldsymbol{\theta})$

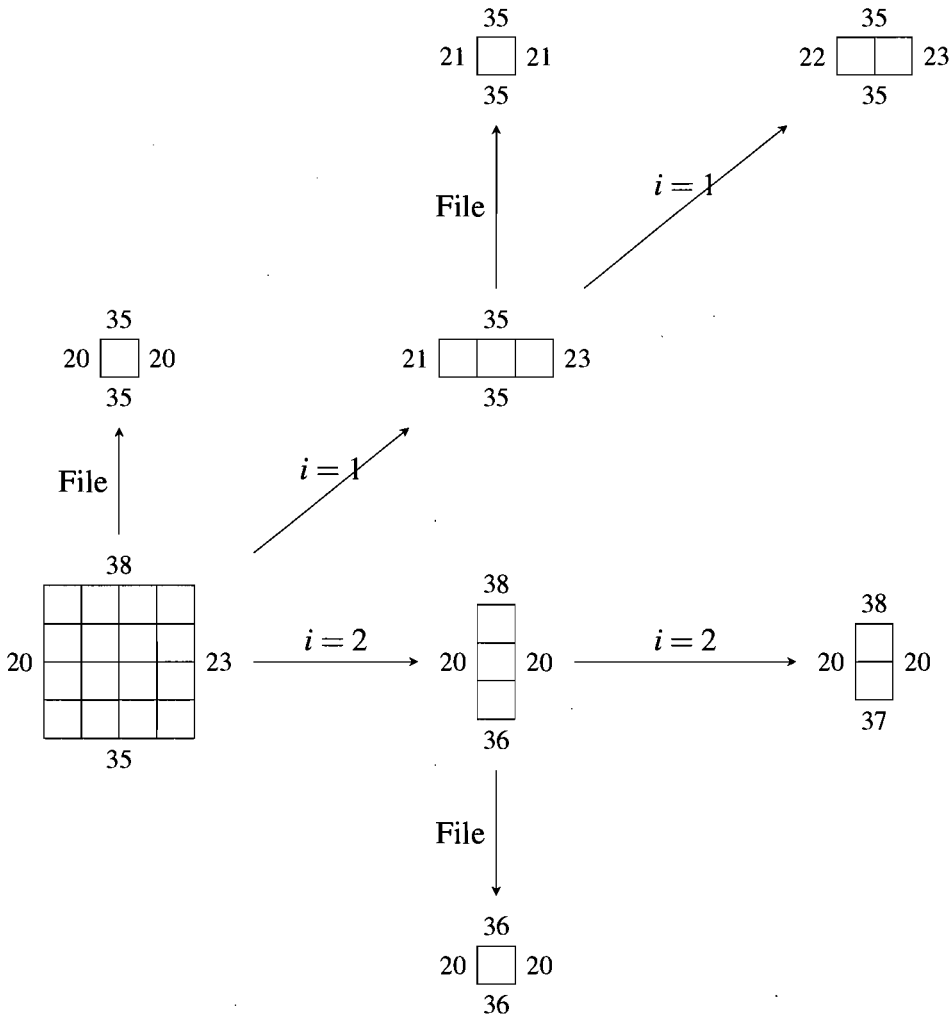

Figure 6.5 – Exemple d'arbre de scission, dans le cas où seul un sous-gradient est estimé

par une moyenne. Nous supposons que $\theta_{i,p} \in \{a_{i,p}, \ldots, b_{i,p}\}$. En plus de multiplier le nombre de configurations possibles, soit $\prod_{p=1}^{P} \prod_{i=1}^{I} (b_{i,p} - a_{i,p} + 1)$, l'application de la technique pose problème si le nombre d'agents change d'une période à l'autre, ce qui bien entendu se produit souvent.

Supposons qu'au début de la période p, la file d'attente contient au moins un contact et nous ajoutons un agent. Un contact est alors retiré de la file pour être servi par ce nouvel agent. Si nous ajoutons deux agents, deux contacts sont retirés. Ainsi, l'évolution du système dépend non pas seulement du fait d'ajouter des agents ou pas mais aussi du nombre exact d'agents ajoutés. De plus, si nous retirons suffisamment d'agents pour que $B_i > \theta_{i,p}$, les $B_i - \theta_{i,p}$ fins de service suivant le début de la période p ne retirent aucun contact de la file. Le comportement du modèle dépend donc du nombre exact d'agents retirés. Cela peut bien entendu produire un grand nombre de copies parallèles au début des périodes.

En général, au début d'un nouvelle période, chaque valeur de $N_i \in \{0, \ldots, M_i\}$ peut engendrer une trajectoire distincte. Ici, $M_i = B_i + \sum_{k=1}^{K} m_{i,k} Q_k$ avec $m_{i,k} = 1$ si un agent du groupe i peut servir un contact de type k et 0 si tel n'est pas le cas. La valeur M_i donne la somme du nombre d'agents occupés dans le groupe i et du nombre de contacts en attente qu'un agent du groupe i peut servir.

Si $a_{i,p} > M_i$, étant donné que $b_{i,p} \geq a_{i,p}$, le comportement du système suivant immédiatement le changement de période ne dépend pas de la valeur exacte de N_i si bien qu'il n'est pas nécessaire de faire une scission dès le début de la période. Si $a_{i,p} \leq M_i$ mais $b_{i,p} > M_i$, il faut créer une copie parallèle pour chaque valeur de $N_i \in \{a_{i,p}, \ldots, M_i\}$ ainsi qu'une copie regroupant les valeurs $\{M_i + 1, \ldots, b_{i,p}\}$. Enfin, si $a_{i,p} \leq b_{i,p} \leq M_i$, nous devons créer une copie parallèle pour chaque valeur de $N_1 \in \{a_{i,p}, \ldots, b_{i,p}\}$. Dans le cas où un seul paramètre est considéré, le nombre de copies parallèles possible est donc borné par $\min(M_i, b_{i,p} - a_{i,p} + 1)$.

Par contre, si nous avons un vecteur de paramètres, nous devons décomposer l'ensemble $\{a_{i,p}, \ldots, b_{i,p}\}$ en utilisant M_i pour chacun des groupes et énumérer toutes les

combinaisons possibles de composantes. Cela revient souvent à considérer toutes les $\prod_{i=1}^{I}(b_{i,p}-a_{i,p}+1)$ combinaisons d'agents. Nous perdons alors beaucoup des bénéfices de la scission étant donné que, dès le début de la période p, il faut créer autant de trajectoires qu'il y a de configurations différentes pour cette période. Bien souvent, beaucoup de ces trajectoires sont identiques par la suite et les copies parallèles associées se recombinent. Ainsi, la méthode générale de scission et de recombinaison ne fonctionnera bien, dans un cas multi-périodes, que pour I petit, voire $I = 1$.

Concentrons-nous maintenant sur l'estimation du sous-gradient par la méthode des différences finies. L'estimé sera alors une matrice dont l'élément concernant le groupe d'agents i et la période p est

$$q_{i,p}(\boldsymbol{\theta}) = \frac{f(\boldsymbol{\theta}+d_{i,p}\mathbf{e}_{i,p})-f(\boldsymbol{\theta})}{d_{i,p}}.$$

Ici, $d_{i,p}$ est une constante souvent égale à 1 et $\mathbf{e}_{i,p}$ est une matrice de $I \times P$ dont l'élément (i,p) est 1 et tous les autres éléments sont 0.

Pendant la première période, au plus $1+\sum_{i=1}^{I} d_{i,1}$ trajectoires sont créées : les trajectoires pour estimer $f(\boldsymbol{\theta}+s\mathbf{e}_{i,1})$, avec $i = 1,\ldots,I$ et $s = 1,\ldots,d_{i,1}$, ainsi qu'une trajectoire pour la simulation de base destinée à estimer $f(\boldsymbol{\theta})$. Au début de la seconde période, seule cette dernière trajectoire est clonée en au plus $1+\sum_{i=1}^{I} d_{i,2}$ nouvelles trajectoires, afin d'estimer $f(\boldsymbol{\theta}+s\mathbf{e}_{i,2})$, pour $i = 1,\ldots,I$ et $s = 1,\ldots,d_{i,2}$. Encore une fois, il faut partitionner $\{a_{i,p},\ldots,b_{i,p}\}$ en fonction de M_i mais cette fois pour chaque groupe séparément ; il n'est pas nécessaire d'énumérer toutes les combinaisons. Les trajectoires créées pendant la première périodes ne sont pas clonées à nouveau. Ainsi, à la fin de la simulation, nous avons un total de $1+\sum_{i=1}^{I}\sum_{p=1}^{P} d_{i,p}$ trajectoires.

Tout le travail de simulation effectué jusqu'au début d'une période $p > 1$ est le même quelle que soit la valeur de $\boldsymbol{\theta}+s\mathbf{e}_{i,p}$. La méthode de scission et de recombinaison permet d'éviter de répéter ce travail et peut donc être très efficace pour estimer le sous-gradient dans un contexte multi-périodes.

Un autre cas pour lequel la méthode est prometteuse est celui où nous estimons le sous-gradient par rapport au nombre d'agents dans des quarts de travail. Un quart est défini ici comme un vecteur de variables binaires de taille P dont la composante d'indice p, pour $p = 1,\ldots,P$, détermine si un agent affecté à ce quart est en service pendant la période p. Ajouter un agent sur un tel quart augmente alors le nombre d'agents dans une ou plusieurs périodes de la journée.

Dans un tel contexte, $\boldsymbol{\theta}$ est une matrice de dimensions $I \times Q$, où Q est le nombre total de quarts, et dont chaque élément $\theta_{i,q}$ donne le nombre d'agents du groupe i affectés au quart q. L'estimé du sous-gradient devient alors

$$q_{i,q}(\boldsymbol{\theta}) = \frac{f(\boldsymbol{\theta} + d_{i,q}\mathbf{e}_{i,q}) - f(\boldsymbol{\theta})}{d_{i,q}}.$$

Il peut exister un grand nombre de quarts différents, parfois des milliers, ce qui engendre beaucoup de configurations différentes à simuler. Par contre, beaucoup de ces configurations se ressemblent, car il arrive souvent que des quarts ne se distinguent que par une période de travail en plus, en moins ou placée ailleurs pendant la journée. La méthode de scission et de recombinaison pourrait permettre d'identifier ces configurations similaires et ainsi de réduire substantiellement le temps de simulation.

Pendant la première période, il existe au pire cas $I+1$ copies parallèles : une copie de base simulant la configuration $\boldsymbol{\theta}$ et une copie pour laquelle il y a un agent de plus dans le groupe i, pour $i = 1,\ldots,I$. Chacune de ces copies regroupe plusieurs trajectoires représentant des quarts de travail. Au début de la seconde période, la copie de base peut de nouveau se scinder en $I+1$ copies parallèles, chaque copie correspondant à un agent ajouté dans un groupe ou à la configuration de base. La copie i créée pendant la première période, pour sa part, se scinde en deux : une copie pour laquelle le quart de travail des agents du groupe i inclut la période 1 mais pas la période 2, et une pour laquelle les deux périodes sont incluses dans le quart. Cela crée un total de $3(I+1)$ copies parallèles, mais les copies partageant le même vecteur d'agents pendant la deuxième période devraient se

recombiner éventuellement. Seuls $I+1$ vecteurs d'agents différents sont possibles pour une période donnée : le vecteur de base, et le vecteur pour lequel un agent est ajouté dans le groupe i, pour $i = 1,\ldots,I$. Ainsi, la méthode ne simule habituellement pas plus de $I+1$ copies parallèles, peu importe le nombre de quarts de travail dans le modèle.

6.3 Exemples numériques

Dans cette section, nous testons notre méthode de scission et de recombinaison, d'abord sur un exemple avec un seul type de contact et un seul groupe d'agents, puis sur des exemples avec plusieurs groupes. Nous évaluons la performance de notre méthode pour estimer des sous-gradients dans le cas où il y a plusieurs groupes d'agents. Nous étudions aussi empiriquement l'effet d'un taux de transition adaptatif.

Nous verrons que notre méthode réduit souvent efficacement le nombre moyen de copies parallèles et donc le travail de simulation par rapport à la technique Monte Carlo classique qui simule chaque configuration séparément. Mais la méthode de scission et de recombinaison impose un coût en temps de calcul qui fait en sorte qu'il faut simuler beaucoup de configurations semblables simultanément pour observer une diminution du temps de calcul par rapport à la simulation séparée de toutes les configurations.

Pour les exemples de cette section, le vecteur $\boldsymbol{\theta} = (N_1,\ldots,N_I)$ correspond au vecteur d'agents.

6.3.1 Test avec un type de contact et un groupe d'agents

Revenons à l'exemple 1a de la section 5.4.3, qui comporte un seul type de contact et un groupe d'agents. Nous avons testé avec $\theta = N_1$ variant entre 52 et 55. Cela nous donne un total de quatre configurations si bien que le gain maximal qui peut être obtenu avec la méthode, par rapport à la simulation séparée de toutes les configurations, est 4.

Pour que la méthode soit efficace, il faut que les quatre configurations à simuler se ressemblent. Si le taux d'arrivée des contacts est petit, nous obtenons avec 52 agents un

niveau de service élevé et un taux d'occupation bas. Dans un tel cas, ajouter un agent ne change pas beaucoup l'évolution du système. D'un autre côté, si le taux d'arrivée est élevé, le niveau de service avec 52 agents est plus bas et ajouter des agents affecte davantage la performance du centre de contacts.

Pour illustrer cela, nous avons appliqué la méthode de scission et de recombinaison pour l'exemple avec un nombre moyen d'arrivées de $\lambda T = 15\ 000$, 19 118 et 25 000, pour les quatre configurations. Le tableau 6.I présente le temps total nécessaire pour simuler toutes les configurations avec différentes variantes de la méthode de scission ainsi qu'avec la méthode Monte Carlo classique appliquée sur chaque configuration séparément, avec variables aléatoires communes. Le tableau donne aussi le niveau de service et le taux d'occupation obtenus en simulant avec 52 agents, le nombre moyen de copies parallèles observées lorsque la recombinaison est permise et lorsqu'elle n'est pas appliquée, le nombre moyen de recombinaisons et le nombre espéré de transitions. Les résultats empiriques ont été obtenus avec 1 000 macro-réplications indépendantes de la méthode.

Comme le montrent les temps d'exécution, lorsque le taux d'arrivée est petit pour cet exemple, la méthode de scission et de recombinaison est environ 1,5 fois plus rapide que la simulation de toutes les configurations séparément, avec variables aléatoires communes. Par contre, le gain diminue si le taux d'arrivée augmente. Si le taux est suffisamment élevé, la méthode devient même plus lente que simuler les configurations séparément. Comme nous pouvons nous y attendre, la recombinaison permet de gagner en temps d'exécution dans certains cas, mais le gain pour cet exemple est assez faible.

Tentons maintenant de cerner les facteurs pouvant affecter l'efficacité de la méthode proposée. D'abord, nous devons garder à l'esprit que la méthode de scission et de recombinaison impose inévitablement un coût en temps de calcul en raison des vérifications nécessaires lors de chaque transition pour déterminer si une scission ou une recombinaison a lieu. Nous pouvons certes alléger ces coûts en tenant à jour des variables indicatrices permettant de tester rapidement si $\boldsymbol{\theta} \in \boldsymbol{\Upsilon}_j(t)$ pour toute trajectoire, mais nous ne

Tableau 6.I – Performance de la méthode de scission et de recombinaison pour un exemple avec un type de contact et un groupe d'agents

λT	15 000	19 118	25 000
Niveau de service	100%	100%	81%
Taux d'occupation	62%	78%	95%
Temps simulation séparée	22s	24s	32s
Temps scission/recombinaison	15s	30s	54s
Temps scission sans recombinaison	17s	31s	49s
Nombre moyen de copies parallèles	1,00	2,57	2,91
Nombre moyen de copies parallèles sans recombinaison	2,10	3,92	3,98
Nombre moyen de recombinaisons	7,48	215	2 116
Nombre moyen de transitions	44 484	48 602	54 484

pouvons pas l'éliminer complètement.

Ce coût augmente avec le nombre moyen de transitions et de copies parallèles. Comme le taux de transition augmente avec le taux d'arrivée, plus λT est grand, plus le coût imposé par la méthode est élevé. D'un autre côté, pour que le nombre moyen de copies parallèles soit petit, les configurations à simuler doivent se ressembler et la recombinaison doit être appliquée. Ces conditions sont satisfaites avec un taux d'arrivée petit.

Si le taux d'arrivée est élevé, le taux d'occupation des 52 agents est lui aussi élevé. Ajouter un ou deux agents aide alors beaucoup à réduire ce taux d'occupation et à augmenter le niveau de service. Les configurations simulées sont alors si différentes les unes des autres que la méthode revient en pratique à simuler chaque configuration de façon séparée, mais avec un coût en temps de calcul plus élevé. Du temps peut également être perdu si des recombinaisons sont immédiatement suivies de scissions, ce qui est susceptible de survenir avec des configurations engendrant des trajectoires significativement différentes. De plus, même si les recombinaisons se produisent, il faut quand même mettre à jour davantage de compteurs, soit un ensemble de compteurs pour chaque trajectoire.

Pour les trois taux d'arrivée testés, le nombre moyen de recombinaisons observées est largement inférieur au nombre moyen de transitions. Cela peut mener à croire que la

recombinaison ne permet pas de gain important pour cet exemple, mais d'un autre côté, un nombre élevé de recombinaisons pourrait indiquer que beaucoup de ces recombinaisons sont suivies d'une scission.

Pour mieux étudier le gain apporté par la recombinaison, examinons le nombre moyen de copies parallèles pendant la simulation, c'est-à-dire la moyenne de $\int_0^T R(t)dt/T$ sur toutes les macro-réplications, où $R(t)$ est le nombre de copies parallèles au temps t. Pour les trois taux d'arrivée, la recombinaison réduit le nombre moyen de copies parallèles, mais cette réduction diminue si le taux d'arrivée augmente. Sans la recombinaison, le nombre moyen de copies parallèles est habituellement près de la borne supérieure donnée par le nombre total de configurations, 4 dans ce cas. Par contre, ce n'est pas le cas avec $\lambda T = 15\ 000$. Cela peut s'expliquer par le fait qu'avec ce petit taux d'arrivée, pour certaines macro-réplications, il n'est pas nécessaire de créer une trajectoire pour chacune des quatre configurations. Cela signifie que pour certaines macro-réplications, aucun contact n'arrive à un moment où N_1 agents sont occupés, où N_1 est un certain nombre inférieur à 55.

Ces résultats montrent que la recombinaison, même si elle n'a pas lieu très souvent, permet de réduire le nombre de copies parallèles pendant la simulation et est donc utile pour éviter de répéter du travail de simulation. Pourtant, elle n'apporte pas un gain important en temps d'exécution.

Ce faible gain peut s'expliquer de la façon suivante. Permettre la recombinaison implique des tests à chaque fin de service et chaque abandon, ainsi que la construction et la mise à jour d'une liste ordonnée de copies parallèles utilisée pour déterminer quelles paires de copies tester. Cela impose un coût de calcul additionnel qui est compensé partiellement par le fait que la recombinaison sauve du travail.

Ainsi, s'il y a peu de groupes d'agents, la méthode n'est intéressante que si le taux d'arrivée est petit, au moins dans certains intervalles de temps. Pour illustrer cela, prenons le même exemple, avec un taux d'arrivée faible la plupart du temps, sauf pendant deux périodes achalandées d'une demi-heure chacune. Le tableau 6.II montre le nombre

moyen d'arrivées dans les différentes périodes de la journée. Chaque ligne correspond à un intervalle avec ses extrémités et le nombre d'arrivées moyen.

Le tableau 6.III, semblable au tableau 6.I, présente les résultats pour cet exemple modifié, avec 1 000 macro-réplications. La méthode de scission et de recombinaison permet cette fois un gain en temps d'exécution de 2,3. Examinons maintenant les causes de cette différence de performance par rapport à l'exemple précédent.

D'abord, pour simuler l'exemple, nous avons utilisé un taux de transition plus élevé que pour l'exemple précédent, non pas seulement pour les périodes achalandées mais pour l'horizon entier. Il est ainsi normal que le temps total nécessaire pour exécuter les quatre configurations séparément soit supérieur à celui observé lorsque le taux d'arrivée est fixe.

Pendant les périodes normales, il se produit davantage de transitions fictives qu'avec l'exemple original, car nous avons augmenté le taux de transition. Le temps de calcul lors de telles transitions est dominé par celui nécessaire pour générer les nombres aléatoires. Lorsque la méthode de scission est employée, pour chaque transition, un nombre aléatoire est généré une seule fois et utilisé pour faire avancer chaque copie de la CMTC. Ainsi, plus le taux de transition fictive est élevé, plus la méthode permet de sauver du temps.

Encore une fois, la recombinaison permet de réduire le nombre moyen de copies parallèles tout en sauvant un peu sur le temps d'exécution. Sans la recombinaison, le nombre de copies parallèles est très près du nombre de configurations si bien que seule la réutilisation des nombres aléatoires peut justifier le gain en temps d'exécution par

Tableau 6.II – Variation du nombre moyen d'arrivées dans le temps pour l'exemple 1a

t_1	t_2	$\mathbb{E}[A(t_1,t_2)]$
08:00	09:00	1 471
09:00	09:30	3 676
09:30	13:00	5 147
13:00	13:30	3 676
13:30	21:00	11 029

Tableau 6.III – Performance de la méthode de scission et de recombinaison pour un exemple avec un type de contact, un groupe d'agents et un taux d'arrivée variant dans le temps

λT	25 000
Niveau de service	70%
Niveau de service 8:00 – 9:00	100%
Niveau de service 9:00 – 9:30	1%
Taux d'occupation	80%
Taux d'occupation 8:00 – 9:00	76%
Taux d'occupation 9:00 – 9:30	100%
Temps simulation séparée	3min16s
Temps scission/recombinaison	1min25s
Temps scission sans recombinaison	1min56s
Nombre moyen de copies parallèles	1,46
Nombre moyen de copies parallèles sans recombinaison	3,92
Nombre moyen de recombinaisons	412
Nombre moyen de transitions	125 072

rapport à la simulation de configurations séparées.

6.3.2 Simulation avec plusieurs groupes d'agents

Prenons maintenant les exemples 4 et 5 de la section 5.4.3 avec trois types de contacts et trois groupes d'agents pour l'exemple 4 et six groupes pour l'exemple 5. Comme nous avons vu précédemment, la méthode est plus efficace lorsque le taux d'arrivée est petit. Nous utilisons alors les taux d'arrivée du tableau 6.IV pour les expérimentations.

Pour l'exemple 4, la simulation de base est effectuée avec le vecteur d'agents $\boldsymbol{\theta} = (20, 12, 20)$. Nous avons augmenté le nombre d'agents jusqu'à 21 pour les groupes 1 et 3 et jusqu'à 13 pour le groupe 2. Le nombre résultant de configurations est alors 4.

Tableau 6.IV – Nombre moyen d'arrivées, pendant l'intervalle $[0, T]$, pour les exemples 4 et 5

Exemple	Type 1	Type 2	Type 3
4	4 779	7 169	7 169
5	3 976	3 976	2 982

Pour l'exemple 5, la simulation de base est effectuée avec $\boldsymbol{\theta} = (36,35,27,3,5,4)$. Nous avons testé avec un agent de plus pour chaque groupe, ce qui donne sept configurations.

Le tableau 6.V, dont le format est le même que celui du tableau 6.I avec une colonne pour chaque exemple, compare la méthode de scission et de recombinaison pour estimer un sous-gradient avec la simulation séparée de chaque configuration. Pour les deux exemples, le niveau de service de la configuration de base est donné pour chaque type de contact tandis que le taux d'occupation est indiqué pour chaque groupe d'agents. Encore une fois, nous avons effectué 1 000 macro-réplications indépendantes.

La méthode de scission et de recombinaison réduit le temps de calcul par rapport à la simulation des configurations séparément d'un facteur 1,6 pour l'exemple 5, mais n'apporte aucune réduction pour l'exemple 4. Par contre, la méthode réduit le nombre moyen de copies parallèles pour les deux exemples. De plus, pour les deux exemples, le nombre moyen de copies parallèles lorsque la recombinaison n'est pas utilisée est près de la borne supérieure donnée par le nombre de configurations.

Dans le cas de l'exemple 4, la recombinaison ne réduit pas beaucoup le nombre de copies parallèles si bien que nous ne pouvons pas nous attendre à ce qu'elle réduise le temps d'exécution. Dans cet exemple, seul le taux d'occupation du premier groupe d'agents est faible. Ceci est causé par le fait que le premier groupe reçoit en priorité des contacts du premier type, dont le taux d'arrivée est inférieur à celui des deux autres types. Les contacts des deux autres types, dont le taux d'arrivée est plus élevé, vont en priorité vers les agents des deux autres groupes qui parviennent à les servir. Ainsi, la politique de routage utilisée fait en sorte qu'il y a peu de débordement vers le premier groupe.

Par contre, avec l'exemple 5, nous avons un taux d'occupation bas dans tous les groupes, particulièrement les trois derniers. La méthode est alors plus efficace. De plus, pour cet exemple, les trajectoires pour les groupes $i = 1,2,3$ naissent plus rapidement que celles pour les groupes $i = 4,5,6$ étant donné que tous les agents d'au moins un des

Tableau 6.V – Performance de la méthode de scission et de recombinaison lors d'une estimation de sous-gradient

	Exemple 4	Exemple 5
Niveau de service	(99%,97%,99%)	(100%,100%,100%)
Taux d'occupation	(78%,91%,90%)	(75%,77%,74%, 19%,19%,19%)
Temps simulation séparée	54s	52s
Temps scission/recombinaison	1min7s	33s
Temps scission sans recombinaison	56s	49s
Nombre moyen de copies parallèles	3,47	2,16
Nombre moyen de copies parallèles sans recombinaison	3,98	6,51
Nombre moyen de recombinaisons	164	106
Nombre moyen de transitions	48 601	32 008

trois premiers groupes doivent être occupés avant que les contacts soient envoyés vers les agents des trois derniers groupes. Cela réduit le nombre de copies parallèles qui peuvent se créer d'un coup au moment d'une arrivée.

Habituellement, lors d'une scission, seules deux copies parallèles sont créées à partir de la copie mère tandis qu'avec l'exemple 4, une scission peut créer jusqu'à quatre copies parallèles d'un coup. Ceci est dû à la politique de routage et au fait que le nombre d'agents dans les trois premiers groupes est beaucoup plus élevé que celui dans les trois derniers.

En conclusion, la méthode de scission et de recombinaison est efficace pour estimer le sous-gradient sur certains exemples de centres de contacts. La méthode fonctionne particulièrement bien s'il y a beaucoup de groupes d'agents, donc beaucoup de configurations, et que la charge de travail de ces agents est souvent faible. Par contre, comme nous avons vu au chapitre précédent, un trop grand nombre de groupes d'agents rend inefficace la méthode d'uniformisation et de conversion en temps discret utilisée pour la simulation de la CMTC.

6.3.3 Impact d'un taux de transition adaptatif sur les recombinaisons

Dans la section 5.2.3, nous avons abordé une méthode permettant de bénéficier d'un taux de transition adaptatif qui consistait à partitionner l'ensemble d'états $\mathcal{S}$ et à associer, à chaque sous-ensemble, un taux de transition et un index de recherche. Toutefois, les possibilités de recombinaison diminuent à partir du moment où les CMTC simulées en parallèle ne sont plus dans le même sous-ensemble d'états r. En effet, comme nous avons vu au chapitre précédent, dès que r diffère d'une configuration à l'autre, la synchronisation de l'état du système par rapport au temps en souffre. Cela réduit les chances que deux trajectoires soient dans le même état à une même étape. De plus, le nombre de transitions fictives successives entre chaque transition principale dépendant de r, les CMTC en parallèle avec des modes d'opération différents ne sont plus synchronisées dans le temps. Il faut donc comparer le numéro de transition courant au moment de la recombinaison afin de vérifier que les copies recombinées sont à la même étape.

Pour illustrer cela, nous ajoutons des seuils sur la taille de la file d'attente dans l'exemple 5, avec le taux d'arrivée original de la section 5.4.3. Nous avons appliqué la méthode avec $R = 2$, puis avec $R = 3$. Dans le premier cas, le seuil sur la taille de la file est 10 tandis que dans le second, nous avons 10 et 25 comme seuils. Le tableau 6.VI montre le temps d'exécution, le nombre moyen de recombinaisons et le nombre moyen de copies parallèles, pour $R = 1, 2$ et 3. Comme nous pouvions nous en attendre, augmenter le nombre de sous-ensembles de $\mathcal{S}$ réduit le nombre moyen de recombinaisons et augmente par le fait même le nombre moyen de copies parallèles.

Le temps d'exécution avec $R = 2$ est plus élevé que celui avec $R = 1$, car le nombre moyen de recombinaisons est plus petit avec $R = 2$. Par contre, passer à $R = 3$ sous-

Tableau 6.VI – Effet du partitionnement de $\mathcal{S}$ sur les recombinaisons

R	Temps d'exécution	Nombre moyen de recombinaisons	Nombre moyen de copies parallèles
1	62s	108	5,54
2	67s	8	6,75
3	63s	14	9

ensembles de S permet pour cet exemple particulier de compenser cette diminution d'efficacité de la recombinaison et de retrouver à peu près le même temps d'exécution qu'avec $R = 1$.

Cette expérience montre qu'il est important de ne pas abuser du taux de transition adaptatif. Il faut, comme expliqué dans le chapitre précédent, utiliser un petit nombre de sous-ensembles et faire en sorte que la CMTC soit la plupart du temps dans le même sous-ensemble.

CHAPITRE 7

AMÉLIORATION DU LOGICIEL DE SIMULATION

En plus d'améliorer l'efficacité des simulations, nous avons aussi travaillé sur l'architecture de notre outil de simulation pour le rendre plus flexible, extensible et plus facile à utiliser. Comme expliqué dans la section 1.3, nous avons développé une bibliothèque nommée ContactCenters permettant de construire des simulateurs de centres de contacts. En utilisant cette bibliothèque, nous avons construit un certain nombre de simulateurs qui peuvent être configurés à l'aide de fichiers dans le format XML sans programmation Java. Au cours de ce projet de doctorat, nous avons apporté plusieurs améliorations à ContactCenters. En particulier, nous avons conçu et implanté une politique de routage permettant d'ajuster les priorités pour les contacts et les agents ainsi que les délais de débordement entre les groupes d'agents pour un contact attendant en file. Nous avons implanté un simulateur utilisant le modèle de CMTC développé au chapitre 5 ainsi que la méthode de scission et de recombinaison du chapitre 6, ce qui nous a permis de mieux comprendre les problèmes posés par ces méthodes. Le simulateur générique principal a été amélioré pour prendre en charge le transfert de contacts entre groupes d'agents, les rappels par l'entremise d'une file d'attente virtuelle et la production de rapports dans le format Excel de Microsoft en plus du format textuel déjà présent. De plus, la validation des fichiers de paramètres XML se fait désormais en utilisant un Schéma XML. Plusieurs de ces améliorations ont été apportées en réponse à des commentaires des gens de Bell Canada qui ont utilisé le simulateur au cours de ce projet.

Nous n'expliquons pas ici en détails l'architecture du logiciel ainsi que les améliorations apportées. Pour cela, voir la documentation de ContactCenters qui est accessible depuis sa page Web [14]. Dans ce chapitre, nous faisons plutôt un bref survol de l'architecture du logiciel pour ensuite examiner ses principales fonctionnalités et problèmes les plus importants qui sont apparus pendant son développement. Nous traitons dans

l'ordre des problèmes affectant les résultats de simulation, la performance du logiciel et sa simplicité d'utilisation.

Les deux sections suivantes sont consacrées respectivement à l'architecture de ContactCenters et à celle du simulateur générique utilisant des fichiers XML. La section 7.3 décrit les principales fonctionnalités du simulateur générique, notamment la prise en charge de plusieurs types de contacts et groupes d'agents, les lois de probabilité admises pour les différentes variables aléatoires, le modèle utilisé pour les abandons, un survol des différentes politiques de routage et processus d'arrivées possibles, etc. Nous découvrons ensuite, dans la section 7.4, que modéliser les groupes d'agents avec des compteurs plutôt qu'avec un objet pour chaque agent a un impact sur la performance et le réalisme. La section 7.5 étudie quant à elle les difficultés soulevées par la prédiction des temps d'attente en général et tente de trouver une heuristique convenant pour la plupart des cas sans trop affecter la performance. Nous y découvrons que prendre le dernier temps d'attente observé avant le début d'un service fournit en général une bonne prédiction. La section 7.6 présente les mécanismes que nous avons mis en œuvre pour rendre notre simulateur générique plus extensible, avec un minimum de programmation Java, sans en affecter sa performance. Nous expliquons ensuite dans la section 7.7 les stratégies utilisées pour mieux exploiter le format XML de façon à simplifier l'entrée des paramètres du simulateur. Nous discutons aussi, dans la section 7.8, des moyens mis en œuvre pour améliorer la clarté des messages d'erreurs affichés par notre programme de simulation.

L'architecture et l'implantation de ContactCenters représentent une contribution importante de cette thèse, car cet outil est un des plus rapides au monde et permet d'effectuer des expériences numériques qu'il n'était pas possible de mener efficacement auparavant. En particulier, tous les exemples numériques dans cette thèse ont été simulés avec ContactCenters. De plus, certaines idées de ce chapitre, sans être totalement nouvelles, sont applicables à d'autres logiciels pour mieux tirer parti des outils existants.

7.1 Vue d'ensemble de l'architecture de ContactCenters

Comme nous l'avons expliqué dans la section 1.3, ContactCenters est formée d'objets représentant des composantes du centre de contacts et reliés entre eux par des observateurs. La bibliothèque de base, à partir de laquelle les simulateurs sont construits, comprend 122 classes réparties en sept paquets.

Le diagramme UML de la figure 7.1 montre comment les composantes de la bibliothèque sont reliées entre elles pour permettre la génération d'arrivées et la composition d'appels sortants. Sur cette figure, chaque rectangle représente une classe tandis que les liens entre les rectangles correspondent à des relations entre les classes. Les étiquettes sur les liens décrivent la nature des relations tandis que le texte aux extrémités indiquent la cardinalité. Un losange au bout d'un lien indique une relation de composition, c'est-à-dire que l'objet concerné ne peut pas exister sans un autre objet auquel il est relié. Un triangle au bout d'un lien correspond à une relation d'héritage entre classes.

Comme le montre le diagramme, les contacts, représentés par des objets de la classe `Contact`, sont construits à l'aide d'usines abstraites représentées par des implantations de l'interface `ContactFactory`. Comme l'indique le lien sur le diagramme, chaque processus d'arrivées, représenté par un objet de la classe `ContactArrivalProcess`, contient une et une seule usine abstraite pour construire les contacts. D'un autre côté, un composeur d'appels sortants, représenté par un objet de type `Dialer`, contient une politique de composition (`DialerPolicy`) qui contient à son tour une liste de clients à contacter (`DialerList`). Souvent, une usine abstraite est utilisée pour produire les contacts peuplant une telle liste. Le processus d'arrivées et le composeur d'appels sortants, qui correspondent tous deux à des sources de contacts (`ContactSource`), annoncent tous les contacts produits à zero ou plusieurs observateurs enregistrés. Ces observateurs sont des objets implantant l'interface `NewContactListener`.

La figure 7.2 montre quant à elle comment le routeur, une instance de la classe `Router`, est connecté aux groupes d'agents et aux files d'attente du modèle. Cette fi-

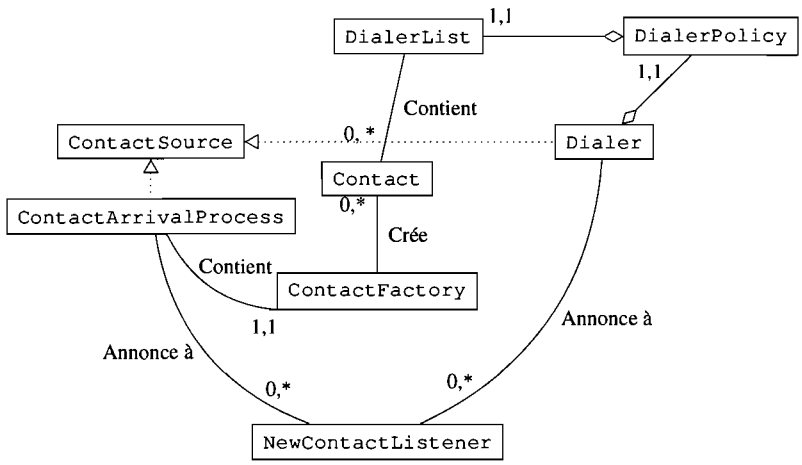

Figure 7.1 – Diagramme UML décrivant les relations existant entre les contacts, les processus d'arrivées et les composeurs d'appels sortants

gure montre que le routeur possède un ou plusieurs groupes d'agents (`AgentGroup`) et files d'attente (`WaitingQueue`). Chaque fois qu'un contact débute son service, un objet de type `EndServiceEvent` est créé pour représenter le service par un agent d'un groupe. De façon semblable, la mise en attente d'un contact engendre la création d'un objet `DequeueEvent` qui représente cette attente. Ces deux derniers objets représentent des événements qui se produisent respectivement au moment de la fin du service et de l'abandon du contact. Comme le montre la figure, des observateurs peuvent être enregistrés auprès des groupes d'agents, des files d'attente et du routeur.

7.2 Grandes lignes de l'architecture du simulateur générique

La figure 7.3 montre comment le simulateur générique, construit à partir des composantes de ContactCenters est utilisé. D'abord, des fichiers de configuration dans le format XML sont créés par l'utilisateur. Le contenu acceptable pour ces fichiers est indiqué dans la documentation du simulateur. Le fichier décrivant les paramètres du modèle (types d'appels, groupes d'agents, routage, etc.) est analysé pour produire un objet de type `CallCenterParams` tandis que le fichier abritant les paramètres de simulation

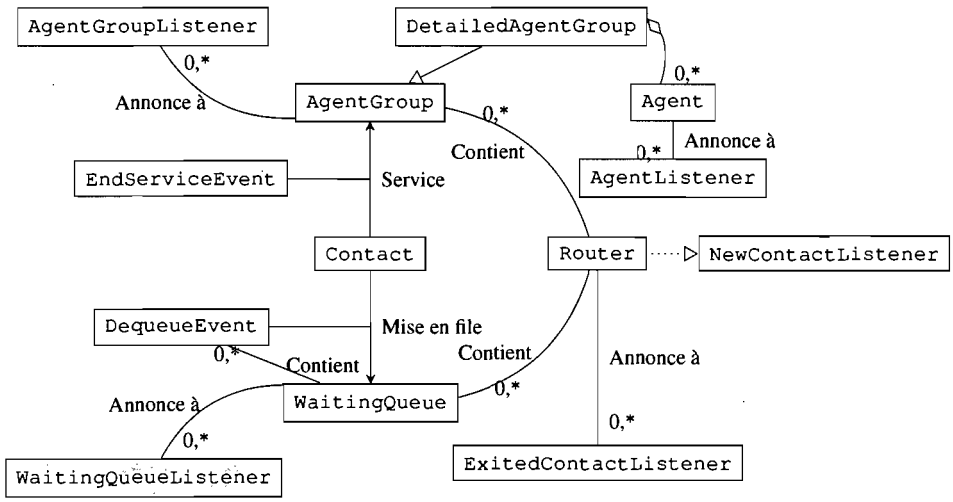

Figure 7.2 – Diagramme UML décrivant l'architecture du routeur de ContactCenters

(nombre de réplications, statistiques à inclure dans le rapport, etc.) est converti en un objet de type `SimParams`. Un objet de la sous-classe `RepSimParams` est produit si des réplications indépendantes sont utilisées pour simuler le modèle tandis qu'un objet `BatchSimParams` est créé dans le cas où une seule période est simulée comme si elle était infinie dans le modèle. Les deux objets de paramètres sont enfin utilisés pour construire le simulateur, un objet de type `CallCenterSim`, qui permet d'effectuer la simulation et de produire les résultats.

Notre première version du simulateur générique était un prototype composé d'une

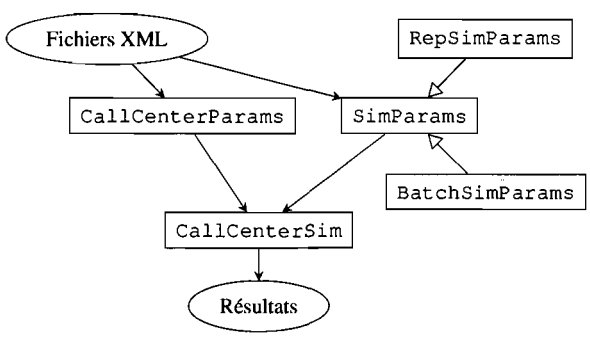

Figure 7.3 – Utilisation du simulateur générique

seule classe qui contenait tout le nécessaire pour effectuer la simulation et collecter les statistiques. Par contre, l'importance de ce simulateur s'est accrue avec le temps pour devenir l'outil le plus utilisé de ContactCenters. La classe le représentant est vite devenue très complexe si bien qu'il a été nécessaire de la séparer en plusieurs classes. La version actuelle du simulateur générique comprend 172 classes réparties en neuf paquets.

Comme le montre la figure 7.4, le simulateur générique actuel comporte trois composantes principales : un modèle simulé construit à partir des paramètres chargés depuis un fichier XML, une logique de simulation qui peut exécuter des réplications indépendantes ou traiter une période comme si elle était infinie dans le modèle, ainsi qu'un système de collecte statistique. Examinons ces composantes plus en profondeur.

Nous avons déjà abordé le modèle simulé dans la section 1.3. Ce modèle est représenté par une instance de la classe `CallCenter` regroupant tous les paramètres lus à partir des fichiers de configuration sous une forme facile à manipuler par le programmeur. Pour chaque type de contact, le modèle prévoit une usine abstraite permettant de créer les objets `Contact` mais regroupant aussi les paramètres spécifiques à ce type. Pour chaque groupe d'agents, un objet gestionnaire effectue la mise à jour de l'état au cours de la simulation. Des objets de gestion sont également prévus pour la construction et la mise à jour des paramètres des processus d'arrivées, des composeurs d'appels sortants et du routeur.

La logique de simulation est représentée par un objet de type `SimLogic`. Un tel objet permet de simuler un certain nombre d'étapes avec le modèle et d'associer une période statistique à chaque contact quittant le système. Lorsque la simulation se fait par réplications indépendantes, sur horizon fini, chaque étape correspond à une réplication et la période statistique est habituellement la période d'arrivée. Avec une simulation sur horizon infini, chaque étape simule une partie d'une longue et unique réplication et la période statistique est 0 puisque le modèle ne prévoit alors qu'une seule période.

Le système de collecte statistique est quant à lui composé d'observateurs qui mettent à jour des matrices de compteurs durant la simulation. Habituellement, ces matrices

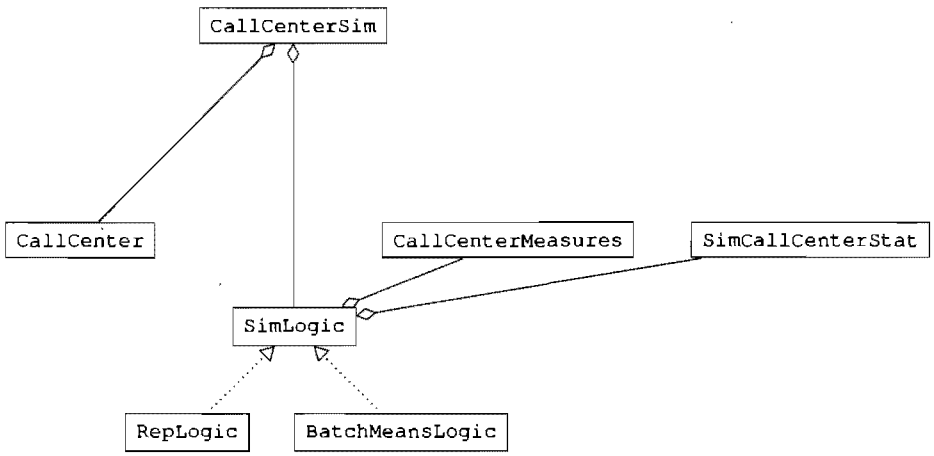

Figure 7.4 – Diagramme UML décrivant la structure du simulateur générique

comportent une rangée par type de contact ou groupe d'agents et une colonne par période. Ces matrices de compteurs sont regroupées dans un objet de type `CallCenterMeasures`. À la fin de chaque étape de simulation, les matrices sont obtenues et stockées dans des matrices de collecteurs statistiques afin de calculer des moyennes, des variances empiriques, des intervalles de confiance, etc. Ces dernières matrices sont regroupées à l'intérieur d'un objet de type `SimCallCenterStat`. Ces deux derniers objets sont contenues dans la logique de simulation étant donné que le comptage des contacts et le traitement final des matrices de mesures à la fin des étapes de simulation dépend de la logique de simulation. Les résultats de simulation sont ainsi stockés dans des matrices qui peuvent être retrouvées par le système de construction de rapports statistiques, des programmes d'optimisation, etc.

Bien entendu, la performance de ce simulateur générique est inférieure à celle d'un simulateur adapté à une situation particulière. Cette différence est causée principalement par le grand nombre de statistiques calculées par le simulateur générique. Pour traiter ce problème, nous avons ajouté une option permettant à l'utilisateur de choisir quelles statistiques il souhaite calculer.

7.3 Fonctionnalités principales du simulateur générique

Comme nous l'avons mentionné dans la section précédente et dans la section 1.3, le simulateur générique permet de traiter des modèles avec plusieurs types d'appels et groupes d'agents. Des paramètres distincts peuvent être donnés, dans le fichier de configuration XML représentant le modèle, pour chaque type, chaque groupe ainsi que pour le routeur. Dans cette section, nous nous contentons de résumer les fonctionnalités principales du simulateur. Pour plus de détails et des exemples de fichiers de configuration, voir le guide d'utilisation du simulateur, qui fait partie de la documentation de Contact-Centers [14].

Les paramètres d'un type d'appel permettent en particulier de déterminer la probabilité d'abandon immédiat et les loi de probabilité utilisées pour les durées de service et de patience. Dans le modèle implanté par le simulateur, les durées de service pour les appels de type k arrivés pendant la période p et servis par des agents du groupe i sont i.i.d. et suivent une loi de probabilité définie par l'utilisateur.

Les appels ne pouvant être servis immédiatement sont insérés dans une file d'attente et peuvent abandonner immédiatement avec une probabilité dépendant à la fois de leur type et de leur période d'arrivée. Les appelants décidant d'attendre peuvent devenir impatients et abandonner. Pour les appels de type k arrivant pendant la période p, les temps de patience sont i.i.d. et suivent une loi de probabilité quelconque.

Pratiquement n'importe quelle loi de probabilité de SSJ peut être utilisée pour ces durées. Cela inclut en particulier les lois courantes comme l'exponentielle, la gamma, la lognormale, etc. L'utilisateur peut également, avec un peu de programmation Java, fournir sa propre loi. Nous verrons comment ceci est implanté à la section 7.6.1.

Les appels sont produits par des sources de contacts pouvant être activée ou désactivée durant la simulation, à des moments fixés par l'utilisateur. Les paramètres de telles sources sont également inclus dans le fichier de configuration XML.

Dans le cas des appels entrants, la source de contacts correspond à un processus

d'arrivées. Le simulateur prend bien entendu en charge le processus de Poisson couramment utilisé, mais plusieurs variantes de ce processus sont également disponibles. En particulier, le taux d'arrivée peut varier dans le temps, de façon constante par morceaux ou continue. Le taux d'arrivée peut également, pour chaque jour simulé, être multiplié par un facteur d'achalandage aléatoire comme dans l'exemple de la section 3.5.1 ; il en résulte un processus doublement stochastique. La loi de probabilité de ce facteur d'achalandage est elle aussi fixée par l'utilisateur. D'autres processus plus complexes, comme ceux proposés dans [5], sont également pris en charge. Enfin, l'utilisateur peut, à l'aide de programmation Java, construire son propre processus d'arrivées et l'intégrer au simulateur. Le mécanisme permettant cela sera expliqué dans la section 7.6.2.

Dans le cas des appels sortants, la source de contacts est un composeur puisant les contacts dans une liste supposée infinie. Un composeur détermine, après chaque fin de service, un nombre d'appels à tenter de composer en utilisant une politique de numérotation. Chaque appel tenté est dit réussi avec une certaine probabilité dépendant de son type et de la période courante. Un appel réussi est envoyé au routeur après un certain temps de composition aléatoire tandis qu'un appel échoué est perdu. Conditionnellement au type d'appel k et à la période de composition p, les temps de composition sont eux aussi i.i.d. et suivent une loi de probabilité déterminée par des paramètres dans le fichier de configuration XML.

Plusieurs politiques de numérotation sont fournies, la plupart s'inspirant de [25]. Dans la politique la plus courante, aucun appel n'est composé si le nombre total d'agents libres ou le nombre $N_{\mathrm{F}}^{\mathrm{D}}(t)$ d'agents libres pouvant servir les appels composés sont inférieurs à des seuils fixés par l'utilisateur et dépendant de la période courante. Si le nombre d'agents libres est suffisant, le nombre d'appels tentés est déterminé par la formule $\max(0, \mathrm{round}(\kappa N_{\mathrm{F}}^{\mathrm{D}}(t)) + c)$ où $\kappa \in \mathbb{R}$ et $c \in \mathbb{N}$ sont des constantes fixées par l'utilisateur et $\mathrm{round}(\cdot)$ est une fonction arrondissant son argument à l'entier le plus près.

Pour chaque groupe d'agents du modèle, il est possible de fixer le nombre de membres pour chaque période. Au lieu d'un nombre d'agents, l'utilisateur peut également définir

des types de quarts de travail et indiquer combien d'agents du groupe il veut affecter à chaque type de quart. Les quarts de travail sont décrits à l'aide de vecteurs de taille P dont chaque composante p est une variable binaire qui indique si l'agent sur ce quart travaille pendant la période principale p.

Dans ce modèle, un agent peut se déconnecter pour une durée aléatoire lorsqu'il termine un service, de façon indépendante des autres agents. C'est pourquoi il est également possible mais optionnel de fixer pour chaque période la probabilité avec laquelle un agent se déconnecte à la fin d'un service ainsi que la loi de probabilité pour les durées de déconnexion. Aucune déconnexion n'a lieu si ces paramètres ne sont pas fixés.

Le routeur reçoit tous les appels entrants et les appels sortants réussis afin de leur affecter un agent libre ou une file d'attente. Il est aussi averti chaque fois qu'un agent est ajouté ou devient libre afin de tenter de lui affecter un appel en attente. Ces deux opérations sont accomplies par la politique de routage. Plusieurs politiques couramment utilisées dans les centres de contacts réels sont disponibles pour le simulateur générique.

Par exemple, lors d'une arrivée, la politique la plus simple parcourt une liste de groupes d'agents et choisit le premier groupe de la liste comportant au moins un agent libre. Une autre politique implantée sélectionne le groupe d'agents ayant la plus haute valeur de préférence pour le nouveau contact et prend le groupe comportant l'agent avec le plus long temps d'inactivité si plusieurs groupes partagent la même valeur maximale de préférence. Le premier exemple de politique utilise une liste de groupes d'agents pour chaque type de contact tandis que le second emploie une matrice donnant une valeur de préférence pour chaque type de contact et groupe d'agents. Tous ces paramètres sont ajustés par l'utilisateur dans le fichier de configuration XML. De plus, la politique du modèle de CMTC de la section 5.2 est disponible avec le simulateur générique.

Il est également possible, avec certaines politiques, d'indiquer le temps minimal qu'un appel d'un type k doit passer en file avant d'être autorisé à être servi par un agent d'un groupe i. Avec de la programmation Java, l'utilisateur peut également implanter des politiques de routage plus complexes impliquant, par exemple, des conditions sur la

taille de la file, le nombre d'agents dans les groupes ou même des statistiques sur des intervalles de temps précédant le routage.

Pour couvrir toutes ces possibilités, le simulateur générique doit prendre en charge une large gamme de paramètres et un format de fichier XML relativement complexe. Pour alléger le travail de l'utilisateur créant ou modifiant un fichier de paramètres, certains paramètres peuvent heureusement être omis s'ils sont facilement recalculables. Par exemple, pour les durées de service, il faut en général spécifier une loi de probabilité pour chaque combinaison de type de contact, groupe d'agents et périodes de la journée ! Par contre, il arrive souvent que la loi des durées de service ne dépend que du type de contact si bien qu'une syntaxe est disponible pour ne fournir que cette information. De façon semblable, si la probabilité d'abandon immédiat est la même pour toute la journée, l'utilisateur n'a pas à répéter sa valeur pour chaque période principale.

Par contre, reconstituer de l'information cause problème quand il existe plusieurs façons différentes de le faire. Par exemple, certaines politiques de routage utilisent une matrice de priorités de dimensions $K \times I$ pour affecter un agent à un nouveau contact et une matrice complémentaire de dimensions $I \times K$ pour choisir un contact en attente pour un agent libre. La matrice de priorités pour la sélection des agents peut être reconstituée en transposant la matrice de priorités pour la sélection des contacts ou encore à partir de listes ordonnées de groupes d'agents associés à chaque type de contact. Une méthode qui semble naturelle dans un cas peut ne pas l'être dans un autre, ce qui a posé certains problèmes d'interprétation dans le passé. À présent, dans de tels cas, il faut indiquer dans le fichier de configuration comment les éléments manquants doivent être construits. Rappelons que le format des fichiers de paramètres et la description des paramètres possibles sont spécifiés dans la documentation de ContactCenters [14].

Le simulateur générique estime toutes les mesures de performance dont nous avons discuté dans la section 2.2, pour chacun des types d'appel et chaque période principale. En particulier, le niveau de service peut être estimé avec plusieurs temps d'attente acceptables différents. Les résultats peuvent être affichés à l'écran sous forme textuelle,

sauvegardés dans un fichier au format Microsoft Excel pour être analysés par un tableur ou encore dans le format XML pour être réutilisé par un programme.

Le simulateur a permis de traiter des centres d'appels de diverses tailles et complexités, allant jusqu'à une cinquantaine de types de contacts, de groupes d'agents et de périodes. Aucune limite théorique de capacité n'a été établie ; les limites ne dépendent que de la mémoire disponible.

7.4 Impact de la représentation des groupes d'agents avec des compteurs

Dans la réalité, lorsque la fin du quart de travail d'un agent survient, cet agent termine habituellement le service en cours avant de quitter le centre. Le comportement des autres agents demeure inchangé. Dans notre modèle, ceci se produit uniquement lorsque le nombre d'agents dans un groupe devient inférieur au nombre d'agents occupés dans ce même groupe. Un certain nombre d'agents est alors marqué pour quitter le groupe dès qu'ils deviennent libres tandis que les autres agents se comportent comme si de rien n'était. Notre estimé du taux d'occupation tient bien entendu compte de ces agents fantôme. Si tel n'était pas le cas, il serait possible d'obtenir des taux d'occupation supérieurs à 100% dans certains modèles où les agents sont très occupés.

Simuler cet aspect exige de créer et maintenir à jour un objet distinct pour chaque agent dans le modèle, ce qui accroît bien entendu le temps d'exécution du programme s'il y a beaucoup d'agents. Une solution à ce problème consiste à implanter les groupes d'agents simplement en utilisant des compteurs donnant le nombre de membres dans chaque état. Mais si nous implantons les groupes de cette façon, nous devons simplifier le modèle, car il n'est plus possible de savoir si, à la fin d'un service donné, l'agent concerné est marqué pour quitter le groupe ou pas. Dans ce cas, lorsque $N_i(t) < N_{\mathrm{B},i}(t)$ pour un certain groupe i, les agents occupés terminent les services et le groupe n'accepte aucun contact tant que $N_i(t) \leq N_{\mathrm{B},i}(t)$.

Intuitivement, ceci ne devrait pas trop affecter les résultats. Par contre, nous avons

constaté qu'il y avait bel et bien un impact tandis que nous comparions les résultats de notre simulation avec ceux d'un modèle développé par les gens de Bell Canada avec un autre logiciel. En effet, il se peut qu'un agent qui acceptait un contact dans le modèle détaillé à un temps t le refuse au même moment avec cette approximation. Cela réduit le temps total effectif de disponibilité des agents et peut bien entendu provoquer une diminution du niveau de service. Le taux d'occupation des agents pourrait également être diminué si les contacts qui doivent attendre plus longtemps abandonnent au lieu d'être servis. Mais sachant que l'approximation réduit le temps de disponibilité des agents, il se peut aussi que le taux d'occupation soit plus élevé avec l'approximation qu'avec le modèle détaillé.

Pour illustrer cela, prenons un modèle avec un seul type de contact, un seul groupe d'agents et pour lequel les arrivées suivent un processus de Poisson et les temps de service sont exponentiels. Le temps de patience des contacts qui ne sont pas servis immédiatement est exponentiel avec une moyenne de cinq minutes et il n'y a pas d'abandon immédiat. Les autres paramètres de ce modèle sont donnés dans le tableau 7.I dont chaque ligne correspond à une période de la journée. Les colonnes donnent respectivement l'heure de début, le nombre moyen d'arrivées, le temps moyen de service et le nombre d'agents, pour la période concernée. La dernière ligne donne le nombre total moyen d'arrivées et le temps moyen de service global.

Nous simulons 100 réplications indépendantes avec des objets représentant chaque agent ainsi qu'avec des compteurs. Les résultats donnés au tableau 7.II confirment notre raisonnement : le niveau de service est plus bas avec l'approximation qu'avec le modèle détaillé. Parfois, l'écart entre les deux modèles est 10% si bien que la baisse est significative. De plus, le taux d'occupation diffère légèrement entre les deux modèles. Comme nous pouvions nous y attendre, les résultats ne commencent à différer entre les deux modèles qu'au moment où le nombre d'agents diminue entre deux périodes. Mais contrairement à ce qu'on pourrait imaginer, la différence est assez importante.

Ainsi, il vaut habituellement mieux utiliser des objets pour représenter chaque agent

Tableau 7.I – Paramètres du centre de contacts utilisé pour examiner l'impact de l'implantation des groupes d'agents

Période	$\mathbb{E}[A_p]$	$1/\mu_p$	N_p
08:00	328	495	78
08:30	462	618	131
09:00	693	618	213
09:30	779	665	263
10:00	825	677	286
10:30	817	676	297
11:00	816	669	294
11:30	770	682	288
12:00	679	683	262
12:30	688	672	259
13:00	698	651	253
13:30	714	658	257
14:00	685	680	270
14:30	668	671	254
15:00	654	660	255
15:30	654	659	244
16:00	650	664	248
16:30	601	680	224
17:00	386	743	154
17:30	310	662	119
18:00	254	672	91
18:30	227	660	80
Journée	13 355	663	—

Tableau 7.II – Comparaison des deux modèles de groupes d'agents

	Modèle détaillé		Modèle approximatif	
Période	Niveau de service	Taux d'occupation	Niveau de service	Taux d'occupation
08:00	74%	80%	74%	80%
08:30	62%	92%	62%	92%
09:00	70%	93%	70%	93%
09:30	71%	96%	71%	96%
10:00	60%	99%	60%	99%
10:30	69%	98%	69%	98%
11:00	73%	98%	72%	98%
11:30	82%	98%	80%	98%
12:00	82%	97%	72%	97%
12:30	83%	97%	82%	97%
13:00	79%	97%	77%	97%
13:30	83%	97%	83%	97%
14:00	94%	94%	94%	94%
14:30	85%	97%	80%	97%
15:00	84%	97%	81%	97%
15:30	88%	96%	87%	96%
16:00	88%	95%	88%	95%
16:30	80%	97%	71%	98%
17:00	74%	97%	42%	99%
17:30	67%	97%	45%	98%
18:00	63%	97%	43%	98%
18:30	57%	97%	46%	98%
Journée	77%	96%	73%	97%
Temps d'exécution		46,3s		35,66s

puisque ce modèle est plus proche de la réalité. Comme le montre la dernière ligne du tableau, le modèle détaillé prend un peu plus de temps à exécuter que le modèle approximatif, mais cette différence de temps n'est pas très importante. Nous avons évalué ce temps avec 1 000 réplications, car les temps étaient trop semblables avec 100 réplications.

7.5 Prédiction des temps d'attente

Il devient de plus en plus populaire, dans les centres de contacts, d'annoncer aux clients une prédiction de leur temps d'attente au moment de leur arrivée dans le système. Cela incite certains clients à abandonner plutôt qu'attendre et augmente la satisfaction des clients restants qui savent davantage à quoi s'attendre. Les prédictions peuvent également influencer certaines politiques de routage et déterminer si un client se voit offert la possibilité d'être rappelé plus tard.

En général, obtenir une prédiction exacte des temps d'attente est difficile. Mais si tous les temps de service sont indépendants avec une moyenne de $1/\mu$, s'il n'y a aucun abandon et si tous les agents peuvent servir tous les contacts, le temps d'attente espéré peut être prédit par la formule $(Q+1)/(N\mu)$ où Q est le nombre de contacts en attente au moment de la prédiction et N est le nombre d'agents [39]. Ceci se justifie par le fait que sous ces hypothèses, le temps d'attente obéit à une loi Erlang de paramètres $Q+1$ et $1/(N\mu)$. Par contre, comme le montre [39], l'analyse devient beaucoup plus complexe dès qu'il y a des abandons, Dans ce dernier cas, le temps d'attente suit une somme d'exponentielles avec des paramètres différents. S'il y a plusieurs types de contacts et plusieurs groupes d'agents, seules des approximations sont disponibles. Les articles [43, 44] proposent de telles approximations se fondant par exemple sur le dernier temps d'attente observé.

Il n'existe aucune approximation universelle si bien qu'il faut permettre à l'utilisateur de choisir comment les temps d'attente sont estimés. Par contre, nous avons voulu fournir

une approximation par défaut. Pour ce faire, nous en avons comparé plusieurs et choisi celle qui donnait les meilleurs résultats sur des exemples de test. Dans cette section, nous décrivons les estimateurs testés, les comparons sur un modèle simple avec un type de contact, puis sur un modèle avec plusieurs types de contacts.

7.5.1 Estimateurs comparés

Temps d'attente précédent. Nous pouvons estimer le temps d'attente d'un nouveau contact en prenant soit le dernier temps d'attente observé au début d'un service, soit celui observé après la fin d'un service. Dans ces deux cas, nous pouvons ou non considérer le dernier temps d'attente observé après un abandon en plus du temps observé avant ou après un service. Dans le cas où les contacts sont de plusieurs types, pour prédire le temps d'attente d'un contact de type k, nous pouvons prendre le dernier temps d'attente d'un contact de type k ou encore le dernier temps d'attente quel que soit le type.

Plus long temps d'attente actuel. L'estimation est donnée par le plus long temps d'attente parmi tous les contacts en file au moment de la prédiction. Dans le cas d'une file premier arrivé premier servi, cela correspond au temps d'attente du premier contact de la file. Dans le cas où il y a plusieurs types de contacts, nous pouvons prendre le contact de type k en file qui a attendu le plus longtemps ou encore le contact avec le plus long temps d'attente, quel que soit son type.

Temps d'attente espéré. Dans le cas où il n'y a qu'un seul type de contact, nous pouvons utiliser la formule $(Q+1)/(N\mu)$ pour obtenir une prédiction du temps d'attente. Cette approximation devrait être bonne s'il y a peu d'abandons et si les temps de service sont exponentiels. Pour généraliser l'approximation au cas où $K > 1$, nous devons supposer que chaque type de contact k possède un groupe d'agents k qui est dédié au service des contacts de ce type. Le temps d'attente espéré devient alors $(Q_k+1)/(N_k\mu_k)$ où Q_k est le nombre de contacts de type k en attente au moment de la prédiction, N_k est le

nombre d'agents capables de servir des contacts de type k et $1/\mu_k$ est le temps de service moyen des contacts de type k.

Temps d'attente espéré tenant compte des abandons. S'il y a des abandons, la formule précédente sur-estime le temps d'attente espéré. Pour remédier à ce problème, nous pouvons utiliser la formule $\beta(Q+1)/(N\mu)$ présentée dans [44], où β est un facteur de correction calculé comme suit. Soit $F_{\mathrm{P}}(x)$ la fonction de répartition du temps de patience des contacts et λ le taux d'arrivée. Le facteur de correction est donné par $\beta = wN\mu/q$ où w et q sont le temps d'attente et la taille de la file observés à l'état stationnaire. Les constantes w et q sont obtenues en résolvant les équations

$$\frac{\lambda(1-F_{\mathrm{P}}(w))}{N\mu} = 1$$

et

$$q = \lambda \int_0^w (1-F_{\mathrm{P}}(x))dx.$$

Par contre, il faut que $\lambda/(N\mu) > 1$ pour qu'il y ait une solution à l'équation permettant de trouver w. Ainsi, la méthode ne s'applique que pour certaines valeurs des paramètres du modèle.

7.5.2 Comparaison des estimateurs sur un modèle simple

Ces estimateurs sont faciles à implanter et ne ralentissent pas beaucoup la simulation. Pour obtenir le dernier temps d'attente, il est nécessaire d'enregistrer un observateur auprès de chacune des files d'attente afin d'être averti dès qu'un contact quitte la file, pour en obtenir son temps d'attente. Mais le dernier temps d'attente est obtenu en temps constant lorsqu'il est demandé. Obtenir le temps d'attente espéré se fait aussi en temps constant si Q_k et N_k sont calculables dans $\mathcal{O}(1)$. Cette hypothèse est réaliste, car habituellement, le modèle est organisé de telle sorte que les contacts en attente sont répartis en K files et que chaque file ne contient des contacts que d'un seul type. Sous cette

même hypothèse, le plus long temps d'attente actuel pour les contacts de type k peut être obtenu en temps constant. Par contre, obtenir le plus long temps d'attente parmi tous les contacts se fait dans $\mathcal{O}(K)$.

Pour comparer ces approximations en pratique, prenons d'abord le modèle simple de la section 3.5.1, sans abandon. Pour chaque contact simulé, nous avons calculé les différentes prédictions W_P du temps d'attente et les avons comparées avec le temps d'attente W observé au moment du début du service ou de l'abandon. Nous avons calculé la moyenne des temps d'attente prédits pour la comparer à la moyenne des temps d'attente observés. Nous avons également calculé l'erreur quadratique moyenne (EQM) afin d'estimer $\mathbb{E}[(W - W_P)^2]$ pour chaque méthode testée. Le temps d'attente et l'erreur quadratique moyens sont estimés sur tous les temps d'attente de 100 réplications.

Le tableau 7.III montre les résultats avec ces paramètres. Dans cet exemple, la valeur moyenne de la prédiction est proche de la moyenne des temps observés. La formule $(Q+1)/(N\mu)$ a la plus petite erreur quadratique moyenne parmi les quatre estimateurs testés et est alors le meilleur estimateur du temps d'attente pour cet exemple puisque les hypothèses sur laquelle elle est fondée sont presque vérifiées. La formule ne permet pas d'obtenir une erreur de prédiction de 0, car elle calcule l'espérance et non pas le temps d'attente exact. De plus, la formule n'est pas exacte si le nombre d'agents du modèle change pendant l'attente, ce qui se produit si un client arrive pendant une période et est servi pendant la suivante.

Prendre le plus long temps passé en file jusque-là ou le dernier temps d'attente observé avant le début d'un service est à peu près équivalent du point de vue de l'erreur de prédiction. Le bon fonctionnement de ces approximations pourrait s'expliquer par le fait que deux contacts arrivant à des moments semblables entrent dans un système dont l'état est semblable si bien que leur temps d'attente se ressemblent. Plus précisément, si nous utilisons le temps d'attente observé d'un contact C_1 pour prédire le temps d'attente du nouveau contact C_2, plus les instants d'arrivée de C_1 et de C_2 sont proches, plus la prédiction du temps d'attente de C_2 en utilisant le temps de C_1 est précise. Si nous

prenons comme prédiction le dernier temps d'attente d'un contact servi, il peut exister des contacts en cours de service qui sont arrivés après ce contact et qui fourniraient une prédiction plus exacte. Ainsi, l'erreur de prédiction est plus grande si nous prenons le temps d'attente du dernier contact servi que si nous utilisons celui du dernier contact ayant quitté la file.

Si nous modélisons les abandons, lorsque la prédiction du temps d'attente est donnée par le dernier temps d'attente observé, nous pouvons choisir d'inclure ou non les abandons dans les observations. Le tableau 7.IV présente les résultats pour le modèle de la section 3.5.1 avec les abandons. Nous n'avons pas testé l'estimateur $\beta(Q+1)/(N\mu)$, car nous n'avons pas pu calculer la constante β avec les paramètres de notre modèle. L'erreur de prédiction est plus petite que dans le tableau précédent, car les temps d'attente avec abandons sont plus petits et donc moins variables.

Encore une fois, la formule $(Q+1)/(N\mu)$ fournit la meilleure approximation, probablement en raison du faible pourcentage d'abandons dans ce modèle. Comme dans le cas précédent, le plus long temps d'attente est la meilleure prédiction après $(Q+1)/(N\mu)$. Quant à la prédiction utilisant le dernier temps d'attente observé, il faut encore une fois prendre celui avant le début d'un service. Tenir compte des temps avant abandon dans la prédiction augmente l'erreur de prédiction, car vu la faible fréquence des abandons, un temps d'attente conditionnel au service est beaucoup plus représentatif de la distribution du temps d'attente inconditionnel que ne l'est un temps d'attente avant abandon.

Tableau 7.III – Comparaison des estimateurs de temps d'attente sur l'exemple de la section 3.5.1, sans abandon

Estimateur	$\bar{W}_P$	EQM
Temps d'attente observé	37s	0
$(Q+1)/(N\mu)$	37s	638
Dernier temps d'attente avant début de service	33s	1 435
Dernier temps d'attente après la fin de service	33s	2 076
Plus long temps d'attente actuel	32s	1 283

Tableau 7.IV – Comparaison des estimateurs de temps d'attente sur l'exemple de la section 3.5.1, avec abandons

Estimateur	$\bar{W}_P$	EQM
Temps d'attente observé	9s	0
$(Q+1)/(N\mu)$	11s	286
Dernier temps d'attente avant début de service	8s	464
Dernier temps d'attente après la fin de service	8s	702
Dernier temps d'attente avant début de service ou abandon	7s	500
Dernier temps d'attente après la fin de service ou abandon	7s	719
Plus long temps d'attente actuel	7s	375

7.5.3 Comparaison des estimateurs sur un modèle avec plusieurs types de contacts

Prenons maintenant un modèle de centre de contacts avec trois types de contacts, trois groupes d'agents et 18 périodes d'une demi-heure qui s'inspire d'un centre d'appel réel. Pour $k = 1,2,3$, les contacts de type k arrivent selon un processus de Poisson avec taux $\lambda_{k,p}$ pendant la période p et les temps de service sont i.i.d. et exponentiels de moyenne $1/\mu_k$. Les temps de patience sont i.i.d. exponentiels avec moyenne de 5 minutes quel que soit le type de contact. Pour $i = 1,2,3$, le nombre d'agents dans le groupe i pendant la période p est fixé à $N_{i,p}$.

Lorsqu'un contact de type 1 arrive, un agent libre est recherché dans les groupes 1, 2 puis 3, dans cet ordre. Si aucun agent libre n'est trouvé, le contact est mis en attente. Pour les contacts de type 2, l'ordre de parcours des groupes d'agents est 2, 3, puis 1. L'ordre de parcours pour le type 3 est quant à lui 3, 1 et 2. Lorsqu'un agent devient libre, il choisit le contact qui a attendu le plus longtemps en file. Le tableau 7.V donne les paramètres pour cet exemple. Chaque ligne décrit une période tandis que chaque colonne correspond à un type de paramètre. Les taux d'arrivées donnés correspondent au nombre moyen d'arrivées par période.

Tableau 7.V – Paramètres pour l'exemple avec trois types de contacts et trois groups d'agents

Temps de patience moyen	5min
(μ_1, μ_2, μ_3)	(650s, 630s, 660s)

p	t_{p-1}	$\lambda_{1,p}$	$\lambda_{2,p}$	$\lambda_{3,p}$	$N_{1,p}$	$N_{2,p}$	$N_{3,p}$
1	08:00	187	93	22	64	31	16
2	08:30	242	121	28	82	40	19
3	09:00	355	178	41	121	59	25
4	09:30	395	198	46	134	65	28
5	10:00	415	207	48	140	68	29
6	10:30	422	211	49	143	69	29
7	11:00	411	206	48	139	68	29
8	11:30	376	188	44	127	62	27
9	12:00	344	172	40	117	57	25
10	12:30	337	168	39	114	55	24
11	13:00	343	171	40	116	56	25
12	13:30	339	169	39	115	56	24
13	14:00	332	166	39	112	55	24
14	14:30	320	160	37	108	53	24
15	15:00	311	156	36	106	51	23
16	15:30	316	158	38	108	52	24
17	16:00	299	150	35	102	50	23
18	16:30	256	128	30	87	42	20
Total		6 000	3 000	700			

Le tableau 7.VI présente les résultats pour ce modèle. Les trois premiers groupes de deux colonnes donnent les résultats pour les trois types de contacts séparément tandis que le dernier groupe de deux colonnes donne les résultats pour tous les contacts. Une ligne du tableau est consacrée à chaque estimateur et fournit le temps d'attente moyen prédit ainsi que l'erreur quadratique moyenne. Nous remarquons que l'approximation du temps d'attente espéré fonctionne moins bien avec cet exemple étant donné que les hypothèses sur lesquelles elle repose ne sont pas du tout satisfaites. Le meilleur estimateur est le dernier temps d'attente observé avant le début de service. Comme dans l'exemple précédent, tenir compte des temps avant abandon dans la prédiction augmente légèrement l'erreur d'estimation.

Nous remarquons aussi que tenir compte du type de contact dans la prédiction augmente l'erreur quadratique moyenne. Cela s'explique par le fait que si nous restreignons le type du contact, il faut effectuer la prédiction avec un contact arrivé plus tôt.

En conclusion, l'estimateur utilisé par défaut devrait être le plus long temps d'attente actuel, sans tenir compte du type de contact. Par contre, dans certains modèles où le temps d'attente diffère beaucoup d'un type de contact à l'autre, il pourrait être bénéfique de prendre le plus long temps d'attente de type k. Évidemment, si $K = I = 1$, le temps d'attente espéré est tout indiqué.

7.6 Extensibilité du simulateur générique

Les classes de la bibliothèque ContactCenters sont extensibles afin de s'adapter à la plus grande gamme possible de centres de contacts. Lorsque nous construisons un simulateur ne nécessitant aucune programmation, seulement la configuration par des fichiers XML, il est bien entendu nécessaire d'en restreindre les possibilités pour avoir un nombre raisonnable de paramètres configurables.

Évidemment, utiliser un simulateur précompilé est beaucoup plus simple que construire un programme Java complet, même en partant d'un exemple existant. Il serait donc dom-

Tableau 7.VI – Comparaison des estimateurs de temps d'attente sur un exemple avec trois types de contacts

	CT1		CT2		CT3		Global	
	Temps	EQM	Temps	EQM	Temps	EQM	Temps	EQM
Temps d'attente observé	10s	0	10s	0	10s	0	10s	0
$(Q+1)/(N\mu)$	13s	103	16s	218	20s	396	15s	160
Dernier temps d'attente avant début de service	9s	82	9s	82	9s	84	9s	82
Dernier temps d'attente avant début de service ou abandon	17s	647	17s	630	17s	639	17s	641
Temps d'attente du premier contact en file	9s	75	9s	75	9s	77	9s	75
Dernier temps d'attente de type k avant début de service	9s	87	9s	97	9s	153	9s	95
Dernier temps d'attente de type k avant début de service ou abandon	9s	97	9s	105	9s	155	9s	104
Temps d'attente du premier contact de type k en file	9s	80	7s	102	3s	230	8s	97

mage de devoir créer un nouveau simulateur à chaque fois que quelque chose manque dans les simulateurs génériques existants. Il est ainsi nécessaire que le simulateur générique aussi soit extensible.

L'héritage est une solution naturelle en Java pour rendre un programme ou une bibliothèque extensible. Par contre, le simulateur générique de centres d'appels doit être utilisable et extensible avec le moins de programmation Java possible. En particulier, nous avons voulu éviter qu'il soit nécessaire de construire une sous-classe de simulateur chaque fois qu'il est nécessaire d'utiliser une loi de probabilité, une politique de routage, un processus d'arrivées ou une politique de composition d'appels sortants personnalisés. Pour permettre cette extensibilité, nous avons combiné le mécanisme de réflexion de Java et le patron de conception des fournisseurs de services. Le premier mécanisme permet en particulier de construire des objets à partir de classes dont le nom n'est connu qu'à l'exécution du programme tandis que le second permet à différentes classes de se greffer au système de façon dynamique. Dans cette section, nous examinons ces mécanismes et leur intégration dans le simulateur générique avant d'aborder d'autres mécanismes

complémentaires que nous avons envisagés.

7.6.1 Réflexion

En général, la réflexion permet de travailler avec des objets dont la classe, les champs et les méthodes ne sont connus qu'à l'exécution. Étant donné que l'usage abusif de ce mécanisme peut dégrader la performance d'un programme, nous ne l'avons utilisé que pour construire certains objets dont la classe est connue à l'exécution seulement.

La réflexion est principalement utilisée pour créer les objets représentant les lois de probabilité et les générateurs de nombres aléatoires. Pour générer des nombres aléatoires avec SSJ et ContactCenters, nous produisons d'abord des nombres suivant la loi uniforme puis les transformons, la plupart du temps par inversion, pour qu'ils suivent la loi souhaitée. L'utilisateur désireux de changer le type de générateur uniforme par défaut spécifie le nom de la classe à utiliser. Grâce à la réflexion, le programme de simulation crée un objet représentant cette classe et utilise cet objet pour créer les générateurs.

Pour représenter une loi de probabilité, dans le fichier de configuration XML, l'utilisateur indique le nom d'une classe représentant la loi voulue ainsi qu'un vecteur donnant les paramètres pour la loi choisie. De façon semblable aux générateurs de nombres aléatoires uniformes, le simulateur crée, par réflexion, un objet représentant la classe de loi de probabilité sélectionnée par l'utilisateur et utilise cet objet pour construire la loi, en recherchant et en appelant un constructeur acceptant les paramètres fournis.

D'une façon semblable, le générateur de nombres aléatoires non uniformes est obtenu à partir d'un nom de classe optionnel fourni par l'utilisateur ainsi que la loi de probabilité et le générateur de nombres uniformes construits au préalable. Le nom de la classe du générateur affecte la méthode utilisée pour générer les nombres aléatoires. Si ce nom est omis, le générateur par défaut obtenu utilise l'inversion.

Le format des paramètres pour les lois de probabilité et les générateurs de nombres aléatoires est spécifié en détails dans le manuel d'utilisation du simulateur générique, inclus dans la documentation de ContactCenters [14]. Les lois de probabilité disponibles et

leurs paramètres ainsi que les différentes méthodes de génération des variables aléatoires sont donnés dans la documentation de SSJ [55].

Avec ce mécanisme, presque toutes les lois de probabilité de SSJ peuvent être utilisées et l'utilisateur peut facilement créer sa propre loi et l'employer dans le simulateur de centres de contacts. Un avantage important de cette technique est qu'aucun code spécifique à chaque loi et à chaque générateur n'est nécessaire pour établir le pont entre les générateurs de nombres aléatoires de SSJ ou de l'utilisateur et le simulateur générique de centres d'appels.

7.6.2 Fournisseurs de services

Le mécanisme de réflexion convient bien pour construire les lois de probabilité et les générateurs de nombres aléatoires, car il a été possible de trouver une signature typique englobant presque tous les constructeurs possibles. Par contre, il existe des cas où une telle signature est inexistante ou si complexe que l'appel du constructeur devient trop lourd. Par exemple, certaines politiques de routage demandent des listes de types de contacts et de groupes d'agents tandis que d'autres nécessitent des matrices donnant des priorités pour chaque paire de types de contacts et de groupes d'agents. Pour appeler un constructeur par réflexion, il faudrait rechercher un constructeur acceptant comme arguments tous les paramètres possibles pour une politique de routage. Une solution possible est de regrouper tous ces arguments dans un objet représentant les paramètres de routage. Mais dans tous les cas, un constructeur de ce genre, avec des paramètres qui ne sont pas forcément tous utilisés, sèmerait la confusion chez une personne utilisant la politique de routage dans un simulateur utilisant ContactCenters directement, sans passer par le simulateur générique.

Plutôt que procéder ainsi, nous avons opté pour utiliser des fournisseurs de services définis comme suit. Un *service* est un ensemble d'interfaces et de classes abstraites accomplissant une action donnée, par exemple construire un objet représentant une politique de routage et retrouver ou construire les paramètres adéquats pour la configurer.

Un *fournisseur de service* [78] est une implantation des interfaces ou des sous-classes des classes abstraites correspondantes à un service donné. Un tel fournisseur peut par exemple se charger de construire une politique de routage bien précise. Java 6 fournit un mécanisme permettant de retrouver tous les fournisseurs correspondant à un service donné, qui peuvent être ajoutés sans recompiler le programme. Les fournisseurs sont placés dans des archives qui peuvent être retrouvées à l'exécution en cherchant à certains endroits prédéfinis.

Dans notre simulateur, la politique de routage est représentée par un objet de la classe `Router`. L'interface `RouterFactory` représente quant à elle un service capable de construire un routeur en utilisant l'objet représentant le modèle de centre d'appels ainsi que l'objet contenant les paramètres de routage extraits depuis le fichier XML. La méthode `createRouter` de cette interface retourne l'objet représentant le routeur ou une référence `null` si le routeur ne peut pas être construit par l'implantation particulière utilisée. Une implantation de `RouterFactory` décide si elle peut construire le routeur en examinant les paramètres de routage, tout particulièrement une chaîne de caractères donnant le nom de la politique voulue. La méthode `createRouter` peut accomplir toute initialisation nécessaire, par exemple retrouver ou construire une matrice de priorités.

Pour construire le routeur, la classe `ServiceLoader` de Java 6 est utilisée pour énumérer toutes les instances de `RouterFactory` enregistrées et les tester une par une pour trouver la première capable de construire le routeur. De cette façon, l'utilisateur peut facilement implanter une nouvelle politique de routage en créant une sous-classe de `Router` puis une implantation de `RouterFactory` pour construire le nouveau routeur. Le manuel d'utilisation du simulateur générique, inclus dans la documentation de ContactCenters [14], fournit un exemple d'une telle politique personnalisée ainsi que des détails additionnels sur l'interface `RouterFactory`.

Un mécanisme semblable est utilisé pour construire les processus d'arrivées et les politiques de composition des appels sortants. Seuls le nom de l'interface et de la méthode de construction changent.

7.6.3 Autres mécanismes envisageables

Il existe deux autres mécanismes que nous avons pensé utiliser pour rendre notre système plus extensible mais que nous n'avons pas mis en place : un langage de scripts et la programmation orientée aspects. Avec la première possibilité, l'utilisateur peut incorporer des scripts dans les fichiers de configuration du simulateur. Ces scripts, qui peuvent par exemple décrire une politique de routage complexe, sont compilés lors de l'initialisation et exécutés pendant la simulation. Par contre, utiliser de tels scripts risque de beaucoup ralentir le simulateur. La gestion des scripts pourrait être implantée avec le Java Scripting API [41], intégré à Java 6. Le simulateur n'utilise pas encore de tels scripts, car le besoin ne s'est pas encore fait sentir.

Nous pouvons également penser à appliquer la programmation orientée aspects [48] pour modulariser le simulateur. Ce paradigme introduit le concept d'*aspect* qui consiste en un ensemble de fragments de code appelés *advice* et se greffant à des endroits du programme appelés *points de jointure* (ou *join points* en anglais). Un aspect sert à définir un comportement couvrant plusieurs endroits dans le programme, par exemple la journalisation d'événements, le traitement d'erreurs, un aspect de modélisation tel que les recours qui nécessiteraient d'être pris en compte à plusieurs endroits pendant la simulation, etc. Nous pourrions mettre en œuvre la programmation orientée aspects en utilisant l'outil AspectJ [2] qui ajoute ce concept à Java. Par contre, avec les outils que nous connaissons pour la programmation orientée aspects en Java, l'incorporation d'un aspect demande la recompilation de tout le simulateur ; l'utilisateur ne peut pas activer ou désactiver un aspect à volonté sans tout recompiler. De plus, la couche ajoutée par la programmation orientée aspects pourrait réduire la performance du simulateur.

7.7 Simplification de l'entrée des données

Le problème principal du simulateur générique réside sans aucun doute dans sa complexité d'utilisation. Écrire un fichier XML est certes plus simple que rédiger un pro-

gramme Java, mais cela pose malgré tout des difficultés. L'idéal serait de disposer d'une interface graphique permettant à la fois de définir les paramètres du simulateur, d'exécuter des simulations et d'examiner les résultats. Cette interface pourrait aussi donner accès aux formules d'approximation les plus courantes, aux algorithmes d'optimisation existants, etc., qui ont déjà été implantés par un autre étudiant. Nous avons supervisé des tentatives de développement d'une telle interface, mais les étudiants qui devaient faire cela disposaient d'une période de temps trop courte.

Même sans interface graphique élaborée, nous pouvons faciliter l'entrée des paramètres pour l'utilisateur en construisant un Schéma XML [51, 77] pour spécifier la structure d'un fichier de paramètres dans un langage compréhensible pour la machine. Un schéma indique quels éléments sont autorisés dans un fichier de paramètres, quels attributs sont permis pour chaque élément et quelles structures hiérarchiques peuvent être construites. Cela permet de valider les fichiers de paramètres de façon plus robuste et normalisée qu'un programme écrit manuellement pour traiter chaque paramètre séparément. Un schéma permet aussi à des éditeurs XML tels que <oXygen/> [69], Eclipse [27], etc. de guider l'utilisateur dans la construction ou la modification d'un fichier de paramètres et de valider eux-mêmes le fichier, sans démarrer pour cela le simulateur. Un schéma peut aussi contenir de la documentation pour chaque élément et attribut spécifié.

La construction d'un schéma XML pour un format complexe n'est pas triviale. Le schéma choisi doit englober toutes les fonctionnalités actuelles du simulateur générique tout en laissant place à l'ajout de nouvelles fonctions. Idéalement, un fichier de configuration XML validé par ce schéma devrait permettre de démarrer le simulateur sans erreur, mais ceci est impossible en raison de contraintes impliquant plusieurs éléments qui ne peuvent pas être décrites par le schéma. Le format des fichiers spécifié par le schéma ne doit pas être trop verbeux puisque les fichiers pourront toujours être édités manuellement. Il est également crucial que le format ne change pas profondément par la suite puisque plusieurs composantes telles que le simulateur, les logiciels d'optimisation, une éventuelle interface graphique pour éditer les paramètres, etc., en dépendent.

Nous avons développé un tel schéma et avons adapté le simulateur pour l'utiliser, ce qui a demandé de remplacer le mécanisme de lecture de paramètres utilisé auparavant, d'écrire des méthodes de support et d'adapter le code construisant le modèle de notre simulateur générique. Enfin, quelques programmes ont été nécessaires pour effectuer la conversion de l'ancien format vers le nouveau.

Tout d'abord, nous avons remplacé notre propre système de lecture de paramètres par *Java Architecture for XML Binding* (JAXB, [47]), un outil de liaison XML intégré à Java 6. JAXB permet de transformer un document XML suivant un schéma déterminé en une hiérarchie d'objets et inversement de produire un document XML depuis des objets. La validation via un schéma XML peut être effectuée lors de ces deux opérations. Plusieurs autres outils semblables à JAXB existent, mais ils comportent souvent des lacunes comme l'impossibilité de générer les classes Java à partir d'un schéma XML, l'absence de prise en charge des nouveautés de Java 5, etc. ou contraignent à produire du code ou des fichiers XML trop verbeux.

Nous avons ensuite dû écrire des méthodes de support pour convertir les objets les plus complexes représentant des tableaux bidimensionnels et des lois de probabilité vers des objets compatibles avec ContactCenters. Cette tâche accomplie, il nous a fallu adapter le simulateur pour utiliser la nouvelle hiérarchie d'objets de paramètres. Nous avons par la même occasion modifié son architecture pour la rendre plus extensible.

7.8 Amélioration de la clarté des messages d'erreur

Un autre problème avec le simulateur générique était l'affichage de messages d'erreur imprécis, parfois même incompréhensibles à moins d'examiner le code source du programme. Par exemple, les paramètres pour les durées de service sont donnés, dans les fichiers de configuration XML destinés au simulateur générique, à l'intérieur d'éléments XML nommés `serviceTime`. Supposons que l'utilisateur ait employé l'élément nommé `servTime` au lieu de `serviceTime`. D'anciennes versions du simula-

teur affichaient alors un message semblable à celui sur la figure 7.5. Sur cette figure et les suivantes, certaines parties des noms de paquets et de fichiers qui sont longs inutilement ont été remplacées par des points de suspension.

Ce message indique que le système de lecture des paramètres n'a pas pu traiter l'élément `servTime` faisant partie de l'élément `inboundType` représentant un type de contact. Le message d'erreur n'était pas clair, car il faisait référence à des éléments internes du système de lecture des paramètres et n'indiquait pas l'emplacement du problème dans le fichier.

Grâce à l'utilisation du schéma XML, le message d'erreur affiché par la version actuelle du simulateur, en cas de problème équivalent, est celui sur la figure 7.6. Le message indique que l'élément `servTime` n'est pas accepté par le schéma XML du fichier de paramètres, propose des noms d'éléments acceptables et indique l'emplacement du problème dans le fichier. Utiliser un schéma XML améliore ainsi la clarté des messages d'erreur.

Par contre, si une erreur survient au moment de la construction du simulateur, après la lecture des paramètres et leur conversion en objets par JAXB, l'emplacement du paramètre causant l'erreur dans le fichier XML ne peut pas être retrouvé. De plus, il arrive souvent qu'une partie du programme dans laquelle une erreur est détectée n'ait pas accès à toutes les informations nécessaires pour produire un message d'erreur précis. Par exemple, si le taux donné par l'utilisateur pour une loi exponentielle est négatif, le constructeur dans lequel l'erreur est détectée, qui se trouve dans SSJ, n'a aucune in-

```
Exception in thread main ...xmlconfig.ParamReadException :
In element mskccparams/inboundType/servTime, Could
not add the subelement, no appropriate adder method
addServTime, creater method createServTime or
dispatcher method defaultNestedElement found in class
...contactcenters.old.msk.InboundTypeParams
```

Figure 7.5 – Exemple de message affiché par d'anciennes versions du simulateur générique lorsqu'un élément inconnu était rencontré

```
The following problem occurred during unmarshalling.
[FATAL ERROR] cvc-complex-type.2.4.a : Invalid
content was found starting with element
'servTime'. One of 'serviceTime, conferenceTime,
preServiceTimeNoConf, serviceTimesMultTransfer,
transferTime, probTransfer, probTransferWait,
transferTarget, expectedWaitingTimeThresh,
expectedWaitingTimesMult, probVirtualQueue,
probVirtualQueueCallBack, patienceTimesMultNoVirtualQueue,
patienceTimesMultCallBack, serviceTimesMultNoVirtualQueue,
serviceTimesMultCallBack, arrivalProcess' is expected. at
file :.../singleType.new.xml, line 29, column 65
```

Figure 7.6 – Exemple de message affiché par la version actuelle du simulateur générique lorsqu'un élément inconnu est rencontré

formation à propos du contexte de simulation mis en place par ContactCenters. Il ne peut alors pas afficher un message indiquant pour quelle loi l'erreur a été commise et pour quel type d'appel. Cela produisait, dans d'anciennes versions du simulateur, des messages ressemblant à celui sur la figure 7.7. Ces messages étaient très difficiles à comprendre, à moins de connaître les détails de l'implantation du simulateur et de son mécanisme de lecture des paramètres depuis les fichiers XML. En plus, ils n'indiquaient rien quant à la source de l'erreur.

Pour traiter ce problème, nous avons tiré parti du mécanisme des exceptions de Java pour conserver toute l'information relative à une erreur. Lorsqu'un problème survient, par exemple un paramètre négatif pour une loi de probabilité, une exception est lancée

```
Exception in thread main ...xmlconfig.ParamReadException :
In text "-8" with parent mskccparams/inboundType/serviceTime,
Exception occurred during call to a method
Caused by ...xmlconfig.DistributionCreationException :
For ...probdist.ExponentialDist (-8),
java.lang.IllegalArgumentException : lambda <= 0
```

Figure 7.7 – Exemple de message d'erreur affiché par d'anciennes versions du simulateur lorsqu'un paramètre négatif imprévu était trouvé

avec un message décrivant le problème. Si une telle exception n'est pas attrapée par la méthode appelante, elle est propagée, ce qui signifie que la méthode appelante lance aussi une exception. Ce processus se répète souvent jusqu'à atteindre la méthode principale du programme et un message d'erreur s'affiche avec la classe de l'exception, son message descriptif et toute la pile d'appels de méthodes observée au moment de l'erreur.

Mais si l'exception est attrapée, un traitement spécial peut être effectué, par exemple créer une nouvelle exception, avec un nouveau message descriptif mais aussi un lien vers l'ancienne exception. Cette nouvelle exception peut être attrapée par une autre méthode appelante qui la traite à son tour en créant encore une autre exception et un nouveau message descriptif ajoutant des détails sur l'origine de l'erreur. Le message peut par exemple indiquer que l'erreur concerne la loi des temps de service pour tel type k d'appel. De cette façon, le programme principal reçoit une chaîne d'exceptions et peut afficher une suite de messages de plus en plus spécifiques permettant à l'utilisateur de retracer la cause de l'erreur. Dans de tels cas, la pile d'appels, superflue, n'est pas affichée pour améliorer la concision des messages.

Avec ce mécanisme de gestion des erreurs, le message obtenu ressemble à celui sur la figure 7.8. La première ligne du message indique que l'erreur s'est produite lors de la création de l'usine abstraite produisant les contacts de type 0 ; le message concerne forcément le premier type de contact déclaré dans le fichier XML de configuration. La deuxième indique que l'objet représentant la loi de probabilité des temps de service ne peut pas être créé. La troisième ligne indique que l'erreur s'est produite lors de l'appel à un constructeur, donne le nom de la classe concernée et les paramètres ayant causé l'erreur. La dernière ligne fournit enfin la cause de l'erreur du constructeur, qui est un argument négatif.

```
...contactcenters.msk.model.CallCenterCreationException :
Cannot create call factory for call type 0

Caused by ...contactcenters.msk.model.CallFactoryCreationException
Cannot create service time distribution

Caused by ...xmlbind.DistributionCreationException :
An error occurred during call to constructor, for
distribution class ...probdist.ExponentialDist with
parameters (-8.0)

Caused by java.lang.IllegalArgumentException : lambda <=
0
```

Figure 7.8 – Exemple de message d'erreur affiché par la version actuelle du simulateur lorsqu'un paramètre négatif imprévu est trouvé

CONCLUSION

Dans cette thèse, nous avons examiné différentes façons d'augmenter l'efficacité des simulations de centres de contacts. Certaines de ces techniques sont même applicables à d'autres modèles. Par contre, aucune technique n'est parfaite pour tous les cas. Nous avons également tenté d'améliorer nos outils de simulation. Dans cette conclusion, nous récapitulons les contributions principales de chacun des chapitres de cette thèse et proposons des pistes de recherche.

D'abord, nous avons étudié des techniques pour rendre plus efficace la simulation d'une configuration d'un modèle particulier. Dans le chapitre 3, nous avons d'abord présenté une méthode permettant de combiner la stratification sur une variable aléatoire continue avec une ou plusieurs variables de contrôle linéaires. Nous avons vu que le coefficient de la variable de contrôle dépend de la variable sur laquelle nous stratifions et qu'il existe plusieurs méthodes différentes pour effectuer la combinaison. Utiliser un coefficient différent pour chaque strate est en général la méthode la plus efficace, même si un coefficient dépendant de la variable de stratification continue semble mieux intuitivement. Trouver de meilleures variables de contrôle que le nombre A d'arrivées pour les exemples de centres de contacts que nous avons testés demeure un problème de recherche encore ouvert. Nous ne sommes pas non plus parvenus à construire un exemple réaliste dans lequel choisir un coefficient pour la variable de contrôle dépendant de la variable continue de stratification est significativement plus avantageux qu'un coefficient dépendant de la strate.

Plusieurs autres techniques pourraient être étudiées pour réduire la variance sur une configuration donnée d'un modèle. En particulier, des méthodes quasi-Monte Carlo [57, 63] seraient applicables. Ces méthodes consistent à remplacer les variables aléatoires suivant la loi uniforme par des variables déterministes distribuées de façon plus uniforme que des variables aléatoires. Les points ainsi générés sont par la suite randomisés afin de pouvoir calculer des intervalles de confiance sur les estimateurs obtenus. Pour rendre

ces méthodes efficaces sur un modèle complexe de centre de contacts, dont la dimension est très élevée, il faudrait toutefois trouver une façon de réduire la dimension effective du problème. Par exemple, au lieu de générer les arrivées de façon séquentielle, nous pourrions générer le nombre total d'arrivées durant toute la journée, le nombre d'arrivées en avant-midi et en après-midi conditionnels à celui de toute la journée, etc.

Une autre technique prometteuse est l'échantillonnage stratégique qui consiste à modifier la densité des lois de probabilité du modèle pour ensuite multiplier les estimateurs obtenus par un rapport de vraisemblance. Le problème est de trouver un tel changement de densité et de prouver qu'il réduit effectivement la variance. Il existe en théorie un changement de paramètres permettant d'appliquer l'échantillonnage stratégique de façon à obtenir une variance nulle pour n'importe quelle chaîne de Markov [67]. Or, une simulation par événements discrets correspond à une chaîne de Markov. La variance nulle peut aussi être obtenue en théorie avec une variable de contrôle appliquée à chaque étape de la chaîne. Habituellement, calculer le changement de paramètres pour l'échantillonnage stratégique ou les coefficients ou les espérances des variables de contrôle est trop complexe en pratique, mais il est parfois possible d'approximer la loi qui donne une variance nulle.

Nous avons aussi examiné comment améliorer l'efficacité de la comparaison de configurations d'un même modèle. Pour cela, dans le chapitre 4, nous avons étudié les variables aléatoires communes pour estimer une différence. Nous avons appliqué la technique sur un exemple de centre de contacts, obtenu une réduction de la variance par rapport à des variables aléatoires indépendantes et comparé diverses méthodes de synchronisation, sans toutefois trouver une méthode se démarquant clairement de toutes les autres en général. Nous avons aussi développé des preuves mathématiques montrant que dans certains modèles, la variance de la différence du nombre de contacts servis ayant attendu moins de s et du nombre d'abandons en fonction de δ, si le temps de service moyen est multiplié par $1-\delta$, est dans $\mathcal{O}(\delta^{1-\varepsilon})$ pour $\varepsilon > 0$. Nous avons aussi montré que la variance de la différence du temps d'attente est dans $\mathcal{O}(\delta^2)$. En pratique, la va-

riance de la différence semble dans $\mathcal{O}(\delta)$ pour les deux premières mesures, ce qui est très près de $\mathcal{O}(\delta^{1-\varepsilon})$ étant donné que nous pouvons choisir ε arbitrairement petit.

Les estimateurs pour le niveau de service et le pourcentage d'abandons varient de façon discontinue par rapport aux paramètres testés. Construire des variantes continues de ces estimateurs est un problème ouvert. Une façon possible d'y parvenir serait d'utiliser une méthode Monte Carlo conditionnel qui consiste à remplacer une variable aléatoire par une espérance conditionnelle. Des preuves de convergence pourraient être développées pour ces nouveaux estimateurs, d'autres mesures de performance et d'autres paramètres.

Pour augmenter encore davantage l'efficacité, nous avons essayé de simplifier notre modèle de simulation. Le chapitre 5 a alors proposé un simulateur simplifié utilisant une chaîne de Markov en temps continu et l'uniformisation pour estimer la performance d'un centre de contacts plus rapidement qu'avec le simulateur détaillé par événements discrets mais avec davantage de précision qu'avec les approximations couramment utilisées. Ce simulateur est avantageux si le centre de contacts ne comporte pas trop de types de contacts ou de groupes d'agents et si le taux d'arrivée est élevé. Plus le modèle dévie de ces hypothèses, plus le gain obtenu en utilisant le simulateur simplifié au lieu du simulateur détaillé par événements discrets est petit. De plus, le simulateur simplifié suppose que les arrivées peuvent être modélisées comme un processus de Poisson et les temps de service et de patience peuvent être considérés exponentiels. Si tel n'est pas le cas, les estimateurs calculés par le simulateur simplifié sont biaisés, mais il faut garder à l'esprit que le simulateur simplifié est principalement utilisé pendant les premières phases d'une optimisation. La simulation par événements discrets permet de raffiner la solution obtenue à la fin du processus.

Par contre, il arrive parfois que le taux de transition soit trop élevé, occasionnant de nombreuses transitions fictives. Nous avons rendu le taux de transition adaptatif sans perdre l'uniformisation en partitionnant l'ensemble des états de la chaîne de Markov et en associant, à chaque sous-ensemble, un taux de transition et un index de recherche.

Cette idée permet en particulier d'utiliser une capacité maximale de file d'attente élevée afin de réduire la probabilité de contacts bloqués. Il suffit pour cela de partitionner l'ensemble d'états en utilisant des seuils sur la taille de la file. Alors, si la taille de la file d'attente est petite la plupart du temps, l'état de la CMTC se trouve souvent dans un sous-ensemble avec un taux de transition petit si bien que la capacité de la file élevée n'augmente pas trop le temps d'exécution. Lorsque les variables aléatoires communes sont utilisées, le taux de transition adaptatif ne nuit pas trop à la synchronisation si chaque configuration est simulée dans le même sous-ensemble d'états. Pour favoriser cela, il est bon de se limiter à un petit nombre de sous-ensembles.

Dans le chapitre 6, nous avons proposé une méthode de scission et de recombinaison permettant d'exploiter la similitude qui existe forcément entre les configurations semblables d'un même modèle. Nous avons généralisé une méthode existante pour prendre en charge des variables de décision entières et des paramètres multiples. Nous avons aussi discuté des problèmes posés par son implantation. Nous avons appliqué cette méthode à un modèle de centres de contacts utilisant une CMTC uniformisée dans le but d'estimer un sous-gradient. Nous avons obtenu un gain en performance intéressant par rapport à la simulation séparée de toutes les configurations s'il y a beaucoup de configurations similaires à traiter. Cela se produit en particulier si le taux d'occupation des agents est petit et s'il y a beaucoup d'agents répartis dans plusieurs groupes. En revanche, si le nombre d'agents varie beaucoup entre les configurations, la méthode revient à simuler chaque configuration séparément avec un coût additionnel en temps de calcul. De plus, la performance de la conversion en temps discret diminue si le nombre de groupes d'agents augmente. Le plus grand potentiel de la méthode réside ainsi dans le cas, que nous n'avons pas implanté, où il y a plusieurs périodes ou plusieurs quarts de travail pour les agents. Dans de tels cas, le nombre de configurations semblables se multiplie même si le nombre de groupes d'agents demeure modéré. Avec des adaptations additionnelles, la méthode pourrait aussi s'appliquer à d'autres modèles que notre CMTC, à condition que dans l'implantation, l'état puisse rapidement être cloné.

Enfin, le chapitre 7 a présenté les grandes lignes de l'architecture et les fonctionnalités principales de notre logiciel ContactCenters qui permet de simuler une grande gamme de centres de contacts. Nous nous sommes penchés sur le simulateur générique précompilé permettant de traiter des centres d'appels mixtes avec plusieurs types d'appels et groupes d'agents. Nous y avons aussi traité des problèmes principaux rencontrés pendant son développement. Nous y avons vu que la façon de gérer les groupes d'agents affecte parfois les résultats de simulation et que la prédiction des temps d'attente en général n'est pas simple bien que nous ayons trouvé une approximation convenant dans la plupart des cas. Nous avons amélioré l'extensibilité de notre outil pour permettre à l'utilisateur de personnaliser le routage, les arrivées et la composition d'appels avec un minimum de programmation Java. Nous avons aussi grandement simplifié l'entrée des paramètres pour l'utilisateur et clarifié les messages d'erreur affichés.

Malgré tout, beaucoup de travail reste à faire sur cet outil de simulation pour le rendre plus facile à utiliser. Un premier problème non traité qui nous semble important est de concevoir et implanter un modèle couvrant la plupart des situations de recours des gestionnaires lorsque la performance du système n'est pas celle escomptée. Ces recours peuvent consister à changer la quantité d'agents affectés aux services, modifier le routage, changer les paramètres du composeur, etc., durant la journée.

Un autre problème important est de fournir un cadre permettant d'implanter de façon simple des politiques de routage complexes dont chaque décision dépend de l'état du système et de sa performance observés autant dans le présent que dans le passé. De telles politiques peuvent malgré tout être implantées avec le système actuel, mais cela nécessite souvent beaucoup de travail. Les paramètres des politiques doivent également être facilement accessibles pour pouvoir être optimisés éventuellement. Un problème semblable pourrait également survenir avec les politiques de composition pour les appels sortants.

Une interface graphique pourrait aussi être développée pour rendre plus aisée l'entrée des paramètres et la construction de politiques de routage complexes pour les gestion-

naires.

BIBLIOGRAPHIE

[1] O. Z. Akşin, M. Armony et V. Mehrotra. The modern call center : A multi-disciplinary perspective on operations management research. *Production and Operations Management*, 16(6), pages 665–688, décembre 2007.

[2] AspectJ. The AspectJ project at Eclipse.org, 2006. Disponible sur `http://www.aspectj.org`.

[3] J. Atlason, M. A. Epelman et S. G. Henderson. Call center staffing with simulation and cutting plane methods. *Annals of Operations Research*, 127, pages 333–358, 2004.

[4] A. N. Avramidis, W. Chan et P. L'Ecuyer. Staffing multi-skill call centers via search methods and a performance approximation. *IIE Transactions*, pages 1–45, 2007.

[5] A. N. Avramidis, A. Deslauriers et P. L'Ecuyer. Modeling daily arrivals to a telephone call center. *Management Science*, 50(7), pages 896–908, 2004.

[6] A. N. Avramidis, M. Gendreau, P. L'Ecuyer et O. Pisacane. Simulation-based optimization of agent scheduling in multiskill call centers. Dans *Proceedings of the 2007 Industrial Simulation Conference*. EUROSIS, 2007.

[7] A. N. Avramidis et P. L'Ecuyer. Modeling and simulation of call centers. Dans M. E. Kuhl, N. M. Steiger, F. B. Armstrong et J. A. Joines, éditeurs, *Proceedings of the 2005 Winter Simulation Conference*, pages 144–152, Pistacaway, New-Jersey, 2005. IEEE Press.

[8] A. N. Avramidis et J. R. Wilson. Integrated variance reduction strategies for simulation. *Operations Research*, 44, pages 327–346, 1996.

[9] T. E. Booth et S. P. Pederson. Unbiased combinations of nonanalog Monte Carlo

techniques and fair games. *Nuclear Science and Engineering*, 110, pages 254–261, 1992.

[10] P. Bratley, B. L. Fox et L. E. Schrage. *A Guide to Simulation*. Springer-Verlag, New York, seconde édition, 1987.

[11] L. Brown, N. Gans, A. Mandelbaum, A. Sakov, H. Shen, S. Zeltyn et L. Zhao. Statistical analysis of a telephone call center : A queueing-science perspective. *Journal of the American Statistical Association*, 100, pages 36–50, 2005.

[12] E. Buist. Conception et implantation d'une bibliothèque pour la simulation de centres de contacts. Mémoire de maîtrise, Département d'Informatique et de Recherche Opérationnelle, Université de Montréal, août 2005.

[13] E. Buist, W. Chan et P. L'Ecuyer. Speeding up call center simulation and optimization by Markov chain uniformization. Dans S. J. Mason, R. R. Hill, L. Moench et O. Rose, éditeurs, *Proceedings of the 2008 Winter Simulation Conference*, pages 1652–1660, Pistacaway, New-Jersey, 2008. IEEE Press.

[14] E. Buist et P. L'Ecuyer. *ContactCenters : A Java Library for Simulating Contact Centers*, 2005. Manuel de l'utilisateur disponible sur `http://www.ericbuist.com/contactcenters`.

[15] E. Buist et P. L'Ecuyer. A Java library for simulating contact centers. Dans M. E. Kuhl, N. M. Steiger, F. B. Armstrong et J. A. Joines, éditeurs, *Proceedings of the 2005 Winter Simulation Conference*, pages 556–565, Pistacaway, New-Jersey, 2005. IEEE Press.

[16] CCmath. CCmath : business consulting and software for call centers, 2008. Disponible sur `http://www.ccmath.com`.

[17] M. T. Ceşik et P. L'Ecuyer. Staffing multiskill call centers via linear programming and simulation. *Management Science*, 54(2), pages 310–323, 2008.

[18] R. C. H. Cheng. Variance reduction methods. Dans J. Wilson, J. Henriksen et S. Roberts, éditeurs, *Proceedings of the 1986 Winter Simulation Conference*, pages 60–68, New York, 1986. ACM.

[19] W. G. Cochran. *Sampling Techniques*. John Wiley and Sons, New York, seconde édition, 1977.

[20] R. B. Cooper. *Introduction to Queueing Theory*. North-Holland, New York, seconde édition, 1981.

[21] T. Cormen, C. Leiserson et R. Rivest. *Introduction à l'algorithmique*. Dunod, Paris, 1994.

[22] CRTC. Final standards for quality of service indicators for use in telephone company regulation and other related matters, 2000. Canadian Radio-Television and Telecommunications Commission, Décision CRTC 2000-24. Voir `http://www.crtc.gc.ca/archive/ENG/Decisions/2000/DT2000-24.htm`.

[23] H. A. David. *Order Statistics*. Wiley, seconde édition, 1981.

[24] C. de Boor. *A Practical Guide to Splines*. Numéro 27 dans Applied Mathematical Sciences Series. Springer-Verlag, New York, 1978.

[25] A. Deslauriers. Modélisation et simulation d'un centre d'appels téléphoniques dans un environnement mixte. Mémoire de maîtrise, Département d'Informatique et de Recherche Opérationnelle, Université de Montréal, février 2003.

[26] A. Deslauriers, J. Pichitlamken, P. L'Ecuyer, A. Ingolfsson et A. N. Avramidis. Markov chain models of a telephone call center with call blending. *Computers and Operations Research*, 34(6), pages 1616–1645, 2007.

[27] Eclipse.org. Eclipse.org home, 2008. Voir `http://www.eclipse.org`.

[28] G. S. Fishman. *Monte Carlo : Concepts, Algorithms, and Applications.* Springer Series in Operations Research. Springer-Verlag, New York, 1996.

[29] B. L. Fox. Generating Markov-chain transitions quickly : I. *Operations Research Society of America Journal on Computing*, 2(2), pages 126–13, 1990.

[30] B. L. Fox et P. W. Glynn. Computing Poisson probabilities. *Communications of the ACM*, 31(4), pages 440–445, avril 1988.

[31] B. L. Fox et P. W. Glynn. Discrete-time conversion for simulating finite-horizon Markov processes. *SIAM Journal on Applied Mathematics*, 50(5), pages 1457–1473, 1990. ISSN 0036-1399.

[32] B. L. Fox et A. R. Young. Generating Markov-chain transitions quickly : II. *Operations Research Society of America Journal on Computing*, 2(1), pages 3–11, 1991.

[33] E. Gamma, R. Helm, R. Johnson et J. Vlissides. *Design Patterns : Elements of Reusable Object-Oriented Software.* Addison-Wesley, Reading, Mass., seconde édition, 1998.

[34] N. Gans, G. Koole et A. Mandelbaum. Telephone call centers : Tutorial, review, and research prospects. *Manufacturing and Service Operations Management*, 5, pages 79–141, 2003.

[35] O. Garnett, A. Mandelbaum et M. Reiman. Designing a call center with impatient customers. *Manufacturing and Service Operations Management*, 4(3), pages 208–227, 2002. ISSN 1253-4614.

[36] P. W. Glynn. Efficiency improvement techniques. *Annals of Operations Research*, 53, pages 175–197, 1994.

[37] P. W. Glynn et R. Szechtman. Some new perspectives on the method of control variates. Dans K.-T. Fang, F. J. Hickernell et H. Niederreiter, éditeurs, *Monte*

Carlo and Quasi-Monte Carlo Methods 2000, pages 27–49, Berlin, 2002. Springer-Verlag.

[38] P. W. Glynn et W. Whitt. Indirect estimation via $L = \lambda w$. *Operations Research*, 37, pages 82–103, 1989.

[39] V. F. C. Goncalves, A. Palma-Dos-Reis, J. Duque, D. L. Keefer et W. Whitt. Predicting queueing delays. *Management Science*, 45(6), pages 870–888, 1999. ISSN 0025-1909.

[40] W. K. Grassmann. Finding transient solutions in Markovian event systems through randomization. *Numerical Solutions of Markov Chains*, pages 357–371, 1991.

[41] M. Grogan. JSR 223 : Scripting for the Java™ platform, décembre 2006. Disponible sur `http://jcp.org/en/jsr/detail?id=223`.

[42] D. Gross et D. R. Miller. The randomization technique as a modeling tool and solution procedure for transient Markov processes. *Operations Research*, 32(2), pages 343–361, 1984.

[43] R. Ibrahim et W. Whitt. Real-time delay estimation based on delay history. *Manufacturing and Service Operations Management*, septembre 2008.

[44] R. Ibrahim et W. Whitt. Real-time delay estimation in call centers. Dans S. J. Mason, R. R. Hill, L. Moench et O. Rose, éditeurs, *Proceedings of the 2008 Winter Simulation Conference*, pages 2876–2883, Pistacaway, New-Jersey, 2008. IEEE Press.

[45] A. Ingolfsson, E. Akhmetshina, S. Budge, Y. Li et X. Wu. A survey and experimental comparison of service-level-approximation methods for nonstationary $m(t)/m/s(t)$ queueing systems with exhaustive discipline. *INFORMS Journal on Computing*, 19(2), pages 201–214, 2007.

[46] A. Jensen. Markoff chains as an aid in the study of Markoff processes. *Skandinavisk Actuarietidskrift*, 36, pages 87–91, 1953.

[47] K. Kawaguchi. *JSR 222 : Java™ Architecture for XML Binding (JAXB) 2.0*, décembre 2006. Disponible sur `http://jcp.org/en/jsr/detail?id=222`.

[48] G. Kiczales, J. Lamping, A. Mendhekar, C. Maeda, C. Videira Lopes, J.-M. Loingtier et J. Irwin. Aspect-oriented programming. Dans *Proceedings of ECCOP '97*, juin 1997.

[49] G. Koole. Redefining the service level in call centers. Rapport technique, Department of Mathematics, Vrije Universiteit, Amsterdam, 2005. Disponible sur `http://www.math.vu.nl/~koole/articles/report03b/art.pdf`.

[50] G. Koole, A. Pot et J. Talim. Routing heuristics for multi-skill call centers. Dans S. Chick, P. J. Sánchez, D. Ferrin et D. J. Morrice, éditeurs, *Proceedings of the 2003 Winter Simulation Conference*, pages 1813–1816, Pistacaway, New-Jersey, 2003. IEEE Press.

[51] G. Lapalme. XML : Looking at the forest instead of the trees. Disponible sur `http://www.iro.umontreal.ca/~lapalme/ForestInsteadOfTheTrees`, 2007.

[52] S. S. Lavenberg et P. D. Welch. A perspective on the use of control variables to increase the efficiency of Monte Carlo simulations. *Management Science*, 27, pages 322–335, 1981.

[53] A. M. Law et W. D. Kelton. *Simulation Modeling and Analysis*. McGraw-Hill, New York, troisième édition, 2000.

[54] P. L'Ecuyer. Tests based on sum-functions of spacings for uniform random numbers. *Journal of Statistical Computation and Simulation*, 59, pages 251–269, 1997.

[55] P. L'Ecuyer. *SSJ : A Java Library for Stochastic Simulation*, 2004. Manuel de l'utilisateur, disponible sur `http://www.iro.umontreal.ca/~simardr/ssj`.

[56] P. L'Ecuyer. Modeling and optimization problems in contact centers. Dans *Proceedings of the Third International Conference on Quantitative Evaluation of Systems (QEST'2006)*, pages 145–154, University of California, Riversdale, 2006. IEEE Computing Society.

[57] P. L'Ecuyer. Quasi-Monte Carlo methods with applications in finance. *Finance and Stochastics*, 2008. Á paraître.

[58] P. L'Ecuyer. Stochastic simulation. Notes pour un cours gradué en simulation, 2008.

[59] P. L'Ecuyer et E. Buist. Simulation in Java with SSJ. Dans M. E. Kuhl, N. M. Steiger, F. B. Armstrong et J. A. Joines, éditeurs, *Proceedings of the 2005 Winter Simulation Conference*, pages 611–620, Pistacaway, New-Jersey, 2005. IEEE Press.

[60] P. L'Ecuyer et E. Buist. Variance reduction in the simulation of call centers. Dans L. F. Perrone, F. P. Wieland, J. Liu, B. G. Lawson, D. M. Nicol et R. M. Fujimoto, éditeurs, *Proceedings of the 2006 Winter Simulation Conference*, pages 604–613, Pistacaway, New-Jersey, 2006. IEEE Press.

[61] P. L'Ecuyer et E. Buist. On the interaction between stratification and control variates, with illustrations in a call center simulation. *Journal of Simulation*, 2(1), pages 29–40, 2008.

[62] P. L'Ecuyer, V. Demers et B. Tuffin. Splitting for rare-event simulation. Dans L. F. Perrone, F. P. Wieland, J. Liu, B. G. Lawson, D. M. Nicol et R. M. Fujimoto,

éditeurs, *Proceedings of the 2006 Winter Simulation Conference*, pages 137–148, Pistacaway, New-Jersey, 2006. IEEE Press.

[63] P. L'Ecuyer et C. Lemieux. Variance reduction via lattice rules. *Management Science*, 46(9), pages 1214–1235, 2000.

[64] P. L'Ecuyer, L. Meliani et J. Vaucher. SSJ : A framework for stochastic simulation in Java. Dans E. Yücesan, C.-H. Chen, J. L. Snowdon et J. M. Charnes, éditeurs, *Proceedings of the 2002 Winter Simulation Conference*, pages 234–242, Pistacaway, New-Jersey, 2002.

[65] P. L'Ecuyer et G. Perron. On the convergence rates of IPA and FDC derivative estimators. *Operations Research*, 42(4), pages 643–656, 1994.

[66] P. L'Ecuyer, R. Simard, E. J. Chen et W. D. Kelton. An object-oriented random-number package with many long streams and substreams. *Operations Research*, 50(6), pages 1073–1075, 2002.

[67] P. L'Ecuyer et B. Tuffin. Approximate zero-variance simulation. Dans S. J. Mason, R. R. Hill, L. Moench et O. Rose, éditeurs, *Proceedings of the 2008 Winter Simulation Conference*, pages 170–181, Pistacaway, New-Jersey, 2008. IEEE Press.

[68] P. L'Ecuyer et F. Vázquez-Abad. Functional estimation with respect to a threshold parameter via dynamic split-and-merge. *Discrete Event Dynamic Systems : Theory and Applications*, 7(1), pages 69–92, 1997.

[69] SyncRO Soft Ltd. <oXygen/> XML editor & XSLT debugger, 2007. Voir `http://www.oxygenxml.com`.

[70] A. Mandelbaum et S. Zeltyn. The Palm/Erlang-a queue, with applications to call centers. Disponible sur `http://ie.technion.ac.il/serveng/References/Erlang_A.pdf`, juin 2005.

[71] V. Mehrotra et J. Fama. Call center simulation modeling : Methods, challenges, and opportunities. Dans S. Chick, P. J. Sánchez, D. Ferrin et D. J. Morrice, éditeurs, *Proceedings of the 2003 Winter Simulation Conference*, pages 135–143, Pistacaway, New-Jersey, 2003. IEEE Press.

[72] B. L. Nelson. Control-variate remedies. *Operations Research*, 38, pages 974–992, 1990.

[73] NovaSim. ccProphet — simulate your call center's performance, 2003. Voir `http://www.novasim.com/CCProphet`.

[74] S. Rajasekaran et K. W. Ross. Fast algorithms for generating discrete random variates with changing distributions. *Modeling and Computer Simulation*, 3(1), pages 1–19, 1993.

[75] Rockwell Automation, Inc. Arena simulation, 2005. Voir `http://www.arenasimulation.com`.

[76] R. J. Serfling. *Approximation Theorems for Mathematical Statistics*. Wiley, New York, 1980.

[77] C. M. Sperberg-McQueen et H. Thompson. W3C XML Schema, avril 2000. Voir `http://www.w3.org/XML/Schema`.

[78] Sun Microsystems, Inc. `ServiceLoader` (Java platform SE 6), 2008. Voir `http://java.sun.com/javase/6/docs/api/java/util/ServiceLoader.html`.

[79] H. M. Taylor et S. Karlin. *An Introduction to Stochastic Modeling*. Academic Press, San Diego, troisième édition, 1998.

[80] P. Vakili. Using a standard clock technique for efficient simulation. *Operations Research Letters*, 10(8), pages 445–452, 1991.

[81] R. B. Wallace et W. Whitt. A staffing algorithm for call centers with skill-based routing. *Manufacturing and Service Operations Management*, 7(4), pages 276–294, 2005.

[82] Wikipedia. Virtual queue, avril 2007. Voir `http://en.wikipedia.org/wiki/Virtual_queue`.

[83] F. Yergeau, T. Bray, J. Paoli, C. M. Sperberg-McQueen et E. Maler. *Extensible Markup Language (XML) 1.0.* W3C Recommendation, troisième édition, février 2004. Aussi disponible sur `http://www.w3.org/TR/REC-xml`.