Université Pierre et Marie Curie – Paris VI

Méthodes de décomposition pour l'analyse des performances des systèmes de production manufacturières en boucle fermée et de topologies générales

THÈSE

présentée et soutenue publiquement le 17 décembre 2003

pour l'obtention du

Doctorat de l'Université Pierre et Marie Curie – Paris VI

(Spécialité Informatique)

par

David Douard

Composition du jury

Président :	Philippe Chrétienne
Rapporteurs :	Maria Di Mascolo Stanley B. Gershwin
Examinateur :	Alain Patchong
Directeur de thèse :	Yves Dallery
Co-directeur de thèse :	Bruno Baynat
Invité :	John Buzzacott

Laboratoire de Génie Industriel – École Centrale de Paris

Mis en page avec la classe thloria.

Remerciements

Je tiens à remercier l'ensemble du Laboratoire de Génie Industriel de l'École Central de Paris qui m'a accueilli, et en particulier son directeur, Jean-Claude Bocquet. Je dois en particulier la qualité de cet accueil à M^{mes} Sylvie Guillemin et Anne Prevot.

Mais il me faut avant tout souligner à quel point mes directeurs de thèse furent humainement importants tout au long ce travail. Bruno Baynat, d'abord, car il a su me supporter, m'accompagner et me soutenir aux moments où il m'était plus difficile d'avancer. Yves Dallery, ensuite, qui m'a toujours laissé une grande liberté de mouvements, me laissant conduire mon travail à ma guise (certainement aussi grâce à son incroyable capacité à se surcharger de travail), tout en gardant un œil attentif à mes travaux.

Merci aussi à Alain Patchong, de PSA Peugeot Citroën, m'a permis de donner un sens pratique à ce travail, il m'a procuré l'essentiel des problématiques que j'ai été amené à traiter.

Et puis tous ceux que j'ai rencontré, dans ce laboratoire comme à Paris 6, je pense à Fikri Karaesman, à Francis de Véricourt, mais aussi à Stefan Mocanu, à Véronique Varenne ou Thierry Lanfoy qui m'ont plusieurs fois sorti de petites contrariétés administratives, ainsi que tous les autres qui, j'espère, me pardonneront de ne les avoir pas cités.

Enfin, il me faut signaler l'importance de mes proches, qui m'ont toujours soutenu, aidé et encouragé à aboutir de travail.

Merci à tous.

Dans le couloir il y a une glace, qui double fidèlement les apparences. Les hommes en tirent conclusion que la Bibliothèque n'est pas infinie ; si elle l'était réellement, à quoi bon cette duplication illusoire ?

Jorge Luis Borges
La Bibliothèque de Babel, Fictions

Table des matières

Première partie

1

Introduction

On s'intéresse au fonctionnement de systèmes de production manufacturières. On entend par là des systèmes physiques complexes comportant des machines opérant de façon séquentielle sur les objets en cours de traitement. Le cas le plus représentatif de ces lignes de production est celui de la ligne de montage automobile, qui nous a servi de fil conducteur pour ce travail. Nous nous sommes en effet intéressé à une nouvelle unité de production automobile que l'entreprise PSA Peugeot Citroën est en train de place en route. Ses particularités font apparaître un certain nombre de problématiques que l'on se propose d'étudier en détails tout au long de ce travail.

L'objectif de cette étude est de déterminer comment cette ligne va se comporter dans son ensemble, connaissant les caractéristiques des différents éléments constituant la ligne. On cherchera en particulier à déterminer ses performances, comme par exemple son taux de production horaire.

1.1 Systèmes de production

Un système de production est constitué de postes, sur lesquels un traitement est effectué (en général par un ou plusieurs robots). Quand une pièce arrive sur un poste, elle subit un traitement spécifique au poste. A la fin de ce traitement, la pièce est amenée au poste suivant par un transporteur. Afin de découpler le postes les uns des autres, on place souvent des zones de stockage entre ces postes robotisés.

Les machines qui opèrent sur les pièces sont susceptibles d'avaries. Ces pannes peuvent aussi bien être de très courte durée (il y a par exemple nécessité de changer régulièrement l'outil d'usinage ou la pointe de soudage), mais aussi de durée plus longue, lorsque la machine connaît une avarie plus importante qui nécessite une intervention par un technicien sur le poste.

Le cas le plus simple de système de production est la ligne simple ouverte [12, 6]. Elle est composée de machines (représentant un poste robotisé) séparées par des zones de stockage. On peut voir Figure 1.1 une représentation d'une telle ligne. Chaque machine est représentée par un carré, chaque zone de stockage est représentée par un cercle. Lorsqu'on observe un tel système

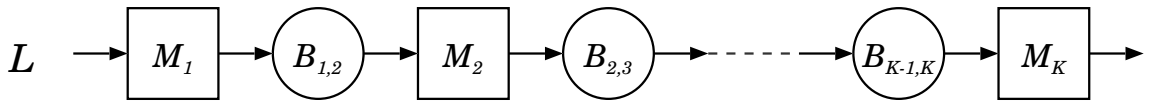

FIG. 1.1 – Représentation symbolique d'une ligne de production simple ouverte.

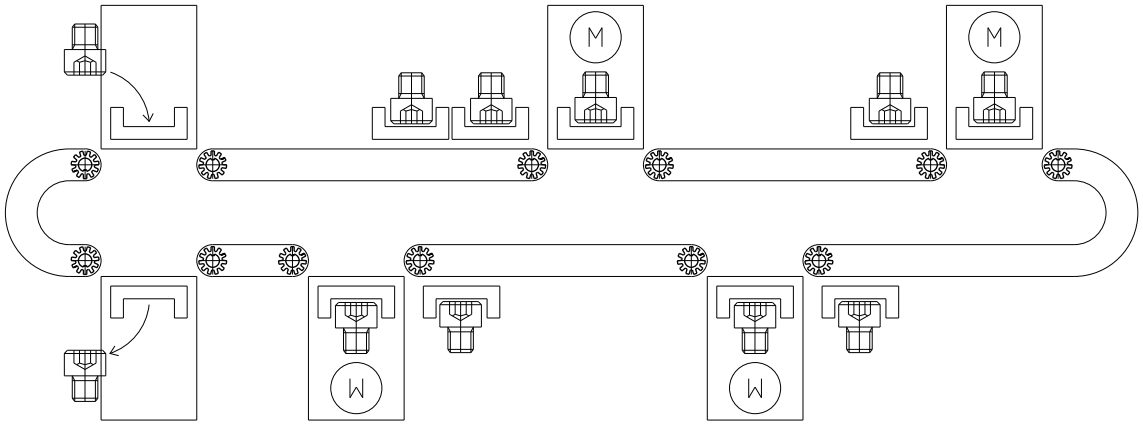

FIG. 1.2 – Exemple de ligne de production industrielle en forme de boucle.

ouvert, on fait toujours l'hypothèse de fonctionnement suivante : la première machine n'est jamais en famine, elle a toujours de quoi être alimentée en amont, et la dernière machine n'est jamais bloquée, elle peut toujours décharger une pièce en fin de traitement. Ce mode de fonctionnement de la ligne est dit saturé.

En milieu industriel, le cas plus courant de ligne de production est la ligne fermée. Il n'y a plus d'entrée ni de sortie. Cela se justifie dans le cas standard où le système de production est le suivant : la ligne est constitué de machines, entre lesquelles se trouvent des chariots transporteurs. Le nombre de ces chariots est fini et ces derniers circulent dans la ligne, emportant les pièces en train d'être traitées. Il s'agit bien d'une ligne fermée du point de vue des chariots. Il y a donc nécessairement un processus de chargement et de déchargement des pièces, considérés alors comme les points d'entrée et de sortie de la ligne (qui est alors ouverte du point de vue des pièces). Mais dans la mesure où l'on fait l'hypothèse que ces mécanismes de chargement et de déchargement ne sont pas bloquant, il suffit d'observer la circulation des chariots dans la ligne pour en analyser le fonctionnement. Il s'agit bien d'un système fermé.

1.2 Évaluation de performances

L'étude d'un tel système de production requiert tout d'abord d'en faire un modèle, c'est-à-dire une représentation simplifiée de ce système. La problématique traditionnelle de la modélisation apparaît ici ; soit on construit un modèle sophistiqué, qui prendra en compte de nombreux phénomènes physiques intervenant dans le fonctionnement de la ligne, soit on construit un modèle simple. Les modèles très réalistes sont en général impossibles à étudier, étant trop complexes. Les modèles trop simplistes sont, quant à eux, facilement analysable, mais on risque alors d'étudier un objet qui n'a que peu de rapport avec le système réel, et les résultats obtenus sont alors sans intérêt. Toute la difficulté de la modélisation consiste à trouver un compromis acceptable entre ces deux situations, à savoir un modèle à la fois assez réaliste pour que les résultats obtenus soient significatifs, mais suffisamment simple pour que l'on puisse, en pratique, les analyser.

Un modèle est formellement une représentation logico-mathématique du comportement du système. Nous avons essentiellement deux familles de modèles pour ce type de systèmes : les modèles déterministes et les modèles stochastiques. Pour notre système, les modèles stochastiques sont mieux adaptés, ne serait-ce que parce que les machines sont susceptibles de subir des

phénomènes aléatoires, comme les pannes de équipements, mais aussi par exemple pour pouvoir disposer de représentations probabilistes d'informations non disponibles a priori.

Ces modèles stochastiques peuvent s'appréhender à l'aide de deux grandes familles de méthodes : la simulation ou les méthodes mathématiques.

1.2.1 La simulation

C'est la méthode qui est actuellement le plus souvent utilisé dans l'industrie pour analyser le fonctionnement des systèmes de production (aussi bien durant la phase de conception que en production, par exemple pour chercher à en modifier le point de fonctionnement). Il s'agit d'outils informatiques qui permettent de reproduire le comportement du modèle.

Pour ce type de modèles, on fait des simulations à événements discrets. Ce sont des simulations où tous les états du modèles sont discrets, les transitions entre ces états pouvant survenir à tout moment (fonctionnement asynchrone). On définit d'abord les règles de transitions entre ces états, on place ensuite le modèle dans un état arbitraire, puis on laisse le simulateur faire évoluer ce modèle selon les règles définies. Les résultats que l'on obtient sont alors une suite précise d'états du modèle. On parle d'une trajectoire. Pour avoir des informations sur le comportement général de la ligne (moyennes, performances stationnaires), il faut lancer plusieurs de ces simulations (plusieurs trajectoires), on obtient ainsi une estimation du fonctionnement stationnaire de la ligne.

Ces techniques de simulation souffrent cependant d'un problème important. Elles nécessitent en effet d'importantes ressource de calcul. Même aujourd'hui, où les ordinateurs disposent de puissance de calcul phénoménales, la simulation d'un système un peu complexe demande beaucoup de temps. Cela peut être un inconvénient rédhibitoire si l'on veut introduire cette estimation des paramètres de performance de la ligne dans une boucle d'optimisation.

1.2.2 Méthodes mathématiques

Ces méthodes consistent à rechercher des équations mathématiques qui permettent de décrire le fonctionnement du modèle. C'est ce que font les physiciens lorsqu'ils cherchent à décrire le fonctionnement du monde réel. Cependant, lorsque le système est complexe, il n'est pas toujours aisé de trouver ces équations qui décrivent son fonctionnement. De plus, même si l'on trouve ces équations, leur résolution est parfois impossible.

Nos modèles sont souvent impossibles à analyser directement de manière exacte, afin d'obtenir les performances du système de production. Aussi, lorsqu'on ne sait pas traiter exactement ces modèles, on cherche à développer des méthodes approximatives. Il existe donc deux familles de méthodes mathématiques : les méthodes exactes, et les méthodes approximatives. D'une manière générale, on ne sait traiter mathématiquement les modèles stochastiques de systèmes de production que lorsqu'ils sont markoviens.

Méthodes exactes

Pour certains modèles, il est possible de construire la chaîne de Markov associée. De telles chaînes de Markov peuvent se résoudre par des techniques numériques qui dépendent du nombre d'états de la chaîne. Mais ces chaînes de Markov sont souvent impossibles à analyser numériquement, à cause de l'explosion combinatoire du nombre des états de la chaîne lorsque le modèle se complexifie.

Il existe cependant des modèles particuliers dont on sait faire l'évaluation exacte des performances par l'analyse. On dispose alors d'équations simples donnant directement les résultats

voulus. Ces modèles sont les réseaux de Jackson [26] ou les réseaux BCMP [2]. Malheureusement, les hypothèses sur ces réseaux sont extrêmement restrictives (comme par exemple le fait que les zones de stockages doivent être de capacité infinie), et empêchent en général leur utilisation dans des cas pratiques de modèles de production (le modèle devient trop simpliste pour que les résultats soient intéressant).

Méthodes approximatives

Lorsqu'on ne peut pas analyser nos modèles stochastiques de façon exacte (que ce soit par une méthode analytique exacte ou numérique), on peut essayer de développer une méthode approximative. L'objectif de ce type de méthodes est de ramener l'étude du système complexe à celle d'un système plus simple, ou d'une combinaison de systèmes simples, chacun d'eux étant aisément analysable par des méthodes exactes. On peut alors envisager d'étudier une classe de modèles beaucoup plus grande que ce qui nous était accessible grâce aux méthodes exactes. Cela signifie aussi que la précision des résultats est moins, puisqu'ils ne sont plus que des approximations des performances du système réel.

La technique la plus efficace pour analyser nos modèles de production est la décomposition. Elle consiste à décomposer la ligne d'origine en sous-systèmes qui sont autant de "visions" locales du système d'origine. Il faut donc en premier lieu déterminer et caractériser l'élément de la décomposition, qui doit être facilement analysable. Ensuite, il faut chercher des équations qui permettent de relier ces sous-systèmes entre eux (et à la ligne d'origine). Cela détermine un système d'équations, dont on cherchera une résolution par des techniques numériques ou, le plus souvent, par un algorithme de type point fixe.

1.3 La problématique PSA : un atelier de ferrage

La ligne conductrice de tout le travail de cette thèse à été fournie par l'entreprise PSA (Peugeot Citroën), puisque l'une de leur nouvelle unité de production est l'objet d'étude principal de ce travail. Plus précisément, le système étudié est un atelier de ferrage, dont la topologie est inhabituelle dans l'industrie automobile.

1.3.1 Description du système et de son environnement

L'atelier de ferrage

La construction d'une automobile est jalonnée de nombreuses étapes. En schématisant, on peut rencontrer quatre grandes étapes :

- emboutissage : préparation des pièces de la caisse ;
- ferrage : assemblage et soudage des la caisse ;
- peinture ;
- montage : assemblage final de la voiture.

L'atelier de ferrage, qui nous intéresse ici, est l'endroit où est assemblée la caisse de la voiture. Il s'agit d'une chaîne de montage très robotisée, comportant de nombreux postes de travail (pour l'essentiel des robots soudeurs). Traditionnellement, cet atelier de ferrage est formé d'une ligne de production manufacturière comportant une boucle dans laquelle circulent un nombre fixé de chariots, qui transportent de poste en poste les caisses en cours de montage.

Le nouvel atelier de ferrage de PSA à été élaboré sur une topologie plus sophistiquée à deux boucles, dites siamoises. On peut voir Figure 1.3 page suivante le schéma de cet atelier. Il est constitué de 76 postes de travail. Chaque poste est symbolisé par la représentation d'une caisse de voiture. Sur certains de ces postes se trouvent des robots. On parle alors de postes robotisés. Les robots apparaissent sur le schéma sous la forme de cercles autour du poste. Les postes non robotisés sont nommés postes morts, et forment de fait des zones de stockage.

Fonctionnement général

Chaque poste correspond à un emplacement sur lequel il peut y avoir un chariot. La ligne commence au poste 1 (poste de chargement). Lorsqu'un chariot se présente à ce poste, un châssis de voiture est déposé. Le poste 2 est un poste de routage. Lorsqu'un chariot y arrive, il est soit transmis vers le poste 3, soit vers le poste 51, selon une politique de routage préétablie par le chef d'atelier, et à la condition que le poste en question ne soit pas occupé.

Au poste 29, un mécanisme de routage dual de celui évoqué pour le poste 2, choisit de récupérer le chariot en provenance soit du poste 28, soit du poste 76, selon la politique de routage et l'approvisionnement de chacun de ces deux postes. Les chariots, de retour sur la branche principale, continuent leur route jusqu'au poste 46, où la caisse est alors déposée (retirée du chariot) pour poursuivre son traitement dans un autre atelier. Le chariot, quant à lui, poursuit son parcours (à vide) et rejoint le poste 1 pour un nouveau cycle.

Dans une telle ligne de production, il est possible d'ajouter ou d'enlever des chariots selon que le chef d'atelier veuille augmenter ou diminuer la productivité de la ligne. Ces chariots sont stockés dans une zone de garage située autour des postes 47 à 50. Dans cette ligne, on peut donc mettre en circulation entre 1 et 75 chariots.

Les postes robotisés Chaque poste qui doit effectuer un traitement comporte entre 1 et 6 robots qui travaillent ensemble. On connaît les caractéristiques de fonctionnement de chacun

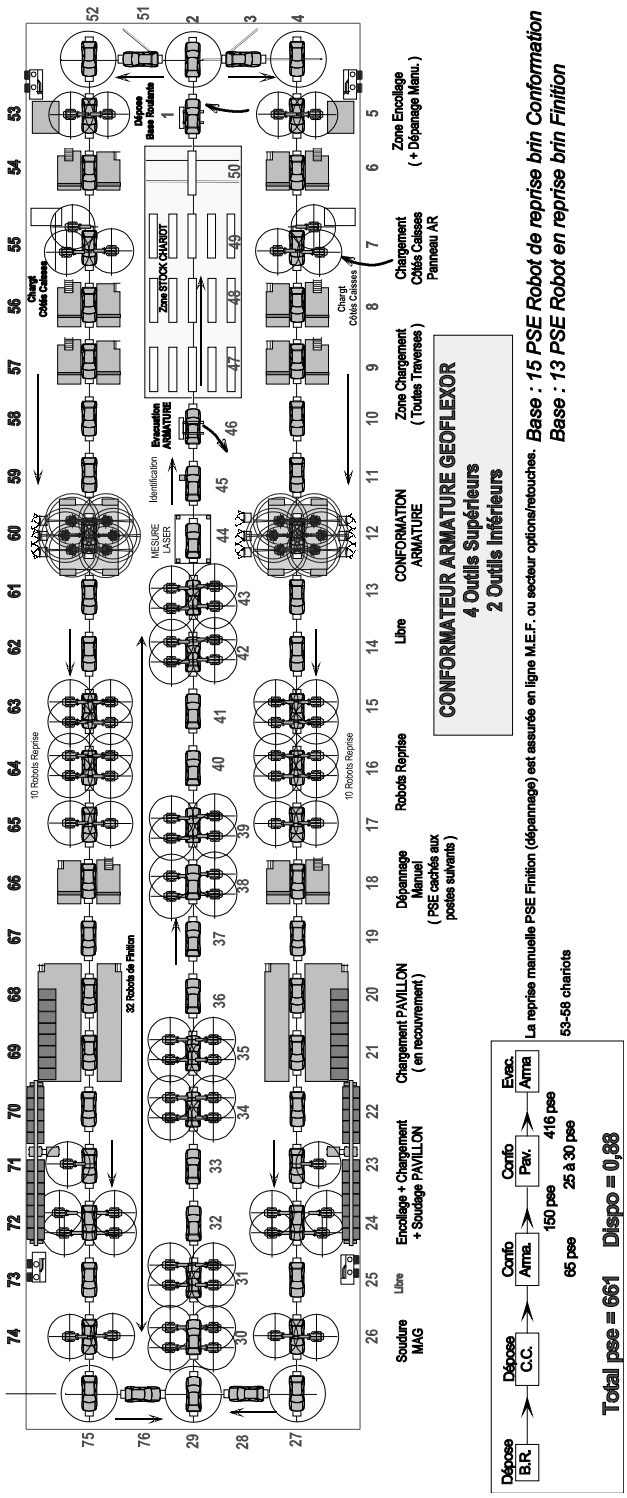

FIG. 1.3 – Vue schématique du nouvel atelier de ferrage de PSA.

de ces robots, c'est-à-dire son temps de cycle (temps de traitement) qui est constant, ainsi que ses paramètres de fiabilité : le temps moyen de bon fonctionnement (MTTF, Mean Time To Failure) et le temps moyen de réparation (MTTR, Mean Time To Repair). On ne connaît en général que ces valeurs. Les moments d'ordres supérieurs des distributions de ces temps de bon fonctionnement et de réparation sont extrêmement difficiles à estimer, et ne sont donc pas disponibles.

Ainsi, lorsqu'un chariot arrive à hauteur d'un poste robotisé, l'ensemble des robots se mettent au travail pour un temps total égal au temps de cycle. Si durant cette période, l'un des robots à une défaillance, tous les robots sont arrêtés, afin qu'un opérateur humain puisse effectuer la maintenance. Une fois la réparation achevée, le travail reprend à l'endroit précis où il avait été interrompu.

Lorsque le travail est terminé sur le poste, le chariot est avancé jusqu'au poste suivant si ce dernier est libre, auquel cas un nouveau cycle de travail peut commencer. Sinon, le chariot reste sur le poste en cours jusqu'à ce que le poste suivant soit disponible. On dit alors que le poste est bloqué.

De même, si après avoir transféré un chariot au poste suivant, nul chariot n'est présent sur le poste précédant, le poste ne peut rien faire, et est dit non-alimenté.

Lorsqu'il est bloqué ou non-alimenté, le poste ne peut pas tomber en panne, puisque les robots ne font rien.

Les chariots et les transporteurs Les pièces à traiter sont posées sur les chariots qui naviguent de poste en poste, tractés par des transporteurs. Il existe plusieurs types de transporteurs :

- ils peuvent prendre l'aspect d'un tapi roulant ou d'un mécanisme de traction à l'image des remontée mécaniques des stations de sport d'hiver ;
- ils peuvent aussi être des petits véhicules robotisés autonomes ;
- ils se peut enfin qu'il y ait un système de stockage entre les postes robotisés constitué d'une étagère, où le poste en amont dépose des pièces (dans la mesure où l'une au moins des cases est libre), et où le poste en aval puise les pièces (dans la mesure où l'une au moins des cases est occupée par une pièce).

En pratique, ces mécanismes de transport des pièces ne sont pas totalement fiables. Ils le sont néanmoins beaucoup plus que les robots de traitement des pièces. On néglige souvent les effets de leurs pannes pour simplifier le modèle.

1.3.2 Modèle canonique extrait

Une telle ligne peut se modéliser "naturellement" par un modèle discret asynchrone tel que (Figure 1.4 page suivante) :

- chaque machine M_i du modèle représente le poste robotisé de numéro correspondant. Cette machine (unique) doit se comporter comme l'ensemble des robots en parallèle sur le poste robotisé correspondant. On donne ses paramètres de fonctionnement : le temps de traitement T_i, le temps moyen entre deux pannes $MTTF_i$ et le temps moyen de réparation $MTTR_i$;
- chaque stock $B_{i,j}$ représente l'ensemble des postes morts situés entre deux postes robotisés. On peut, à ce niveau, vouloir prendre en compte les pannes des transporteurs, qui seront alors prises en compte dans les stocks ;
- les postes 2 et 29 ne sont pas à proprement des postes robotisés, en ce sens qu'il n'y a pas de robot y travaillant sur les pièces. Cependant, ce sont des tables tournantes susceptibles

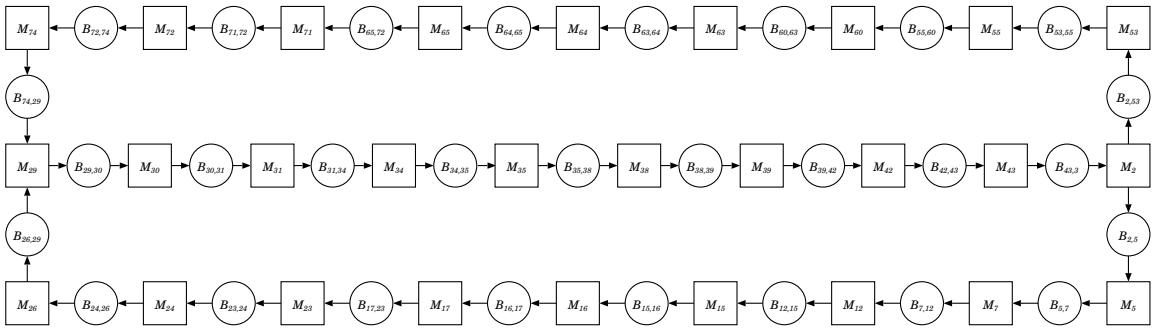

FIG. 1.4 – Modèle asynchrone discret extrait de la ligne réelle représentée Figure 1.3 page 8.

d'avaries, et on peut donc les modéliser par des machines telles que définies ci-dessus. Ils fonctionnent selon une politique de routage préétablie ;
– au sein de cette ligne circulent un maximum de 75 chariots (en général autour de 40 à 50). Ce nombre est connu a priori.

1.3.3 Problématiques

Lorsqu'on cherche à analyser un tel modèle, il y a plusieurs solutions, dont la plus courante est la simulation à évènements discrets. L'intérêt de cette méthode est que l'on peut simuler un modèle aussi précis que l'on veut, moyennant des temps de calculs adéquats. Les inconvénients sont ceux inhérents à la simulation, à savoir des temps de calcul qui restent encore maintenant importants, voire rédhibitoires, et la difficulté qu'il y a à interpréter les résultats obtenus.

Une autre approche consiste à chercher des techniques analytiques qui, après une analyse fine du modèle étudié, permettent d'avoir un résultat quasi instantanément. Cela implique, en général, de déterminer un système d'équations dont la résolution donne pour résultats les performances de notre ligne de production, tel que le taux de production.

Lorsque l'on veut utiliser ces techniques numériques sur le modèle qui nous intéresse, il faut être capable de traiter un certain nombre de points, certains ayant souvent été traités, mais d'autres restent peu explorés. On sait en pratique bien traiter les cas de lignes simples et ouvertes par des techniques analytiques basées sur la décomposition d'un modèle fluide dérivé de ce modèle asynchrone discret.

Ces méthodes reposent sur les principes développés par Stan B. Gershwin dans [21], puis Dallery et al. dans [11]. L'algorithme de base est nommé DDX (en référence aux auteurs de l'article ayant présenté cet algorithme). Il utilise activement une brique de base : l'analyse exacte du dipôle (ligne composée d'un stock et de deux machines) par Dubois et Forrestier dans [18]. Il permet l'étude approximative d'une ligne simple et ouverte, dont les machines ont un temps de traitement constant, mais qui sont susceptibles de tomber en panne, et dont les stocks sont fiables mais de capacité limitée. L'analyse d'un dipôle, ainsi que le DDX seront rappelés en détail dans la Chapitre 3 où sont présentés ces résultats préliminaires, Section 3.1 et Section 3.2.

Ces méthodes peuvent être étendues a des topologies fermées, en introduisant la contrainte de population comme critère de convergence du DDX. Cela est étudié en détail Chapitre 4.

Le routage est difficile à traiter directement, et on présentera Chapitre 8 une méthode pour appréhender ce mécanisme en transformant le modèle avec routage en modèle avec assemblages/désassemblages. Ces derniers mécanismes sont étudiés lorsqu'on s'intéresse à l'extension de la méthode proposée pour les lignes fermées simples aux cas des lignes fermées comportant

plusieurs boucles, comme c'est le cas cas de l'atelier de ferrage, après transformation du modèle (du routage en assemblage/désassemblage) Chapitre 5.

La prise en compte des pannes des zones de stockage est étudiée, quant à elle, Chapitre 7, dans le cas des lignes simples et ouvertes. On y propose une analyse de ces mécanismes de panne, puis une méthode simple pour traiter ces pannes.

Enfin, on présentera un cas d'application pratique de ces méthodes à l'atelier de ferrage de PSA, au Chapitre 8, au moyen d'un logiciel développé au cours de ce travail pour PSA, DispoPSA.

2

État de l'art

Notre objectif étant de couvrir l'ensemble des problématiques exposées dans l'introduction, il convient d'abord de faire le point sur les travaux antérieurs qui ont pu traiter, au moins partiellement, certaines des problématiques que l'on veut aborder. On s'intéressera aux différents modèles que l'on peut extraire du système que l'on veut analyser, puis on verra comment chacun de ces points peut être analysé.

Sommaire

2.1 Modèles

Comme nous l'avons vu Section 1.3.2, nous allons analyser le modèle asynchrone discret de la ligne de production manufacturière. Ce modèle est extrêmement classique aujourd'hui, et est généralement considéré comme un bon modèle, c'est-à-dire comme un compromis acceptable entre la fidélité au système réel et la possibilité d'analyse de ce modèle [4, 5, 12].

2.1.1 Modèle asynchrone discret

Dans ce modèle, les entités sont discrètes et sont appelées clients (ou encore palettes ou chariots, pour reprendre la terminologie usuelle de l'atelier de production). L'ensemble des évé-

nements pouvant survenir à tout moment, le modèle est asynchrone. Plus précisément, les caractéristiques de notre modèle sont les suivantes :

- Le système est constitué de K machines et d'un certain nombre de zones de stockage, le tout formant en ensemble fermé et connexe.
- Entre deux machines peut se trouver une zone de stockage. Chaque zone de stockage $B_{i,j}^A$ à une capacité finie $C_{i,j}^A$. Sur chaque machine M_i^A, on considère qu'il y a une place de stockage (pour pouvoir traiter la pièce), et que cette place n'est pas comptée dans les zones de stockage voisines. Lorsqu'une machine M_i^A prélève une pièce de la zone de stockage $B_{h,i}^A$ qui est située entre les machines M_h^A et M_i^A, le nombre de pièces dans cette zone $n_{h,i}$ est décrémenté, tandis que si une machine M_i dépose une pièce dans la zone de stockage $B_{i,j}^A$ située entre les machines M_i^A et M_j^A, le nombre de pièces de la zone $n_{i,j}$ est incrémenté. La zone de stockage étant finie, les valeurs possibles de $n_{i,j}$ sont donc $n_{i,j} \in \left\{0, \ldots, C_{i,j}^A\right\}$. Lorsque la zone de stockage dans laquelle la machine M_i veut déposer une pièce est pleine, cette dernière se trouve alors bloquée. La pièce en question reste sur la machine, qui attend de pouvoir s'en débarrasser pour reprendre un nouveau traitement. De même, si la zone de stockage dans laquelle la machine M_i^A veut prélever une pièce est vide, cette dernière se trouve non-alimentée (ou encore en famine).
- Le système ne traite qu'une seule catégorie de pièce. Le système étant fermé, il y a un mécanisme, non détaillé ici, qui pose les pièces à traiter sur un système de chariots qui tournent dans la ligne de production. Le nombre total N de chariots présents dans la ligne est fixé. De même, chaque pièce ayant fait le tour de la ligne et donc ayant été transformée, se trouvent retirée du chariot sur lequel elle se trouve, et libère ainsi ce chariot qui peut alors recevoir une nouvelle pièce à traiter par la ligne. Ce processus est supposé fiable, et de façon à ce qu'il y ait toujours des pièces disponibles en entrée de ligne, et que le processus de récupération de pièces en fin de traitement ne soit jamais bloqué.
- Le temps de traitement d'une pièce sur la machine M_i^A est constant T_i^A. En général, ce temps inclut le traitement proprement dit par la machine, mais aussi les temps de chargement et de déchargement sur la machine. On se limitera dans cette étude au cas où toutes les machines ont le même temps de traitement (lignes homogènes).
- Les temps de transfert à travers les zones de stockage sont considérés comme nuls.
- Les machines ne sont pas fiables : lorsqu'elles sont en train de travailler sur une pièce, elles sont susceptibles de tomber en panne. Ce type de pannes (pannes dépendant des opérations) est considéré comme le plus courant dans l'industrie [5].
- Le temps de bon fonctionnement de la machine (avant qu'elle ne tombe en panne) est un processus stochastique, dont on ne connaît en général que le premier moment de la distribution, $MTTF_i^A$ (Mean Time to Failure, temps moyen avant une panne).
- Lorsque la machine tombe en panne, elle a besoin d'être réparée. Le temps de réparation de la machine est aussi un processus stochastique, dont on ne connaît en général que le premier moment de la distribution, $MTTR_i^A$ (Mean Time to Repair, temps moyen de réparation).
- Lorsqu'une machine est réparée, elle reprend le traitement sur la pièce en cours à l'endroit où celui-ci s'était interrompu lorsque la panne est survenue.
- Lorsqu'une machine est suivie de plusieurs zones de stockage, elle place la pièce en fin de traitement sur l'une de ces zones de stockage. Le choix de cette zone est aléatoire, et on connaît la probabilité $p_{i,j}$ pour la machine M_i^A de déposer la pièce dont elle vient de terminer le traitement dans la zone de stockage $B_{i,j}^A$.
- De même, si la machine M_i^A est précédée de plusieurs zones de stockage, elle prélève une pièce, au moment où elle entame un nouveau cycle de travail, dans l'un de ces stocks, de

façon aléatoire.

Nous souhaitons obtenir les paramètres de performance de ce modèle. Ces paramètres sont pour l'essentiel le débit moyen de la ligne à chaque poste, et le nombre moyen de clients dans chaque zone de stockage.

Ce modèle est en général inaccessible à l'analyse directe. C'est en effet un modèle qui n'est pas markovien, en raison des temps de traitement constants sur les machines. Une façon de contourner cela est de remplacer ces temps de traitement par des processus stochastiques dont les temps suivent des lois markoviennes (exponentielles ou composées d'exponentielles). Le modèle obtenu est alors bien markovien, mais d'une part on s'éloigne fortement du système réel, et d'autre part, le modèle markovien résultant est souvent d'un complexité telle qu'il est difficile à analyser. La méthode la plus courante, celle qui donne les meilleurs résultats, est d'approcher ce modèle asynchrone discret par un modèle asynchrone continu.

2.1.2 Modèle continu

Dans ce modèle, on considère que le matériel circulant dans la ligne est continu. Il s'agit donc non plus de pièces discrètes, mais d'un fluide circulant continûment de stock en stock. Chacun de ces stocks peut être vu, en poussant l'analogie avec un système hydraulique, comme un réservoir de capacité finie, et chaque machine peut être vue comme une pompe, récupérant ce fluide du réservoir situé avant elle, et rejettent le fluide dans le réservoir après elle, avec un débit fixé U_i^C (relativement au temps de traitement T_i^A de la machine équivalente du modèle asynchrone discret).

Si l'on observe le cas d'une ligne en forme de boucle simple (sans mécanismes de routage, ceux-ci seront évoqués plus loin), notre modèle fluide est très similaire au modèle asynchrone discret que l'on cherche à approcher. La capacité des stocks en modèle continu va cependant être choisie par : $C_{i,j}^C = C_{i,j}^A + 1$. Il s'agit en effet de prendre en compte la place de stockage implicitement disponible sur la machine M_j^A du modèle asynchrone discret, dont il n'y a plus d'équivalent sur la machine M_j^C du modèle fluide.

Les machines M_i^C ne sont pas fiables, comme leurs équivalents dans le modèle discret, et sont donc sujettes à des pannes, selon les mêmes mécanismes que pour le modèle discret. La transformation du modèle asynchrone discret en modèle fluide procède donc comme suit :

- chaque machine M_i^C a une vitesse déterministe constante :

$$U_i^C = \frac{1}{T_i^A}$$

- les mécanismes de panne et de réparation des machines M_i^C suivent les mêmes lois de distribution que pour les machines du modèle discret M_i^A, en particulier :

$$MTTF_i^C = MTTF_i^A \tag{2.1a}$$

$$MTTR_i^C = MTTR_i^A \tag{2.1b}$$

- la taille de la zone de stockage dans le modèle continu est choisie supérieure d'une unité à celle de la zone de stockage équivalente du modèle asynchrone discret, afin de prendre en compte la place cise sur la machine située juste après :

$$C_{i,j}^C = C_{i,j}^A + 1$$

  Notons que si, dans le cas des lignes ouvertes, on avait le choix d'ajouter cette place ou non (cela est discuté à plusieurs reprises dans la littérature, voir [14, 12]), dans le cas des lignes

fermées, on n'a plus le choix, car négliger ces places de stockage implicitement disponibles sur les machines implique une diminution de la capacité totale de la ligne — c'est-à-dire du nombre maximal de clients pouvant y circuler — ce qui diminue fortement la productivité calculée.

Distributions des temps Nous n'avons fait, jusqu'ici, aucune hypothèse sur les distributions des mécanismes de panne des machines. En général, seules les valeurs moyennes de ces temps sont disponibles. Afin de simplifier encore ce modèle, on approxime ces distributions des temps de réparation et de bon fonctionnement par des distributions exponentielles de taux respectifs μ_i^C et λ_i^C :

$$\lambda_i^C = \frac{1}{MTTF_i^A} \tag{2.2a}$$

$$\mu_i^C = \frac{1}{MTTR_i^A} \tag{2.2b}$$

On définit l'efficacité de la machine en isolation par :

$$e_i^C = \frac{\frac{1}{\lambda_i^C}}{\frac{1}{\lambda_i^C} + \frac{1}{\mu_i^C}} \tag{2.3}$$

Avec ces hypothèses, notre modèle continu se comporte comme un processus markovien à espace d'états mixte, à la fois discret (état des machines) et continu (état des stocks). On ne sait en général pas le résoudre, sauf dans les cas très simples, comme par exemple le cas du dipôle, ligne ouverte composée de deux machines et d'un stock intermédiaire, observé à saturation.

2.2 Analyse des lignes simples ouvertes par décomposition

Ces modèles (que ce soit le modèle asynchrone discret ou le modèle continu) sont en général inaccessibles à une analyse directe exacte. Dans le cas du modèle asynchrone, puisqu'il n'est pas markovien, on ne peut pas faire grand chose. Des travaux ont proposés des les transformer en modèles synchrones en discrétisant le temps, ce qui permet de décrire le modèle par une chaîne de Markov à temps discret. Le cas du dipôle est traité dans [5] et [24]. Le modèle continu est, quant à lui, bien markovien, mais il est en pratique impossible à traiter de façon directe, car trop complexe (explosion combinatoire) et par son aspect mixte, qui induit pour sa résolution un système d'équations différentielles de taille importante.

Une façon d'aborder un tel problème est de le segmenter, pour le réduire à plusieurs problèmes plus petits mais accessibles à l'analyse. C'est l'idée de la décomposition introduite par Gershwin [21] sur le modèle asynchrone discret de la ligne ouverte en tandem. Cette méthode à été ensuite améliorée dans [10] par l'introduction d'un algorithme itératif de résolution. Cette décomposition repose sur la capacité à analyser un cas simple de ligne, qui sert de brique de base à la décomposition.

2.2.1 Le dipôle

Le modèle de ligne le plus simple est le dipôle, qui est constitué de deux machines séparées d'un stock. Il est étudié en détails dans [18], où est proposée l'analyse exacte du modèle continu du dipôle avec des distributions des temps de panne et de réparation exponentielles. Cette étude

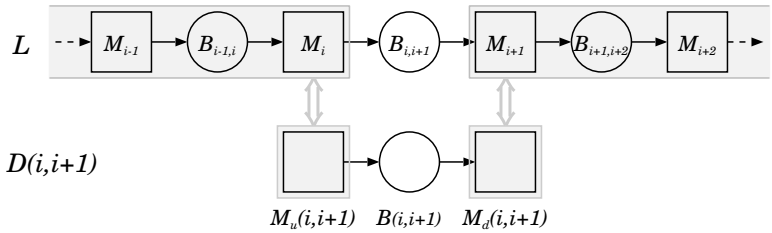

FIG. 2.1 – Chaque machine d'un dipôle doit modéliser toute une partie de ligne.

a, par la suite, été étendue en utilisant un modèle de dipôle où les machines ont des distributions exponentielles des temps de bon fonctionnement, et des distributions hyper-exponentielles à deux étages des temps de réparation par Le Bihan dans [27]. Ces méthodes seront rappelées en détails Section 3.1. Cette analyse de dipôle sert aussi bien à l'analyse par décomposition du modèle asynchrone discret de Gershwin, où le modèle continu du dipôle sert d'approximation du modèle asynchrone discret de ce dernier, qu'à son extension la plus courante, la décomposition du modèle continu.

2.2.2 Résolution par extension de la méthode de décomposition au modèle continu

La technique de décomposition et l'algorithme itératif évoqués plus haut ont ensuite été appliqués au modèle continu, afin de corriger les problèmes de précision dont souffre la méthode de décomposition du modèle asynchrone. Les fondements de cette méthode sont les mêmes que l'idée de départ de Gershwin, mais les équations ont été réécrites et simplifiées. Cela rend cet algorithme simple, efficace et robuste. Il est souvent appelé algorithme DDX (initiales de ses auteurs). Cet algorithme sera rappelé en détail dans la Section 3.2.

Le principe de la décomposition consiste à associer à chaque zone de stockage $B_{i,j}$ un dipôle, que l'on sait résoudre exactement [18], et de trouver des paramètres pour les deux machines qui le composent de façon à ce que le flux de matière traversant le stock du dipôle se comporte comme celui qui traverse le stock correspondant de la ligne $B_{i,j}$. Ainsi, chaque dipôle $L(i,j)$, associé à la zone de stockage $B_{i,j}$, est constitué de deux machines $M_u(i,j)$ et $M_d(i,j)$, dont les paramètres $\lambda_u(i,j)$, $\mu_u(i,j)$, $U_u(i,j)$, $\lambda_d(i,j)$, $\mu_d(i,j)$ et $U_d(i,j)$ sont à déterminer. On se place dans l'hypothèse de lignes homogènes, donc dont tous les temps de traitement sont égaux (arbitrairement égaux à 1.0). Les vitesses $U_u(i,j)$ et $U_d(i,j)$ sont donc posées égales à 1.0. Les paramètres restant à déterminer sont alors estimés de façon à ce que les flux de matériels entrant et sortant de la zone $B(i,j)$ du dipôle soient le plus proche possible de ceux entrant et sortant de $B_{i,j}$.

Cela peut se faire en observant que chaque machine amont $M_u(i,j)$ d'un dipôle $L(i,j)$ doit se comporter de façon à modéliser l'ensemble de la partie de la ligne située en amont de la zone de stockage $B_{i,j}$ (Figure 2.1 . De même, $M_d(i,j)$ doit modéliser au mieux le fonctionnement de la portion de la ligne L située en aval de $B_{i,j}$. On arrive ainsi à établir des équations qui donnent les paramètres $\lambda_u(i,j)$ et $\mu_u(i,j)$ en fonction des paramètres de la portion de la ligne située en amont de $B_{i,j}$, donc en particulier en fonction du dipôle $L(h,i)$. De même $\lambda_d(i,j)$ et $\mu_d(i,j)$ peuvent être mis en relation avec les paramètres de la portion de la ligne située en aval de $B_{i,j}$. On obtient, au final, un système d'équations que l'on pourra résoudre par une méthode itérative de point fixe.

2.2.3 Extension de la méthode

Une façon d'améliorer un peu cette méthode de décomposition du modèle continu à été apportée Dallery dans [9] dans un premier temps, puis par Le Bihan dans [27]. Il s'agit d'essayer d'améliorer la façon dont chaque machine d'un dipôle modélise la portion de la ligne qui lui est dévolue en utilisant des distributions générale-exponentielles [9] ou hyper-exponentielles à deux étages (HE_2) [27] pour les temps de réparation. On montre dans [9] qu'il est en général peu utile d'augmenter la précision des distributions des temps de panne des machines des dipôles, mais qu'il peut être par contre plus intéressant d'utiliser des distributions plus fines des temps de réparation. Cela se justifie parce qu'on peut montrer que ces temps suivent en réalité des distributions hyper-exponentielles à n étages, n étant le nombre de machines appartenant à la portion de la ligne que notre machine doit modéliser. Or, s'il est inutile de prendre en compte tous ces moments de la distribution, une approximation aux deux [9] ou trois [27] premiers moments par une distribution GE ou HE_2 est un bon compromis. Cela implique tout de même de revoir la méthode d'analyse exacte d'un dipôle pour l'adapter à ces dipôles [27].

2.3 Mécanismes d'assemblages et de désassemblages

Un mécanisme d'assemblage est une machine de la ligne qui comporte plusieurs zones de stockage en amont, et dont chaque traitement commence par prendre un client dans chacun de ces stocks pour les assembler (considérant le modèle discret). De même la machine de désassemblage dépose en fin de traitement un client dans chacun des stocks en aval. On a vu que cette machine ne peut commencer un travail que lorsque tous les stocks en amont sont achalandés, et ne peut décharger un client en fin de traitement que si tous les stocks en aval ont au moins une place disponible.

Une extension de l'algorithme DDX pour les lignes ouvertes comportant de tels mécanismes à été proposée dans [15, 16]. Le principe est exactement le même que pour le DDX simple, appliquant les mêmes raisonnements qui ont permis l'établissement des équations du DDX à ces mécanismes.

Une autre approche à été proposée par [22], puis étendue aux lignes non homogènes par [23].

2.4 Lignes fermées

La technique de décomposition du modèle fluide introduite ci-dessus ne s'applique qu'aux cas de lignes ouvertes. Si on s'intéresse aux lignes fermées, il faut de plus tenir compte du fait que l'on a un nombre fixé de clients qui circulent dans la ligne.

Une extension de l'algorithme DDX aux cas des lignes simples fermées à été proposée Frein et al. dans [20]. Le principe de base est le même que pour le DDX, sauf que, lorsqu'on écrit les équations à résoudre pour le DDX en ouvert, les deux machines aux bords de la ligne (la première et le dernière) définissent des conditions aux limites de la ligne qui permettent la résolution du système. Dans une ligne bouclée, ces machines n'existent plus. Par contre, nous avons une contrainte supplémentaire, qui est la contrainte de population.

Le principe de la résolution proposée dans [20] est une dichotomie sur le nombre total de clients estimés. La fonction inversée est en fait le DDX lui-même.

Cet algorithme sera lui aussi étudié et amélioré Section **??**.

Récemment, une autre approche à été étudiée pour traiter ces lignes fermées par [30]. Les fondements de cette nouvelle technique sont très différents des algorithmes types DDX. En effet,

les équations établies pour analyser la ligne sont très différentes de celles du DDX, et s'appuient sur des considérations sur les propagations des blocages et des non-alimentations. Attendu le nombre limité de clients circulant dans la ligne, une panne d'une machine ne pourra pas provoquer de blocage d'un certain nombre de machines, situées assez "loin" de cette dernière (c'est-à-dire dont le nombre de places de stocks cumulées situées entre ces deux machines est supérieure au nombre total de clients dans la ligne). De même pour les non-alimentations.

2.5 Routages

Les mécanismes de routage sont difficiles à appréhender directement. Une machine est une machine de routage lorsqu'elle dispose de plusieurs zones de stock en aval, mais contrairement au désassemblage évoqué plus haut, à la fin de chaque service, la machine choisit un de ces stocks pour y déposer la pièce dont elle vient de terminer le traitement. Ce choix est fait soit de façon aléatoire (on parle alors de routage probabiliste), soit déterminé a priori.

Une technique de traitement de ces mécanismes à été proposée par Helbert et Jusic dans [25]. Elle se limite aux lignes ne comportant pas de boucles, et telles que ces mécanismes de routage, à deux branches seulement, ne soient pas consécutifs (elle ne peut traiter les cas où une machine de routage est séparée d'une autre machine de routage d'un stock). Elle est fondée sur l'étude d'un élément de base plus complexe que le dipôle, comportant trois machines, et modélisant le mécanisme de routage.

Une autre idée, proposée par [**?**], repose sur la transformation du modèle du routage en mécanisme d'assemblage/désassemblage, se ramenant ainsi à un cas que l'on sait traiter lorsque la ligne est ouverte et ne comporte pas de boucle.

2.6 Pannes et délais de transmission des buffers

Les modèles usuels de systèmes de production font l'hypothèse que les zones de stockage sont fiables, et que les transports dans celles-ci sont instantanés.

2.6.1 Délais dans les transporteurs

Une façon de prendre ces délais de transport en compte dans les zones de stockage est proposée par Commault et Semmeray dans [8]. Cette méthode repose sur une transformation de la ligne. Il s'agit de considérer que les stocks dont les délais de transfert sont non nuls (mais dont la durée reste suffisemment courte pour ne pas devenir le principal facteur de ralentissement) peuvent être vus comme des stocks habituels (dont les temps de transfert sont nuls), mais de taille sensiblement réduite (dans la mesure où les délais de transmission sont assez faibles pour ne pas être le phénomène de ralentissement prédominant). Si T_c est le temps de convoyage dans les zones de stockage, et que le temps de traitement des machines est T, alors on fait la transformation $C_{eq} = C(1 - \frac{T}{T_c})$ pour chacun des stocks.

2.6.2 Pannes des transporteurs

En pratique, on a vu que les zones de stockage sont souvent des convoyeurs, qui sont aussi susceptibles de tomber en panne. Si ces convoyeurs sont en général beaucoup plus fiables que les machines, on peut vouloir prendre en compte leurs défaillances. Un travail sur ces pannes de stock sur le dipôle est proposé par Burman dans [3]. Il propose de résoudre complètement le dipôle dont le stock est non fiable, en résolvant la chaîne de Markov modélisant ce système.

Devant la complexité de la méthode, il propose aussi, en fin d'étude, une heuristique simple pour prendre en compte de ces pannes de stock par transformation du modèle, en se ramenant à l'étude bien connue du dipôle dont le stock est fiable. Nous évoquerons ce travail Chapitre **??**, et nous proposerons de nous servir de ce type de transformation pour prendre en compte ces pannes de stocks dans des lignes simples et ouvertes.

2.6.3 Lignes non-homogènes

Lorsque l'on souhaite traiter des lignes où les machines n'ont pas toutes les mêmes temps de traitement, deux solutions sont proposées pour les lignes simples ouvertes.

Techniques d'homogénéisation

L'objectif de ces techniques présentées dans [27] et [13] est de proposer une transformation de la ligne inhomogène d'origine en une ligne homogène, aisément traitable par l'algorithme DDX. Elles sont au nombre de deux :

- méthode CTA : cette méthode (proposée au départ par [29] pour des modèles asynchrones discrets) consiste à "augmenter" la vitesse de traitement de toutes les machines à celle de la machine la plus rapide de la ligne, et de compenser les paramètres de fiabilité de ces machines en conséquence :

$$U^e = max(U_1, U_2, \ldots, U_K) \tag{2.4a}$$

$$\lambda_i^e = \left[\frac{U^e}{U_i}\left(1+\frac{\lambda_i}{\mu_i}\right)-1\right]^2 \frac{U_i}{U^e}\frac{\mu_i^2}{\lambda_i} \tag{2.4b}$$

$$\mu_i^e = \left[\frac{U^e}{U_i}\left(1+\frac{\lambda_i}{\mu_i}\right)-1\right] \frac{U_i}{U^e}\frac{\mu_i^2}{\lambda_i} \tag{2.4c}$$

- méthode RSA : cette méthode (à l'origine proposée dans [15]) consiste à "réduire" la vitesse de toutes les machines à la vitesse de la machine la plus lente :

$$U^e = min(U_1, U_2, \ldots, U_K) \tag{2.5a}$$

$$\lambda_i^e = \lambda_i \frac{U^e}{U_i} \tag{2.5b}$$

$$\mu_i^e = \mu_i \tag{2.5c}$$

Chacune de ces deux méthodes à des avantage et des inconvénients, aussi des méthodes qui combinent ces deux approches ont été proposé dans [13].

Approche directe

Une approche directe a été proposée dans [27]. Le système d'équations usuel de l'algorithme DDX à été revu pour prendre en compte directement l'inhomogénéité de la ligne. Cette méthode fonctionne bien mais complique sensiblement le système d'équations. De plus, H. Le Bihan dans [27] à noté que cette technique directe revenait fondamentalement à faire une transformation telle qu'évoquée ci-dessus (combinaison des deux méthodes proposées ci-dessus), mais au cours du déroulement de l'algorithme.

Deuxième partie

Lignes de production fermées

3

Résultats préliminaires

L'objectif de ce chapitre est, dans un premier temps, de récapituler la technique d'analyse exacte d'un dipôle en modèle continu. Ces résultats ont été publiés pour la première fois dans [18] dans le cas de distributions exponentielles. Une version étendue aux distributions hyper-exponentielles à deux étages a été traitée dans [27]. C'est cette version que nous reprenons ici, dans une présentation un peu modifiée, mais complète ; le version publiée [27] étant incomplète et partiellement erronée.

Ensuite, on se propose de rappeler le principe détaillé de la méthode de résolution d'une ligne simple ouverte ainsi que l'algorithme de résolution appelé DDX. Ces résultats ont été publiés pour la première fois dans [11] dans le cas de distributions exponentielles pour les mécanismes de panne. Une version étendue aux distributions hyper-exponentielles à deux étages a été traitée dans [27]. C'est cette dernière version que nous exposons ici.

Sommaire

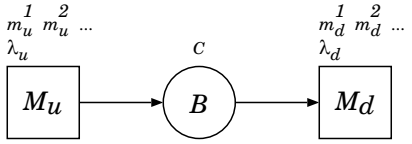

FIG. 3.1 – Modèle de dipôle étudié.

M_u	M_d
productive	productive
en panne	en panne
bloquée	non-alimentée

TAB. 3.1 – États possibles des machines du dipôle.

3.1 Analyse d'un dipôle

L'objectif de cette section est d'analyser le fonctionnement d'un dipôle, ligne élémentaire constituée de deux machines et d'un stock, observé à saturation. On propose pour cela de rappeler la technique d'analyse exacte du modèle fluide extrait de cette ligne proposée par [18] et étendue par [27].

3.1.1 Modèle

On considère un dipôle constitué de deux machines M_u et M_d et d'un stock intermédiaire B (Figure 3.1). Chaque machine produit du matériel fluide à une vitesse constante donnée U_u ou U_d dans la mesure où elle le peut. La machine M_u a toujours du matériel disponible en amont, et déverse du matériel dans le stock B à une vitesse U_u. La machine M_d prend du matériel dans le stock B, et a toujours la possibilité de déverser du matériel. Le stock B a une capacité finie C.

On fait l'hypothèse que la ligne est : $U_u = U_d = U$. On peut alors sans restreindre la généralité de l'analyse poser $U = 1.0$, l'unité de temps étant arbitraire.

Chaque machine peut tomber en panne tandis qu'elle est effectivement en train de travailler. Le temps de bon fonctionnement et le temps de réparation sont aléatoires et suivent des lois de distribution données. On fait l'hypothèse que les temps de bon fonctionnement suivent des lois exponentielles. De manière complètement générique, on caractérise les temps de réparation par une distribution qui est une composition d'exponentielles (Figure 3.2 page ci-contre). Lorsque la machine tombe en panne, elle peut atteindre l'état de panne P_i, qui est une attente de durée exponentiellement distribuée (de taux μ_i) avec la probabilité p_e^i.

Lorsque la machine en amont M_u tombe en panne, la machine M_d continue de fonctionner (sous réserve qu'elle ne tombe pas en panne) jusqu'à ce que le stock B soit vide. A cet instant, M_d arrête de travailler et devient non-alimentée. Lorsque la machine M_d tombe en panne, la machine M_u continue de fonctionner (sous réserve qu'elle ne tombe pas en panne) jusqu'à ce que le stock B soit plein. A cet instant, M_u arrête de travailler et devient bloquée.

Les machines ne peuvent tomber en panne que lorsqu'elles sont effectivement en train de travailler. Leurs états possibles sont ainsi résumés dans la Table 3.1.

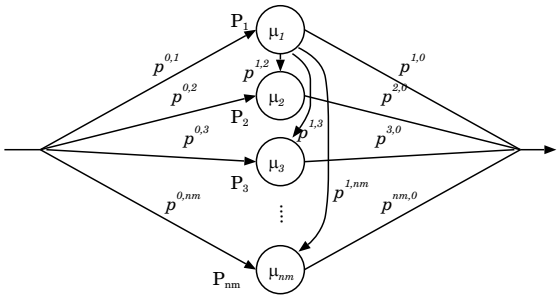

FIG. 3.2 – Distribution du temps de réparation des machines. Elle est constituée de n_m étages exponentiels $(P_1, \ldots, P_{nm})$ de taux $\mu_1, \ldots, \mu_{n_m}$.

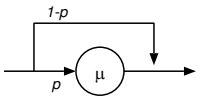

FIG. 3.3 – Distribution exponentielle.

Distributions des temps de réparation

Nous présentons différents cas de distributions des temps de réparation. Les calculs sont présentés dans le cas général. Cependant, en pratique, seules les distributions exponentielle (E), générale-exponentielle (GE) ou hyper-exponentielle à deux étages (HE_2) sont utilisée. On définira la distribution utilisée par le nombre de moments n_m nécessaires à la caractériser.

Cas général Le cas général que l'on observe est celui d'une distribution hyper-exponentielle à n_m étages, telle que présentée Figure 3.2.

Distribution exponentielle Si l'on se limite à une distribution exponentielle le nombre de moments nécessaires à déterminer complètement la distribution est $n_m = 1$. On a la relation simple :

$$\mu = \frac{1}{m}$$

Distributions genérale-exponentielles (GE) Cette distribution est un cas limite de la distribution hyper-exponentielle à deux étages, où l'un des étages à un taux infini (Figure 3.3).

Les deux premiers moments de cette distribution sont :

$$m = \frac{p}{\mu} \text{ et } m^{(2)} = 2\frac{p}{\mu^2}$$

Pour obtenir les paramètres de la distribution à partir des deux premiers moments s'obtient en inversant ces relations :

$$\mu = 2\frac{m}{m^{(2)}} \text{ et } p = 2\frac{m^2}{m^{(2)}}$$

Ce cas est intéressant car il permet de prendre en compte les deux premiers moments de la distribution des temps de réparation (améliorant la précision de l'estimation de ces temps de

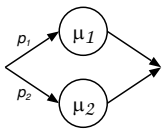

FIG. 3.4 – Distribution hyper-exponentielle à deux étages.

réparation) sans générer de surcoût algorithmique, puisque l'analyse d'un tel dipôle revient à une analyse de dipôle dont les temps de réparation sont exponentiels (cela est démontré dans [9]). Les transformations à faire pour analyser une dipôle en GE par la méthode classique (dipôle dont les temps de réparation sont exponentiels) sont :

$$\lambda^e = \lambda \tag{3.1a}$$
$$\mu^e = p\mu \tag{3.1b}$$

Distributions hyper-exponentielles à 2 étages (HE_2) Cette distribution comporte 2 étages exponentiels, tel que présenté Figure 3.4. Elle est constituée de deux étages exponentiels de taux respectifs μ_1 et μ_2, avec les probabilités respectives p_1 et p_2 (on a bien sûr la contrainte $p_1 + p_2 = 1.0$). Il y a donc trois paramètres indépendants qui déterminent la distribution HE_2. Une telle distribution nécessite donc les trois premiers moments pour être complètement déterminée. Si on connaît les paramètres p_1, p_2, μ_1 et μ_2, on peut déterminer les trois premiers moments par les relations suivantes :

$$m = \frac{p_1}{\mu_1} + \frac{p_2}{\mu_2} \tag{3.2a}$$
$$m^{(2)} = 2\left(\frac{p_1}{\mu_1^2} + \frac{p_2}{\mu_2^2}\right) \tag{3.2b}$$
$$m^{(3)} = 6\left(\frac{p_1}{\mu_1^3} + \frac{p_2}{\mu_2^3}\right) \tag{3.2c}$$

De même lorsqu'on connaît les trois premiers moments de la distribution, on peut obtenir les paramètres de la distribution par les équations suivantes [27, 1] ; si on pose :

$$cv^2 = \frac{m^{(2)} - m^2}{m^2} \tag{3.3a}$$
$$D = \frac{1}{3}\frac{m^{(3)} - 6m^3}{m^3(cv^2 - 1)} - 3 \tag{3.3b}$$
$$\Delta_D = 1 - \left[\frac{D^2}{2(cv^2 - 1)} + 1\right]^{-1} \tag{3.3c}$$

alors :

$$p_1 = \frac{1 + \sqrt{\Delta_D}}{2} \tag{3.4a}$$
$$p_2 = 1 - p_1 \tag{3.4b}$$

si $D > 0$ alors :

$$\frac{1}{\mu_1} = m\left(1 - \sqrt{\frac{p_2}{2p_1}(cv^2 - 1)}\right) \tag{3.5a}$$

$$\frac{1}{\mu_2} = m\left(1 + \sqrt{\frac{p_1}{2p_2}(cv^2 - 1)}\right) \tag{3.5b}$$

si $D < 0$ alors :

$$\frac{1}{\mu_1} = m\left(1 + \sqrt{\frac{p_2}{2p_1}(cv^2 - 1)}\right) \tag{3.6a}$$

$$\frac{1}{\mu_2} = m\left(1 - \sqrt{\frac{p_1}{2p_2}(cv^2 - 1)}\right) \tag{3.6b}$$

Paramètres

Les paramètres d'entrée de notre modèle sont donc les suivants :

- C : capacité du stock B,
- λ_u : taux du temps de bon fonctionnement de M_u,
- nm_u : nombre de serveurs exponentiels de la distribution du temps de réparation de la machine M_u,
- $\mu_u^1, \ldots, \mu_u^{nm_u}$: taux des étages exponentiels de la distribution du temps de réparation de la machine M_u,
- $\mathbf{T}_u = [p_u^{i,j}]_{i,j\in\{1,nm_u\}}$: matrice carrée (dimension nm_u) des probabilités des transitions de la distribution du temps de réparation de la machine M_u,
- $\mathbf{E}_u = [p_u^{0,1}, \ldots, p_u^{0,nm_u}]$: vecteur (dimension nm_u) des probabilités d'entrée de la distribution du temps de réparation de la machine M_u,
- $\mathbf{S}_u = [p_u^{1,0}, \ldots, p_{out_u}^{nm_u,0}]$: vecteur (dimension nm_u) de probabilités de sortie de la distribution du temps de réparation de la machine M_u,
- λ_d : taux du temps de bon fonctionnement de M_d,
- nm_d : nombre de serveurs exponentiels de la distribution du temps de réparation de la machine M_d,
- $\mu_d^1, \ldots, \mu_d^{nm_d}$: taux des étages exponentiels de la distribution du temps de réparation de la machine M_d,
- $\mathbf{T}_d = [p_d^{i,j}]_{i,j\in\{1,nm_d\}}$: matrice carrée (dimension nm_d) des transitions de la distribution du temps de réparation de la machine M_d,
- $\mathbf{E}_d = [p_d^{0,1}, \ldots, p_d^{0,nm_d}]$: vecteur de probabilités d'entrée de la distribution du temps de réparation de la machine M_d,
- $\mathbf{S}_d = [p_d^{1,0}, \ldots, p_d^{nm_d,0}]$: vecteur de probabilités de sortie de la distribution du temps de réparation de la machine M_d.

Paramètres de performance

L'objectif de l'analyse est d'obtenir les paramètres de performance du dipôle. Au nombre de ceux-ci on cherchera en particulier :

- X : taux de production ; c'est la quantité moyenne de matériel produite par le dipôle par unité de temps,
- p_w : probabilité que le dipôle soit en train de produire (donc que M_d le soit),
- p_s : probabilité de non-alimentation de M_d,

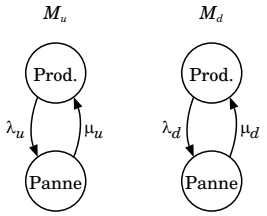

FIG. 3.5 – Chaînes de Markov des deux machines isolées.

- p_b : probabilité de blocage de M_u,
- h_m : en-cours moyen du stock B.

Remarque 1 : p_s (resp. p_b) est la probabilité de non-alimentation (resp. blocage) de M_d (resp. M_u), car le dipôle est saturé, donc M_u (resp. M_d) ne peut jamais être non-alimentée (resp. bloquée).

3.1.2 Analyse du fonctionnement du système

Ce modèle est à la fois discret et continu ; discret car les changements d'état des machines et du buffer (plein, vide, normal) sont de nature discrète, tandis que le variations du niveau de stock sont continues.

C'est donc un modèle markovien à espace d'états mixte $E = \{0, 1, \ldots, nm_u\} \times \{0, 1, \ldots, nm_d\} \times [0, C]$, où $\{0, 1, \ldots\}$ est l'ensemble des états d'une machine : $n > 0 \Rightarrow$ en panne, à l'étape n de la distribution du temps de panne, et $0 \Rightarrow$ productive. $[0, C]$ est l'espace d'états du stock.

Il s'agit donc d'un système ayant des modes de fonctionnement (états) stables dans le temps, pendant lesquels il peut y avoir variation continue de l'état du stock, et des transitions entre ces modes de fonctionnement. Les cas où le stock est plein ou vide sont des cas particuliers de ces modes de fonctionnement, que l'on nomme états externes par opposition aux états internes. On définit l'espace d'états internes par :

$$E_{int} = \{0, 1, \ldots, nm_u\} \times \{0, 1, 2, \ldots, nm_d\} \times]0, C[\tag{3.7}$$

tandis que l'espace d'états externes est donné par :

$$E_{ext} = \{0, 1, \ldots, nm_u\} \times \{0, 1, 2, \ldots, nm_d\} \times \{0, C\} \tag{3.8}$$

Les probabilités associées à ces états sont :
- $f_{a,b}(h)$: densité de probabilité de l'état interne (a, b, h),
- $P_{a,b}(0)$: probabilité de l'état externe $(a, b, 0)$,
- $P_{a,b}(C)$: probabilité de l'état externe (a, b, C).

Fonctionnement discret

Les machines isolées peuvent se trouver dans deux états : *productives* ou *en panne*. On note que l'état *en panne* est en fait constitué lui-même d'un ensemble de sous-états correspondants aux différents étages de la distribution des temps de réparation. La suite de ces deux états est un processus markovien (CMTC) à deux états (Figure 3.5). Dans cette chaîne, l'ensemble des états que peut prendre le mécanisme de panne (autant d'états possibles que d'étages dans la distribution du temps de panne) ont été agrégés en un seul état pour la clarté de la représentation.

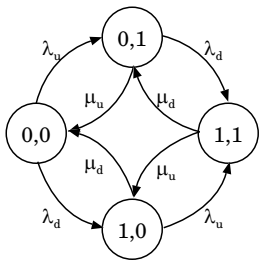

FIG. 3.6 – Chaîne de Markov du tandem constitué des deux machines du dipôle.

Chacune de ces CMTC est caractérisée par son générateur infinitésimal Q_u (on définit Q_d de façon exactement similaire) défini par :

$$Q_u = \begin{bmatrix} -\lambda & p_u^{0,1}\lambda_u & p_u^{0,2}\lambda_u & \dots & p_u^{0,nm}\lambda_u \\ p_u^{1,0}\mu_u^1 & -\mu_u^1 & p^{1,2}\mu_u^1 & \dots & p^{1,nm}\mu_u^1 \\ \dots & \dots & \dots & \dots & \dots \\ p_u^{nm,0}\mu_u^{nm} & p_u^{nm,1}\mu_u^{nm} & \dots & p_u^{nm,nm-1}\mu_u^{nm} & -\mu_u^{nm} \end{bmatrix} \tag{3.9}$$

Le fonctionnement du tandem constitué des deux machines est modélisé par une CMTC à quatre états (si l'on agrège les états en panne de chaque machine) visible Figure 3.6.

Le générateur infinitésimal correspondant est donné par :

$$Q_{dipole} = (Q_u \otimes I_{nm_d}) + (I_{nm_u} \otimes Q_d) \tag{3.10}$$

où $\otimes$ est le produit Kronecker (produit tensoriel), et I_n est la matrice identité d'ordre n (il ne s'agit là que de la réécriture matricielle des équations développées dans [27]).

Fonctionnement dynamique

Chaque fois que le système se trouve dans un des états possibles tels que décrits dans la section précédente, la valeur du niveau du stock évolue de manière continue jusqu'à un changement d'état (une machine tombe en panne ou se trouve réparée), ou jusqu'à ce qu'on ait atteint un état externe (buffer plein ou vide).

Les évolutions dynamiques sont :

- les deux machines sont productives : $\Delta h_m = 0$ (variation de l'encours) ; c'est un état stable. C'est ici vrai car le dipôle est homogène, donc les deux machines produisent à la même vitesse, et le niveau de stock ne varie pas ;
- la machine M_u est productive, M_d est en panne : $\Delta h_m = U\Delta t$; le buffer se remplit à la vitesse U, tant qu'il n'est pas plein ;
- la machine M_u est en panne, M_d est productive : $\Delta h_m = -U\Delta t$; le buffer se vide à la vitesse U, tant qu'il n'est pas vide ;
- les deux machines sont en panne : $\Delta h_m = 0$; c'est un état stable.

On peut modéliser l'ensemble du comportement du système par des chaînes de Markov à temps continu et à espace d'états mixte.

Espaces d'états internes Dans le cas où le stock n'est ni vide ni plein, la chaîne de Markov à espace d'états mixte est donné Figure 3.7 page suivante. Sur cette représentation, les états

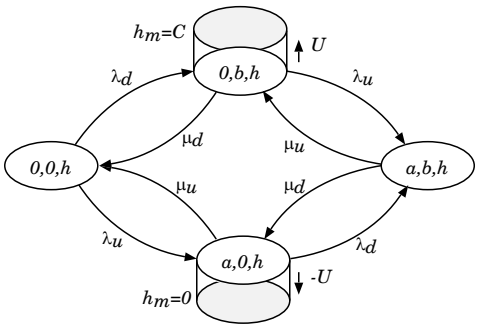

FIG. 3.7 – Chaîne de Markov à temps continu et espace d'états mixte modélisant le fonctionnement du dipôle (espace d'états internes).

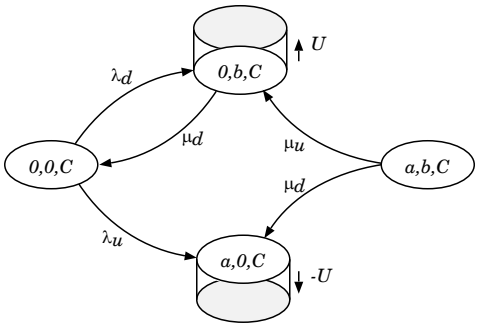

FIG. 3.8 – Chaîne de Markov à temps continu et espace d'états mixte modélisant le fonctionnement du dipôle lorsque le stock est plein (états externes).

où une machine est en panne, qui sont normalement multiples (puisque dépendant du serveur exponentiel qui est en cours), sont représentés de façon agrégé en un seul état *en panne*. Il faut donc comprendre les valeurs μ comme le taux du temps total de réparation. De plus, les ellipses grisées représentent les sens de variation continue du stock, à la vitesse indiquée (U ou $-U$). Ainsi, lorsqu'on est dans l'état $(0, b, h)$ (où M_d est en panne), le stock croît continûment à la vitesse U.

Espaces d'états externes Ceux-ci sont caractérisés par le fait que le stock est soit plein, soit vide :

- Stock plein : Dans le cas où le stock est plein, la chaîne de Markov modélisant le système est montré Figure 3.8.
- Stock vide : Dans le cas où le stock est vide, la chaîne de Markov modélisant le système est donné Figure 3.9 page suivante.

3.1.3 Résolution

La résolution de la chaîne de Markov à espace d'états mixte (états internes du dipôle) permet d'obtenir un système d'équations différentielles donnant les fonctions densités de probabilité des différents états du système. Les espaces d'états externes permettent d'extraire les conditions aux limites donnant les constantes d'intégration.

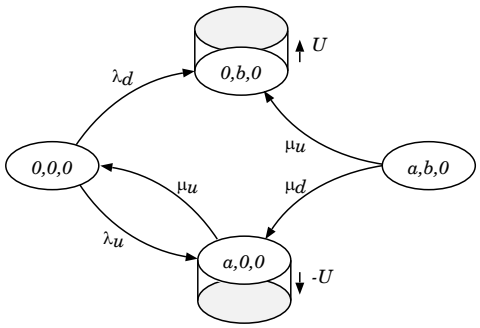

FIG. 3.9 – Chaîne de Markov à temps continu modélisant le fonctionnement du dipôle lorsque le stock est vide (états externes).

Système d'équations différentielles

Le système d'équations différentielles, dont l'établissement est proposé par [18] dans les cas simples de temps de réparation exponentiels, est donné sous forme matricielle par :

$$D_{dipole}\frac{d\mathbf{f}(h)}{dh} = \mathbf{f}(h)Q_{dipole} \tag{3.11}$$

avec $D_{dipole} = D_u \otimes I_{nm_d} + I_{nm_u} \otimes D_d$ où $D_u = \overbrace{\begin{pmatrix} U & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & \dots & \dots & 0 \end{pmatrix}}^{nm_u+1}$ et $D_d = \overbrace{\begin{pmatrix} -U & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & \dots & \dots & 0 \end{pmatrix}}^{nm_d+1}$

et $\mathbf{f}(h)$ est le vecteur des densités de probabilité (fonctions de l'encours h du stock) : $\mathbf{f}(h) = \begin{bmatrix} f_{0,0}(h) & f_{0,1}(h) & f_{0,2}(h) & \dots & f_{0,nm_d}(h) & f_{1,0}(h) & \dots & f_{nm_u,nm_d}(h) \end{bmatrix}$, où $f_{a,b}(h)$ est la fonction densité de probabilité de l'état (a, b), où a est l'état de la machine M_u, et b celui de la machine M_d.

Ce système d'équations différentielles est un système linéaire simple, dont la résolution se fait de façon élémentaire (voir par exemple [34] ou [7]). Le résultat est le vecteur des densités de probabilités $\mathbf{f}(h)$. Ce sont des fonctions de l'encours h, qui comportent un certain nombre de constantes d'intégration à déterminer.

Remarque 2 : La forme générale de la solution de ce système est, pour chaque composante de la solution $f_{i,j}(h)$, une somme de fonctions exponentielles $f_{i,j}(h) = \sum_l k_l e^{r_l h}$. Cependant, la forme particulière de la matrice D_{dipole}, car D_u et D_d sont nulles presque partout, fait qu'en réalité, seul un sous-système de ce système d'équations est le système d'équations différentielles proprement dit, tandis que le reste est un systèmes linéaire simple. En effet, le produit $D_{dipole}\frac{d\mathbf{f}(h)}{dh}$ est un vecteur nul presque partout. C'est seulement lorsqu'une composante de ce vecteur est non nulle que l'on à réellement une équation différentielle. Ainsi, seules les composantes de la forme $f_{0,j}$ et $f_{i,0}$ de la solution f(h) forment un système d'équations différentielles, toutes les autres composantes de f(h) pouvant s'exprimer comme combinaison linéaire ce celles-ci. Les détails de la résolution de ce système sont donnés en Annexe A.

Système d'équations des états externes

Pour établir les équations concernant les états externes, on observe chaque cas individuellement :

– états $(a, 0, C)$: ces états sont transitoires (Figure 3.8 page 30) et donc on a :

$$P_{a,0}(C) = 0 \tag{3.12}$$

– états (a, b, C) : ces états ne sont accessibles que par des états transitoires, donc :

$$P_{a,b}(C) = 0 \tag{3.13}$$

– états $(0, 0, C)$: d'après la Figure 3.8 page 30,

$$(\lambda_u + \lambda_d)P_{0,0}(C) = \sum_{j=1}^{nm_d} p_d^{j,0}\mu_d^j P_{0,j}(C) \tag{3.14}$$

– états $(0, b, C)$: d'après la Figure 3.8 page 30,

$$p_d^{b,0}\mu_d^b P_{0,b}(C) + \sum_{j\neq b} p_d^{b,j}\mu_d^b P_{0,j}(C) = p_u^{0,b}\lambda_d P_{0,0}(C) + Uf_{0,b}(C) \tag{3.15}$$

– états $(0, b, 0)$: ces états sont transitoires (Figure 3.9 page précédente), donc :

$$P_{0,b}(0) = 0 \tag{3.16}$$

– états $(a, b, 0)$: ces états ne sont accessibles que par des états transitoires, donc :

$$P_{a,b}(0) = 0 \tag{3.17}$$

– états $(0, 0, 0)$: d'après la Figure 3.9 page précédente,

$$(\lambda_u + \lambda_d)P_{0,0}(0) = \sum_{i=1}^{nm_u} p_u^{i,0}\mu_u^i P_{i,0}(0) \tag{3.18}$$

– états $(a, 0, 0)$: d'après la Figure 3.9 page précédente,

$$p_u^{0,a}\lambda_u P_{0,0}(0) - Uf_{a,0}(0) = p_u^{a,0}\mu_u^a P_{a,0}(0) + \sum_{i\neq a} p_u^{a,i}\mu_u^i P_{i,0}(0) \tag{3.19}$$

Équations de transitions entre les états internes et externes

Quand notre système est à l'état frontière $(0, 0, C)$, il peut "entrer" dans l'état interne $(a, 0, C)$ au moment où M_u tombe en panne. L'écriture de l'équilibre des flux de cette transition (Figure 3.8 page 30, l'état $(a, 0, C)$ est ici considéré comme étant un état interne) donne :

$$p_u^{0,a}\lambda_u P_{0,0}(C) = Uf_{a,0}(C) \tag{3.20}$$

De même, lorsque le système se trouve être dans l'état frontière $(0, 0, 0)$, il peut "entrer" dans l'état interne $(0, b, 0)$ au moment où M_d tombe en panne. L'équilibre des flux de cette transition (Figure 3.9 page précédente, l'état $(0, b, 0)$ est ici considéré comme étant un état interne) donne :

$$p_d^{0,b}\lambda_d P_{0,0}(0) = Uf_{0,b}(0) \tag{3.21}$$

Synthèse

Ce système d'équations différentielles se résout complètement en déterminant les constantes apparues dans l'expression élémentaire des fonctions $f_{i,0}(h)$ et $f_{0,j}(h)$. Celles-ci peuvent être déterminées grâce à l'ensemble des équations sur les états frontière exposées ci-dessus (équations (3.12) à (3.19)), puis en exprimant la normalisation (ces fonctions étant des densités de probabilité, la somme des probabilités d'être dans les différents états doit être 1). Tous les détails de la résolution en HE_2 sont exposés Annexe A.

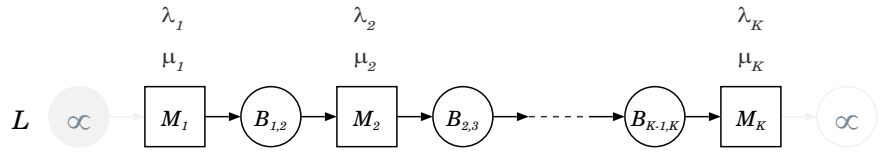

FIG. 3.10 – Ligne de production simple (tandem) ouverte en saturation.

3.2 Analyse d'une ligne simple ouverte : l'algorithme DDX

On cherche à analyser le modèle continu d'une ligne de production manufacturière simple et ouverte L (Figure 3.10). Les hypothèses de fonctionnement de cette ligne sont celles données Section 2.1.2. On rappelle que la ligne fonctionne à saturation (la première machine c'est jamais en famine et la dernière machine n'est jamais bloquée).

3.2.1 Paramètres de la ligne

La ligne L est complètement caractérisée par ses paramètres :
- K : nombre de machines de la ligne ; le nombre de buffers est alors $K-1$;
- λ_i : taux du temps de bon fonctionnement de la machine M_i ;
- μ_i : taux du temps de réparation de la machine M_i ;
- U_i : vitesse de traitement de la machine M_i. On fait l'hypothèse que la ligne L est homogène, de ce fait on pose $\forall i, U_i = 1.0$;
- $C_{i,i+1}$: taille de la zone de stockage $B_{i,i+1}$ située entre les machines M_i et M_{i+1}.

3.2.2 Analyse du modèle continu

Équations de base

Certaines équations peuvent être directement déduites des propriétés élémentaires du système.

La loi de conservation du flux implique que chaque machine doit avoir le même taux de production :

$$X_i = X_j \quad \forall i, j \in \{1, \ldots, M\} \tag{3.22}$$

On en déduit l'égalité des efficacités (sachant que $X_i = Pw_i U_i$, avec $U_i = 1.0$ car la ligne est homogène) :

$$pw_i = pw_j \quad \forall i, j \in \{1, \ldots, M\} \tag{3.23}$$

Les pannes étant dépendantes des opérations par hypothèse, une machine ne peut se trouver que dans un seul des quatre états (productive, en panne, bloquée ou non-alimentée) :

$$pw_i + pf_i + pb_i + ps_i = 1 \tag{3.24}$$

Tant que la machine M_i est en train de produire, elle peut tomber en panne. Le temps de bon fonctionnement suit une loi exponentielle de taux λ_i. Lorsqu'elle est en panne, le temps de

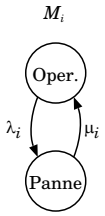

FIG. 3.11 – Chaîne de Markov à temps continu de l'évolution temporelle de l'état (opérationnelle ou en panne) d'une machine M_i dans la ligne

réparation suit aussi une loi exponentielle de taux μ_i (Figure 3.11). En conséquence, l'évolution temporelle de ce système suit un processus markovien à deux états, dont la solution à l'équilibre est donnée par [14] :

$$pw_i\lambda_i = pf_i\mu_i \tag{3.25}$$

Des relations (3.25) et (3.24) et (2.3) , on peut facilement déduire l'équation liant l'efficacité d'une machine dans la ligne L à son efficacité en isolation :

$$pw_i = e_i(1 - ps_i - pb_i) \tag{3.26}$$

3.2.3 La méthode de décomposition

L'analyse directe de la ligne L ainsi définie n'est en général pas possible (sauf pour $K = 2$, voir Section 3.1, ou pour $K = 3$ [5]). L'idée est alors de décomposer la ligne d'origine en plusieurs petites lignes, qui seront analysées de manière exacte (donc en lignes de 2 ou 3 machines), puis de trouver un système d'équations reliant ces petites lignes entre elles de façon à approximer au mieux le fonctionnement de la ligne d'origine. Il s'agit ensuite de trouver un algorithme résolvant le système d'équations engendré.

La méthode que l'on va détailler ici a été proposée par Gershwin dans [21]. Il s'agit d'une décomposition en *dipôles*, du modèle asynchrone discret. Cette méthode à été adaptée au modèle fluide dans [10] puis améliorée dans [27].

On décompose ainsi la ligne simple fermée comportant K machines et $K - 1$ stocks en un ensemble de K dipôles $D(i, i+1)$ tel qu'illustré Figure 3.12 page suivante. Chaque dipôle $D(i, i+1)$ est une petite ligne ouverte (modèle continu) qui comporte une machine amont $M_u(i, i+1)$, une machine aval $M_d(i, i+1)$ et un stock intermédiaire $B(i, i+1)$. La technique de décomposition consiste à déterminer les paramètres de ce dipôle de façon à ce que le flux traversant le stock $B(i, i+1)$ soit le plus proche possible de celui qui traverse $B_{i,i+1}$ dans la ligne originale L. Pour ce faire, on cherche à établir un système d'équations ainsi qu'un algorithme de résolution de ce système de manière à établir les paramètres des dipôles.

On rappelle que chaque dipôle est analysé à saturation : la machines $M_u(i, i+1)$ n'est jamais en famine, de même que la machine $M_d(i, i+1)$ n'est jamais bloquée.

Afin de remplir les objectifs de la décomposition (faire correspondre au mieux les flux traversant $B(i, i+1)$ à ceux traversant $B_{i,i+1}$), la machine $M_u(i, i+1)$ doit produire (et donc alimenter $B(i, i+1)$) de façon la plus semblable possible à la manière dont M_i produit (et donc alimente $B_{i,i+1}$) dans la ligne L.

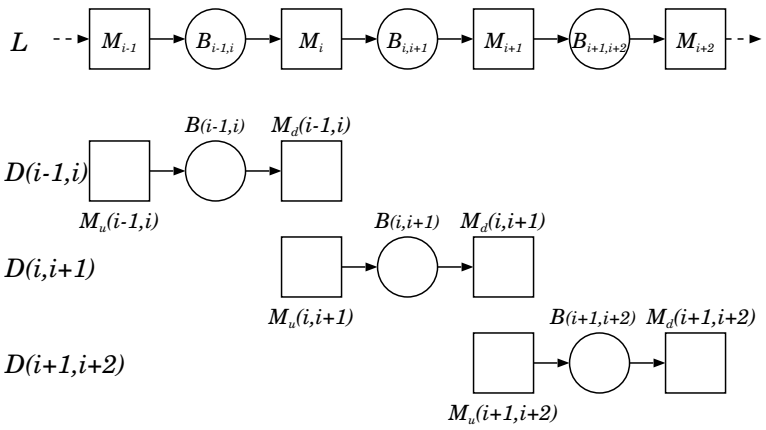

FIG. 3.12 – Principe de la décomposition (sur une partie de la ligne L).

La machine M_i arrête de produire pour trois raisons :

1. lorsqu'elle est en panne ;
2. lorsqu'elle est non-alimentée ;
3. lorsqu'elle est bloquée.

Ce troisième cas est directement modélisé dans $D(i, i+1)$ par un blocage de la machine $M_u(i, i+1)$ (ce qui correspond à une panne de la machine $M_d(i, i+1)$ avec le stock $B(i, i+1)$ plein). La machine $M_u(i, i+1)$ ne pouvant jamais être en famine, cette machine n'arrête de produire que lorsqu'elle tombe en panne (en dehors des blocages déjà évoqués). En conséquence, une panne de la machine $M_u(i, i+1)$ doit représenter à la fois une panne de la machine M_i et une non-alimentation de cette dernière. Comme une non-alimentation de la machine M_i dans la ligne L est une conséquence d'une panne d'une machine située en amont, on peut affirmer que la machine $M_u(i, i+1)$ représente le comportement de la portion de ligne située en amont de M_i.

De plus, comme une panne de la machine $M_u(i, i+1)$ est la conséquence d'une panne d'une des machines précédant le stock $B_{i,i+1}$ dans la ligne L, la réparation de $M_u(i, i+1)$ doit intervenir quand la machine M_j en cause dans le ligne L est réparée. Or le temps résiduel de réparation de le machine M_j au moment de la non-alimentation de M_i (et donc au moment de la panne de $M_u(i, i+1)$) est exponentiellement distribué avec le même taux que celui de la distribution des temps de réparation de la machine M_j (grâce à la propriété sans mémoire de la distribution exponentielle). Une représentation exacte de la distribution des temps de réparation de la machine $M_u(i, i+1)$ est donc une distribution hyper-exponentielle à i étages, où chaque étage correspond à la distribution des temps de réparation des machines $M_j, 1 \leq j < i$. Il est par contre plus compliqué d'estimer la distribution des temps de bon fonctionnement de la machine $M_u(i, i+1)$. En effet, cette dernière n'est en général pas markovienne. Elle est constitué de la distribution du temps de bon fonctionnement de la machine M_i (qui est bien markovien), et de la distribution des temps séparant les pannes des machines amont vidant le stock $B_{i,i+1}$ (qui n'est en générale pas markovienne). Nous choisissons d'approximer la distribution du temps de bon fonctionnement de $M_u(i, i+1)$ par une distribution exponentielle.

De même, une panne de la machine $M_d(i, i+1)$ représente à la fois une panne et un blocage de la machine M_{i+1}. La machine $M_d(i, i+1)$ va donc modéliser le fonctionnement de la portion de ligne située en aval de M_{i+1}. Et une représentation exacte de la distribution des temps de réparation de la machine $M_d(i, i+1)$ est une distribution hyper-exponentielle à $K - i$ étages,

chaque étage correspondant à la distribution des temps de réparation des machines $M_j, i < j \leq K$. On choisit aussi une distribution exponentielle comme distribution des temps de bon fonctionnement de la machine $M_d(i, i+1)$.

Caractérisation des dipôles

Afin de caractériser un dipôle $D(i, i+1)$, il faut déterminer la capacité $C(i, i+1)$ du stock $B(i, i+1)$, les taux de production ainsi que les paramètres des mécanismes de panne et de réparation des machines $M_u(i, i+1)$ et $M_d(i, i+1)$. La capacité $C(i, i+1)$ doit naturellement être choisie égale à celle du stock $B_{i,i+1}$ de la ligne L, de même que les taux de production des machines vont être choisis égaux à $U = 1.0$. Les temps de bon fonctionnement sont décrits pas des lois exponentielles, de taux $\lambda_u(i, i+1)$ pour la machine $M_u(i, i+1)$, et $\lambda_d(i, i+1)$ pour la machine $M_d(i, i+1)$. On a vu ci-dessus qu'une caractérisation exacte du temps de réparation est une distribution hyper-exponentielle à i étages pour $M_u(i, i+1)$ et à $K-i$ étages pour $M_d(i, i+1)$. Cependant, on sait (voir la Section 3.1) que la résolution du dipôle est d'autant plus complexe (dimension du système d'équations différentielles) que les nombres d'états des distributions des temps de réparation des deux machines sont grands. Or, l'objectif est de proposer une méthode simple et rapide (en temps de calcul). On va donc approximer ces distributions des temps de réparation par des distributions exponentielles (E), générale-exponentielles (GE) ou au plus, hyper-exponentielle de à deux étages (HE_2). On notera n_m le nombre de moments nécessaire à caractériser ce temps de réparation (1 pour du E, 2 pour du GE et 3 pour du HE_2). La caractérisation des temps de réparation par des exponentielles est en fait la méthode originale proposée par [10], tandis que les deux autres caractérisations ont été proposées par [27]. On note donc $m_u^{(n)}(i, i+1)$ le moment d'ordre n de la distribution du temps de réparation de la machine $M_u(i, i+1)$, et $m_d^{(n)}(i, i+1)$ le moment d'ordre n de la distribution du temps de réparation de la machine $M_d(i, i+1)$.

Remarque 3 : les moments d'ordre 1 (moyenne) seront écrits sans l'exposant (1).

On note par $\mathcal{F}_u(i, i+1)$ l'ensemble des paramètres de la machine $M_u(i, i+1)$ ($\lambda_u(i, i+1)$ et $m_u^{(n)}(i, i+1)$) et $\mathcal{F}_d(i, i+1)$ l'ensemble des paramètres de la machine $M_d(i, i+1)$ ($\lambda_d(i, i+1)$ et $m_d^{(n)}(i, i+1)$).

On peut alors définir les paramètres suivants, comme on l'a fait pour la ligne L :

- $e_u(i, i+1)$: efficacité en isolation de la machine $M_u(i, i+1)$:

$$e_u(i, i+1) = \frac{1}{1 + \lambda_u(i, i+1) m_u(i, i+1)} \tag{3.27}$$

- $I_u(i, i+1)$:

$$I_u(i, i+1) = \lambda_u(i, i+1) m_u(i, i+1) = \frac{1}{e_u(i, i+1)} - 1 \tag{3.28}$$

- $e_d(i, i+1)$: efficacité en isolation de la machine $M_d(i, i+1)$:

$$e_d(i, i+1) = \frac{1}{1 + \lambda_d(i, i+1) m_d(i, i+1)} \tag{3.29}$$

- $I_d(i, i+1)$:

$$I_d(i, i+1) = \lambda_d(i, i+1) m_d(i, i+1) = \frac{1}{e_d(i, i+1)} - 1 \tag{3.30}$$

Pour chaque dipôle, il y a entre quatre et six paramètres à estimer (selon la caractérisation choisie des temps de réparation). Si l'on suppose que ces inconnues ont été estimées, chaque dipôle peut être analysé en isolation à l'aide de la technique présentée Section 3.1. Les résultats des cette analyse sont les paramètres de performance du dipôle suivants :

- $p_w(i, i+1)$: efficacité du dipôle en isolation. La loi de conservation du flux s'applique au dipôle et implique que $p_w(M_u(i, i+1)) = p_w(M_d(i, i+1))$; on peut alors définir $p_w(i, i+1) = p_w(M_u(i, i+1)) = p_w(M_d(i, i+1))$;
- $p_b(i, i+1)$: proportion du temps où le dipôle est bloqué. Comme la machine aval du dipôle ne peut jamais être bloquée, $p_b(M_d(i, i+1)) = 0$; on définit alors la probabilité de blocage du dipôle comme celle de sa machine amont : $p_b(i, i+1) = p_b(M_u(i, i+1))$;
- $p_s(i, i+1)$: proportion du temps où le dipôle est non-alimenté. Comme la machine amont du dipôle ne peut jamais être en famine, $p_s(M_u(i, i+1)) = 0$; on peut ainsi définir la probabilité de non-alimentation du dipôle comme celle de sa machine aval : $p_s(i, i+1) = p_s(M_d(i, i+1))$;
- $X(i, i+1)$: taux de production du dipôle. Comme les deux machines du dipôles ont le même taux de production (régime permanent), on écrit que $X(i, i+1) = X_u(i, i+1) = X_d(i, i+1)$;
- $h_m(i, i+1)$: en-cours moyen du stock $B(i, i+1)$.

Notation : Cet ensemble de paramètres de performance des dipôles $D(i, i+1)$ sera noté de manière synthétique par $\mathcal{P}(i, i+1)$.

En appliquant (3.26) à la machine $M_u(i, i+1)$ (en se souvenant que cette machine n'est jamais non-alimentée), on obtient :

$$p_w(i, i+1) = e_u(i, i+1)(1 - p_b(i, i+1)) \tag{3.31}$$

De même, en appliquant (3.26) à la machine $M_d(i, i+1)$ (cette dernière ne pouvant jamais être bloquée), on a :

$$p_w(i, i+1) = e_d(i, i+1)(1 - p_s(i, i+1)) \tag{3.32}$$

D'un point de vue formel, l'analyse en isolation du dipôle $D(i, i+1)$ peut être exprimée comme une fonction A dont les paramètres sont l'ensemble des paramètres du dipôle $D(i, i+1)$ (soit $\mathcal{F}_u(i, i+1)$ et $\mathcal{F}_d(i, i+1)$, de même que les constantes du dipôle, $C(i, i+1)$ et $U(i, i+1)$ qui ne sont pas explicitement incorporées dans la liste des paramètres de la fonction), et dont le résultat est l'ensemble $\mathcal{P}(i, i+1)$ des paramètres de performance de $D(i, i+1)$, soit $p_w(i, i+1)$, $p_b(i, i+1)$, $p_s(i, i+1)$ and $h_m(i, i+1)$:

$$\mathcal{P}(i, i+1) = A\left(\mathcal{F}_u(i, i+1),\ \mathcal{F}_d(i, i+1)\right) \tag{3.33}$$

Signalons enfin que, dès lors que les paramètres de performance de chaque dipôle sont connus, ils constituent une approximation des paramètres de performance de la ligne L (et donc les paramètres recherchés).

Équations

Comme illustré Figure 3.13 page suivante, cela consiste à relier les paramètres des mécanismes de panne et de réparation des machines $M_u(i+1, i+2)$ et ceux de $M_d(i-1, i)$ aux paramètres

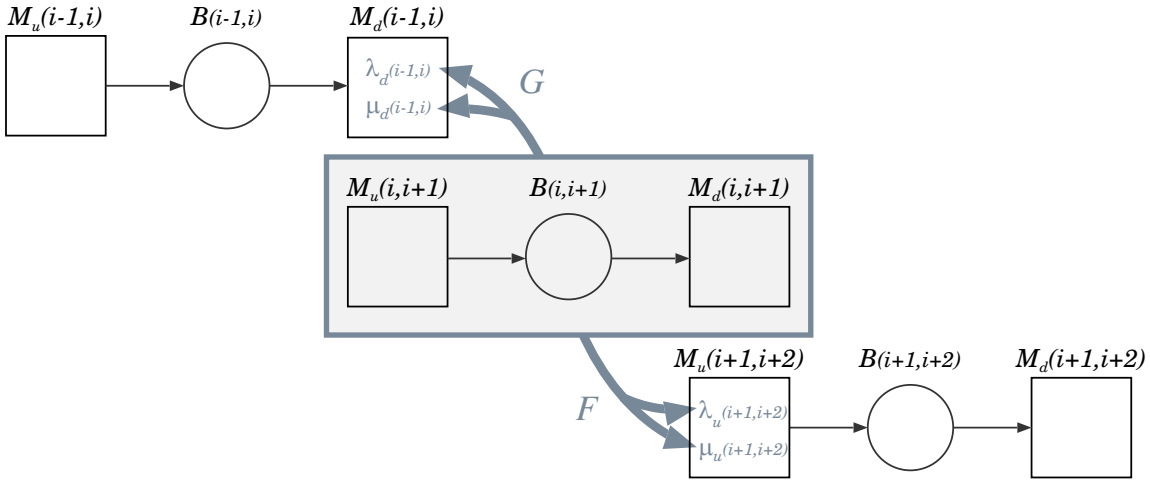

FIG. 3.13 – Illustration graphique des équations $\mathcal{F}_u(i+1,i+2) = F\left(\mathcal{P}\left(i,i+1\right)\right)$ et $\mathcal{F}_d(i-1,i) = G\left(\mathcal{P}\left(i,i+1\right)\right)$

de performance du dipôle $D(i,i+1)$. L'objectif est de déterminer deux fonctions F et G telles que :

$$\mathcal{F}_u(i+1,i+2) = F\left(\mathcal{P}\left(i,i+1\right)\right) \tag{3.34}$$
$$\mathcal{F}_d(i-1,i) = G\left(\mathcal{P}\left(i,i+1\right)\right) \tag{3.35}$$

Les fonctions F et G seront obtenues en utilisant les relations clés suivantes. Premièrement, l'égalité des efficacités des machines $M_u(i,i+1)$, $M_d(i-1,i)$ et M_i s'écrit formellement :

$$pw_i = pw(i,i+1) = pw(i-1,i) \tag{3.36}$$

On note que grâce à la relation (3.23), on peut établir que tous les dipôles ont la même efficacité.

La deuxième relation établit que la machine $M_d(i-1,i)$ doit avoir la même probabilité de non-alimentation que la machine M_i (en effet, si M_{i+1} est non-alimentée, on souhaite que $M_d(i,i+1)$ le soit aussi) :

$$ps_i = ps(i-1,i) \tag{3.37}$$

La troisième relation établit enfin que la machine $M_u(i,i+1)$ doit avoir la même probabilité de blocage que la machine M_i :

$$pb_i = pb(i,i+1) \tag{3.38}$$

À partir des équations (3.23), (3.26), (3.31), (3.32), (3.36), (3.37) et (3.38), on peut facilement déduire les relations suivantes :

$$pw(i,i+1) = e_i(1 - ps(i-1,i) - pb(i,i+1)) \tag{3.39}$$

$$1 + I_d(i-1,i) + I_u(i,i+1) = \frac{1}{p_w(i,i+1)} + I_i = \frac{1}{p_w(i-1,i)} + I_i \tag{3.40}$$

Cette dernière équation est un premier pas vers l'établissement des fonctions F et G. En effet, elle met en relation $I_d(i-1,i)$ avec les paramètres de performance de $D(i,i+1)$ (via $p_w(i,i+1)$ et $I_u(i,i+1)$). En réécrivant cette équation à l'indice $i+1$ au lieu de i, on met en relation $I_u(i+1,i+2)$ avec les paramètres de performance de $D(i,i+1)$ (via $p_w(i,i+1)$ et $I_d(i,i+1)$).

Rappelons qu'une panne de la machine $M_u(i+1,i+2)$ représente à la fois une panne et une famine de M_{i+1}. Soit $\alpha_u(i+1,i+2)$ la proportion des pannes de $M_u(i+1,i+1)$ qui ne sont pas dues à une panne de M_{i+1}. Ces pannes sont donc dues à une non-alimentation de la machine M_{i+1}, cette dernière étant la conséquence soit d'une panne, soit d'une non-alimentation de la machine M_i. Or ces deux événements sont représentés par une panne de la machine $M_u(i,i+1)$. $\alpha_u(i+1,i+2)$ peut alors être considéré comme la proportion de pannes de $M_u(i+1,i+2)$ dues à une panne de $M_u(i,i+1)$.

La distribution $f_u(i+1,i+2)$ des temps de réparation de la machine $M_u(i+1,i+2)$ est la somme pondérée de la distribution $f_u^r(i,i+1)$ des temps résiduels de réparation de la machine $M_u(i,i+1)$ au moment de la non-alimentation de la machine M_i.

On peut alors estimer la distribution des temps de réparation de la machine $M_u(i+1,i+2)$ comme :

$$f_u(i+1,i+2) = \alpha_u(i,i+1) f_u^r(i,i+1) + (1 - \alpha_u(i,i+1))\, f_i \tag{3.41}$$

où f_i est la fonction densité de probabilité du temps de réparation de la machine M_i. D'où l'on tire les relations suivantes (en se souvenant que la distribution des temps de réparation de M_i est exponentielle) :

$$\begin{aligned} m_u^{(n)}(i+1,i+2) = {} & \alpha_u(i,i+1) m_u^{r(n)}(i,i+1) + \\ & (1 - \alpha_u(i,i+1)) n! \left(\frac{1}{\mu_i}\right)^n, \quad \forall i \in \{1,\dots,K-2\} \end{aligned} \tag{3.42}$$

où $m_u^{r(n)}(i,i+1)$ est le moment d'ordre n de la distribution du temps résiduel de réparation de la machine $M_u(i,i+1)$ au moment de la non-alimentation de la machine M_i.

Contrairement aux autres relations écrites jusqu'ici, cette dernière relation n'est qu'approximative. En effet, si une panne d'une machine située en amont de la zone de stockage $B(i,i+1)$ vide celle-ci, il y a de fortes chances qu'une panne ultérieure d'une machine située avant M_i vide aussi ce stock. Cette dépendance n'est pas prise en compte ici.

$\alpha_u(i+1,i+2)$ doit maintenant être estimé. Ce paramètre peut être exprimé comme le rapport entre le nombre moyen de pannes par unité de temps de la machine $M_u(i+1,i+2)$ qui sont dues à une panne de la machine $M_u(i,i+1)$ et le nombre total de pannes par unité de temps de la machine $M_u(i+1,i+2)$.

Le nombre de pannes par unité de temps de la machine $M_u(i+1,i+2)$ (qui est égal, en régime permanent, au nombre de réparations par unité de temps de cette machine) est :

$$\frac{pf_u(i+1,i+2)}{m_u(i+1,i+2)} \tag{3.43}$$

La probabilité que la machine $M_u(i+1,i+2)$ soit en panne et que cette panne soit une conséquence d'une panne de la machine $M_u(i,i+1)$ est égale à $ps(i,i+1)$ (qui est exactement la probabilité que $M_u(i,i+1)$ soit en panne et que le stock $B(i,i+1)$ soit vide). Le nombre de pannes par unité de temps de la machine $M_u(i+1,i+2)$ provenant d'une panne de la machine $M_u(i,i+1)$ (qui est égal au nombre de réparations par unité de temps, en régime permanent), a donc pour valeur :

$$\frac{ps(i,i+1)}{m_u^r(i,i+1)} \tag{3.44}$$

On déduit de (3.43) et (3.44) l'expression de $\alpha_u(i,i+1)$:

$$\alpha_u(i,i+1) = \frac{ps(i,i+1)m_u(i+1,i+2)}{pf_u(i+1,i+2)m_u^r(i,i+1)}, \quad \forall i \in \{1,\dots,K-2\} \tag{3.45}$$

Les transitions entre les états *en panne* et *productive* permettent d'exprimer la probabilité pour que la machine $M_u(i,i+1)$ soit en panne comme :

$$pf_u(i+1,i+2) = I_u(i+1,i+2)pw(i+1,i+2) \tag{3.46}$$

On tire des relations (3.42), (3.45) et (3.46) le résultat suivant :

$$\alpha_u(i+1,i+2) = \left[\mu_{i+1}m_u^r(i,i+1)\left(\frac{I_u(i+1,i+2)pw(i,i+1)}{ps(i,i+1)} - 1\right) + 1\right]^{-1} \tag{3.47}$$

Il est important de noter que l'expression de $\alpha_u(i+1,i+2)$ ne dépend que des paramètres de performance de $D(i,i+1)$ (qui sont déterminable dès que l'on connaît les paramètres de $D(i,i+1)$ et de $I_u(i+1,i+2)$ (dont on connaît une estimation via (3.40)).

Nous obtenons finalement le système d'équations caractérisant la fonction F.

Les machines aux bords de la ligne (M_1 et M_k) n'étant respectivement jamais en famine et jamais bloquée, on déduit les équations concernant les dipôles $L(1,2)$ et $L(K-1,K)$:

$$\lambda_u(1,2) = \lambda_1 \tag{3.48a}$$

$$m_u^{(n)} = m_1^{(n)} \, \forall n \in \{1,\dots,n_m\} \tag{3.48b}$$

Les autres équations s'écrivent, pour $i \in 1,\dots,K-2$:

$$I_u(i+1,i+2) = \frac{1}{pw(i,i+1)} + I_{i+1} - I_d(i,i+1) - 1 \tag{3.49a}$$

$$\alpha_u(i+1,i+2) = \left[\mu_{i+1}m_u^r(i,i+1)\left(\frac{I_u(i+1,i+2)pw(i,i+1)}{ps(i,i+1)} - 1\right) + 1\right]^{-1} \tag{3.49b}$$

$$m_u^{(n)}(i+1,i+2) = \left[\alpha_u(i+1,i+2)m_u^{r(n)}(i,i+1) + (1-\alpha_u(i+1,i+2))\,\frac{n!}{{\mu_{i+1}}^n}\right]^{-1} \tag{3.49c}$$

$$\lambda_u(i+1,i+2) = I_u(i+1,i+2)\frac{1}{m_u(i+1,i+2)} \tag{3.49d}$$

De manière similaire, on peut obtenir le système d'équations caractérisant la fonction G.

$$\lambda_d(K-1,K) = \lambda_K \tag{3.50a}$$
$$m_d^{(n)} = m_K^{(n)} \, \forall n \in \{1,\dots,n_m\} \tag{3.50b}$$

et, pour $i \in 2,\dots,K-1$:

$$I_d(i-1,i) = \frac{1}{pw(i,i+1)} + I_i - I_u(i,i+1) - 1 \tag{3.51a}$$
$$\alpha_d(i-1,i) = \left[\mu_i m_d^r(i,i+1)\left(\frac{I_d(i-1,i)pw(i,i+1)}{pb(i,i+1)} - 1\right) + 1\right]^{-1} \tag{3.51b}$$
$$m_d^{(n)}(i-1,i) = \left[\alpha_d(i,i+1)m_d^{r(n)}(i,i+1) + (1-\alpha_d(i,i+1))\,\frac{n!}{{\mu_i}^n}\right]^{-1} \tag{3.51c}$$
$$\lambda_d(i-1,i) = I_d(i-1,i)\frac{1}{m_d(i-1,i)} \tag{3.51d}$$

Remarquons que les équations (3.49a), (3.49b), (3.51a) et (3.51b) ne sont que des équations intermédiaires.

On note que ces équations font intervenir les temps résiduels de réparation, qu'il s'agit de les déterminer.

Cas de temps de réparation exponentiels. Dans ce cas, les dipôles sont constitués de machines ayant des temps de réparation exponentiels. Le temps résiduel est alors très simple à exprimer. En effet, la propriété sans mémoire de la distribution exponentielle nous permet d'affirmer que le temps résiduel de réparation de la machine $M_u(i,i+1)$ au moment de la non-alimentation de M_{i+1} suit une distribution exponentielle de même taux que la distribution des temps de réparation de $M_u(i,i+1)$. On a ainsi simplement :

$$m_u^r(i+1,i+2) = \frac{1}{\mu_u(i,i+1)} \tag{3.52}$$

De même :

$$m_d^r(i-1,i) = \frac{1}{\mu_d(i,i+1)} \tag{3.53}$$

Cas de temps de réparation générales exponentiels. Les machines des dipôles ont maintenant des temps de réparation qui suivent des lois générales exponentielles. Ces dernières sont complètement caractérisées par la connaissance de leurs deux premiers moments. Il nous faut donc évaluer les deux premiers moments de la distribution des temps résiduels de réparation de la machine $M_u(i+1,i+2)$ à l'instant de la non-alimentation de M_{i+1}. Les temps de réparation de $M_u(i,i+1)$ suivent une distribution exponentielle de taux $\mu_u(i,i+1)$ avec une probabilité $q_u(i,i+1)$, et sont nuls avec une probabilité $1-q_u(i,i+1)$. Or pour qu'une panne de la machine $M_u(i,i+1)$ entraîne une panne de la machine $M_u(i+1,i+2)$, il faut que le temps de réparation de la première soit non nul. En conséquence, la distribution des temps résiduels de réparation de $M_u(i,i+1)$ au moment de la non-alimentation de M_{i+1} suit une loi exponentielle de taux $\mu_u(i,i+1)$. On a donc :

$$m_u^r(i+1,i+2) = \frac{1}{\mu_u(i,i+1)} \tag{3.54a}$$

$$m_u^{r(2)}(i+1,i+2) = \frac{2}{\mu_u(i,i+1)^2} \tag{3.54b}$$

de même que :

$$m_d^r(i-1,i) = \frac{1}{\mu_d(i,i+1)} \tag{3.55a}$$

$$m_d^{r(2)}(i-1,i) = \frac{2}{\mu_d(i,i+1)^2} \tag{3.55b}$$

Cas de temps de réparation hyper-exponentiels à deux étages. Nous envisageons maintenant des temps de réparation des machines des dipôles qui suivent des lois hyper-exponentielles à deux étages. Pour déterminer complètement ces distributions, nous avons besoin de connaître ses trois premiers moments. Du fait de la propriété sans mémoire de la distribution exponentielle, la distribution des temps résiduels de réparation de $M_u(i,i+1)$ est elle aussi une distribution hyper-exponentielle à deux étages. De plus, les taux des deux étages de la distribution du temps résiduel de réparation sont les mêmes taux que ceux du temps de réparation. Seuls changent les probabilité $p_{u1}^r(i,i+1)$ (probabilité que le temps résiduel de réparation de $M_u(i,i+1)$ au moment de la non-alimentation de M_{i+1} soit exponentiel de taux $\mu_{u1}(i,i+1)$) et $p_{u2}^r(i,i+1)$ (probabilité que ce temps soit exponentiel de taux $\mu_{u2}(i,i+1)$) [27].

La probabilité $p_{u1}^r(i,i+1)$ est égale au rapport du nombre de pannes de la machine $M_u(i,i+1)$ qui vident le stock $B(i,i+1)$ et dont la réparation est effectuée par l'étage 1 de la distribution sur le nombre total de pannes de $M_u(i,i+1)$ qui vident le stock.

Soit $P_{1,0}(i,i+1)$ la probabilité que dans le dipôle $D(i,i+1)$, la machine $M_u(i,i+1)$ soit en panne et réparée par l'étage 1, pendant que la machine $M_d(i,i+1)$ fonctionne et que le stock est vide. On raisonne exactement de la même manière avec $P_{2,0}(i,i+1)$ pour l'étage 2.

Le nombre de pannes passant par l'étage 1 est égal au nombre de pannes en sortant. Ce dernier est égal à $P_{1,0}(i,i+1)\mu_{u2}(i,i+1)$. De même, le nombre de pannes passant par l'étage 2 est égal à $P_{0,2}(i,i+1)\mu_{u2}(i,i+1)$. D'où l'on tire :

$$p_{u1}^r(i,i+1) = \frac{P_{1,0}(i,i+1)\mu_{u1}(i,i+1)}{P_{1,0}(i,i+1)\mu_{u1}(i,i+1) + P_{2,0}(i,i+1)\mu_{u2}(i,i+1)} \tag{3.56a}$$

$$p_{u2}^r(i,i+1) = \frac{P_{2,0}(i,i+1)\mu_{u2}(i,i+1)}{P_{1,0}(i,i+1)\mu_{u1}(i,i+1) + P_{2,0}(i,i+1)\mu_{u2}(i,i+1)} \tag{3.56b}$$

Les probabilités $P_{1,0}(i,i+1)$ et $P_{2,0}(i,i+1)$ sont des paramètres de performance que l'on peut obtenir à partir de l'analyse du dipôle $D(i,i+1)$ tel que décrit dans la Section 3.1. Elles sont donc connues.

On peut donc établir complètement les trois premiers moments de la distribution des temps résiduels de réparation de la machine $M_u(i,i+1)$ au moment de la non-alimentation de M_{i+1} :

$$m_u^r(i,i+1) = \frac{p_{u1}^r(i,i+1)}{\mu_{u1}(i,i+1)} + \frac{p_{u2}^r(i,i+1)}{\mu_{u2}(i,i+1)} \tag{3.57a}$$

$$m_u^{r(2)}(i,i+1) = \frac{2p_{u1}^r(i,i+1)}{(\mu_{u1}(i,i+1))^2} + \frac{2p_{u2}^r(i,i+1)}{(\mu_{u2}(i,i+1))^2} \tag{3.57b}$$

$$m_u^{r(3)}(i,i+1) = \frac{6p_{u1}^r(i,i+1)}{(\mu_{u1}(i,i+1))^3} + \frac{6p_{u2}^r(i,i+1)}{(\mu_{u2}(i,i+1))^3} \tag{3.57c}$$

De même :

$$p_{d1}^r(i,i+1) = \frac{P_{0,1}(i,i+1)\mu_{d1}(i,i+1)}{P_{0,1}(i,i+1)\mu_{d1}(i,i+1) + P_{0,2}(i,i+1)\mu_{d2}(i,i+1)} \tag{3.58a}$$

$$p_{d2}^r(i,i+1) = \frac{P_{0,2}(i,i+1)\mu_{d2}(i,i+1)}{P_{0,1}(i,i+1)\mu_{d1}(i,i+1) + P_{0,2}(i,i+1)\mu_{d2}(i,i+1)} \tag{3.58b}$$

d'où :

$$m_d^r(i-1,i) = \frac{p_{d1}^r(i,i+1)}{\mu_{d1}(i,i+1)} + \frac{p_{d2}^r(i,i+1)}{\mu_{d2}(i,i+1)} \tag{3.59a}$$

$$m_d^{r(2)}(i-1,i) = \frac{2p_{d1}^r(i,i+1)}{(\mu_{d1}(i,i+1))^2} + \frac{2p_{d2}^r(i,i+1)}{(\mu_{d2}(i,i+1))^2} \tag{3.59b}$$

$$m_d^{r(3)}(i-1,i) = \frac{6p_{d1}^r(i,i+1)}{(\mu_{d1}(i,i+1))^3} + \frac{6p_{d2}^r(i,i+1)}{(\mu_{d2}(i,i+1))^3} \tag{3.59c}$$

3.2.4 Algorithmes de résolution

Nous disposons ici en réalité d'autant d'équations que d'inconnues. En effet, pour chaque dipôle, on dispose de $(n_m+1)2(K-1)$ équations (les équations (3.49a) à (3.51d)), plus les $2(n_m+1)$ équations aux bords (pour $M_u(1,2)$ et $M_d(K-1,K)$). Soit un total de $2K(n_m+1)$ équations indépendantes, et autant d'inconnues (chaque dipôle ayant $2(n_m+1)$ variables à déterminer, à savoir λ_u, λ_d, $m_u^{(1)}$, $m_u^{(2)}$, …, $m_u^{(n_m)}$, $m_d^{(1)}$, $m_u^{(2)}$, …, $m_d^{(n_m)}$).

Nous disposons donc d'un système d'équations bien défini, mais il n'est malheureusement pas simple (car il n'est pas linéaire), en raison de l'expression des paramètres de performance des dipôles. Sa résolution exacte n'est pas à priori accessible. On en cherche donc une solution approximative.

Principe

La méthode proposée est la recherche itérative d'un point fixe. Le système à résoudre étant : pour tout $i \in \{2, K-2\}$

$$\mathcal{P}(i,i+1) = A(\mathcal{F}_u(i,i+1), \mathcal{F}_d(i,i+1)) \tag{3.60a}$$

$$\mathcal{F}_u(i+1,i+2) = F(\mathcal{P}(i,i+1)) \tag{3.60b}$$

$$\mathcal{F}_d(i-1,i) = G(\mathcal{P}(i,i+1)) \tag{3.60c}$$

Comme toujours avec ce type de méthode, le système est résolu en choisissant des valeurs initiales pour les paramètres inconnus ($\mathcal{F}_u(i,i+1)$ et $\mathcal{F}_d(i,i+1)$ pour chaque dipôle), puis à chaque itération de l'algorithme, on applique les équations du système avec les paramètres précédemment estimés afin d'obtenir de meilleures estimations de ces derniers. Le processus

s'arrête lorsque les paramètres calculés au cours de deux itérations successives sont suffisamment proches. On estime alors que la convergence est atteinte, même si aucune preuve formelle peut être apportée pour justifier cette convergence. Les paramètres ainsi obtenus sont considérés comme de bonnes estimations de la solution (supposée unique) du système. A nouveau, aucune preuve de l'unicité de la solution n'a été fournie à ce jour à notre connaissance. Il semblerait même qu'il existe des cas où cela n'est pas vrai [27], mais ces derniers cas sont extrêmement rares.

Algorithmes

L'algorithme est le même quel que soit la caractérisation choisie des distributions des temps de réparation des machines des dipôles (Algorithme 3.1). Par contre, le nombre de paramètres à calculer à chaque itération dépend de cette caractérisation des temps de réparation des dipôles.

Algorithme 3.1 Algorithmes DDX : principe

Requiert : Paramètres de la ligne L (λ_i, μ_i et $C_{i,i+1}$)

Initialiser : Conditions aux limites :
$\lambda_u(1,2) = \lambda_1$
$m_u^{(n)}(1,2) = \frac{n!}{\mu_1^n}$
$\lambda_d(K-1,K) = \lambda_K$
$m_d^{(n)}(K-1,K) = \frac{n!}{\mu_K^n}$
Initialiser :
pour pour $i = 2, \ldots, K-2$: **faire**
  $\lambda_d(i,i+1) = \lambda_{i+1}$
  $m_d^{(n)}(i,i+1) = \frac{n!}{\mu_{i+1}^n}$
fin pour
répéter
  pour $i = 1$ à $K-2$ **faire**
    calculer les paramètres de performance du dipôle $D(i,i+1)$
    mettre à jour $\lambda_u(i+1,i+2)$ et $m_u^{(n)}(i+1,i+2)$ à partir de (3.49b), (3.49c) et (3.49d)
  fin pour
  pour $i = K-1$ à 2 **faire**
    calculer les paramètres de performance du dipôle $D(i,i+1)$
    mettre à jour $\lambda_d(i-1,i)$ et $m_d^{(n)}(i-1,i)$ à partir de (3.51b), (3.51c) et (3.51d)
  fin pour
jusqu'à convergence

Algorithme $DDX - E$ L'algorithme Algorithme 3.2 donne le détail de la procédure pour le cas des temps de réparation des machines des dipôles exponentiels.

Algorithme $DDX - GE$ Pour des distributions générales exponentielles des temps de réparation des machines des dipôles, il faut évaluer à chaque étape deux moments des distributions. Chaque temps de réparation de la machine $M_u(i,i+1)$ peut prendre un temps exponentiellement distribué de taux $\mu_u(i,i+1)$ avec la probabilité $q_u(i,i+1)$, et un temps nul avec la probabilité $1 - q_u(i,i+1)$. De même pour les machines $M_d(i,i+1)$.

L'algorithme correspondant est donné algorithme Algorithme 3.3.

Algorithme 3.2 Algorithme $DDX - E$

Requiert : Paramètres de la ligne L (λ_i, μ_i et $C_{i,i+1}$)

Initialiser : Conditions aux limites :
$\lambda_u(1,2) = \lambda_1$
$\mu_u(1,2) = \mu_1$
$\lambda_d(K-1,K) = \lambda_K$
$\mu_d(K-1,K) = \mu_K$

Initialiser : pour $i = 2, \ldots, K-2$:
$\lambda_d(i,i+1) = \lambda_{i+1}$
$\mu_d(i,i+1) = \mu_{i+1}$

répéter

  pour $i = 1$ à $K-2$ **faire**

    calculer les paramètres de performance du dipôle $D(i,i+1)$ ($pw(i,i+1)$, $ps(i,i+1)$)

    calculer :

$$I_u(i+1,i+2) = \frac{1}{pw(i,i+1)} + I_{i+1} - I_d(i,i+1) - 1$$

$$\alpha_u(i+1,i+2) = \left[\frac{\mu_{i+1}}{\mu_u(i,i+1)}\left(\frac{I_u(i+1,i+2)pw(i,i+1)}{ps(i,i+1)} - 1\right) + 1\right]^{-1}$$

$$\mu_u(i+1,i+2) = \left[\frac{\alpha_u(i+1,i+2)}{\mu_u(i,i+1)} + \frac{1-\alpha_u(i+1,i+2)}{\mu_{i+1}}\right]^{-1}$$

$$\lambda_u(i+1,i+2) = I_u(i+1,i+2)\mu_u(i+1,i+2)$$

  fin pour

  pour $i = K-1$ à 2 **faire**

    calculer les paramètres de performance du dipôle $D(i,i+1)$ ($pw(i,i+1)$, $pb(i,i+1)$)

    calculer :

$$I_d(i-1,i) = \frac{1}{pw(i,i+1)} + I_{i+1} - I_u(i+1,i+2) - 1$$

$$\alpha_u(i-1,i) = \left[\frac{\mu_{i+1}}{\mu_d(i,i+1)}\left(\frac{I_u(i-1,i)pw(i,i+1)}{pb(i,i+1)} - 1\right) + 1\right]^{-1}$$

$$\mu_d(i-1,i) = \left[\frac{\alpha_u(i-1,i)}{\mu_u(i,i+1)} + \frac{1-\alpha_u(i+1,i+2)}{\mu_{i+1}}\right]^{-1}$$

$$\lambda_d(i-1,i) = I_u(i-1,i)\mu_u(i-1,i)$$

  fin pour

jusqu'à convergence des paramètres

Algorithme 3.3 Algorithme $DDX - GE$

Requiert : Paramètres de la ligne L (λ_i, μ_i et $C_{i,i+1}$)

Initialiser : Conditions aux limites :
$\lambda_u(1,2) = \lambda_1$
$\mu_u(1,2) = \mu_1$
$q_u(1,2) = 1.0$
$\lambda_d(K-1,K) = \lambda_K$
$\mu_d(K-1,K) = \mu_K$
$q_d(K-1,K) = 1.0$

Initialiser : pour $i = 2, \ldots, K-2$:
$\lambda_d(i,i+1) = \lambda_{i+1}$
$\mu_d(i,i+1) = \mu_{i+1}$
$q_d(i,i+1) = 1.0$

répéter
  pour $i = 1$ à $K-2$ **faire**
    calculer les paramètres de performance du dipôle $D(i,i+1)$ ($pw(i,i+1)$, $ps(i,i+1)$)
    calculer :
    $I_u(i+1,i+2) = \frac{1}{pw(i,i+1)} + I_{i+1} - I_d(i,i+1) - 1$
    $\alpha_u(i+1,i+2) = \left[\frac{\mu_{i+1}}{\mu_u(i,i+1)}\left(\frac{I_u(i+1,i+2)pw(i,i+1)}{ps(i,i+1)} - 1\right) + 1\right]^{-1}$
    $m_u(i+1,i+2) = \frac{\alpha_u(i+1,i+2)}{\mu_u(i,i+1)} + \frac{1-\alpha_u(i+1,i+2)}{\mu_{i+1}}$
    $m_u^{(2)}(i+1,i+2) = \frac{2\alpha_u(i+1,i+2)}{(\mu_u(i,i+1))^2} + 2\frac{1-\alpha_u(i+1,i+2)}{\mu_{i+1}^2}$
    $\mu_u(i+1,i+2) = 2\frac{m_u(i+1,i+2)}{m_u^{(2)}(i+1,i+2)}$
    $q_u(i+1,i+2) = 2\frac{(m_u(i+1,i+2))^2}{m_u^{(2)}(i+1,i+2)}$
    $\lambda_u(i+1,i+2) = \frac{I_u(i+1,i+2)}{m_u(i+1,i+2)}$
  fin pour
  pour $i = K-1$ à 2 **faire**
    calculer les paramètres de performance du dipôle $D(i,i+1)$ ($pw(i,i+1)$, $pb(i,i+1)$)
    calculer :
    $I_d(i-1,i) = \frac{1}{pw(i,i+1)} + I_{i+1} - I_u(i+1,i+2) - 1$
    $\alpha_u(i-1,i) = \left[\frac{\mu_{i+1}}{\mu_d(i,i+1)}\left(\frac{I_u(i-1,i)pw(i,i+1)}{pb(i,i+1)} - 1\right) + 1\right]^{-1}$
    $m_d(i-1,i) = \frac{\alpha_d(i-1,i)}{\mu_d(i,i+1)} + \frac{1-\alpha_d(i-1,i)}{\mu_{i+1}}$
    $m_d^{(2)}(i-1,i) = \frac{2\alpha_d(i-1,i)}{(\mu_d(i,i+1))^2} + 2\frac{1-\alpha_d(i-1,i)}{\mu_{i+1}^2}$
    $\mu_d(i-1,i) = 2\frac{m_d(i-1,i)}{m_d^{(2)}(i-1,i)}$
    $q_d(i-1,i) = 2\frac{(m_d(i-1,i))^2}{m_d^{(2)}(i-1,i)}$
    $\lambda_d(i-1,i) = \frac{I_d(i-1,i)}{m_d(i-1,i)}$
  fin pour
jusqu'à convergence des paramètres

Algorithme $DDX-HE_2$ Pour des distributions hyper-exponentielles à deux étages des temps de réparation des machines des dipôles, il faut évaluer à chaque étape trois moments des distributions. Chaque temps de réparation de la machine $M_u(i, i+1)$ peut prendre un temps exponentiellement distribué de taux $\mu_{u1}(i, i+1)$ avec la probabilité $p_{u1}(i, i+1)$, et un temps temps exponentiellement distribué de taux $\mu_{u2}(i, i+1)$ avec la probabilité $p_{u2}(i, i+1) = 1.0 - p_{u1}(i, i+1)$. De même pour les machines $M_d(i, i+1)$.

L'algorithme correspondant est donné Algorithme 3.4.

Algorithme 3.4 Algorithme $DDX - HE_2$

Requiert : Paramètres de la ligne L (λ_i, μ_i et $C_{i,i+1}$)

Initialiser : Conditions aux limites :

$$\lambda_u(1,2) = \lambda_1 \qquad \lambda_d(K-1,K) = \lambda_K$$
$$\mu_{u1}(1,2) = \mu_{u2}(1,2) = \mu_1 \qquad \mu_{d1}(K-1,K) = \mu_{d2}(K-1,K) = \mu_K$$
$$p_{u1}(1,2) = 1.0 \qquad p_{d1}(K-1,K) = 1.0$$
$$p_{u2}(1,2) = 0.0 \qquad p_{d2}(K-1,K) = 0.0$$

Initialiser : pour $i = 2, \ldots, K-2$:

$\lambda_d(i,i+1) = \lambda_{i+1}$

$\mu_{d1}(i,i+1) = \mu_{d2}(i,i+1) = \mu_{i+1}$

$p_{d1}(i,i+1) = 1.0$

$p_{d2}(i,i+1) = 0.0$

répéter

  pour $i = 1$ à $K-2$ **faire**

    calculer les paramètres de performance du dipôle $D(i,i+1)$ ($pw(i,i+1)$, $ps(i,i+1)$, $p^r_{u1}(i,i+1)$, $p^r_{u2}(i,i+1)$)

    calculer :

$$I_u(i+1,i+2) = \frac{1}{pw(i,i+1)} + I_{i+1} - I_d(i,i+1) - 1$$

$$\alpha_u(i+1,i+2) = \left[\frac{\mu_{i+1}}{\mu_u(i,i+1)}\left(\frac{I_u(i+1,i+2)pw(i,i+1)}{ps(i,i+1)} - 1\right) + 1\right]^{-1}$$

    pour $n = 1$ à 3 **faire**

$$m_u^{(n)}(i+1,i+2) = \alpha_u(i+1,i+2)\left(\frac{n!p^r_{u1}(i,i+1)}{(\mu_{u1}(i,i+1))^n} + \frac{n!p^r_{u2}(i,i+1)}{(\mu_{u2}(i,i+1))^n}\right) + (1-\alpha_u(i+1,i+2))n!\left(\frac{1}{\mu_{i+1}}\right)^n$$

    fin pour

    calculer : $\mu_{u1}(i+1,i+2)$, $\mu_{u2}(i+1,i+2)$, $p_{u1}(i+1,i+2)$ et $p_{u2}(i+1,i+2)$ à partir de la méthode détaillée en Annexe A

    évaluer $\lambda_u(i+1,i+2) = I_u(i+1,i+2)\frac{1}{m_u(i+1,i+2)}$

  fin pour

  pour $i = K-1$ à 2 **faire**

    calculer les paramètres de performance du dipôle $D(i,i+1)$ ($pw(i,i+1)$, $pb(i,i+1)$)

    calculer :

$$I_d(i-1,i) = \frac{1}{pw(i,i+1)} + I_{i+1} - I_u(i+1,i+2) - 1$$

$$\alpha_u(i-1,i) = \left[\frac{\mu_{i+1}}{\mu_d(i,i+1)}\left(\frac{I_u(i-1,i)pw(i,i+1)}{pb(i,i+1)} - 1\right) + 1\right]^{-1}$$

    pour $n = 1$ à 3 **faire**

$$m_d^{(n)}(i-1,i) = \alpha_d(i-1,i)\left(\frac{n!p^r_{d1}(i,i+1)}{(\mu_{d1}(i,i+1))^n} + \frac{n!p^r_{d2}(i,i+1)}{(\mu_{d2}(i,i+1))^n}\right) + (1-\alpha_d(i-1,i))n!\left(\frac{1}{\mu_{i+1}}\right)^n$$

    fin pour

    calculer : $\mu_{d1}(i-1,i)$, $\mu_{d2}(i-1,i)$, $p_{d1}(i-1,i)$ et $p_{d2}(i-1,i)$ à partir de la méthode détaillée en Annexe A

    évaluer $\lambda_d(i-1,i) = I_d(i-1,i)\frac{1}{m_d(i-1,i)}$

  fin pour

jusqu'à convergence des paramètres

4

Cas d'une ligne de production constituée d'une boucle simple

Nous proposons de traiter les boucles fermées simples (en tandem). La technique employée dérive directement de ce qui a été présenté Section 3.2. Nous allons d'abord présenter la méthode qui avait été étudiée par [20], puis nous étudierons les raisons de la piètre efficacité (robustesse et vitesse d'exécution) de la méthode de résolution existante. Enfin nous proposerons une autre méthode de résolution qui tente de contourner les difficultés à l'origine des problèmes de la méthode initiale.

Sommaire

4.1 Le modèle

On considère une ligne de production fermée simple constituée de K machines $\{M_i\}_{i\in\{1..K\}}$ séparées pas K stocks intermédiaires tel que montré Figure 4.1 page suivante. Les éléments sont transportés par un nombre fixé N de palettes au travers de la ligne de production. La ligne est supposée homogène, en conséquence, toutes les machines ont le même temps de traitement noté T. On peut poser $T = 1.0$, l'unité de temps étant arbitraire [12]. Les machines sont sujettes à des pannes dépendant des opérations. Le temps de bon fonctionnement de la machine M_i est supposé suivre une loi de distribution exponentielle de taux λ_i. Le temps de réparation de la machine M_i est de même supposé exponentiellement distribué (de taux μ_i).

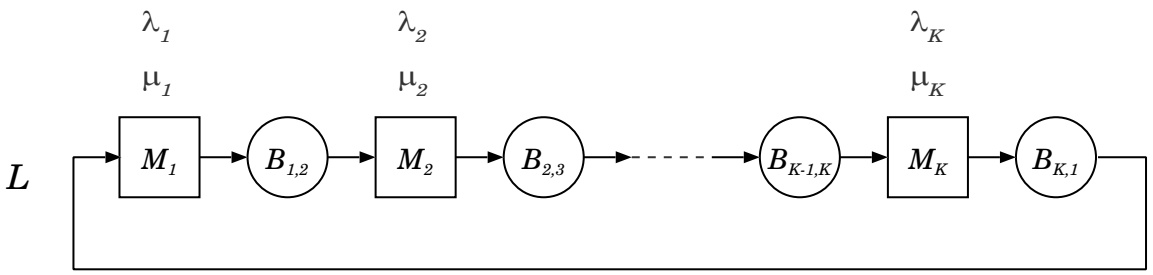

FIG. 4.1 – Modèle de la ligne de production

Remarque 4 : les machines sont numérotées de 1 à K. Dans le reste de ce chapitre, on adopte la convention suivante :

$$i = K \Rightarrow i + 1 = 1$$

et

$$i = 1 \Rightarrow i - 1 = K$$

Il est important de rappeler qu'une machine M_i ne peut pas atteindre simultanément les états bloqué et non-alimenté, car cet événement correspond à l'occurrence simultanée de deux événements indépendants : le fait que $B_{i-1,i}$ et $B_{i,i+1}$ deviennent simultanément vide et plein. En effet, ce double événement a une probabilité d'occurrence qui est du second ordre et est donc négligeable. Cependant, la configuration en boucle simple fermée autorise la situation suivante (exemple d'une ligne de trois machines) : les deux stocks encadrant une machine ont la même capacité, et sont également remplis. Si la machine située entre ces deux stocks tombe en panne assez longtemps, le stock amont se remplira complètement en même temps exactement que le stocks aval se videra complètement. On se retrouve donc avec deux événements simultanés. Mais en pratique, cette situation ne peut survenir que de manière transitoire (suite à des conditions initiales particulières) et ne saurait affecter les performances du système en régime permanent. On rappelle de plus qu'une machine ne peut tomber en panne que lorsqu'elle est effectivement en train de produire.

4.1.1 Équations de base et décomposition

On cherche à établir l'équivalent de la méthode de décomposition proposée par [11] et détaillée Section 3.2 à ce système bouclé. Une première version de cette idée avait été proposée par [20]. Nous utiliseront ici une caractérisation exponentielle des distributions des temps de panne des machines des dipôles de la décomposition, ceci afin de simplifier les écritures des systèmes et équations, mais notons la technique s'adapte directement aux distributions GE et HE_2 comme pour l'algorithme DDX présenté Section 3.2.

L'ensemble des raisonnements tenus pour exprimer les systèmes d'équations F et G pour la ligne simple ouverte (voir section Section 3.2.3) peuvent être tenus à l'identique ici. Le décomposition s'adapte au cas fermé simplement (Figure 4.2 page 54). A chaque stock $B_{i,i+1}$ de la ligne L on associe un dipôle $D(i, i+1)$, dont les paramètres seront choisis de façon à ce que le flux de matériel traversant le stock $B(i, i+1)$ du dipôle soit le plus proche possible du flux traversant le stock $B_{i,i+1}$ dans L.

Le système d'équations est construit exactement de la même manière que pour la ligne ouverte, mais les équations sont valables pour toutes les valeurs de i (il n'y a pas de première ou

dernière machine). On peut directement repérer les équations : pour tout $i \in \{1, K\}$

$$I_u(i+1,i+2) = \frac{1}{pw(i,i+1)} + I_{i+1} - I_d(i,i+1) - 1 \tag{4.1a}$$

$$\alpha_u(i+1,i+2) = \left[\mu_{i+1} m_u^r(i,i+1)\left(\frac{I_u(i+1,i+2)pw(i,i+1)}{ps(i,i+1)} - 1\right) + 1\right]^{-1} \tag{4.1b}$$

$$\mu_u(i+1,i+2) = \left[\frac{\alpha_u(i+1,i+2)}{\mu_u(i,i+1)} + \frac{1-\alpha_u(i+1,i+2)}{\mu_{i+1}}\right]^{-1} \tag{4.1c}$$

$$\lambda_u(i+1,i+2) = I_u(i+1,i+2)\mu_u(i+1,i+2) \tag{4.1d}$$

$$I_d(i-1,i) = \frac{1}{pw(i,i+1)} + I_i - I_u(i,i+1) - 1 \tag{4.2a}$$

$$\alpha_d(i-1,i) = \left[\mu_i m_d^r(i,i+1)\left(\frac{I_d(i-1,i)pw(i,i+1)}{pb(i,i+1)} - 1\right) + 1\right]^{-1} \tag{4.2b}$$

$$\mu_d(i-1,i) = \left[\frac{\alpha_d(i,i+1)}{\mu_d(i,i+1)} + \frac{1-\alpha_d(i,i+1)}{\mu_i}\right]^{-1} \tag{4.2c}$$

$$\lambda_d(i-1,i) = I_d(i-1,i)\mu_d(i-1,i) \tag{4.2d}$$

Cependant, nous perdons par rapport à la version ouverte les valeurs des paramètres $\lambda_u(1,2)$, $\mu_u^{(n)}(1,2)$, $\lambda_d(K-1,K)$ et $\mu_u^{(n)}(K-1,K)$, qui étaient connus, étant donné que nous n'avons plus de "début" ni de "fin" de ligne. Dans le cas de la ligne simple, ces paramètres étaient parfaitement déterminés, et ils n'évoluaient pas pendant l'évolution de l'algorithme. Dans le cas de la ligne bouclée, aucun des paramètres des dipôles n'est ainsi fixé dès le départ. Il n'y a donc plus de "conditions aux limites".

Mais nous avons une contrainte supplémentaire : la somme des en-cours dans les stocks est constante et égale à N. C'est une contrainte permanente (vraie à chaque instant), donc l'égalité reste vraie pour la somme des en-cours moyens :

$$\sum_{i=1}^{K} hm_{i,i+1} = N \tag{4.3}$$

Rappelons que nous avons choisi une caractérisation exponentielle des distributions des temps de panne. Donc, les systèmes F et G constituent un ensemble de $4K$ équations. Rappelons que les équations (4.1a), (4.1b), (4.2a) et (4.2b) ne sont que des intermédiaires de calcul.

Dans le contexte de lignes fermées, on sait d'après [20] que parmi ces $4K$ équations, l'une de celles-ci est linéairement dépendante des autres. Cette dépendance résulte de la structure en boucle, et elle s'explique assez simplement. Pour chaque dipôle, nous avons utilisé l'équation $X_{i+1} = X_i$ pour établir la relation clé (3.40) que l'on retrouve dans nos systèmes d'équations sous la forme des relations (4.1a) et (4.2a). Pour $i < K$, à chaque fois que l'on écrit $X_{i+1} = X_i$, on ajoute bien une *nouvelle* équation, indépendante des autres relations déjà relevées. Mais par

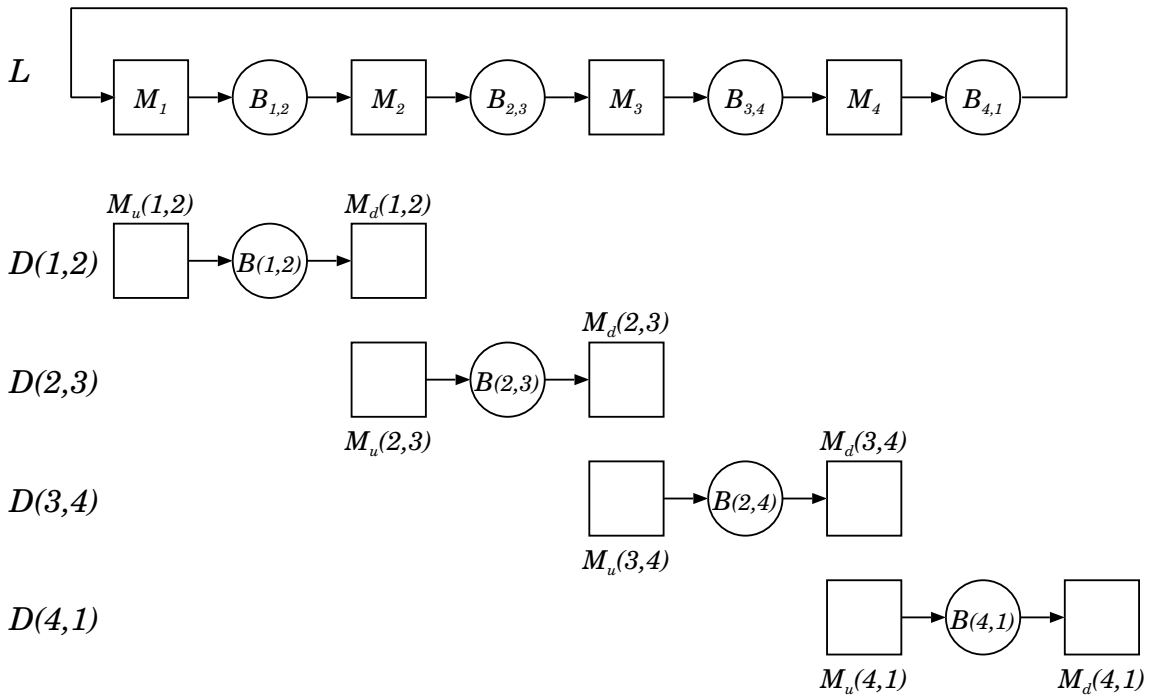

FIG. 4.2 – Principe de la décomposition pour une ligne simple fermée (de 4 machines).

rapport à la version originale (ligne ouverte), nous écrivons une fois de plus ces relations, pour le dipôle $D(K,1)$. Or au moment où nous posons, pour établir les équations de la décomposition de ce dipôle, la relation $X_1 = X_K$, nous n'écrivons pas une nouvelle équation, indépendante des autres relations. Elle peut en effet se déduire par transitivité de l'ensemble des autres relations $X_{i+1} = X_i$ déjà posées.

Ce système forme donc un ensemble de $4K-1$ équations indépendantes à $4K$ inconnues (les paramètres $\lambda_u(i,i+1)$, $\mu_u(i,i+1)$, $\lambda_d(i,i+1)$ et $\mu_d(i,i+1)$). Mais nous n'avons pas encore utilisé une propriété de la ligne : la contrainte de population (4.3). Afin de prendre cette contrainte en compte, on note que la somme des en-cours moyens des stocks des dipôles doit être égale au nombre de clients N circulant dans la ligne :

$$\sum_{i=1}^{K} hm(i,i+1) = N \tag{4.4}$$

En définitive, nous obtenons un système d'équations (relations (4.1a) à (4.2d)) dans lequel l'une des équations à été "éliminée" au profit de la contrainte de population. La résolution de ce système consiste donc à choisir quelle relation doit être éliminée, et comment prendre en compte la contrainte de population (4.4) dans l'algorithme.

4.2 Méthodes de résolution

Le principe de base de toutes les méthodes de résolution du système que l'on présente ici est celui de la ligne simple ouverte présenté Section 3.2. C'est une technique de résolution d'équation par recherche de point fixe [10], qui consiste à résoudre itérativement le système suivant :

$$\mathcal{P}(i,i+1) = A\left(\mathcal{F}_u(i,i+1),\ \mathcal{F}_d(i,i+1)\right) \tag{4.5a}$$
$$\mathcal{F}_u(i+1,i+2) = F\left(\mathcal{P}\left(i,i+1\right)\right) \tag{4.5b}$$
$$\mathcal{F}_d(i-1,i) = G\left(\mathcal{P}\left(i,i+1\right)\right) \tag{4.5c}$$

auquel la contrainte de population (4.4) doit être ajouté.

Le système est résolu en choisissant arbitrairement des valeurs initiales pour les paramètres inconnus ($\mathcal{F}_u(i,i+1)$ et $\mathcal{F}_d(i,i+1)$ pour chaque dipôle), puis à chaque itération de l'algorithme, on applique les équations du système avec les paramètres précédemment estimés afin d'obtenir de meilleures estimations de ces derniers. Le processus s'arrête les paramètres calculés au cours de deux itérations successives sont suffisamment proches. On estime alors que la convergence est atteinte, même si aucune preuve formelle ne peut être apportée pour justifier cette convergence. Les paramètres ainsi obtenus sont considérés comme de bonnes estimations de la solution (supposée unique) du système. A nouveau, à notre connaissance, aucune preuve de l'unicité de la solution n'a été fournie à ce jour.

4.2.1 Principes communs

Les différentes techniques de résolution utilisent toutes une propriété essentielle de la ligne fermée. Lorsqu'on propose d'éliminer une équation à l'ensemble du système qu'on cherche à résoudre, il y a plusieurs façons de le faire. L'une d'elles est de fixer un des paramètres $I_u(i,i+1)$, donc une des équations (4.1a) (ou $I_d(i,i+1)$, soir une des équations (4.2a)). Nous sommes alors devant un système de $4K-1$ équations à $4K-1$ inconnues. À partir du moment où, pour i_0 donné, le paramètre $I_u(i_0,i_0+1)$ est fixé et connu, le système a une solution unique.

Remarque 5 : par la suite, on raisonnera en ayant choisi pour paramètre fixé $I_u(1,2)$. Le choix de l'indice est totalement arbitraire mais sans conséquence, étant donné la topologie en boucle. De plus, le raisonnement est similaire si on choisit de fixer $I_d(1,2)$.

Naturellement, comme on n'a pas pris en compte la contrainte de population (4.4), il n'y a aucune raison pour que ce système donne la solution recherchée. Cependant, nous avons un paramètre libre ($I_u(1,2)$). On note $Q(I_u(1,2))$ la fonction qui renvoie la somme des en-cours moyens des dipôles en fonction de la valeur du paramètre fixé $I_u(1,2)$. Cette fonction est parfaitement définie et calculable (approximativement), en appliquant la méthode de l'algorithme DDX classique (voir Section 3.2). La résolution du problème consiste maintenant à trouver $I_u(i_0,i_0+1)$ tel que $Q(I_u(i_0,i_0+1)) = N$. Cela est assez simple, et on se base pour ce faire sur les propriétés suivantes de cette fonction :

Pour tout i_0 fixé,
- $Q(I_u(i,i+1))$ est une fonction décroissante de $I_u(i_0,i_0+1)$;
- $Q(I_d(i,i+1))$ est une fonction croissante de $I_d(i_0,i_0+1)$.

Cette propriété n'a jamais été démontrée formellement, mais s'avère toujours vraie expérimentalement.

4.2.2 La méthode originale

La méthode qui est présentée ici fut à l'origine proposée par [20]. Il s'agit d'une dichotomie sur le paramètre $I_u(1,2)$. Pour chaque itération de la dichotomie, on fixe $I_u(1,2)$. Cela revient donc à éliminer l'une des équations (4.1a) du système d'équations, afin de retrouver un système bien défini.

L'algorithme est donc constitué de deux boucles d'itérations imbriquées. La boucle interne est simplement une instance (à peine modifiée) de l'algorithme DDX standard, tel que présenté Section 3.2. L'exécution de cet algorithme donne une valeur Q (en fonction du paramètre fixé $I_u(1,2)$). La boucle externe fait alors une dichotomie sur Q. La boucle interne est référencée et donnée dans l'algorithme 4.1.

Algorithme 4.1 FCD_I : boucle d'itérations interne

Requiert : $I_u(1,2)$
1: **Initialiser :** $\lambda_u(i,i+1)$, $\lambda_d(i,i+1)$, $\mu_u(i,i+1)$ and $\mu_d(i,i+1)$ aux valeurs correspondantes de la ligne L
2: **Initialiser :** $\lambda_u(1,2) = \mu_u(1,2)_u(1,2)$
3: **répéter**
4: **pour** $i = 1$ à K **faire**
5: évaluer les performances de $D(i,i+1)$
6: **si** $i \neq K$ **alors**
7: calculer $I_u(i+1,i+2)$ à partir de ((4.1a))
8: **fin si**
9: mettre à jour les valeurs de $\lambda_u(i+1,i+2)$ et $\mu_u(i+1,i+2)$ à partir de ((4.1b)), ((4.1c)) et ((4.1d))
10: **fin pour**
11: **pour** $i = K$ à 1 **faire**
12: évaluer les performances de $D(i,i+1)$
13: calculer $I_d(i-1,i)$ depuis ((4.2a))
14: mettre à jour les paramètres $\lambda_d(i-1,i)$ et $\mu_d(i-1,i)$ à partir de ((4.2b)), ((4.2c)) et ((4.2d))
15: **fin pour**
16: **jusqu'à** convergence
17: $Q(I_u(1,2)) = \sum_{i=1}^{K} hm(i,i+1)$

La boucle externe utilise la monotonie de la fonction $Q(I_u(1,2))$ pour faire une dichotomie sur l'intervalle $[I_u(1,2)_{inf}, I_u(1,2)_{sup}]$ tel que, à chaque itération $Q(I_u(1,2)_{inf}) \geq N$ et $Q(I_u(1,2)_{sup}) \leq N$. La boucle principale (externe) est donnée dans l'algorithme 4.2 et sera appelé par la suite "**FCD_I**".

Cette méthode est simple, mais souffre de plusieurs inconvénients. Premièrement, il est parfois impossible de trouver un couple de valeurs $I_u(1,2)_{inf}$ et $I_u(1,2)_{sup}$ telles que les valeurs correspondantes $Q(I_u(1,2))$ forment un intervalle encadrant N (algorithme 4.2, lignes 2 à 14). Et cela même avec des paramètres de ligne parfaitement usuels. Deuxièmement, l'utilisation de deux boucles d'itérations imbriquées est assez inefficace. Enfin, le choix de la machine utilisée pour faire la dichotomie est très sensible sur la robustesse de l'algorithme. Nous avons été plusieurs fois confrontés à des cas (correspondant à des situations réelles) où un mauvais choix de la machine servant pour la dichotomie rend l'algorithme non-convergent. Il ne semble pas y avoir d'heuristique simple pour choisir cette machine. Enfin, cette méthode n'est pas simplement

Algorithme 4.2 FCD_I : boucle externe

Requiert : N, ϵ

Initialiser : $I_u(1,2)$ à une valeur assez petite
répéter
  exécuter la boucle interne (algorithme 4.1)
  si $Q(I_u(1,2)) < N$ **alors**
    choisir une valeur de $I_u(1,2)$ plus petite (mais toujours positive)
  fin si
jusqu'à $Q(I_u(1,2)) > N$
répéter
  $I_u(1,2)_{inf} = I_u(1,2)$ et $I_u(1,2)_{sup} = 2.I_u(1,2)$
  exécuter la boucle interne (algorithme 4.1) avec $I_u(1,2) = I_u(1,2)_{sup}$
  si $Q(I_u(1,2)) > N$ **alors**
    $I_u(1,2)_{inf} = I_u(1,2)_{sup}$ et $I_u(1,2)_{sup} = 2.I_u(1,2)_{sup}$
  fin si
jusqu'à $Q(I_u(1,2)) < N$
répéter
  exécuter la boucle interne avec $I_u(1,2) = \frac{I_u(1,2)_{inf} + I_u(1,2)_{sup}}{2}$ (algorithme 4.1)
  si $Q < N$ **alors**
    $I_u(1,2)_{sup} = I_u(1,2)$
  sinon
    $I_u(1,2)_{inf} = I_u(1,2)$
  fin si
jusqu'à $Q(I_u(1,2))$ est assez proche de N (à ϵ près)

adaptable aux cas de lignes fermées de topologie générale.

4.2.3 Amélioration de la méthode

Une amélioration de la méthode décrite ci-dessus à été proposée par H. Le Bihan, mais n'a jamais été publiée. L'idée est essentiellement, la même, sauf que la dichotomie est remplacée par une technique où un facteur correctif est appliqué au paramètre $I_u(1,2)$ fixé, dont le but est de contraindre la solution vers la valeur N. La correction appliquée à chaque itération n de la boucle externe est :

$$I_u^{(n)}(1,2) = \left(\frac{Q(I_u^{(n-1)}(1,2))}{N} \right)^{1/p} I_u^{(n-1)}(1,2) \tag{4.6a}$$

Le facteur correctif augmente $I_u(1,2)$ si la dernière valeur de Q est plus grande que N, et le diminue sinon. La boucle interne est donc exactement la même que précédemment. La boucle externe est maintenant donnée dans l'algorithme 4.3 et sera notée "**FCD_M**". Cette méthode utilise un facteur p qui ralentit la convergence pour éviter de voir apparaître des phénomènes oscillatoires (donc il ralentit la convergence quand une nouvelle itération donne une valeur de Q plus éloignée de N que la précédente).

Remarque 6 : cette méthode consiste en fait à faire localement l'approximation de la courbe de la fonction $Q = f(I_u(1,2))$ par une fonction du type $Q = k\,x^{-p}$.

Algorithme 4.3 FCD_M : boucle externe

Requiert : N, ϵ
1: **Initialiser :** $I_u(1,2)$
2: **Initialiser :** p à 1
3: **Initialiser :** $iter$ à 0
4: **répéter**
5: exécuter la boucle interne (algorithme 4.1)
6: **si** $iter > 1$ et $\frac{|Q(I_u(1,2)-N|}{max(Q(I_u(1,2),N)} > \frac{|Q_{old}-N|}{max(Q_{old},N)}$ **alors**
7: $p = p + 1$
8: **fin si**
9: $I_u(1,2) = \left(\frac{Q(I_u(1,2)}{N}\right)^{1/p} I_u(1,2)$
10: $iter = iter + 1$
11: **jusqu'à** $Q(I_u(1,2)$ est assez proche de N (à ϵ près)

Cette méthode est en pratique plus efficace que la technique initiale (Section 4.2.2), mais n'élimine en réalité aucun des défauts de cette dernière. En effet, elle oblige toujours à choisir une machine de la ligne comme "pivot" de la méthode de résolution par approximations successives. De plus, est reste sujette a quelques instabilités numériques et autres problèmes de convergence.

4.2.4 Les fonctions $Q = f(I)$

Les problèmes principaux des deux méthodes évoquées ci-dessus proviennent en fait de la façon dont le paramètre (I_u ou I_d) qui dirige la convergence de la boucle externe est lié à Q (donc

TAB. 4.1 – Paramètres utilisés pour l'étude des courbes $Q = f(I_u)$.

i	1	2	3	4	5	6	7	8	9	10
λ_i	0.02	0.07	0.02	0.1	0.5	0.005	0.07	0.03	0.02	0.001
μ_i	0.1	0.4	0.9	1.5	0.1	0.01	0.4	0.4	0.3	0.5
$C_{i,i+1}$	20	10	8	25	15	10	30	35	40	50

à la forme des relations $Q = f(I_u(i, i+1))$. Pour illustrer cela, on montre à la Figure 4.3 page suivante les tracés des courbes $Q = f(I_u(i, i+1))$ pour $i = 1, \ldots, K$ obtenus pour une ligne de 10 machines dont les paramètres sont donnés Table 4.1.

Remarque 7 : les courbes $Q = f(I_d(i, i+1))$ sont très semblables aux courbes $Q = f(I_u(i, i+1))$ présentées dans Figure 4.3 page suivante, à l'exception du fait que ce sont des fonctions croissantes.

On observe que chacune de ces courbes est une succession de portions de courbes "quasi-verticales" et "quasi-horizontales". Un algorithme cherchant (aussi bien une dichotomie simple, telle que l'algorithme **FCD_I**, qu'une méthode comme l'algorithme **FCD_M**) à résoudre l'équation $Q(I_u) = N$ devra donc opérer sur une portion de courbe assez "plate"de manière à être efficace et robuste. Si la valeur de N recherchée correspond à une partie "verticales" de la courbe choisie (c'est-à-dire que l'on a $\left| \frac{dQ(I_u)}{dI_u} \Big|_{I_u \mid Q(I_u)=N} \right| >> 1$), une très faible variation de I_u entraînera une forte variation de Q, rendant la convergence difficile, car induisant des oscillations et des instabilités numériques. On comprend maintenant mieux pourquoi les algorithmes **FCD_I** et **FCD_M** rencontrent parfois des problèmes de convergence, étant donné qu'ils choisissent toutes deux arbitrairement une machine, donc l'une des courbes. Et selon la courbe choisie et la valeur de N, on peut tout aussi bien se trouver au voisinage d'une portion favorable à la convergence de la méthode que le contraire. Malheureusement, il n'y a pas de méthode simple (heuristique) pour déterminer quelle sera la bonne machine à choisir a priori pour être sûr de converger efficacement, en fonction de la valeur de N.

4.2.5 Nouvelle méthode de résolution

Principe de la méthode

L'objectif est à la fois d'éliminer la structure de l'algorithme à deux boucles d'itérations imbriquées, et surtout de trouver une méthode qui ne brise pas la symétrie de la ligne (dans un telle boucle, aucune machine n'a une position spécifique, contrairement aux lignes ouvertes où la position d'une machine dans la ligne à un sens), de façon à obtenir une méthode efficace et robuste. Pour ce faire, on propose de ne plus fixer l'un des paramètres I_u (ou I_d), mais de forcer la convergence vers la solution cherchée N, en utilisant une contrainte qui s'applique à tout un ensemble de paramètres. On transforme donc le système d'équations pour ajouter cette contrainte. En fait, nous ne modifions pas le système d'équations que l'on cherche à résoudre proprement dit, mais la façon dont les différentes équations sont utilisées dans la résolution du problème. Les deux systèmes ont exactement la même solution, car ils sont bien le même système, mais présentés différemment.

On propose de modifier les relations (4.1a) et (4.2a) en :

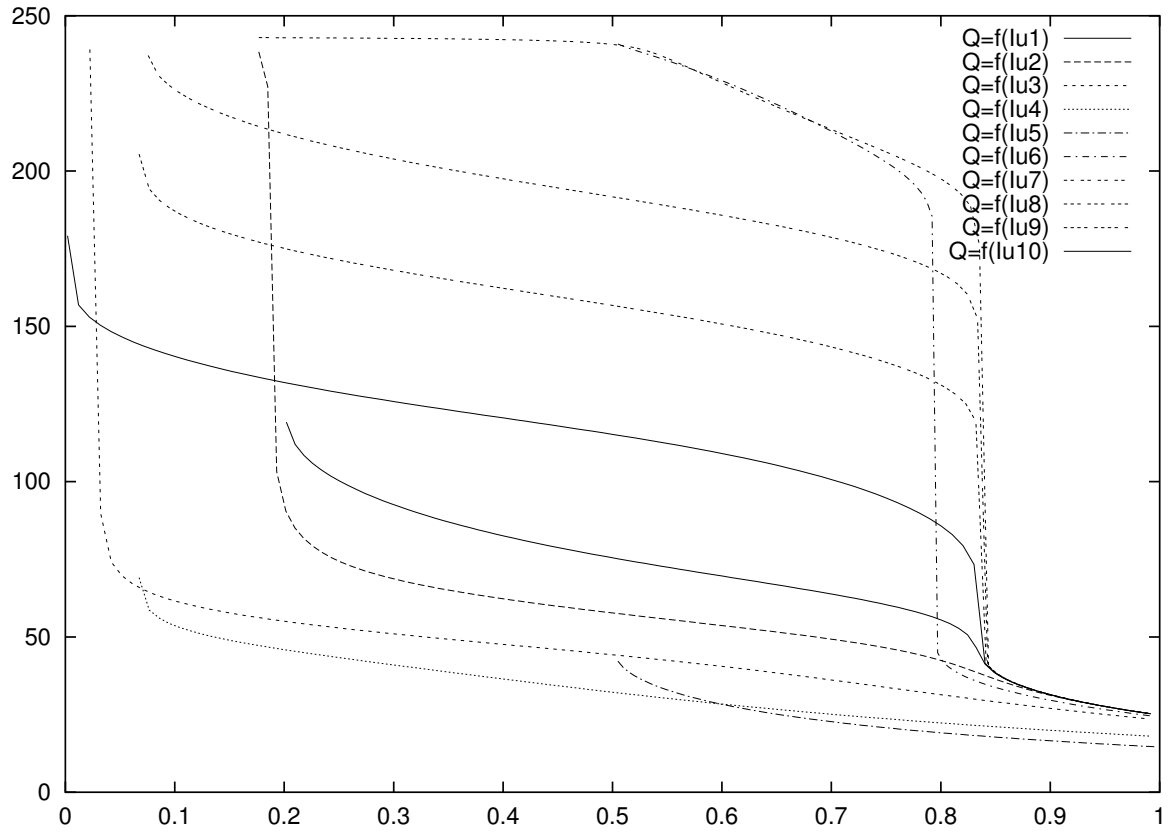

FIG. 4.3 – Courbes $Q = f(I_u)$ pour chaque machine de la ligne dont les paramètres sont donnés en Table 4.1 page précédente.

$$I_u(i+1,i+2) = \beta_u \left(\frac{1}{pw(i,i+1)} + I_{i+1} - I_d(i,i+1) - 1 \right) \tag{4.7a}$$

$$I_d(i-1,i) = \beta_d \left(\frac{1}{pw(i,i+1)} + I_i - I_u(i,i+1) - 1 \right) \tag{4.7b}$$

Il suffit par la suite de rendre les paramètres ajoutés (β_u et β_d) dépendants du rapport Q/N dans le processus de convergence de façon à ce qu'a convergence, ces deux paramètres soient égaux à 1, auquel cas, nous retrouvons exactement le système initial. Comme pour le cas de l'algorithme **FCD_M**, on cherche, par ces paramètres correctifs, à surévaluer I_u (et sous-évaluer I_d) lorsque $Q > N$, attendu que Q est une fonction décroissante de I_u (et une fonction croissante de I_d). De même, quand $Q < N$, on fait l'inverse. Contrairement aux deux méthodes présentée précédemment, nous ajustons tous les paramètres $I_u(i,i+1)$ et $I_d(i,i+1)$ en même temps, sans en choisir un particulièrement. On élimine de cette façon au moins l'une des faiblesses des méthodes existantes, à savoir la "désymétrisation" de la ligne (il n'est plus nécessaire de choisir une machine arbitrairement). On applique un facteur correctif similaire à celui de la méthode **FCD_M** :

$$\beta_u = \left(\frac{Q}{N} \right)^{1/p} \tag{4.8a}$$

$$\beta_d = \left(\frac{N}{Q} \right)^{1/p} \tag{4.8b}$$

Cette idée repose fortement sur l'unicité de la solution et la monotonie des courbes $Q = f(I_u)$ et $Q = g(I_d)$, propriétés non démontrées. Le principe de l'algorithme proposé est alors :

Répéter jusqu'à convergence :

- Pour $i = 1$ à K, faire :

$$\mathcal{P}^{(n)}(i,i+1) = A\left(\mathcal{F}_u^{(n)}(i,i+1),\ \mathcal{F}_d^{(n-1)}(i,i+1) \right) \tag{4.9a}$$

$$\mathcal{F}_u^{(n)}(i+1,i+2) = F\left(\mathcal{P}^{(n)}(i,i+1), \beta_u \right) \tag{4.9b}$$

- Pour $i = K$ à 1, faire :

$$\mathcal{P}'^{(n)}(i,i+1) = A\left(\mathcal{F}_u^{(n)}(i,i+1),\ \mathcal{F}_d^{(n)}(i,i+1) \right) \tag{4.10a}$$

$$\mathcal{F}_d^{(n)}(i-1,i) = G\left(\mathcal{P}'^{(n)}(i,i+1), \beta_d \right) \tag{4.10b}$$

Première version de l'algorithme

Une première version de l'algorithme est donnée algorithme 4.4 et est notée "**DB_I**". Dans cet algorithme, on peut utiliser soit le facteur correctif (21) seul, soit le facteur (22) seul, ou encore les deux à la fois. De plus, dans cette version, ces facteurs ne sont calculés qu'une fois par cycle (parcours complet de la ligne par l'algorithme DDX).

Algorithme 4.4 DB_I

Initialiser : $\lambda_u(i,i+1)$, $\lambda_d(i,i+1)$, $\mu_u(i,i+1)$ et $\mu_d(i,i+1)$ aux valeurs correspondantes de la ligne L
Initialiser : β_u et β_d à 1.0
Initialiser : p à 1
Initialiser : Q_{old} à 0
répéter
  pour $i = 1$ à K **faire**
    évaluer les paramètres de performance du dipôle $D(i,i+1)$
    calculer $I_u(i+1,i+2)$ depuis (4.7a)
    mettre à jour les valeurs des paramètres $\lambda_u(i+1,i+2)$ et $\mu_u(i+1,i+2)$ à partir de (4.1b), (4.1c) et (4.1d)
  fin pour
  pour $i = K$ à 1 **faire**
    évaluer les paramètres de performance du dipôle $D(i,i+1)$
    calculer $I_d(i-1,i)$ depuis (4.7b)
    mettre à jour les valeurs des paramètres $\lambda_d(i-1,i)$ et $\mu_d(i-1,i)$ à partir de (4.2b), (4.2c) et (4.2d)
  fin pour
  $Q = \sum_{i=1}^{K} hm(i,i+1)$
  si $|Q-N| > |Q_{old}-N|$ **alors**
    $p = p+1$
  fin si
  $Q_{old} = Q$
  éventuellement : $\beta_u = \left(\frac{Q}{N}\right)^{1/p}$
  éventuellement : $\beta_d = \left(\frac{N}{Q}\right)^{1/p}$
jusqu'à convergence de tous les paramètres

Politique d'incrémentation du paramètre p. L'idée sous-jacente qui justifie ce paramètre p est de réduire la "vitesse de convergence" de façon à éliminer d'éventuels problèmes oscillatoires. En fait, nous avons observé, sur plusieurs exemples, de véritables comportements de "sous-amortissement", donc induisant des phénomènes oscillatoires, détériorant de manière drastique la convergence de l'algorithme. Un tel exemple est donné Figure 4.4 page suivante. Dans ce cas, il s'agit d'une ligne dont les paramètres sont donnés Table 4.2 page ci-contre. Le graphe représente la valeur de Q en fonction de l'itération en cours (conformément au principe décrit ci-dessus, une itération correspond à un parcours aller et retour de l'ensemble des dipôles dans l'ordre de la ligne). Il y a deux courbes : dans la première courbe, il a été fixé à 2, et dans la seconde, il a été laissé "libre", c'est-à-dire susceptible d'être incrémenté en fin d'itération si nécessaire (donc si on a tendance à aller trop vite). La manière dont on détecte que l'on va trop

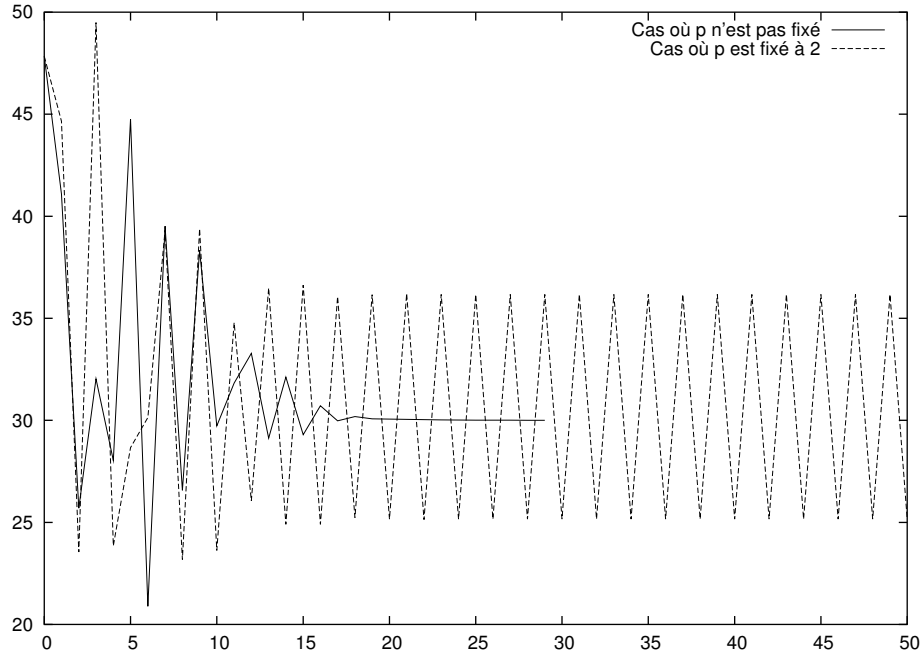

FIG. 4.4 – Étude de la convergence de l'algorithme 4.4 pour différentes politiques d'incrémentation du paramètre p. Les paramètres de la ligne sont donnés tableau Table 4.2.

TAB. 4.2 – Paramètres utilisés pour l'étude de l'impact du paramètre p dans la convergence de l'algorithme 4.4.

i	1	2	3	4	5	6
λ_i	9.07e-3	3.64e-4	1.37e-3	4.56e-5	1.45e-2	8.57e-4
μ_i	1.82e-2	1.55e-3	7.04e-2	1.19e-3	1.42e-1	1.13e-3
$C_{i,i+1}$	20	10	24	15	27	28
i	7	8	9	10	11	
λ_i	5.00e-1	7.63e-4	7.51e-5	1.30e-1	2.65e-3	
μ_i	6.38e-1	4.96e-3	5.59e-3	9.45e-1	2.67e-2	
$C_{i,i+1}$	17	29	22	14	29	

vite dans l'algorithme 4.4 est la plus simple : on vérifie à chaque itération que la nouvelle valeur de Q ne s'est pas éloignée de N par rapport à l'itération précédente, auquel cas on diminue la vitesse de convergence en incrémentant p d'une unité. C'est la technique qui donne les meilleurs résultats concernant le robustesse et l'efficacité de l'algorithme. Il reste cependant des cas où on ne parvient pas à faire converger l'algorithme.

Remarque 8 : on pourrait être tenté de décrémenter p si l'on parvenait à détecter que le processus de convergence est trop "lent". En pratique, il n'est pas trivial de détecter que l'algorithme converge trop lentement. Les essais que nous avons menés se sont presque toujours soldés par des échecs, à cause de phénomènes périodiques induisant des oscillations.

Cette méthode améliore effectivement l'efficacité et la robustesse de la convergence par rapport aux méthodes évoquées plus haut (sections Section 4.2.2 et Section 4.2.3), mais elle ne résout pas tous les problèmes évoqués : premièrement, il est des cas (certes souvent peu réalistes) où l'algorithme ne converge pas, ou mal. De plus, il reste des choix (arbitraires) à faire pour appliquer la méthode, à savoir l'application du facteur correctif : sur le paramètre I des machines $M_u(i, i+1)$ des dipôles (donc utiliser β_u seul ; on notera ce choix "Up"), sur le paramètre I des machines $M_d(i, i+1)$ des dipôles (donc utiliser β_d seul ; on notera ce choix "Down"), ou bien encore l'application des deux coefficients (soit le choix "Both"). En pratique, ce choix peut apparaître crucial sur l'efficacité de l'algorithme. En général, c'est le choix "Up" qui est le meilleur (mais ce n'est pas systématiquement le cas), ce qui peut s'expliquer intuitivement. Lorsque β_u et β_d sont calculés, ils correspondent à une certaine configuration des dipôles, et donc à une valeur de Q correspondant à cette configuration. Quand un dipôle est mis à jour, et par la suite réévalué, la configuration n'est plus la même, et la valeur de Q diffère quelque peu. Ainsi, au fur et à mesure que les dipôles sont analysés au cours d'une itération, et que les dipôles sont mis à jour (selon que l'on est dans la phase i croissant , on met à jour les paramètres λ_u et μ_u, et dans la phase i décroissant, on met à jour les paramètres λ_d et μ_d), on s'éloigne un peu plus de la configuration du début de l'itération. La valeur courante de Q est de plus en plus fausse par rapport à la configuration en cours, donc les valeurs des paramètres β_u et β_d sont de plus en plus "fausses" par rapport à la configuration en cours. Ainsi, comme le paramètre β_u est utilisé avant β_d au cours d'une itération de l'algorithme, il est "meilleur" que ce dernier, ceci expliquant pourquoi c'est en général le choix "Up" qui est plus efficace.

Version finale de l'algorithme

On constate donc, d'après les réflexions précédentes, qu'il serait souhaitable de faire l'évaluation de Q, donc de β_u et β_d à chaque fois qu'un dipôle est modifié. On propose donc une autre version de l'algorithme où cette idée est mise en pratique. Nous nommerons **DB_M** cette version, qui est détaillée dans l'algorithme 4.5.

Afin de mettre à jour la valeur de Q à chaque étape (lignes 12 et 21 de l'algorithme 4.5), il suffit de "corriger" sa valeur en soustrayant la valeur de l'encours moyen du dipôle juste avant de l'analyser, puis d'ajouter cette même valeur après la réévaluation du dipôle. On refera cependant le calcul de Q à chaque itération pour éviter des problèmes liés à la précision numérique (propagation d'erreurs) de la machine (étape 25).

4.3 Résultats numériques : efficacité et robustesse

On cherche à déterminer les performances de la technique proposée (**DB_M**) par rapport aux anciennes méthodes. On comparera donc uniquement les performances "informatiques" de

Algorithme 4.5 DB_M

Initialiser : $\lambda_u(i,i+1)$, $\lambda_d(i,i+1)$, $\mu_u(i,i+1)$ et $\mu_d(i,i+1)$ aux valeurs correspondantes des machines de la ligne L
Initialiser : β_u et β_d à 1.0
Initialiser : p à 1
Initialiser : Q_{old} à 0
Initialiser : $iter$ à 0
répéter
  pour $i = 1$ à K **faire**
    évaluer les paramètres performances du dipôle $D(i,i+1)$
    calculer $I_u(i+1,i+2)$ à partir de (4.7a)
    mettre à jour les valeurs de $\lambda_u(i+1,i+2)$ et $\mu_u(i+1,i+2)$ depuis (4.1b), (4.1c)) et (4.1d)
    si $iter > 0$ **alors**
      mettre à jour Q
      $\beta_u = \left(\frac{Q}{N}\right)^{1/p}$
    fin si
  fin pour
  pour $i = K$ à 1 **faire**
    évaluer les paramètres de performance du dipôle $D(i,i+1)$
    calculer $I_d(i-1,i)$ à partir de (4.7b)
    mettre à jour les valeurs de $\lambda_d(i-1,i)$ et $\mu_d(i-1,i)$ depuis (4.2b), (4.2c) et (4.2d).
    si $iter > 0$ **alors**
      mettre à jour Q
      $\beta_d = \left(\frac{N}{Q}\right)^{1/p}$
    fin si
  fin pour
  $Q = \sum_{i=1}^{K} hm(i,i+1)$
  $iter = iter + 1$
  si $|Q - N| > |Q_{old} - N|$ **alors**
    $p = p + 1$
  fin si
  $Q_{old} = Q$
  $\beta_u = \left(\frac{Q}{N}\right)^{1/p}$
  $\beta_d = \left(\frac{N}{Q}\right)^{1/p}$
jusqu'à convergence de tous les paramètres

ces méthodes, à savoir la robustesse de l'algorithme et son efficacité.

Remarque 9 : on rappelle que ces différents algorithmes, dans la mesure où ils convergent, résolvent le même système d'équations, et donc évaluent les mêmes paramètres de performance.

Remarque 10 : les résultats concernant les performances de l'algorithme **DB_I** ne sont pas exposés ici car le seconde version de l'algorithme, **DB_M** est bien meilleure dans tous les cas (en robustesse comme en vitesse d'exécution).

4.3.1 Exemples générés

La vitesse de convergence est évaluée en calculant le nombre d'analyses de dipôles nécessaires pour atteindre la convergence pour les différentes techniques. Les algorithmes ne convergent pas toujours, en raison de problèmes numériques évoqués Section 4.2.4. Nous avons donc limité artificiellement ce nombre d'analyses en isolation de dipôles à 20000. Cette limite apparaît clairement sur plusieurs graphes, signifiant donc que dans ces cas, l'algorithme n'a pas convergé (en un temps raisonnable), et que les résultats obtenus n'ont donc pas de sens. Remarquons qu'en utilisant cette valeur limite du nombre d'itérations, on classe comme "non-convergeant" deux types de lignes : les lignes pour lesquelles l'algorithme est effectivement "non-convergeant", en général à cause de phénomènes oscillatoires, et les lignes pour lesquelles l'algorithme étudié pourrait converger si on le laissait travailler suffisamment longtemps. Nous proposons de mesurer la robustesse de chaque méthode comme le pourcentage de fois que cette méthode ne parvient pas à convergence (en moins de 20000 analyses) par rapport au nombre total de tests effectués. Cette valeur est obtenue sur des lignes dont les paramètres sont déterminés aléatoirement.

Exemples symétriques

On teste ici des lignes totalement symétriques (c'est-à-dire dont tous les paramètres sont identiques). Notre expérience à montré que ce sont les situations les plus favorables pour tous les algorithmes. Toutes les machines ont les mêmes paramètres de panne et de réparation : $\lambda_i = 0.01$ et $\mu_i = 0.1$ pour toute valeur de i. De même, tous les stocks ont même capacité. Nous avons testé les cas où $C_{i,i+1} = 2$ et $C_{i,i+1} = 10$. Nous avons fait des tests sur des lignes de dimensions $K = 3$ et $K = 10$. Les résultats sont résumés par la Figure 4.5 page suivante. Chacun des quatre graphes à la Figure 4.5 page ci-contre correspond à un couple (K, C), où C est la capacité de chacun des stocks de la ligne. L'axe des x est le nombre N de clients dans la ligne. Ce nombre varie de 1 à $C_{tot} - 1$, où C_{tot} est la capacité totale de la ligne ($C_{tot} = C \times K$). L'axe des y représente le nombre n'analyses en isolation de dipôles, pour les différentes techniques.

On note en premier les piètres performances de l'algorithme original **FCD_I**, puisque ce dernier nécessite entre 5000 et 15000 analyses de dipôles pour converger, mais parfois ne converge pas du tout. La version modifiée de ce dernier, **FCD_M**, donne de bien meilleurs résultats, principalement pour les zones centrales du spectre des valeurs de N (qui sont en général les valeurs utiles, puisque au voisinage de la zone maximisant le taux de production de la ligne). Finalement, on note que la technique proposée **DB_M** converge toujours rapidement vers la solution, souvent 10 à 100 fois plus vite que l'algorithme **FCD_I**, et de 2 à 5 fois plus vite que la version modifiée **FCD_M**.

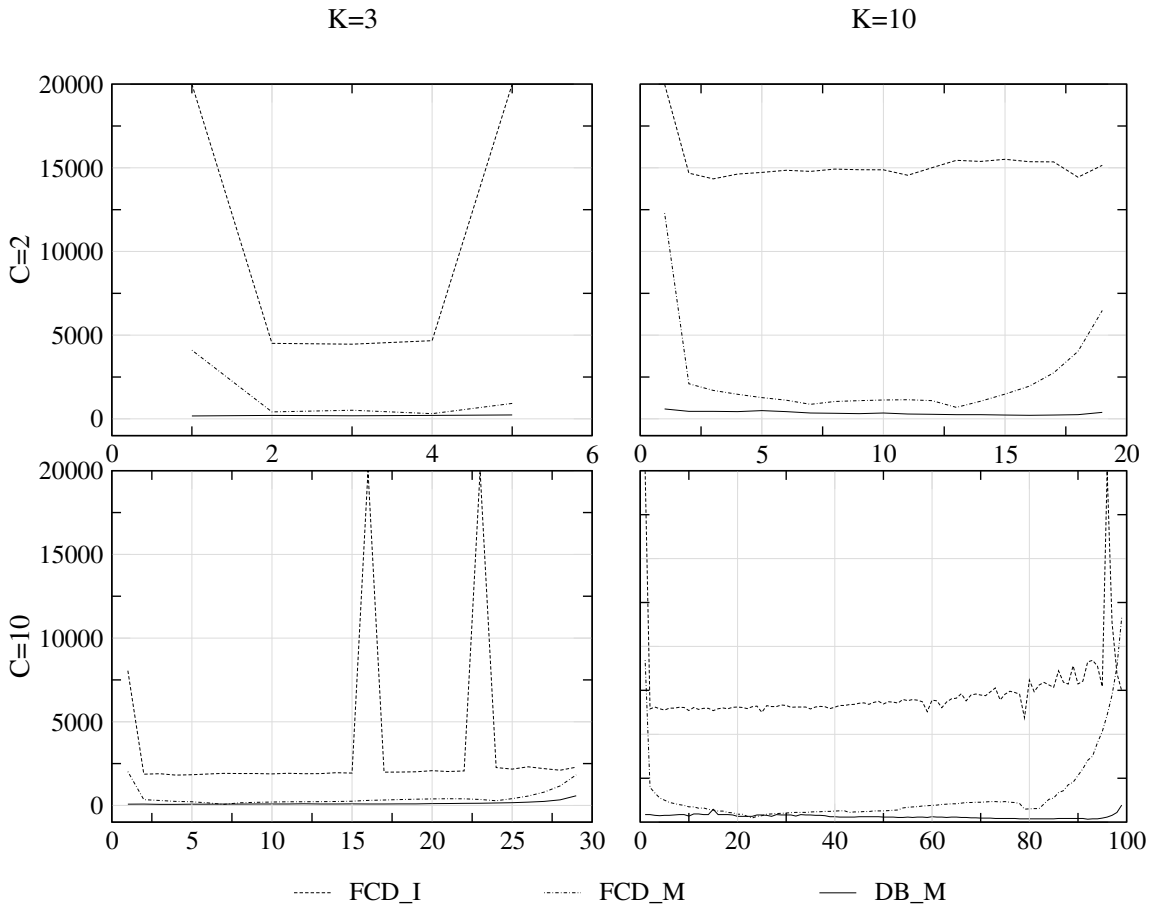

FIG. 4.5 – Exemple 1 : lignes totalement symétriques.

TAB. 4.3 – Exemple 2.

i	1	2	3	4	5
λ_i	0.018	0.010	0.055	1.6e-3	0.017
μ_i	0.220	0.250	0.590	0.087	0.350
I_i	0.081	0.041	0.092	0.018	0.048
$C_{i,i+1}$	2	12	16	23	24

TAB. 4.4 – Exemple 3.

i	1	2	3	4	5	6	7	8	9	10
λ_i	0.016	0.074	0.086	0.081	0.020	1.3e-3	0.063	8.3e-3	0.089	0.093
μ_i	0.230	0.840	0.920	0.810	0.420	0.026	0.880	0.750	0.990	1.000
I_i	0.070	0.088	0.094	0.100	0.047	0.048	0.071	0.011	0.090	0.094
$C_{i,i+1}$	24	27	21	12	29	12	25	29	13	9

Exemples aléatoires

Pour mesurer la robustesse et le comportement général des différentes techniques, on étudie des exemples dont les paramètres ont été générés aléatoirement. Tous les paramètres sont tirés selon des distributions uniformes sur un intervalle donné. Pour chaque machine M_i de la ligne, on tire donc les paramètres λ_i et I_i (on en déduit μ_i) ainsi que pour chaque buffer $B_{i,i+1}$, on tire sa capacité $C_{i,i+1}$.

Nous avons étudiés ainsi deux types de lignes correspondant à des variabilités différentes des paramètres. Les premiers sont référencés comme "lignes à faible variabilité", tandis que les seconds sont nommés "lignes à haute variabilité". Pour chacune de ces deux types de lignes, nous avons fait 2000 tirages aléatoires. Un premier ensemble de 1000 lignes correspond à des lignes de $K = 5$ machines, tandis que le second ensemble de 1000 lignes est constitué de lignes de $K = 10$ machines. Chacun des exemples tirés est testé sur l'ensemble de son spectre du nombre de clients possible, (de $N = 1$ à $N = C_{tot} - 1$). Enfin, on présente deux types de résultats :

- les résultats d'un tirage particulier choisi parmi les 1000 tirages générés, afin de montrer le fonctionnement des différentes techniques sur un exemple,
- la valeur moyenne (robustesse) sur les 1000 tirages, pour donner la robustesse de chacune des techniques présentées.

Lignes à faible variabilité de paramètres. Nous avons tirés les paramètres des lignes en utilisant des distributions uniformes sur les intervalles suivants :

- $\mu_i \in [0.01, 1.0]$
- $I_i \in [0.01, 0.1]$
- $C_{i,i+1} \in \{1, \ldots, 30\}$

Exemple spécifique. Les paramètres de la ligne (choisie parmi les exemples générés) comportant $K = 5$ machines sont donnés dans le Table 4.3. Ce cas est nommé Exemple 2. Les paramètres correspondant à la ligne de $K = 10$ machines (Exemple 3) sont donnés dans le Table 4.4. La capacité totale de la ligne de l'exemple 2 est 77, tandis que celle de l'exemple 3 est 201.

Les résultats numériques pour les exemples 2 et 3 sont résumés aux Figure 4.6 page suivante et Figure 4.7 page 70. On remarque que la différence d'efficacité entre les techniques précédentes (**FCD_I** et **FCD_M**) et la méthode **DB_M** est bien plus importante que pour les cas de

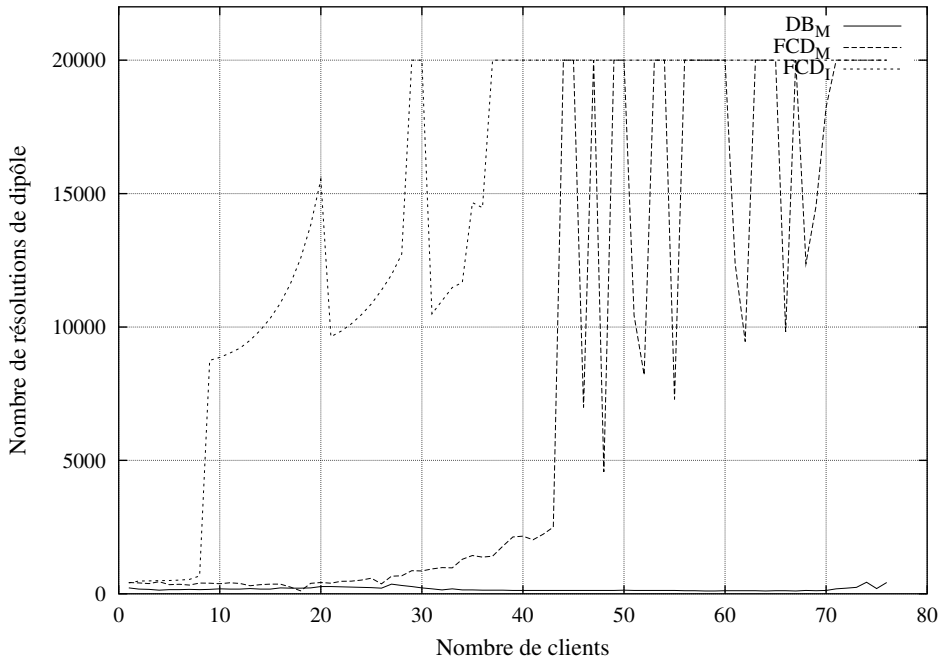

FIG. 4.6 – "Vitesse" de convergence pour l'exemple 2 (Table 4.3 page précédente).

lignes symétriques (Exemple 1). De plus, les méthodes **FCD_I** et **FCD_M** faillissent bien plus souvent (cas visibles sur les graphiques par un nombre d'analyses de dipôles atteignant 20000), tandis que la méthode **DB_M** converge souvent en moins de 200 analyses de dipôle. Plus précisément, pour $K = 5$ (Figure 4.6), les méthodes **FCD** ont un comportement très irréguliers, et pour $K = 10$ (Figure 4.7 page suivante), elles ne parviennent pas à converger pour une proportion très importante des valeurs de N possibles, situés au milieu de l'intervalle des valeurs de N possibles.

Résultats statistiques. Les résultats résumant les performances des techniques sur les 2000 exemples aléatoires (1000 cas de lignes de $K = 5$, et 1000 cas de lignes de $K = 10$) sont présentés Figure 4.8 page suivante et Figure 4.9 page 71. L'axe des ordonnées de ces courbes représente maintenant le nombre moyen d'analyses en isolation pour tous les exemples générés. Comme la capacité totale de chaque ligne générée est spécifique à celle-ci, l'axe des abscisses représente la valeur relative de N par rapport à la capacité totale de la ligne (donc le rapport $\frac{N}{C_{tot}}$ exprimé en pourcentage de la capacité totale C_{tot} de chaque ligne). De plus, chaque point des courbes est accompagné d'une valeur numérique (en pourcentage). Cette valeur correspond à la proportion de cas où l'algorithme ne parvient pas à converger, c'est-à-dire qui n'a pas convergé en 20000 analyses en isolation. Il est important de noter que les cas où un algorithme n'a pas convergé en moins de 20000 itérations ne sont pas pris en compte pour l'évaluation du nombre moyen d'itérations de la méthode.

Les conclusions sont très claires, à la fois concernant l'efficacité et la robustesse de la technique proposée :

- efficacité : pour les petites lignes ($K = 5$), si on se concentre sur la partie centrale du spectre des valeurs de N (soit entre 30% et 70% de la capacité totale), la technique **DB_M** est de 4 à 10 fois plus rapide que la technique **FCD_M**, et de 30 à 100 fois plus rapide

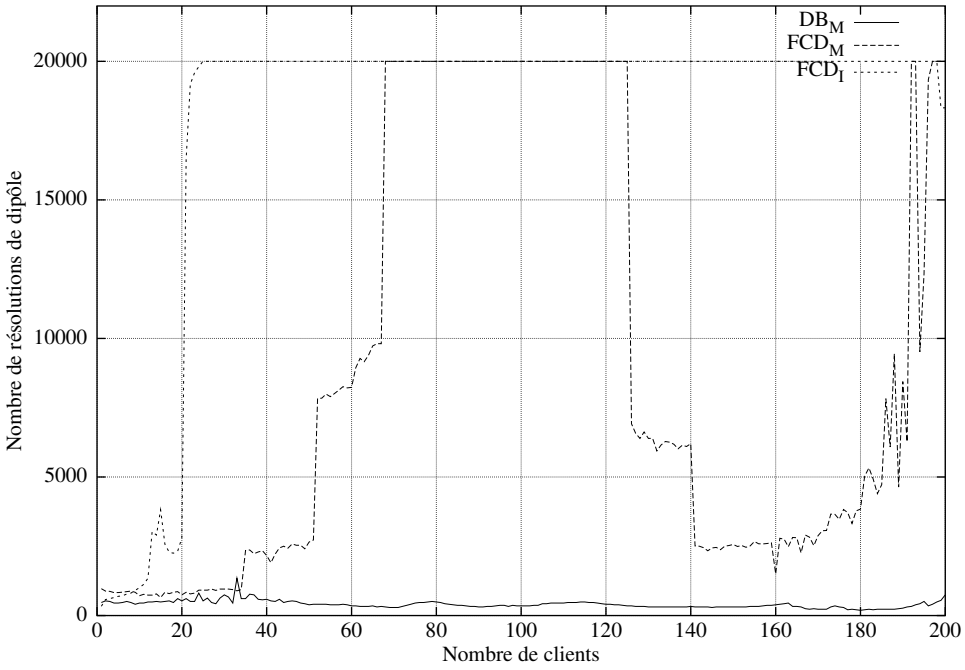

FIG. 4.7 – "Vitesse" de convergence pour l'exemple 3 (Table 4.4 page 68).

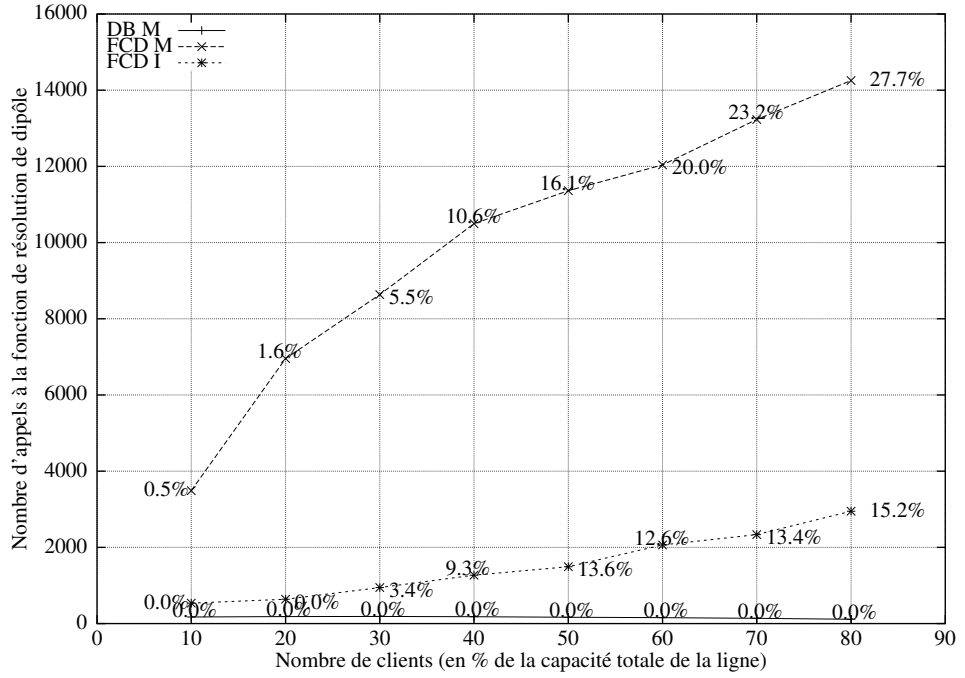

FIG. 4.8 – Statistiques pour les lignes à faible variabilité de paramètres de $K = 5$ machines.

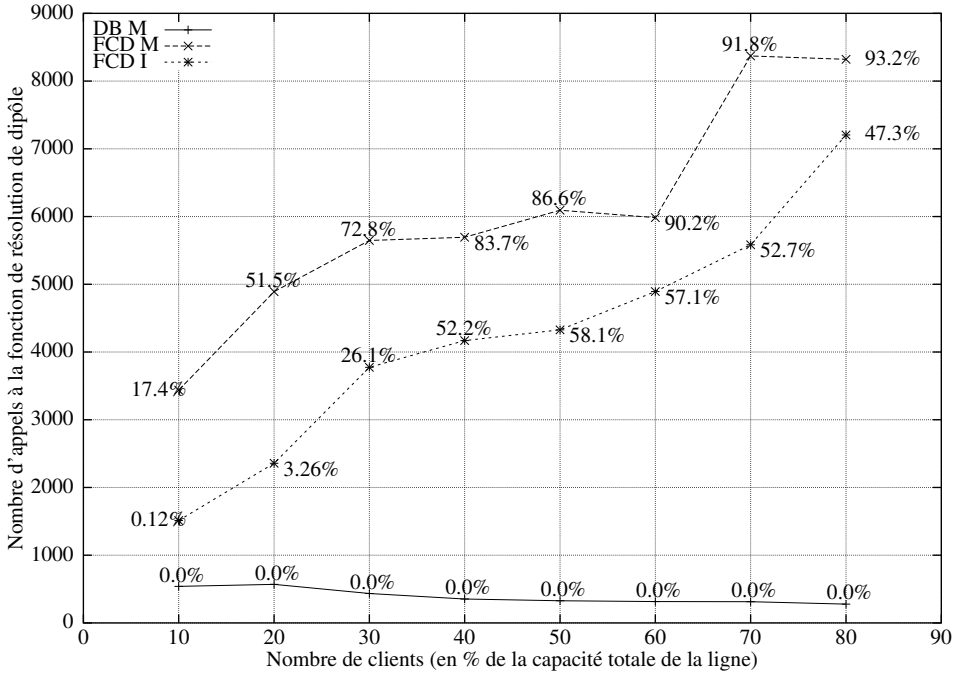

FIG. 4.9 – Statistiques pour les lignes à faible variabilité de paramètres de $K = 10$ machines.

TAB. 4.5 – Exemple 4.

i	1	2	3	4	5
λ_i	1.600	0.930	0.140	8.700	1.400
μ_i	2.500	8.400	6.500	9.800	3.000
I_i	0.640	0.110	0.022	0.890	0.450
$C_{i,i+1}$	27	20	26	1	7

que la technique originale **FCD_I**. Rappelons que l'on ne s'intéresse ici qu'aux cas où les techniques convergent.

- robustesse : sur toutes les lignes testées, la technique **DB_M** n'a pas été mise en défaut une seule fois, tandis que les techniques **FCD** faillissent dans près de 10% des cas pour $K = 5$ (pour $N = 0.5\ C_{tot}$), et dans plus de 50% des cas pour $K = 10$.

Lignes à haute variabilité des paramètres. Les paramètres dans ce cas sont aussi générés par des distributions uniformes, mais avec des intervalles plus grands. Nous avons utilisés les intervalles suivants :

- $\mu_i \in [0.0001, 10.0]$,
- $I_i \in [0.001, 1.0]$,
- $C_{i,i+1} \in \{1, \ldots, 30\}$.

Nous avons tiré 1000 lignes de $K = 5$ machines, et 1000 lignes de $K = 10$ machines.

Une ligne de chaque cas ($K = 5$ et $K = 10$) est présentée Figure 4.10 page suivante et Figure 4.11 page 73. Les paramètres de chacune de ces deux lignes sont donnés Table 4.5 et Table 4.6 page suivante.

TAB. 4.6 – Exemple 5.

i	1	2	3	4	5	6	7	8	9	10
λ_i	2.200	0.049	3.600	0.220	1.100	1.100	0.660	4.900	2.000	1.800
μ_i	8.500	1.700	6.000	1.800	1.500	4.500	1.000	6.300	2.600	2.000
I_i	0.260	0.029	0.600	0.120	0.700	0.230	0.650	0.770	0.790	0.940
$C_{i,i+1}$	29	12	26	24	12	19	13	18	19	23

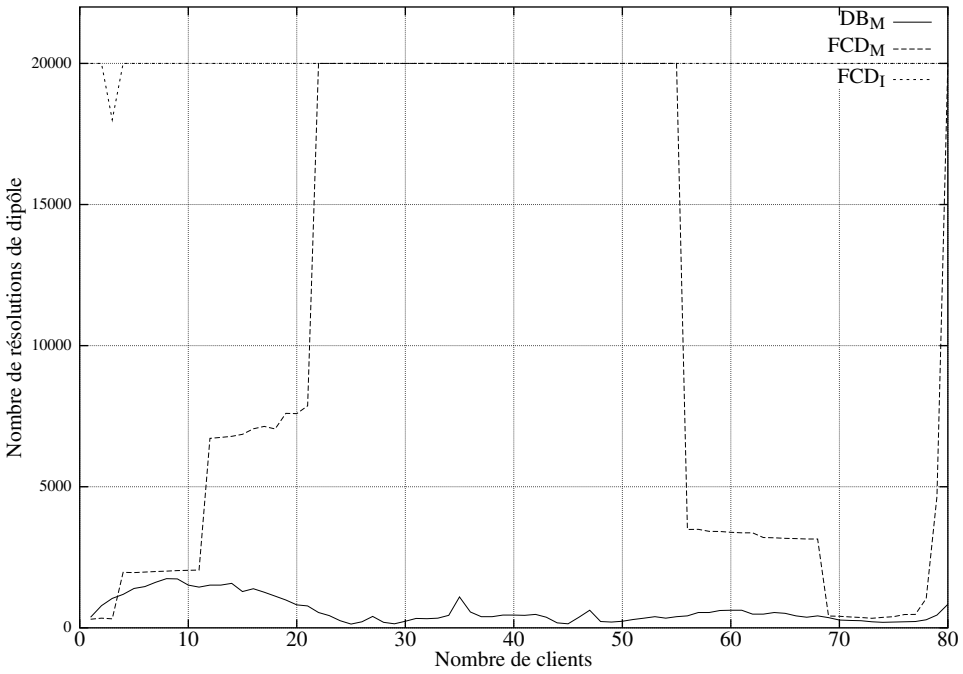

FIG. 4.10 – "Vitesse" de convergence pour l'exemple 4 (Table 4.5 page précédente).

Les résultats confirment les conclusions tirées des cas précédents. Ils montrent, pour des lignes à plus grande variabilité des paramètres, que les techniques **FCD** échouent dans la majorité des cas, tandis que la technique **DB_M** parvient à converger en général en moins de 2000 analyses de dipôles.

Les résultats moyens (robustesse) pour les deux ensembles de 1000 lignes dont les paramètres ont été aléatoirement tirés, sont donnés dans la Figure 4.12 page suivante (lignes de $K = 5$ machines) et Figure 4.13 page 75 (lignes de $K = 10$ machines).

Ces graphes montrent que la technique **DB_M** échoue (ne converge pas) pour moins de 0.4% de ces lignes, tandis que les techniques **FCD** échouent dans 30% à 99% des cas.

4.3.2 Cas réel

On présente maintenant un cas de lignes dont les paramètres proviennent d'une unité de production de l'entreprise de construction automobile française PSA. La ligne de production manufacturière est constituée de 55 postes de travail, et est capable de traiter deux types de produits (lignes multi-produits), les produits "U" et les produits "V". Les paramètres de cette ligne sont donnés Table 4.7 page 74. Les deux premières colonnes donnent le temps de traitement

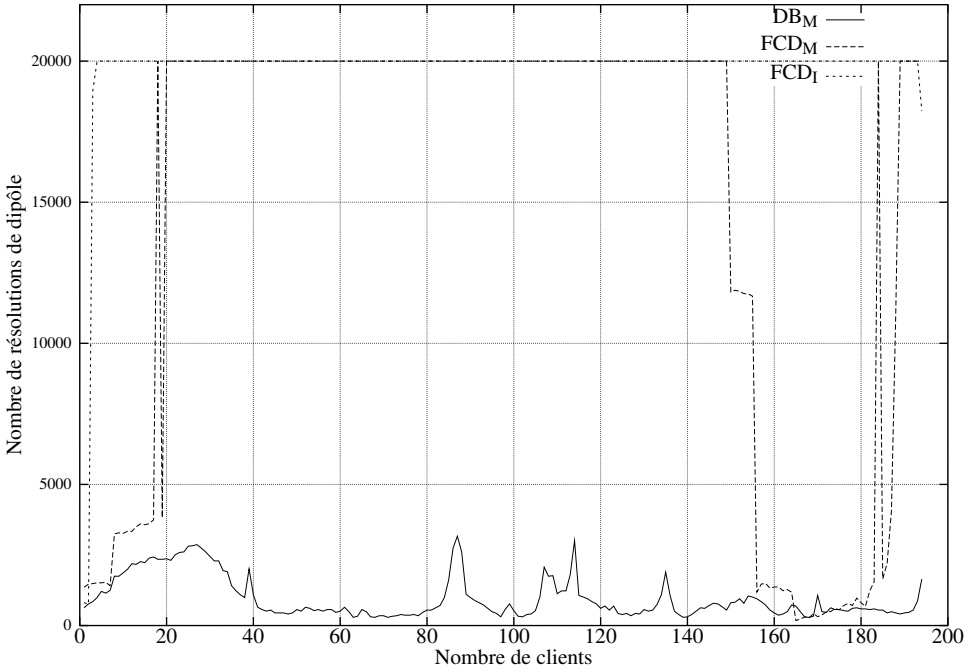

FIG. 4.11 – "Vitesse" de convergence pour l'exemple 5 (Table 4.6 page ci-contre).

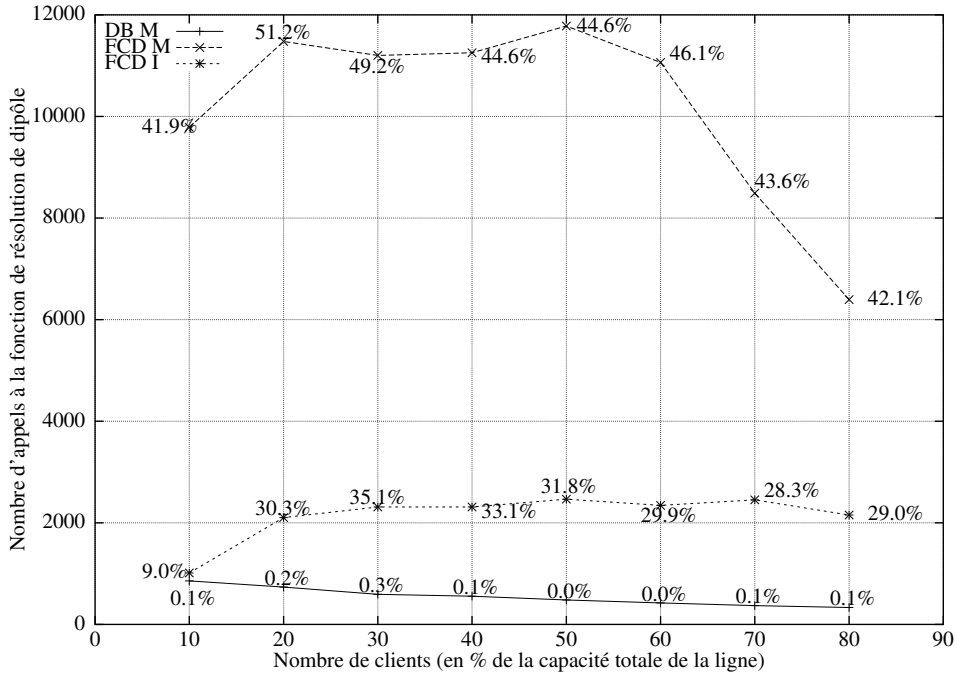

FIG. 4.12 – Robustesse des algorithmes pour des lignes à plus grande variabilité de paramètres comportant $K = 5$ machines.

TAB. 4.7 – Paramètres d'un modèle de ligne de chez PSA.

Post	Tcy U	Tcy V	MTTR	MTBF
1	0.650	0.650	10	1660
2	1.150	1.150	1.500	58.6
3	1.150	1.150	1.500	58.6
4	0	0	0	0
5	1.150	1.150	1.550	59.2
6	1.150	1.150	1.550	59.2
7	0	0	0	0
8	1.150	1.150	1.500	58.6
9	1.150	1.150	1.500	72.4
10	0	0	0	0
11	0.750	1.150	1.500	298
12	1.150	1.150	1.500	108
13	0	0	0	0
14	0.860	1.150	10	495
15	0	0	0	0
16	0	1.150	1.530	36.8
17	0	0	0	0
18	1.250	0	2.080	32.6
19	0	0	0	0
20	0	0	0	0
21	1.150	1.150	1.780	51.8
22	1.150	1.150	1.500	149
23	0	0	0	0
24	0	0	0	0
25	0.175	0.175	10	20000
26	0	0	0	0
27	0.175	0.175	10	20000
28	0	1.150	1.750	70.3

Post	Tcy U	Tcy V	MTTR	MTBF
29	0	0	0	0
30	1.150	0	1.830	81.5
31	0	0	0	0
32	1.150	1.150	1.590	59.7
33	0	0	0	0
34	1.150	1.150	1.630	60.2
35	1.150	1.150	1.590	120
36	0	0	0	0
37	1.150	1.150	1.620	70.3
38	1.150	1.150	1.560	69.3
39	0	0	0	0
40	1.150	1.150	10	20000
41	0	0	0	0
42	0	0	0	0
43	0	0	0	0
44	0.650	0.650	10	1660
45	0	0	0	0
46	0	0	0	0
47	0	0	0	0
48	0	0	0	0
49	0	0	0	0
50	0	0	0	0
51	0	0	0	0
52	0	0	0	0
53	0.175	0.175	10	20000
54	0	0	0	0
55	0.175	0.175	10	20000

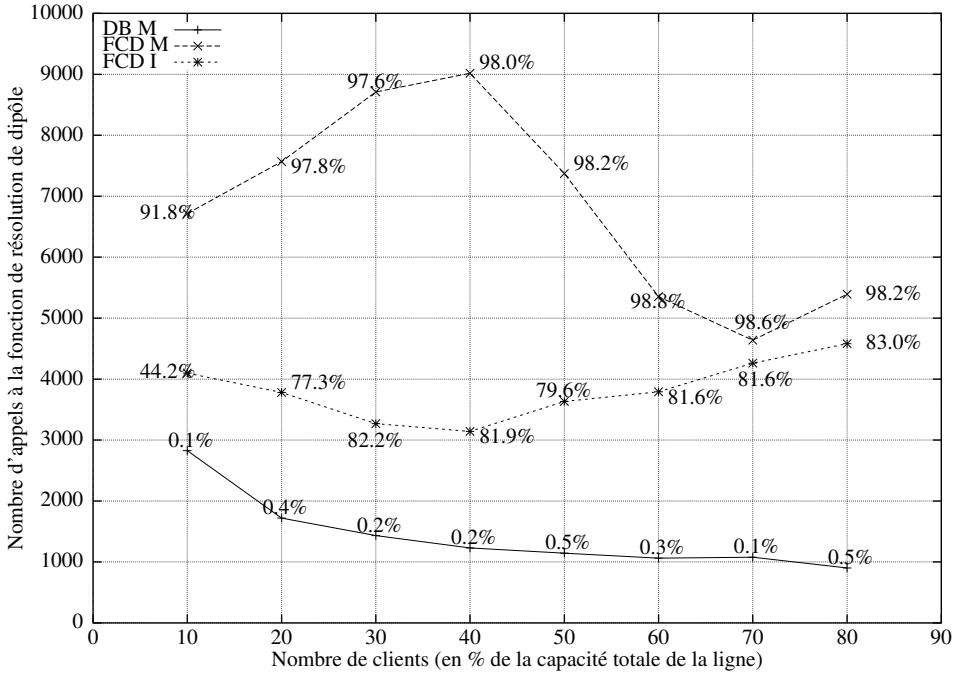

FIG. 4.13 – Robustesse des algorithmes pour des lignes à plus grande variabilité de paramètres comportant $K = 10$ machines.

pour chacun des deux produits. Les troisième et quatrième colonnes donnent le $MTTR$ et $MTTF$ de la machine. Notons qu'un poste n'effectue pas nécessairement un traitement. On le nomme dans ce cas poste mort ; il constitue alors une zone de stockage.

Comme le modèle discret n'est pas homogène, il doit d'abord être transformé en modèle homogène par une technique d'homogénéisation, tel que décrit par [13]. Plusieurs techniques sont disponibles pour faire cette homogénéisation. Nous avons utilisé ici la technique nommée "RSA" décrite dans [27], qui tend à diminuer la vitesse de traitement de toutes les machines à celle de la machine la plus lente, et d'adapter le $MTTF$ en conséquence. Les paramètres résultants de ces deux transformations (tailles des stocks et homogénéisation) pour le modèle continu sont donnés dans le Table 4.8 page suivante.

Tous les résultats présentés ici concernent uniquement le produit "U" (les résultats pour les produits "V" seuls sont très similaires). La figure 4.14 montre la comparaison entre les vitesses de convergence des méthodes **FCD_M** et **DB_M**. La technique originale **FCD_I** n'est pas montrée car elle ne parvient pas à converger pour une majorité des valeurs de N possibles. On remarque sur cet exemple tiré d'un cas réel, que la technique **FCD_M** échoue à converger pour toute valeur de N plus grande que 25 (pour une capacité totale de 55), tandis que la technique **DB_M** converge toujours quasi instantanément. Par exemple, pour $N = 30$, cette dernière technique atteint le point de convergence après 624 analyses de dipôles et prend environ 1.5 ms pour obtenir les paramètres de performance de cette ligne sur un PC cadencé à 800MHz. La Figure 4.15 page 77 montre le taux de production évalué pour ces deux techniques. Lorsque les deux techniques convergent, elles donnent exactement le même résultat, puisqu'elles résolvent le même système.

TAB. 4.8 – Exemple 6 : paramètres du modèle continu pour la ligne de PSA, pour le produit "U".

i	1	2	3	4	5	6	7	8	9
T_i	0.65	1.1	1.1	1.1	1.1	1.1	1.1	0.75	1.1
λ_i	0.1	0.67	0.67	0.65	0.65	0.67	0.67	0.67	0.67
μ_i	0.0006	0.017	0.017	0.017	0.017	0.017	0.014	0.0034	0.0093
$C_{i,i+1}$	1	1	1	2	1	2	1	2	1
i	10	11	12	13	14	15	16	17	18
T_i	0.86	1.2	1.1	1.1	0.17	0.17	1.1	1.1	1.1
λ_i	0.1	0.48	0.56	0.67	0.1	0.1	0.55	0.63	0.61
μ_i	0.002	0.031	0.019	0.0067	5e-05	5e-05	0.012	0.017	0.017
$C_{i,i+1}$	2	4	3	1	3	2	3	2	2
i	19	20	21	22	23	24	25		
T_i	1.1	1.1	1.1	1.1	0.65	0.17	0.17		
λ_i	0.63	0.62	0.64	0.1	0.1	0.1	0.1		
μ_i	0.0083	0.014	0.014	5e-05	0.0006	5e-05	5e-05		
$C_{i,i+1}$	1	2	1	2	4	9	2		

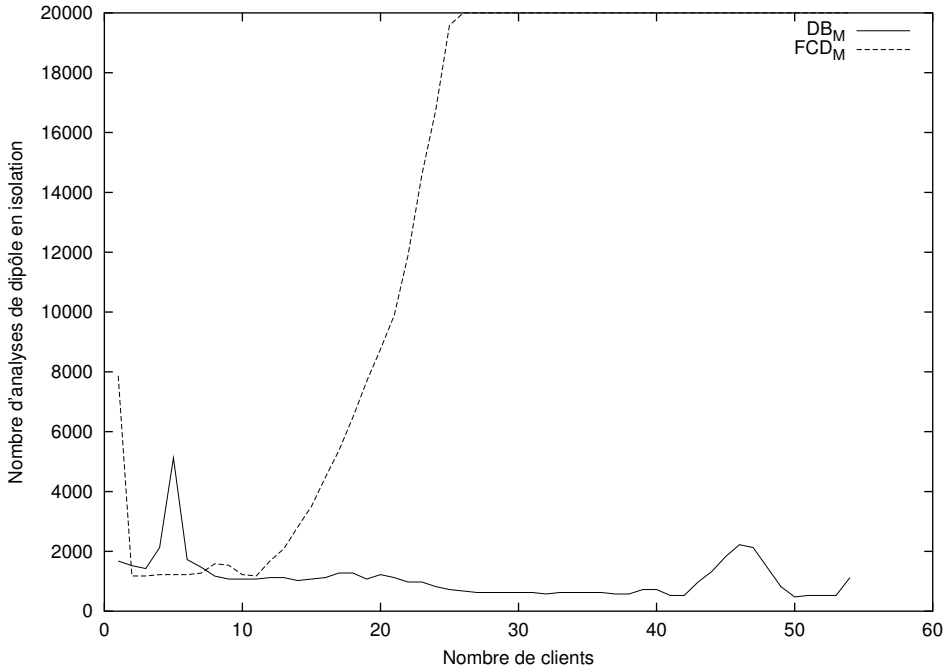

FIG. 4.14 – "Vitesse" de divergence pour l'exemple 6 (Table 4.8).

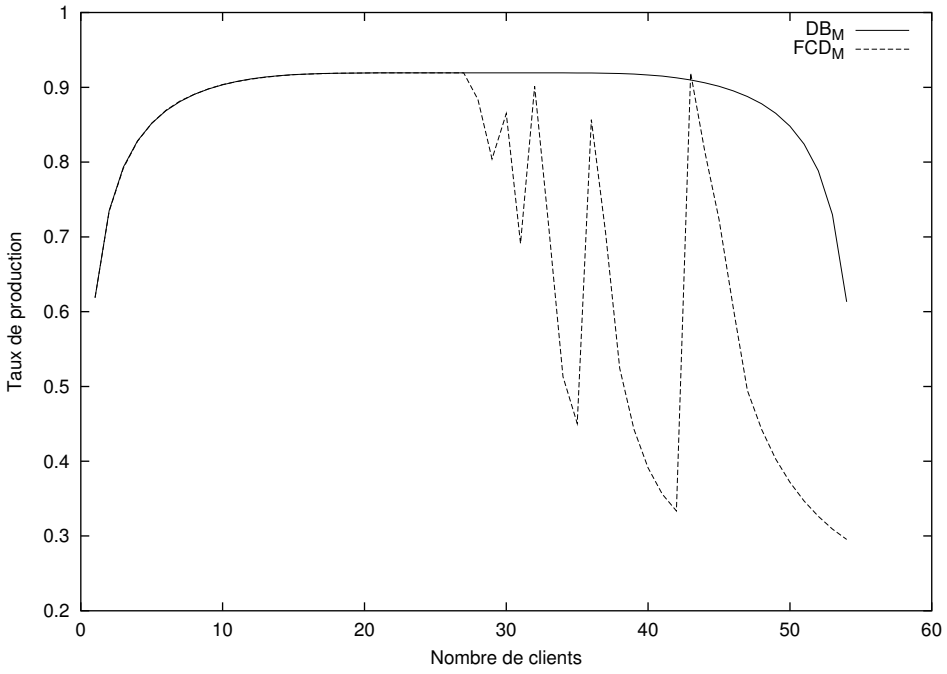

FIG. 4.15 – Taux de production évalué pour l'exemple 6 (Table 4.8 page précédente).

5

Cas de lignes de production de topologies quelconques

On se propose d'étudier une ligne de production manufacturière fermée dont la topologie est la plus générale possible. L'objectif est de généraliser la technique qui est présentée Chapitre 4 aux lignes fermées de topologies quelconques (comportant plusieurs boucles). On reprend alors de manière assez détaillée l'établissement des systèmes d'équations qu'on cherche à résoudre pour donner une estimation des paramètres de performance de la ligne, le contexte d'une topologie générale induisant des modifications sensibles (et une complexité d'écriture des équations plus importante), ainsi que certains aspects à résoudre qui n'apparaissaient pas du tout lorsque la ligne était simple. De plus, nous décrivons l'établissement du système d'équations pour avec des distributions des temps de réparation des dipôles exponentielles, générales-exponentielles ou hyper-exponentielles à deux étages.

Sommaire

5.1 Topologies quelconques : modèle et notations

Des exemples de modèles de lignes que l'on veut étudier sont donnés Figure 5.1 page suivante et Figure 5.2 page suivante. Le cadre d'études est celui évoqué Section 2.1, c'est-à-dire le modèle fluide d'une ligne homogène fermée comportant des mécanismes d'assemblages/désassemblages.

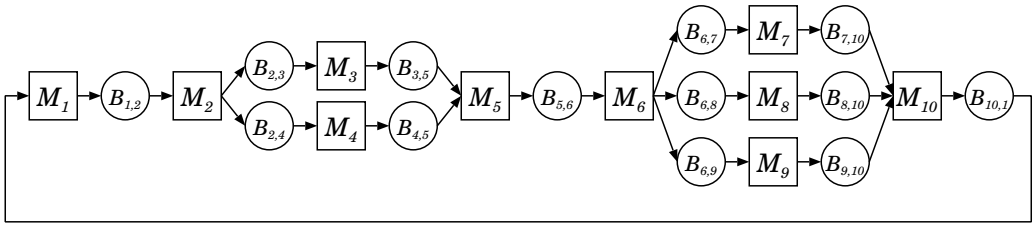

FIG. 5.1 – Exemple de ligne de production comportant plusieurs boucles. Chaque machine est représentée par un carré, chaque buffer par un cercle.

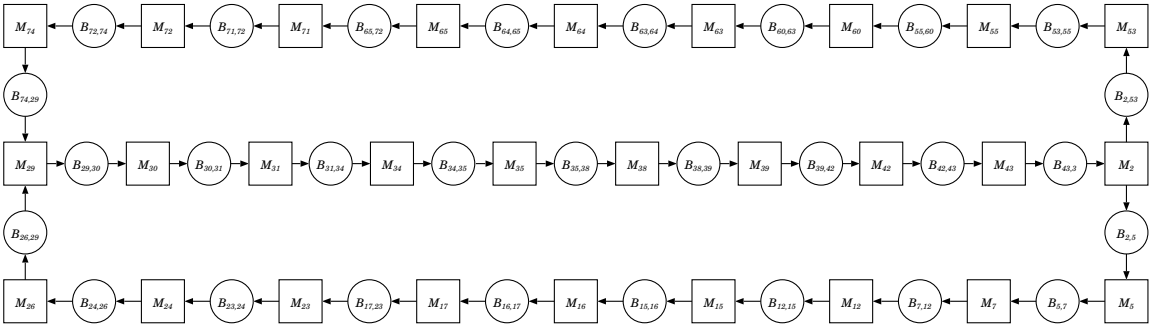

FIG. 5.2 – Ligne de production siamoise (utilisée par PSA dans certains de leurs ateliers de ferrage (Section 1.3)).

Remarque 11 : Lorsqu'une panne survient (par exemple, la machine M_3 de la ligne Figure 5.1), si cette panne dure assez longtemps, et que, de plus, aucune autre panne ne survient, arrive alors un moment où les buffers $B_{2,3}$ et $B_{3,4}$ deviennent simultanément et respectivement vide et plein. Cette situation n'est cependant possible qu'à la suite de conditions initiales favorables, et n'est pas représentative du fonctionnement stationnaire de notre ligne. En effet, la probabilité qu'une machine (si elle n'a qu'un buffer amont et un buffer aval) voit ses buffers amont et aval être au même "niveau" (au sens de la capacité restante à faire fonctionner la machine) est négligeable (du second ordre).

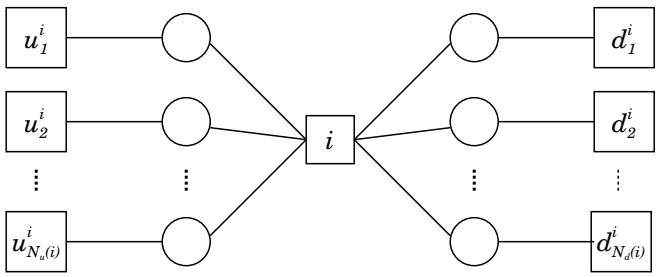

FIG. 5.3 – Partie de ligne de production autour de la machine M_i

5.1.1 Notations

Nous utilisons de nombreux ensembles ; ces derniers seront donc notés en caractère gras. On utilise les notations suivantes :

- K : nombre de machines dans la ligne ;
- $\mathbf{I}$: ensemble des indices de machines de la ligne L. On a $Card(\mathbf{I}) = K$;
- M_i : machine numéro i, $i \in I$;
- $B_{i,j}$: buffer situé entre les machines M_i et M_j ;
- λ_i : taux de la distribution exponentielle représentant le temps de bon fonctionnement de la machine M_i : $\lambda_i = \frac{1}{MTTF_i}$;
- μ_i : taux de la distribution exponentielle représentant le temps de réparation de la machine M_i : $\mu_i = \frac{1}{MTTR_i}$;
- U_i : vitesse de traitement de la machine M_i. On pose $U_i = U = 1.0$ (ligne homogène) ;
- N_u^i (resp. N_d^i) : nombre de machines situées directement en amont (resp. en aval) de la machine M_i ;
- $\mathbf{U_i} = \{u_1^i, \ldots, u_{N_u^i}^i\}$: ensemble des indices des N_u^i machines situées directement en amont de M_i ;
- $\mathbf{D_i} = \{d_1^i, \ldots, d_{N_d^i}^i\}$: ensemble des indices des N_d^i machines situées directement en aval de M_i ;
- $C_{i,j}$: capacité (nombre de places disponibles) du stock $B_{i,j}$.

Les paramètres de performance du modèle continu sont :

- pw_i : efficacité de la machine M_i. C'est la proportion du temps où elle travaille effectivement dans la ligne L ;
- X_i : taux de production de la machine M_i dans la ligne L ;
- $ps_{h,i}$: probabilité que la machine M_i soit non alimentée du fait du buffer $B_{h,i}$;
- $pb_{i,j}$: probabilité que la machine M_i soit bloquée du fait du buffer $B_{i,j}$;
- pI_i : probabilité que la machine M_i soit inutilisée (en attente) dans la ligne L ;
- pf_i : probabilité que la machine M_i soit en panne dans la ligne L ;
- $hm_{i,j}$: en-cours moyen du buffer $B_{i,j}$.

Ces paramètres sont ceux que l'on cherche à calculer.

5.1.2 Équations de la ligne

On a les relations suivantes :

$$\forall i \in I,\ X_i = U\ pw_i = pw_i \tag{5.1}$$

La conservation du flux :

$$\forall i \in I,\ \forall j \in I,\ X_i = X_j \Rightarrow pw_i = pw_j \tag{5.2}$$

Une machine ne pouvant être qu'exclusivement en panne, en train de produire ou en attente, on a la relation :

$$\forall i \in I,\ pI_i + pw_i + pf_i = 1 \tag{5.3}$$

Tant qu'une machine est en train de fonctionner, elle est sujette à défaillance. Le temps de bon fonctionnement (précédant une panne) est distribué exponentiellement, avec un taux λ_i. Lorsqu'elle est en panne, et donc en phase de réparation, le temps de la réparation suit aussi une loi exponentielle, de taux μ_i. Le processus alternant les états *en panne* et *actif* est donc markovien, et on en connaît les paramètres. On en déduit :

$$\forall i \in I,\ \lambda_i pw_i = \mu_i pf_i \tag{5.4}$$

De ces relations, on déduit aisément le relation liant l'efficacité en isolation de la machine et son efficacité dans la ligne :

$$pw_i = e_i(1 - pI_i) \tag{5.5}$$

On obtient, après avoir déterminé que (par définition) $pI_i = \sum_{m \in \mathbf{U_i}} ps_{m,i} + \sum_{n \in \mathbf{D_i}} pb_{i,n}$ [15] :

$$\forall i \in I,\ pw_i = e_i \left(1 - \sum_{m \in \mathbf{U_i}} ps_{m,i} - \sum_{n \in \mathbf{D_i}} pb_{i,n}\right) \tag{5.6}$$

Enfin, étant donné que l'on traite des lignes fermées de topologies quelconques, on sait que l'on a des contraintes de populations liées aux invariants de population de la ligne (dans le cas d'une boucle simple, nous n'avons qu'une seule contrainte). Ces contraintes de populations sont valables à chaque instant, donc aussi en valeur moyenne.

5.2 Principe et équations de la décomposition

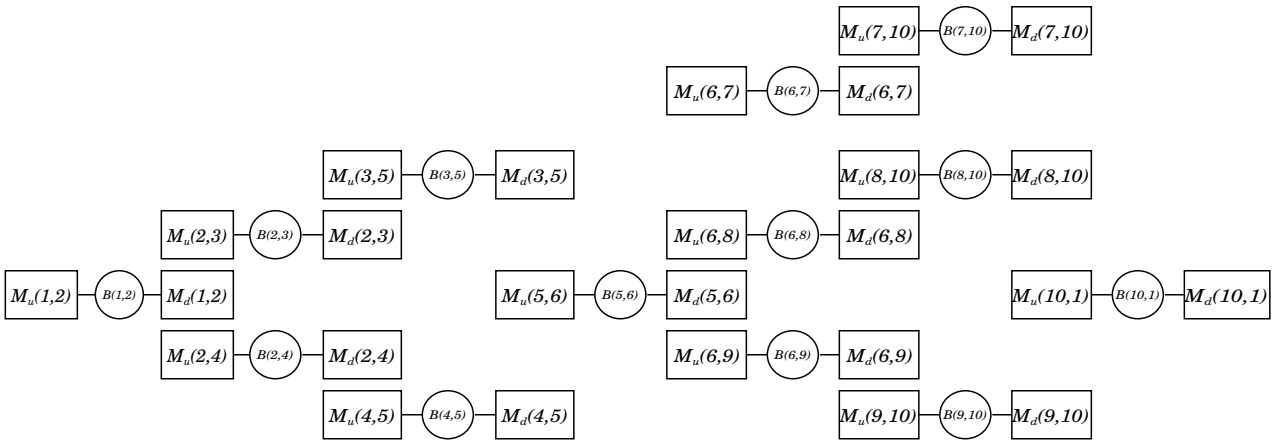

FIG. 5.4 – Décomposition de la ligne Figure 5.1 page 80.

A nouveau, le principe de la décomposition consiste à réduire l'analyse de la ligne initiale L à celle de systèmes plus simples, qui seront eux accessibles à une analyse exacte. La ligne est donc

décomposée en dipôles $D(i,j)$ de la même façon qu'avec un système constitué d'une boucle simple [17] (la décomposition de l'exemple 1 (Figure 5.1 page 80) est donnée Figure 5.4 page précédente). Chaque dipôle est constitué de deux machines $M_u(i,j)$ et $M_d(i,j)$, et d'un stock intermédiaire $B(i,j)$. L'objectif de la méthode de décomposition est de faire en sorte que le flux traversant le buffer $B(i,j)$ du dipôle soit le plus proche possible de celui qui traverse le dipôle équivalent $B_{i,j}$ dans la ligne originale L. Pour ce faire, nous devons établir un ensemble d'équations ainsi qu'un algorithme de résolution de ce système qui permette d'estimer les paramètres des machines des dipôles.

Rappelons que notre objectif est d'approcher le flux traversant les buffers de la ligne, par celui traversant le stock du dipôle équivalent. La machine $M_u(i,j)$ doit donc produire dans le buffer $B(i,j)$ de façon aussi semblable que possible que ce que fait la machine M_i dans la ligne L. Cette machine M_i arrête de produire dans trois cas : 1) quand elle tombe en panne ; 2) quand elle est non alimentée, et 3) quand elle est bloquée. Ce troisième cas peut être directement pris en compte dans $D(i,j)$ par un blocage de la machine $M_u(i,j)$ suite à la panne de $M_d(i,j)$. Cependant, $M_u(i,j)$ n'est jamais en famine. Une panne de $M_u(i,j)$ doit donc à la fois représenter une panne de la machine M_i et une non-alimentation de M_i. Une famine de M_i étant une conséquence d'une panne d'une machine située en amont de M_i, la machine $M_u(i,j)$ modélise l'ensemble de la partie de la ligne située en amont de M_i.

Remarque 12 : si dans le cas d'une boucle simple, il n'y a pas d'ambiguïté quant aux notions d'*ensemble de la partie de la ligne* situé en amont ou en aval d'une machine donnée, le cas multi-boucle est plus délicat. En effet, les mécanismes de blocage ou de famine sont plus compliqués [15]. Un exemple est donné Figure 5.5 :

Au temps $t < t_0$, la machine $M_{d_1^i}$ est tombée en panne. Au temps $t = t_0$, le buffer situé en amont de cette dernière est plein, provoquant le blocage de la machine M_i. Ensuite, le buffer situé avant M_i continue de se remplir. De même, le buffer B_{i,d_2^i} se vide *via* la machine $M_{d_2^i}$ (dont on suppose qu'elle ne tombe pas en panne pendant cette observation). Au moment t_1 où ce dernier buffer est vide, la machine $M_{d_2^i}$ va donc se trouver en famine, du fait de la panne de $M_{d_1^i}$. Il faut donc bien considérer que la machine $M_{d_1^i}$ fait partie de cet ensemble de la partie de la ligne située en amont de $M_{d_2^i}$, et réciproquement. On définit donc la notion de machine en amont (resp. en aval) d'une machines M_i par le fait qu'elle est susceptible de provoquer une non-alimentation (resp. un blocage) de M_i (cette définition est donc dépendante du nombre de clients qui circulent).

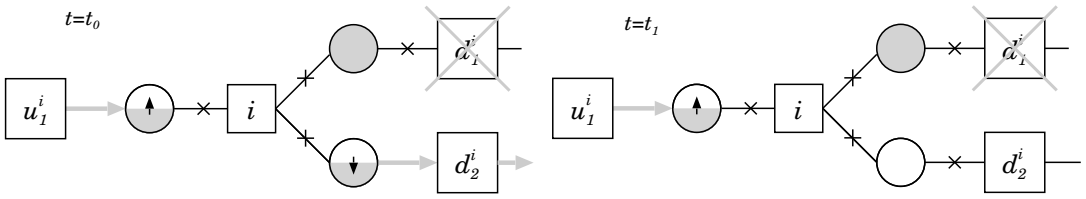

FIG. 5.5 – Détail d'un mécanisme de blocage autour d'une machine de désassemblage.

De même, $M_d(i,j)$ modélise l'ensemble de la partie de la ligne située en aval de M_j.

Pour chaque machine $M_{u_k^i}$ en amont direct de M_i, on définit un dipôle $D(u_k^i, i)$, ainsi que pour chaque machine $M_{d_l^i}$ directement en aval de la machine M_i, on définit le dipôle $D(i, d_l^i)$.

La décomposition dans le cas général est représentée Figure 5.3 page 80 et Figure 5.6 page suivante.

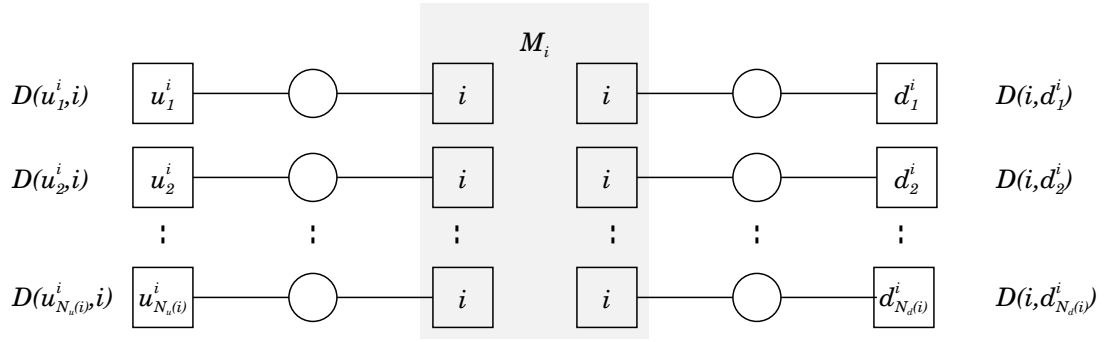

FIG. 5.6 – Décomposition.

Notation : on appelle dipôle connexe à la machine $M_u(i,j)$ d'un dipôle $D(i,j)$ tout dipôle $D(h,i),\ h \in \mathbf{U_i}$ ou $D(i,k),\ k \in \mathbf{D_i}$. On étendra aussi cette définition pour la machine M_i de la ligne L. Ainsi, les dipôles connexes à la machine M_i sont tous les dipôles représentés sur la figure 5.6.

Le nombre total de dipôles est dépendant de la topologie de la ligne. On note N_{dip} ce nombre de dipôles.

Le système d'équations est la synthèse de celui qui présenté dans [15] pour un système d'assemblage/désassemblage, dans une version étendue aux distributions des temps de réparations à plusieurs moments, et des équations d'un système bouclé présentées dans [20], [17] et rappelées Section **??**. Seules les grandes étapes de l'établissement de ces équations sont donc présentées ici.

On peut aisément établir les relations suivantes :

$$\begin{aligned} N_{dip} &= \sum_{i=1}^{i \leq K} N_u(i) \\ &= \sum_{i=1}^{i \leq K} N_d(i) \end{aligned} \tag{5.7}$$

5.2.1 Caractérisation des dipôles

Un dipôle $D(i,j)$ est totalement déterminé par la capacité $C(i,j)$ de son buffer, par les taux de pannes $\lambda_u(i,j)$ et $\lambda_d(i,j)$, et par les paramètres des distributions du temps de réparation.

On notera ainsi $m_u^{(n)}(i,j), n < n_m$ le moment d'ordre n de la distribution du temps de réparation de $M_u(i,j)$. On rappelle aussi que n_m est le nombre de moments nécessaires à caractériser complètement la distribution des temps de panne des machines des dipôles. Pour simplifier, on notera $m_u(i,j)$ le moment d'ordre 1 (la moyenne). On fait de même pour $M_d(i,j)$.

Les vitesses de traitement des machines du dipôles seront les mêmes que pour les machines d'origine, à savoir $U = 1.0$.

On peut maintenant définir les paramètres d'un dipôle de la même façon que pour la ligne entière L :

l'efficacité en isolation $e_u(i,j)$ de la machine $M_u(i,j)$ s'écrit :

$$e_u(i,j) = [1 + \lambda_u(i,j)m_u(i,j)]^{-1} \tag{5.8a}$$

de même :

$$e_d(i,j) = [1 + \lambda_u(i,j)m_u(i,j)]^{-1} \tag{5.8b}$$

Pour chaque dipôle, on a donc $2(n_m + 1)$ inconnues à déterminer. Si l'on suppose que l'on connaît ces paramètres, on peut analyser ce dipôle en isolation (on sait le faire pour des distributions exponentielles [18], générales exponentielles (qui se ramènent à des exponentielles), ou HE_2 [27] (voir Section 3.1). Les résultats de cette analyse sont les paramètres de performance du dipôle :

- $pw(i,j)$: efficacité du dipôle. La conservation du flux appliquée au dipôle implique que $pw(M_u(i,j)) = pw(M_d(i,j))$; on définit donc $pw(i,j) = pw(M_u(i,j)) = pw(M_d(i,j))$;
- $pb(i,j)$: proportion du temps où le dipôle est bloqué. Comme la machine $M_d(i,j)$ n'est jamais bloquée (par hypothèse), on a $pb(M_d(i,j)) = 0$, et on définit donc $pb(i,j) = pb(M_u(i,j))$;
- $ps(i,j)$: proportion du temps où le dipôle est non-alimenté. Comme la machine $M_u(i,j)$ n'est jamais en famine, $ps(M_u(i,j)) = 0$, et on définit $ps(i,j) = pb(M_d(i,j))$;
- $h_m(i,j)$: en-cours moyen du buffer du dipôle.

On peut ainsi écrire un ensemble de relations équivalentes à celles écrites pour la ligne originale (Section 5.1.2) :

$$pw(i,j) = e_u(i,j)(1 - pI_u(i,j)) \tag{5.9a}$$

de même :

$$pw(i,j) = e_d(i,j)(1 - pI_d(i,j)) \tag{5.9b}$$

5.2.2 Équations de la décomposition

Le système d'équations est directement dérivé de la méthode de décomposition avec mécanisme d'assemblage/désassemblage en ouvert décrit dans [15]. Le principe est de mettre en relation les paramètres (de panne et réparation) d'une machine $M_u(i,j)$ avec les paramètres de performance des dipôles connexes à $M_u(i,j)$, et de même pour $M_d(i,j)$. Pour cela, on remarque que, pour un indice i fixé, tous les dipôles connexes à cette machine (donc de la forme $D(i,j)$ ou $D(h,i)$) doivent avoir la même efficacité que M_i dans la ligne L :

$$\forall i, \forall h \in \mathbf{U_i}, \forall j \in \mathbf{D_i}, \quad pw_i = pw(h,i) = pw(i,j) \tag{5.10}$$

On peut déduire que tous les dipôles doivent donc avoir la même efficacité. Ensuite, la probabilité de non-alimenation du dipôle $D(h,i)$ doit être la même que la probabilité que la machine M_i soit en famine à cause d'une panne de M_h :

$$ps_{h,i} = ps(h,i) \tag{5.11}$$

De même, la probabilité d'un blocage du dipôle $D(i,j)$ doit être égale à la probabilité que la machine M_i soit bloquée du fait d'une panne de M_j :

$$pb_{i,j} = pb(i,j) \tag{5.12}$$

On peut déduire de (5.2), (5.6), (5.9), (5.11) et (5.12) les relations :

$$\forall j \in \mathbf{D_i} \quad \sum_{m \in \mathbf{U_i}} I_d(m,i) + \sum_{n \in \mathbf{D_i}} I_u(i,n) - I_i + 1 - N_u(i) - N_d(i) = \frac{N_u(i) + N_d(i) - 1}{E_u(i,j)} \tag{5.13a}$$

$$\forall h \in \mathbf{U_i} \quad \sum_{m \in \mathbf{U_i}} I_d(m,i) + \sum_{n \in \mathbf{D_i}} I_u(i,n) - I_i + 1 - N_u(i) - N_d(i) = \frac{N_u(i) + N_d(i) - 1}{E_u(h,i)} \tag{5.13b}$$

avec :

$$Eu_{i,j} = \frac{\displaystyle\sum_{m \in \mathbf{U_i}} pw(m,i) + \sum_{\substack{n \in \mathbf{D_i}, \\ n \neq j}} pw(i,n)}{N_u(i) + N_d(i) - 1} \tag{5.14a}$$

$$Ed_{h,i} = \frac{\displaystyle\sum_{\substack{m \in \mathbf{U_i}, \\ m \neq h}} pw(m,i) + \sum_{n \in \mathbf{D_i}} pw(i,n)}{N_u(i) + N_d(i) - 1} \tag{5.14b}$$

D'où l'on tire directement :

$$\forall j \in \mathbf{D_i},\ I_u(i,j) = \frac{N_u(i) + N_d(i) - 1}{Eu_{i,j}} + I_i - \sum_{m \in \mathbf{U_i}} I_d(m,i) - \sum_{\substack{n \in \mathbf{D_i}, \\ n \neq j}} I_u(i,n) - N_u(i) - N_d(i) + 1 \tag{5.15a}$$

$$\forall h \in \mathbf{U_i},\ I_d(h,i) = \frac{N_u(i) + N_d(i) - 1}{Ed_{h,i}} + I_i - \sum_{\substack{m \in \mathbf{U_i}, \\ m \neq h}} I_d(m,i) - \sum_{n \in \mathbf{D_i}} I_u(i,n) - N_u(i) - N_d(i) - 1 \tag{5.15b}$$

Remarque 13 : on peut noter que les expressions $E_u(i,j)$ ou $E_d(h,i)$ sont, au final, égales à $pw(i,j)$ et $pw(h,i)$, du fait de l'égalité de toutes les efficacités en jeu ici. Cependant, on garde ces expressions sous cette forme non simplifiée, en vue du processus itératif de résolution du problème. En effet, si en théorie (et donc à convergence de l'algorithme), toutes les efficacités sont égales, ce n'est pas vrai au cours du processus de convergence. Ces expressions (qui sont des moyennes des efficacités) prennent donc leur sens.

Pour finir, on doit se souvenir des principes de la décomposition. Rappelons qu'une défaillance de la machine $M_u(i,j)$ représente à la fois une panne et une non alimentation de M_i. De plus, une famine de M_i peut être provoquée soit par une panne d'une machine représentée par $M_u(m,i), m \in \mathbf{U_i}$, soit par une panne d'une machine représentée par $M_d(i,k), k \in \mathbf{D_i} \setminus \{j\}$ (voir Section 5.2 pour les détails de ces mécanismes). On note $\alpha_u(m,i,j), m \in \mathbf{U_i}$ la proportion stationnaire de pannes de la machine $M_u(i,j)$ dues aux pannes de la machine $M_u(m,i)$, et $\alpha'_u(i,j,n), n \in \mathbf{D_i} \setminus \{j\}$ la proportion stationnaire de pannes de la machine $M_u(i,j)$ dues aux pannes de la machine $M_d(i,n)$. On peut alors écrire les relations suivantes (dont l'établissement n'est pas détaillé, mais qui forment la synthèse de [27] et [15]) :

$$\forall h \in \mathbf{U_i},\ \alpha_u(h,i,j) = \frac{m_i}{m_u^r(h,i)} \frac{ps(h,i)}{\left(\frac{1}{e_u(i,j)} - 1\right) pw(h,i) + \sum_{m \in \mathbf{U_i}} ps(m,i) \left(\frac{m_i}{m_u^r(m,i)} - 1\right) + \sum_{\substack{n \in \mathbf{D_i}, \\ n \neq j}} pb(i,n) \left(\frac{m_i}{m_d^r(i,n)} - 1\right)} \tag{5.16a}$$

$$\forall k \in \mathbf{D_i}, k \neq j,\ \alpha'_u(i,j,k) = \frac{m_i}{m_d^r(i,k)} \frac{pb(i,k)}{\left(\frac{1}{e_u(i,j)} - 1\right) pw(i,k) + \sum_{m \in \mathbf{U_i}} ps(m,i) \left(\frac{m_i}{m_u^r(m,i)} - 1\right) + \sum_{\substack{n \in \mathbf{D_i}, \\ n \neq j}} pb(i,n) \left(\frac{m_i}{m_d^r(i,n)} - 1\right)} \tag{5.16b}$$

où $m_u^{r(n)}(m,i)$ (resp. $m_d^{r(n)}(i,n)$) sont les moments d'ordre n de la distribution du temps résiduel de réparation de la machine $M_u(m,i)$ (resp. $M_d(i,n)$) au moment de la non alimentation (resp. du blocage) de M_i.

De même, on note $\alpha_d(m,j,i)$ la proportion stationnaire de pannes de la machine $M_d(j,i)$ dues aux pannes de la machine $M_u(m,i)$, et $\alpha'_u(j,i,n)$ la proportion stationnaire de pannes de la machine $M_d(j,i)$ dues aux pannes de la machine $M_d(i,n)$:

$$\forall h \in \mathbf{U_i}, h \neq j,\ \alpha_d(h,j,i) = \frac{m_i}{m_u^r(h,i)} \frac{ps(h,i)}{\left(\frac{1}{e_d(j,i)} - 1\right) pw(h,i) + \sum_{\substack{m \in \mathbf{U_i}, \\ m \neq j}} ps(m,i) \left(\frac{m_i}{m_u^r(m,i)} - 1\right) + \sum_{n \in \mathbf{D_i}} pb(i,n) \left(\frac{m_i}{m_d^r(i,n)} - 1\right)} \tag{5.17a}$$

$$\forall k \in \mathbf{D_i},\ \alpha'_d(j,i,k) =$$

$$\frac{m_i}{m_d^r(i,k)} \frac{pb(i,k)}{\left(\frac{1}{e_d(j,i)} - 1\right) pw(i,k) + \sum\limits_{\substack{m \in \mathbf{U_i},\\ m \neq j}} ps(m,i) \left(\frac{m_i}{m_u^r(m,i)} - 1\right) + \sum\limits_{n \in \mathbf{D_i}} pb(i,n) \left(\frac{m_i}{m_d^r(i,n)} - 1\right)} \tag{5.17b}$$

D'où l'on tire (en se souvenant de la définition des paramètres α et α') :

$$m_u^{(n)}(i,j) = \sum_{m \in \mathbf{U_i}} \alpha_u(m,i,j) \overset{(n)}{m_u^r}(m,i) + \sum_{\substack{n \in \mathbf{D_i},\\ n \neq j}} \alpha'_u(i,j,n) \overset{(n)}{m_d^r}(i,n) \tag{5.18a}$$

$$+ \left(1 - \sum_{m \in \mathbf{U_i}} \alpha_u(m,i,j) - \sum_{\substack{n \in \mathbf{D_i},\\ n \neq j}} \alpha'_u(i,j,n)\right) n!\ m_i^n$$

$$\lambda_u(i,j) = \left(\frac{1}{e_u(i,j)} - 1\right) \frac{1}{m_u(i,j)} \tag{5.18b}$$

$$m_d^{(n)}(j,i) = \sum_{\substack{m \in \mathbf{U_i},\\ m \neq j}} \alpha_d(m,j,i) \overset{(n)}{m_u^r}(m,i) + \sum_{n \in \mathbf{D_i}} \alpha'_d(j,i,n) \overset{(n)}{m_d^r}(i,n) \tag{5.19a}$$

$$+ \left(1 - \sum_{\substack{m \in \mathbf{U_i},\\ m \neq j}} \alpha_d(m,j,i) - \sum_{n \in \mathbf{D_i}} \alpha'_d(j,i,n)\right) n!\ m_i^n$$

$$\lambda_d(j,i) = \left(\frac{1}{e_d(j,i)} - 1\right) \frac{1}{m_d(j,i)} \tag{5.19b}$$

Le système formé des équations (5.15), (5.16), (5.17), (5.18) et (5.19) représente le système d'équations de la décomposition. On notera que les paramètres I_u, I_d, α_u, α'_u, α_d et α'_d ne sont pas des inconnues à proprement parler de notre système. Ils servent d'intermédiaires de calcul. Les inconnues de notre système sont les paramètres λ_u, λ_d, $m_u^{(n)}$ et $m_d^{(n)}$. Elles sont donc au nombre de $(n_m + 1)N_{dip}$.

Les estimations des moments résiduels pour les trois distributions envisagées des temps de réparation sont données Section 3.2.3 (page 42).

5.2.3 Équations de conservation

Dans le cas d'une boucle simple, la contrainte de population est simple à exprimer (voir Chapitre 4, [17] et [20]) : la somme des en-cours des buffers est constante dans le temps. On

sait, de plus, que l'on a une équation "de trop" dans le système de la décomposition (voir Section 4.1.1).

Nous avons proposé une technique simple et efficace pour résoudre alors ce système d'équations complet [17] présentée Section **??**.

Rappel : résolution dans le cas de la boucle simple

La solution que nous avons proposé est de modifier le systèmes d'équations de manière à ajouter une pseudo-inconnue. En pratique, on réécrit les équations (5.15) sous une forme modifiée (ici pour le cas de la boucle simple) :

$$I_u(i+1,i+2) = \beta_u \left(\frac{1}{pw(i,i+1)} + I_{i+1} - I_d(i,i+1) - 1 \right) \tag{5.20a}$$

$$I_d(i-1,i) = \beta_d \left(\frac{1}{pw(i,i+1} + I_i - I_u(i,i+1) - 1 \right) \tag{5.20b}$$

avec (en notant Q la population totale estimée dans la boucle à un instant donné de la convergence) :

$$\beta_u = \frac{Q}{N} \tag{5.21a}$$

$$\beta_d = \frac{N}{Q} \tag{5.21b}$$

Extension au cas général

L'extension de cette idée aux systèmes multi-boucles n'est pas triviale. En effet, la première question est de savoir de combien de "boucles" est constituée notre ligne. Chacune de ces "boucles" est par définition un invariant de population. C'est un ensemble de machines et de buffers dans lequel la somme des clients est constante à chaque instant. En pratique, le nombre de ces invariants est déterminé par la topologie de la ligne, et une étude appropriée nous permettra de les mettre en évidence.

On note **Iv** l'ensemble des invariants. Chaque invariant (donc chaque élément de **Iv**) est lui-même un ensemble (ordonné) de machines de la ligne et des buffers séparant ces machines. On notera que le sous-ensemble d'un invariant constitué des seules machines est suffisant pour définir cet invariant. De même, le sous-ensemble des buffers est suffisant pour définir un invariant. L'ensemble **Iv** est constitué de N_{Iv} invariants notés $\mathbf{Iv_i}$. On note aussi N_{Iv_i} le nombre d'éléments de $\mathbf{Iv_i}$ ($N_{Iv_i} = Card(\mathbf{Iv_i})$).

Notations :

- **Iv** : ensemble des invariants de la ligne ;
- N_{Iv} : nombre d'invariants (cardinal de **Iv**) ;
- N_{Iv_i} : nombre d'éléments de $\mathbf{Iv_i}$ (en réalité, nombre de machines, ou nombre de buffers) ;
- $\mathbf{iv_i}$: $i^{ème}$ invariant ($1 \leq i \leq N_{Iv}$) ; il peut être vu trois formes :

- $\mathbf{iv_i^i} = \{n_1, \ldots, n_{N_{iv_i}}\}$: représentation de $\mathbf{iv_i}$ sous forme de l'ensemble ordonné des indices des machines constituant l'invariant ;
- $\mathbf{iv_i^m} = \{M_{n_1}, \ldots, M_{n_{N_{Iv_i}}}\}$: représentation de $\mathbf{iv_i}$ sous forme de l'ensemble des machines constituant l'invariant ;
- $\mathbf{iv_i^b} = \{B_{n_1,n_2}, \ldots, B_{n_{N_{Iv_i}},n_1}\}$: représentation de $\mathbf{iv_i}$ sous forme de l'ensemble des buffers constituant l'invariant ;

– $N(\mathbf{iv_i})$: la population de l'invariant $\mathbf{iv_i}$ ($\mathbf{iv_i} \in \mathbf{Iv}$). $0 < N(\mathbf{iv_i}) < \sum_{\substack{i,j \\ B(i,j)\in\mathbf{iv_i^b}}} C_{i,j}$

Au final, en plus des équations de le décomposition (5.18) et (5.19), on doit ajouter un ensemble d'équations de conservation dans chacun des invariants (on fait comme d'habitude, et du fait de la nature circulaire de chaque invariant, l'hypothèse que $n^i_{N_{iv_i}+1} = n^i_1$) :

$$\forall i \in \{1, .., N_{Iv}\},\ \sum_{j\in\mathbf{iv_i^i}} h_m(n^i_j, n^i_{j+1}) = N_i \tag{5.22}$$

5.3 Résolution du système

La méthode de résolution de ce système d'équations est issue des travaux exposés dans [20], dans la version que nous avons proposée dans [17] et rappelée en détails Section **??**.

5.3.1 Principe

On utilise une méthode de point fixe pour la résolution de ce système d'équations. Comme toujours avec ces méthodes, on va en premier lieu supposer les paramètres recherchés connus. Ce seront (a priori) de mauvaises estimations de ces paramètres. Puis on va déduire de meilleures estimations au moyen des équations présentées Section 5.2.2.

Pour chaque machine (soit pour i fixé), le système (5.18) comporte $(n_m + 1)N_d(i)$ équations (n_m étant le nombre de moments utilisés pour l'estimation de la distribution des temps de réparation ; en général 1, 2 ou 3), et autant d'inconnues (les valeurs des $\lambda_u(i,j)$ et $m_u^{(n)}(i,j)$). De même le système (5.19) comporte $(n_m + 1)N_u(i)$ équations, et autant d'inconnues.

On dispose donc au total d'un système de $(n_m + 1)\sum_{i=1}^{i\leq K}(N_u(i) + N_d(i)) = 2(n_m + 1)N_{dip}$ équations et d'autant d'inconnues, plus les équations de conservation (5.22). Mais, comme on l'a vu Section 5.2.3, notre système d'équations n'est pas libre ; certaines de ses équations sont linéairement dépendantes des autres. En fait, nous disposons d'autant d'équations dépendantes que nous avons d'invariants indépendants (c'est à dire d'invariants qui ne peuvent pas s'exprimer comme combinaison linéaire des autres). En effet, chaque invariant indépendant implique une nouvelle relation circulaire des équations (*via* les relations d'égalité des efficacités pw_i des machines de la ligne, selon exactement le même processus que pour la boucle simple évoqué à la section Section 4.1.1 du chapitre Section **??**). Mais nous avons aussi autant d'équations de conservation indépendantes que d'invariants indépendants (par définition des invariants).

Au final, on dispose donc bien d'un système complet bien dimensionné, supposé bien posé, que l'on peut donc chercher à résoudre. Remarquons que cela n'est pas formellement démontré, mais cela n'a pas vraiment d'importance, car notre méthode, expliquée Section 4.2.5 pour la ligne simple, ne consiste pas à supprimer les équations excédentaires, mais à ajouter des contraintes (autant que nécessaire) pour en faire un système bien posé.

La première étape consiste à identifier les invariants de notre ligne.

Extraction des invariants

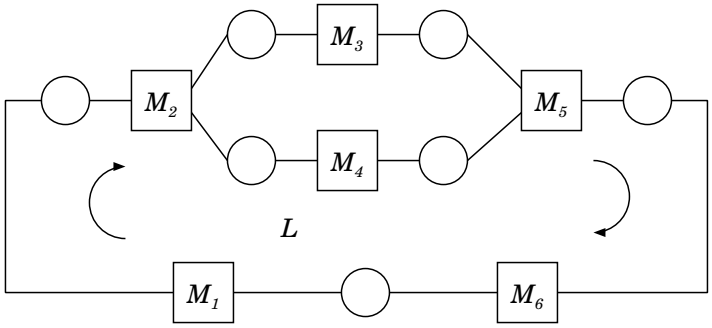

FIG. 5.7 – Exemple de ligne.

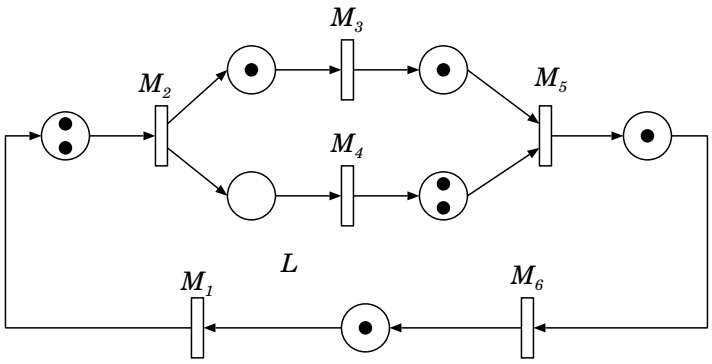

FIG. 5.8 – Réseau de Petri extrait de le ligne de la figure Figure 5.7 (les marquages initiaux représentent les clients initialement présents dans les stocks).

Notre ligne peut se représenter à l'aide d'un réseau de Petri pour pouvoir en extraire les propriétés topologiques (on pourrait se passer du formalisme des réseaux de Petri, mais cela nous permet d'utiliser les nombreux outils existants pour ce formalisme). Chaque transition représente une machine, et chaque stock est représenté par une place (voir Figure 5.7 et Figure 5.8). On omet de la sorte tout l'aspect temporel, mais il ne nous intéresse pas ici, puisqu'on cherche à extraire de notre ligne des propriétés purement topologiques. Par contre, on remarque dans la ligne qu'un stock n'a qu'une seule machine amont, et une seule machine aval. Donc dans le réseau de Petri équivalent, chaque place n'a qu'une seule transition amont et qu'une seule transition aval. Notre réseau de Petri est donc en réalité un graphe d'état.

Pour obtenir les invariants de notre ligne de production, on détermine une plus petite famille génératrice de semi-flots du réseau de Petri extrait de la ligne. En faisant ce choix (extraction d'une famille de semi-flots), on s'assure qu'aucune place ni transition n'est oubliée (aucune machine, ni aucun buffer). Par contre, on ne s'assure pas que notre famille nous donne le nombre minimal de flots, donc d'invariants. Cela n'est cependant pas gênant, on verra plus loin pourquoi (rappelons néanmoins qu'un semi-flots est à coefficients positifs).

La technique de détermination d'une plus petite famille génératrice de semi-flots (PPFGSF) est donnée Annexe B.

5.3.2 Système d'équations modifiées

Une fois cette plus petite famille de semi-flots obtenue, on peut poser le système d'équations complet que l'on cherche à résoudre comme étant le système d'équations (5.18), (5.19) et (5.22). Nous savons que ce système est redondant. Afin de rendre ce système parfaitement bien défini, nous avons deux possibilités (voir Chapitre 4 et [17]) : soit on "élimine" un certain nombre d'équations ((5.18) et (5.19), c'est la méthode originale utilisée pour la résolution d'une boucle simple Frein et al. dans [20]), soit on *ajoute* des variables à notre système telles qu'à l'équilibre, on ait bien résolu le système original (c'est la méthode proposée dans [17] et présentée Chapitre 4).

Nous allons utilisez cette seconde méthode, car elle est nettement plus efficace et plus robuste (voir section Section 4.3), et elle est aisément implémentable dans le cas des lignes comportant de multiples boucles.

Le principe est de remplacer les équations (5.15) par les équations :

$$\forall j \in \mathbf{D_i},\ I_u^{mod}(i,j) = \beta_u(i,j) I_u(i,j) \tag{5.23a}$$

$$\forall h \in \mathbf{U_i},\ I_d^{mod}(h,i) = \beta_d(h,i) I_d(h,i) \tag{5.23b}$$

On ajoute donc un coefficient $\beta_u(i,j)$ (ou $\beta_d(i,j)$), dont la valeur va évoluer au cours du déroulement de l'algorithme pour aboutir (on va tout faire pour), à l'équilibre, à la valeur 1.0, de manière à bien résoudre le système d'équations initial.

Rappelons que nous avons ajouté ces coefficients de façon à faire intervenir les contraintes de population dans le système d'équations. On va donc les définir comme des fonctions des différentes contraintes de populations (donc comme des fonctions de Q, somme des en-cours pour chaque invariant, et de N, valeur de cet invariant). Mais nous devons aussi nous souvenir qu'à convergence, sachant que l'on veut avoir résolu le système d'origine (sans les modifications (5.23)), ces coefficients doivent alors valoir 1.0. Les fonctions définissant ces coefficients doivent ainsi être des fonctions croissantes du rapport Q/N (pour les β_u) ou des fonctions décroissantes du rapport Q/N (pour les β_d), et telles que pour $Q = N$, elles valent 1.0.

Coefficients de convergence

On pourrait croire à la lecture du système modifié (5.23) que l'on ajoute trop de variables supplémentaires à notre système, puisqu'on a formellement défini deux coefficients β par dipôle. En pratique, on va définir les fonctions $\beta_u(i,j)$ et $\beta_d(i,j)$ de façon à ce qu'on n'ait ajouté qu'un seul coefficient par invariant (on notera $\beta(\mathbf{iv})$ le coefficient affecté à l'invariant $\mathbf{iv}$). Chacun de ces coefficients sera estimé, à chaque instant de la convergence, par une relation du type :

$$\beta(\mathbf{iv}) = \frac{N(\mathbf{iv})}{N_{\mathbf{iv}}} \tag{5.24}$$

c'est-à-dire le rapport de la population à un instant donné de la convergence dans un invariant sur sa population à l'équilibre. Ce rapport tend bien vers 1.0 lorsque la méthode converge.

Ensuite, pour chaque dipôle, on va calculer le coefficient $\beta_u(i,j)$ ou $\beta_d(i,j)$ comme une combinaison linéaire de tous les coefficients $\beta(\mathbf{iv})$ des invariants $\mathbf{iv}$ auxquels appartient le dipôle $D(i,j)$ (combinaison normalisée, de façon à ce qu'on ait toujours, à convergence, $\beta_u(i,j) = 1.0$ et $\beta_d(j,i) = 1.0$). Au final, on aura finalement ajouté autant de coefficients (indépendants) que d'invariants indépendants. On voit maintenant pourquoi il est suffisant d'avoir une famille génératrice, et qu'il n'est pas nécessaire d'avoir une famille minimale (une base).

Détermination des coefficients $\beta_u(i,j)$ et $\beta_d(i,j)$

On sait que (voir Section 4.2.4), tout autre paramètre fixé, la population effective moyenne dans un invariant est une fonction décroissante de β_u, et croissante de β_d. Une solution simple pour attribuer une valeur à ces paramètres est alors de choisir :

$$\beta_u(i,j) = f_{ag}(\beta(iv_1(B_{i,j})), \ldots) \tag{5.25a}$$
$$\beta_d(i,j) = 1.0/f_{ag}(\beta(iv_1(B_{i,j})), \ldots) \tag{5.25b}$$

avec :

$$\mathbf{iv}(B_{i,j}) = \{\mathbf{iv} \in \mathbf{Iv} | B_{i,j} \in \mathbf{iv}\} \, (= \{iv_1(B_{i,j}), iv_2(B_{i,j}), \ldots\}) \tag{5.26}$$

où la fonction d'agrégation f_{ag} est une fonction du type moyenne arithmétique, moyenne géométrique, etc. Il suffit qu'elle soit strictement monotone croissante pour chaque paramètre passé (formellement, que chaque dérivée partielle soit monotone croissante), et que $f_{ag}(1,1,...,1) = 1.0$ (normalisée). Le choix de la fonction d'agrégation se fera de façon à ce que la méthode soit la plus robuste possible.

5.4 Algorithme

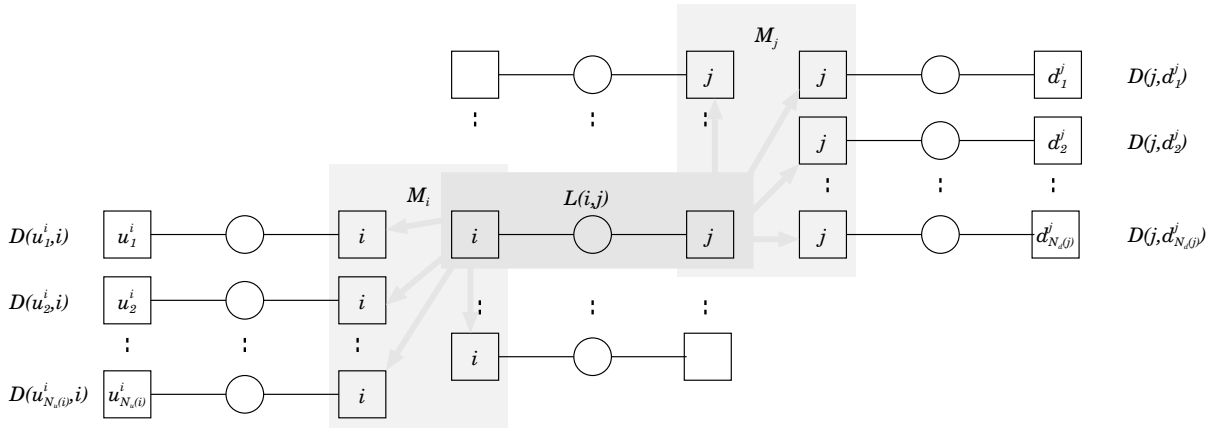

FIG. 5.9 – Principe de mise à jour des paramètres. L'analyse en isolation du dipôle $D(i,j)$ permet d'obtenir de meilleures estimations des paramètres des dipôles autour.

L'analyse en isolation d'un dipôle $D(i,j)$ va permettre de faire une évaluation (meilleure que la précédente) des paramètres λ_d et $m_d^{(n)}$ des dipôles connexes à $M_u(i,j)$, et des paramètres λ_u et

$m_u^{(n)}$ des dipôles connexes à $M_d(i,j)$ (Figure 5.9 page précédente), à l'aide des équations (5.18) et (5.19).

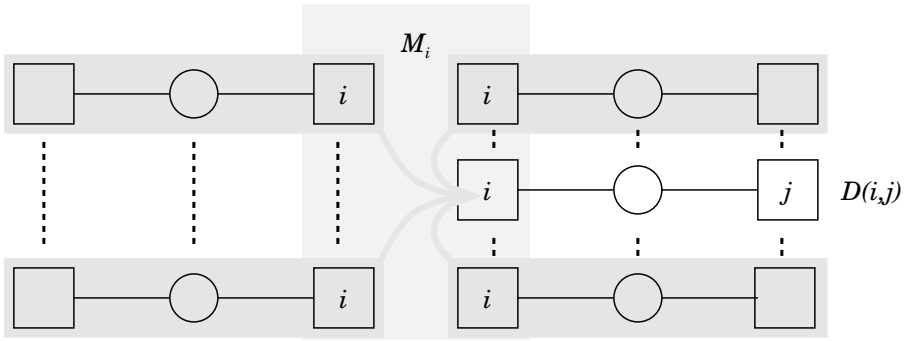

FIG. 5.10 – Mise à jour des paramètres de $M_u(i,j)$. Les paramètres de performance des autres dipôles liés à M_i permettent d'évaluer les paramètres de $M_u(i,j)$.

En réalité, on se rend compte, au regard du système d'équations, que si on veut faire une réévaluation des paramètres $\lambda_u(i,j)$ et $m_u^{(n)}(i,j)$, on a besoin de connaître les paramètres de performance de tous les dipôles connexes à $M_u(i,j)$ (donc de la forme $D(k,i)$ et $D(i,l)$). De même, la mise à jour des paramètres $\lambda_d(i,j)$ et $m_d^{(n)}(i,j)$ nécessite la connaissance des paramètres de performance de tous les dipôles connexes à la machine $M_d(i,j)$. La vision correcte, définissant ainsi une "causalité", est alors plutôt celle présentée Figure 5.10 : la connaissance des paramètres de performance de tous les dipôles connexes à une machine d'un dipôle permet une meilleure estimation des paramètres de cette machine. On parle de "causalité" en ce sens qu'une nouvelle évaluation des paramètres d'un dipôle entraîne une modification de ses paramètres de performance, et donc qu'il est possible d'obtenir une meilleure estimation des paramètres de tous les dipôles connexes. Cela va guider la façon dont on va parcourir l'ensemble des dipôles pour faire les estimations successives des paramètres de ces derniers.

5.4.1 Parcours de la ligne

L'algorithme repose sur l'idée que l'évaluation des paramètres de performance d'un ou plusieurs dipôles permet de faire une meilleure estimation des paramètres d'un dipôle connexe donné. En appliquant cela de manière récursive, on arrive successivement à réévaluer tous les paramètres des dipôles la ligne. Il nous faut pour cela un parcours de la ligne (donc un ensemble ordonné de dipôles), que l'on va emprunter une fois en sens direct (afin de mettre à chaque à jour les paramètres des machines $M_u(i,j)$ des dipôles), et une fois en sens inverse (pour mettre à jour les paramètres des machines $M_d(i,j)$).

L'objectif est de trouver un parcours qui respecte au mieux la graphe de dépendance des dipôles (la façon de faire ce parcours s'est révélée très critique quant à la robustesse et à l'efficacité de l'algorithme).

Le parcours proposé est une sorte de parcours en profondeur sensiblement modifié pour tenter de respecter au mieux le graphe des dépendances (tel qu'exprimé Figure 5.10). Il s'agit d'un algorithme récursif s'appliquant à la ligne originale considérée comme un graphe orienté, où les noeuds du graphe sont les machines de la ligne, et les arcs du graphe sont les buffers (voir Figure 5.11 page suivante pour le cas de l'exemple 1). Nous cherchons à obtenir une liste

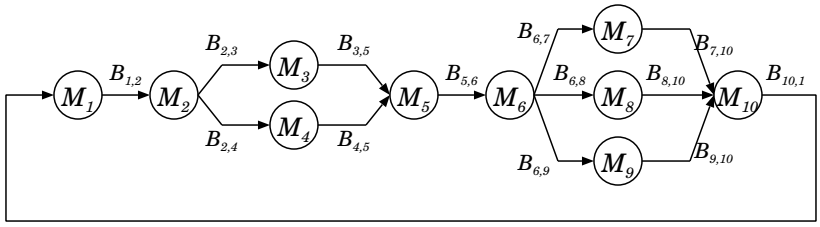

FIG. 5.11 – Graphe équivalent à la ligne de l'exemple 1 représentée Figure 5.1 page 80.

ordonnée des arcs de ce graphe, qui vont correspondre à l'ordre dans lequel on va effectuer les mises à jour des paramètres des machines des dipôles.

Algorithme 5.1 Recherche d'un parcours de la ligne

1: **Initialiser :** $P = ()$ le parcours est initialisé à une liste vide
2: **Initialiser :** les listes $pred(n)$, pour tout noeud n du graphe
3: **tant que** tous les noeuds n'ont pas été visités **faire**
4: choisir un noeud n quelconque non encore visité
5: exécuter Parcours(n)
6: **fin tant que**

Algorithme 5.2 Parcours(n) : la fonction récursive proprement dite

Requiert : un noeud n
1: **pour** tous les successeurs v de n **faire**
2: ajouter (n, v) à P
3: **si** $n \in pred(v)$ **alors**
4: enlever n de la liste $pred(v)$
5: **fin si**
6: **si** $pred(v)$ est vide **alors**
7: **si** v n'a pas encore été visité **alors**
8: exécuter Parcours(v)
9: **fin si**
10: **fin si**
11: **fin pour**
12: marquer n comme visité

On a besoin, pour chaque noeud du graphe, de savoir s'il a déjà été visité, et d'une liste des prédécesseurs du noeud (nommée $pred(n)$, où n est un noeud du graphe).

L'algorithme est donné en Algorithme 5.1 et Algorithme 5.2.

Notation : on définit une branche comme un sous-ensemble connexe du graphe tel que les noeuds n'y ont qu'un seul prédécesseur et une seul successeur.

Ce parcours est construit de façon à respecter au mieux le graphe de dépendance des paramètres (induit par les dépendances données par exemple figure Figure 5.10 page ci-contre). Ainsi, tant que l'on est sur une branche, on poursuit de l'avant. Si jamais on rencontre un noeud qui fait un assemblage, on cherche à ce que toutes les branches qui arrivent sur ce noeud soient

situées avant ce noeud dans le parcours. Et si on rencontre un noeud qui fait un désassemblage, on visite les branches qui en partent successivement.

On remarque aussi que l'algorithme principal (Algorithme 5.1) demande à la ligne 4 de choisir un noeud quelconque du graphe, qui sera la racine du parcours. En pratique, on ne le choisira pas de manière aléatoire.

Choix du noeud racine (Algorithme 5.1, ligne 4)

Étant donné les dépendances pour l'évaluation des paramètres des dipôles (Figure 5.10 page 94), le choix d'un noeud racine se fait par :

- s'il existe, choisir un noeud du graphe qui appartient à tous les invariants (le mieux est de choisir le noeud le plus central d'une branche qui appartient à tous les invariants). Dans ce cas, l'ensemble du graphe sera parcouru à partir de cette unique racine,
- sinon, choisir un noeud qui appartient au plus d'invariants possibles. Le parcours ne sera pas complet à partir de ce noeud, et il faudra donc recommencer à partir d'un autre noeud, en recommençant le même algorithme sur l'ensemble des noeuds encore non visités.

Exemple pratique

Une exécution de l'algorithme détaillé en Algorithme 5.1 sur l'exemple Figure 5.1 page 80 est le suivant (les étapes sont celles de l'Algorithme 5.1) :

- en accord avec Section 5.4.1, on peut choisir la machine M_1 comme point de départ, P est une liste vide, toutes les machines sont marquées comme non visitées, et toutes les listes $pred(M_i)$ sont initialisées à leur valeur, ie. :
  - $pred(M_1) = (M_{10})$
  - $pred(M_2) = (M_1)$
  - $pred(M_3) = pred(M_4) = (M_2)$
  - $pred(M_5) = (M_3, M_4)$,
  - etc.
- Etape 1, avec $v = M_2$: on ajoute $B_{1,2}$ à P : $P = (B_{1,2})$,
- on enlève M_1 de $pred(M_2)$ (Etape 4) ;
- $pred(M_2)$ est vide, M_2 n'a pas été visité, on exécute Parcours(M_2) :
  - Etape 1, avec $v = M_4$: on ajoute $B_{2,4}$ à P : $P = (B_{1,2}, B_{2,4})$,
  - on enlève M_2 de $pred(M_4)$ (Etape 4) ;
  - $pred(M_4)$ est vide, M_4 n'a pas été visité, on exécute Parcours(M_4) :
    - Etape 1, avec $v = M_5$: on ajoute $B_{4,5}$ à P : $P = (B_{1,2}, B_{2,4}, B_{4,5})$,
    - on enlève M_4 de $pred(M_5)$ (Etape 4) ;
    - $pred(M_5)$ est non-vide.
    - il n'y a pas d'autres voisins à M_4, on retourne de Parcours(M_4),
  - Etape 1, avec $v = M_3$: on ajoute $B_{2,3}$ à P : $P = (B_{1,2}, B_{2,4}, B_{4,5}, B_{2,3})$,
  - on enlève M_2 de $pred(M_3)$ (Etape 4) ;
  - $pred(M_3)$ est vide, M_3 n'a pas été visité, on exécute Parcours(M_3) :
    - Etape 1, avec $v = M_5$: on ajoute $B_{3,5}$ à P : $P = (B_{1,2}, B_{2,4}, B_{4,5}, B_{2,3}, B_{3,5})$,
    - on enlève M_3 de $pred(M_5)$ (Etape 4) ;
    - $pred(M_5)$ est vide, M_5 n'a pas été visité, on exécute Parcours(M_5) :
      - etc.

Au final, nous obtenons pour résultat la liste :

$$P = (B_{1,2}, B_{2,4}, B_{4,5}, B_{2,3}, B_{3,5}, B_{5,6}, B_{6,7}, B_{7,10}, B_{6,9}, B_{9,10}, B_{6,8}, B_{8,10}, B_{10,1})$$

.

5.4.2 Algorithme complet

On dispose maintenant de tous les éléments pour résoudre notre système d'équations. L'algorithme proposé est donné en Algorithme 5.3.

Algorithme 5.3 Algorithme complet

1: déterminer les invariants (et obtenir les populations $N(\mathbf{iv})$ des invariants)
2: déterminer l'ordre de parcours de l'ensemble des dipôles à l'aide de l'algorithme 5.1
3: **Initialiser :** λ_u, λ_d, $m_u^{(n)}$ et $m_d^{(n)}$ aux valeurs des paramètres correspondant de la ligne d'origine
4: évaluer les paramètres de performance de tous les dipôles
5: évaluer les valeurs $Q(\mathbf{iv})$ pour tous les invariants **iv**, à l'aide de (5.24)
6: déduire les valeurs de $\beta(\mathbf{iv})$ pour tous les invariants **iv**
7: **répéter**
8: **pour** tout dipôle $D(i,j)$ (dans l'ordre du parcours) **faire**
9: déterminer la valeur de $\beta_u(i,j)$ à l'aide de (5.25a)
10: calculer $e_u(i,j)$ à l'aide de (5.23a)
11: déduire les nouvelles valeurs de $\lambda_u(i,j)$ et $m_u^{(n)}(i,j)$ à l'aide de (5.16) et (5.18)
12: évaluer les nouveaux paramètres de performance de $D(i,j)$
13: corriger les valeurs des $Q(\mathbf{iv})$ pour tout invariant **iv** auquel appartient $D(i,j)$
14: déduire les valeurs $\beta(\mathbf{iv})$ nécessaires
15: **fin pour**
16: **pour** tout dipôle $D(i,j)$ (dans l'ordre inverse du parcours) **faire**
17: déterminer la valeur de $\beta_d(i,j)$ à l'aide de (5.25b)
18: calculer $e_d(i,j)$ à l'aide de (5.23b)
19: déduire les nouvelles valeurs de $\lambda_d(i,j)$ et $m_d^{(n)}(i,j)$ à l'aide de (5.17) et (5.19)
20: évaluer les nouveaux paramètres de performance de $D(i,j)$
21: corriger les valeurs des $Q(\mathbf{iv})$ pour tout invariant **iv** auquel appartient $D(i,j)$
22: déduire les valeurs $\beta(\mathbf{iv})$ nécessaires
23: **fin pour**
24: **jusqu'à** convergence de tous les paramètres

Remarque sur l'algorithme Algorithme 5.3 : ligne 16 : lorsqu'on effectue le parcours en ordre inverse, il faut noter que pour respecter au mieux le graphe de dépendances, et si le premier dipôle du parcours appartient à une branche (de plus de 2 dipôles), il convient de faire un décalage dans le parcours. On commence en effet par le premier élément du parcours (tel que généré par l'algorithme Algorithme 5.1), car c'est celui qui aura été mis à jour en dernier lors du parcours en sens direct, puis on poursuit par le dernier, l'avant dernier, etc.

5.4.3 Choix de la fonction d'agrégation des coefficients de convergence

Cette fonction $f_{ag}(\beta(iv_1(B_{i,j})), \ldots)$, introduite Section 5.3.2, doit être strictement monotone croissante (pour chacun de ses paramètres β) et normalisée $f_{ag}(1, 1, ..., 1) = 1.0$.

Dans l'implémentation que nous avons faite de ces algorithmes, nous avons introduit un certain nombre de garde-fous numériques, afin d'éviter de se trouver avec des paramètres ayant ponctuellement des valeurs impossibles, en début de convergence. C'est en particulier le cas des paramètres β introduits. Nous avons ainsi (arbitrairement) forcé ces paramètres à rester dans une fourchette donnée (en l'occurrence $\beta \in [0.8, 1.2]$). Cela peut légèrement nuire à la vitesse de convergence dans certains cas simples, mais assure une très grande robustesse de l'algorithme.

Mais cela induit aussi une relative indépendance quant au choix de la fonction d'agrégation, puisque au voisinage de 1, toutes ces fonctions sont assez proches, par définition. Finalement, le choix de cette fonction est sans conséquence sur la rapidité de convergence de l'algorithme tel qu'il a été implémenté.

Remarque 14 : une astuce qui a toujours légèrement amélioré la vitesse de convergence de l'algorithme, c'est de poser $\beta = 1.0$ pour tout dipôle qui appartient à tous les invariants de la ligne, dans la mesure où il y en a (ce qui est très généralement le cas des lignes réelles). Cette astuce permet d'améliorer la vitesse de convergence de l'ordre de 5 à 10%. Notons tout de même que cela n'est pas indispensable.

6

Résultats numériques

Afin de tester la justesse de la méthode présentée, on propose un certain nombre de tests, où l'on comparera notre méthode à des simulations numériques à évènements discrets. On ne fera ici que des tests quantitatifs sur les paramètres de performance recherchés, et non sur les performances informatiques de la méthode, qui ont été en détails analysées à la fin du Chapitre 4.

Sommaire

6.1 Le simulateur

Pour toutes les simulations, nous avons utilisé un simulateur à événements discrets que nous avons développé. Il se fonde sur deux projets :

- OMNeT++ [36, 33, 35] : OMNeT++ est un simulateur à événements discrets modulaire et orienté objet, écrit en C++. Il s'agit en fait de bibliothèques et d'un environnement d'exécution (graphique ou en mode texte). On réalise un simulateur en écrivant ses propres classes en C++. Celles-ci dérivent des classes définies dans les bibliothèques fournies, et on définit leur comportement à la réception de messages (événements) en surchargeant la méthode `handleMessage`.
- Akaroa2 [31, 32] : Akaroa2 est un environnement d'exécution de simulations. Il s'agit en fait de l'implémentation d'une approche différente pour les simulations que la réplication parallèle multiple (MRIP, Multiple Replication in Parallel) ; au lieu de diviser la simulation

en "tranches" sur lesquelles on va faire des mesures, et de les considérer comme autant de réplications pour obtenir des résultats statistiques, plusieurs instances du programme sont exécutées sur plusieurs machines, avec un système qui centralise les résultats de chaque instance en cours. C'est ce centre qui décide ensuite, à quel moment il convient d'arrêter la simulation, selon la précision souhaitée.

Pour nos lignes, nous avons modifié ces deux systèmes afin de les faire coopérer selon nos besoins.

6.1.1 OMNeT++

OMNeT++ est un environnement pour implémenter et exécuter des simulateurs à événements discrets. Son domaine d'application privilégié est la simulation de réseaux de communication. Cependant, grâce à son architecture flexible et générique, il est aisé de l'utiliser dans d'autres domaines, comme les réseaux de files d'attentes, les systèmes de production, etc.

OMNeT++ propose une architecture à composants pour les modèles à simuler. Ces composants (appelés modules) sont écrits en C++, puis assemblés dans des composants et des modèles plus grands à l'aide d'un langage de haut niveau (nommé NED).

OMNeT++ a aussi une interface utilisateur graphique, permettant une mise au point aisée des simulateurs. Celle-ci peut cependant ne pas être utilisée, pour plus d'efficacité. Enfin, grâce à son architecture modulaire, le noyau du simulateur peut facilement être intégré à d'autres applications.

C'est un projet développé par l'Université de Karlsruhe, qui est disponible gratuitement pour un usage non commercial, et dont le code source est intégralement disponible. Il peut être utilisé sur quasiment toutes les plates-formes existantes.

OMNeT++ est composé de :

- un noyau de simulation (bibliothèque),
- un compilateur pour le langage de description de modules NED,
- un éditeur graphique pour les fichiers NED,
- une interface utilisateur graphique pour l'exécution des simulations,
- une interface utilisateur en mode textuel pour l'exécution des simulations ,
- divers outils (générateurs de graines pour le générateur de nombre aléatoires, etc.), et une documentation complète.

6.1.2 Akaroa2

Akaroa est un projet du Department of Computer Science, de l'Université de Canterbury, en Nouvelle-Zélande [31]. De manière similaire à OMNeT++, il est disponible gratuitement pour un usage non commercial, avec l'intégralité du code source.

C'est un système de parallélisation cohérente de simulations. L'approche utilisée par Akaroa pour paralléliser les simulations est appelée multiple réplications in parallel (MRIP) [19]. Le principe est d'écrire un simulateur, en utilisant les bibliothèques de Akaroa. Ensuite, un processus central gère la répartition des exécutions des simulations sur les machines disponibles sur le réseau. Il peut aussi fournir aux instances de simulation qu'il contrôle les nombres aléatoires dont a besoin chaque simulateur (de façon à décorréler au maximum les différentes trajectoires, en s'assurant que les graines des générateurs de nombres pseudo-aléatoires sont assez distantes pour chacune des instances de la simulation). Enfin, il s'occupe de récolter les informations de ces différentes instances, afin d'estimer à quel moment il est possible de stopper les simulations, les données observées ayant atteint une précision suffisante.

C'est donc Akaroa qui lance automatiquement un certain nombre d'instances du programme de simulation ; chaque copie est nommée *instance de simulation* et représente une trajectoire du modèle simulé. Chacune de ces instances s'exécute indépendemment des autres, et génère sa propre série d'observations, qui sert à faire des estimations locales du paramètre observé. Akaroa collecte ces estimations locales lorsqu'elles sont produites, et calcule une estimation globale de chacun des paramètres observés.

Il suffit de donner la précision et l'intervalle de confiance désirés pour chaque paramètre. Lorsque les estimations globales des paramètres atteignent la précision suffisante pour chacun des paramètres observés, Akaroa stoppe l'exécution des instances de simulation, puis affiche les résultats globaux. Akaroa propose aussi un mécanisme de détection automatique de fin de période transitoire.

Afin d'avoir une simulation globale de qualité suffisante, le simulateur doit utiliser les générateurs de nombres aléatoires fournis par Akaroa, de façon à éviter autant que possible les corrélations entre les instances de simulations.

6.1.3 Modifications

Il est possible d'utiliser directement OMNeT++ dans Akaroa, mais alors un certain nombre des avantages de Akaroa sont perdus :

- détection automatique de fin de période transitoire ;
- possibilité d'utiliser Akaroa comme détecteur de fin de simulation ;
- possibilité d'utiliser Akaroa comme estimateur pour les paramètres observés.

Pour disposer pleinement des avantage de Akaroa avec OMNeT++, il a fallu modifier un certain nombre d'éléments centraux de ces deux environnements. D'une part, faire en sorte que OMNeT++ utilise Akaroa comme estimateur des paramètres étudiés (aussi bien pour la détection de fin de période transitoire que pour la détection de fin de simulation). D'autre part, il a fallu modifier Akaroa de sorte que :

- on puisse observer un paramètre, sans que celui-ci ne soit utilisé pour déterminer la fin de la période transitoire ;
- on puisse observer un paramètre sans que celui-ci ne soit utilisé pour déterminer le moment de la fin de la simulation.

Le détail de ces modifications est donné en Annexe D.

Au final, nous disposons d'un simulateur capable de distribuer ses calculs, et proposant des résultats de la précision voulue sur les paramètres choisis. Nous avons à chaque fois fait des simulations avec les paramètres (utilisés pour déterminer la précision) suivants :

- probabilité de fonctionnement de chaque machine : 5%, intervalle de confiance à 95% ;
- probabilité de panne : 5%, intervalle de confiance à 95% ;
- probabilité de non-alimentation (s'il y a lieu) : 8%, intervalle de confiance à 90% ;
- probabilité de blocage (s'il y a lieu) : 8%, intervalle de confiance à 90%.

Les résultats présentés sont de deux sortes. D'une part on propose d'observer la productivité (par rapport à la simulation) pour différentes valeurs du nombre de clients circulant dans la ligne. D'autre part, on propose, pour une valeur donnée du nombre de clients circulant dans la ligne, une représentation de la ligne synthétisant tous les résultats (taux de production, probabilités de non-alimentation et de blocage, en-cours moyens, comparés à la simulation). Chaque machine est représentée par un carré divisé en quatre, pour visualiser les erreurs des probabilités de fonctionnement, de panne, de non-alimentation et de blocage (par une échelle de couleur, en %). Chaque stock est représenté par un cercle, dont la couleur du remplissage donne aussi l'erreur d'estimation de l'encours moyen par rapport à la simulation. De plus les valeurs numériques

TAB. 6.1 – Paramètres utilisés pour le premier exemple de ligne.

i	1	2	3	4	5	6	7
λ_i	0.01	0.01	0.05	0.005	0.02	0.003	0.006
μ_i (v1)	0.1	0.06	0.2	0.06	0.1	0.04	0.1
e_i (v1, en %)	90.9	85.7	80.0	92.3	83.3	93.0	94.3
μ_i (v2)	0.05	0.03	0.1	0.03	0.05	0.02	0.05
e_i (v2, en %)	83.3	75.0	66.6	86.7	71.4	86.9	89.3
$C_{i,i+1}$	25	20	15	40	50	40	20

apparaissent dans les stocks (en-cours moyen calculé par la méthode analytique et erreur par rapport à la simulation). Enfin, l'erreur moyenne de probabilité de fonctionnement (donc de la productivité) ainsi que l'erreur quadratique moyenne des en-cours moyens estimés sont notées en bas du schéma.

Remarque 15 : l'erreur des en-cours moyens est calculée par rapport à la capacité totale du stock : $err = \frac{h_m^{analyse} - h_m^{simulation}}{C}$.

6.2 Boucles simples

Nous avons fait un certain nombre de tests sur des boucles simples. En particulier, nous avons repris certains exemples proposés dans [20], afin d'y observer le comportement quantitatif de nos méthodes selon les distributions des mécanismes de réparation utilisées.

6.2.1 Exemple symétrique

Nous analysons une ligne comportant $K = 9$ machines, toutes identiques :
- $\lambda = 0.001$;
- $\mu = 0.1$;
- $C = 5$

La capacité totale de la ligne est donc 54.

Les taux de production estimés par les méthodes analytiques sont comparés à la simulation Figure 6.1 page suivante. On constate que les trois méthodes sont identiques, ce qui est normal, car la ligne étant symétrique, tous les temps de réparation sont les mêmes, donc les trois méthodes sont rigoureusement identiques dans ce cas. L'estimation est correcte, puisque l'erreur est inférieure à 3% (on sous-évalue le taux de production).

6.2.2 Exemple non symétrique

Il s'agit d'une ligne comportant $K = 7$ machines. On en présente deux versions (les temps moyens de réparation sont doublés dans la seconde version). Les paramètres des machines et des stocks sont donnés dans le tableau Table 6.1.

Les résultats sont très corrects, puisque les erreurs sur les taux de production sont, pour des valeurs moyennes du nombre de clients, inférieures à 3%. On constate que les résultats avec des caractérisations exponentielles des temps de réparation des dipôles ont tendance à surévaluer les taux de production, tandis que les deux autres caractérisations sous-évaluent ces taux de

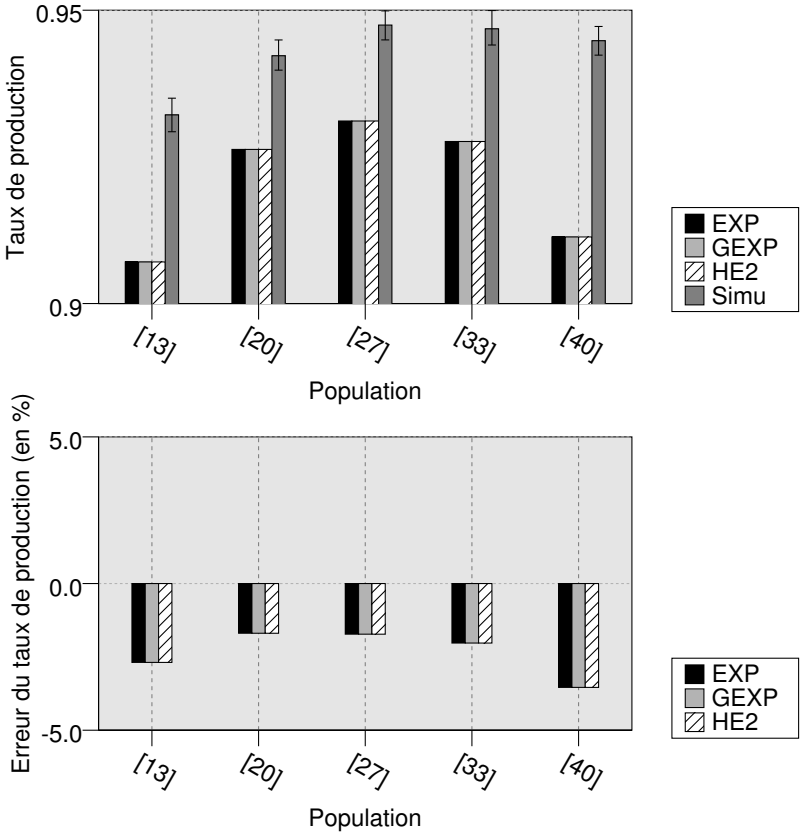

FIG. 6.1 – Comparaison de la productivité pour une symétrique de $K = 9$ machines.

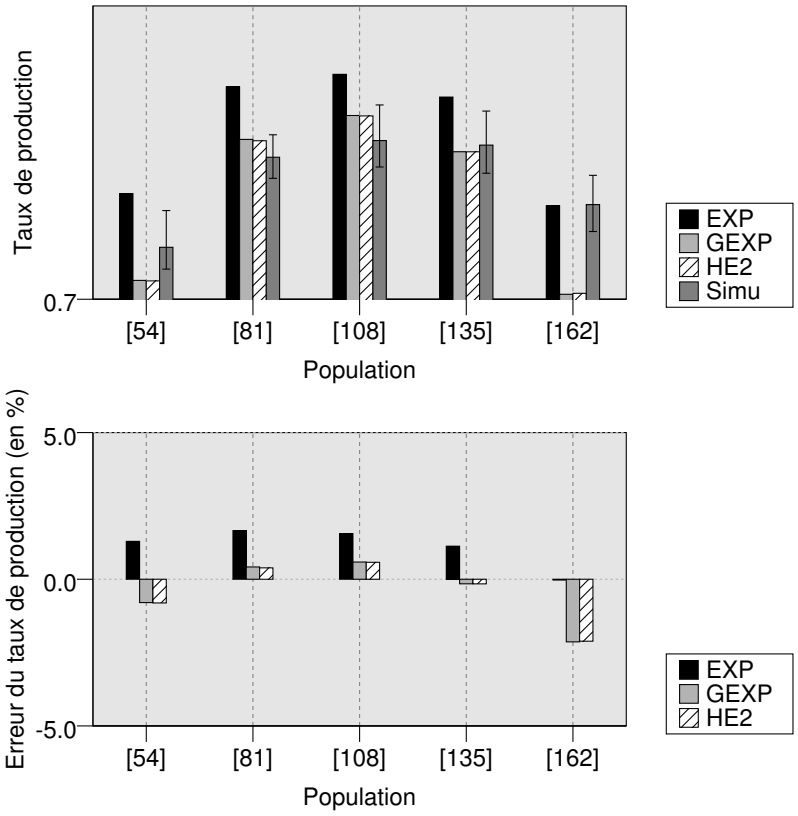

FIG. 6.2 – Comparaison de la productivité de la ligne donnée au tableau Table 6.1 page 102, première version.

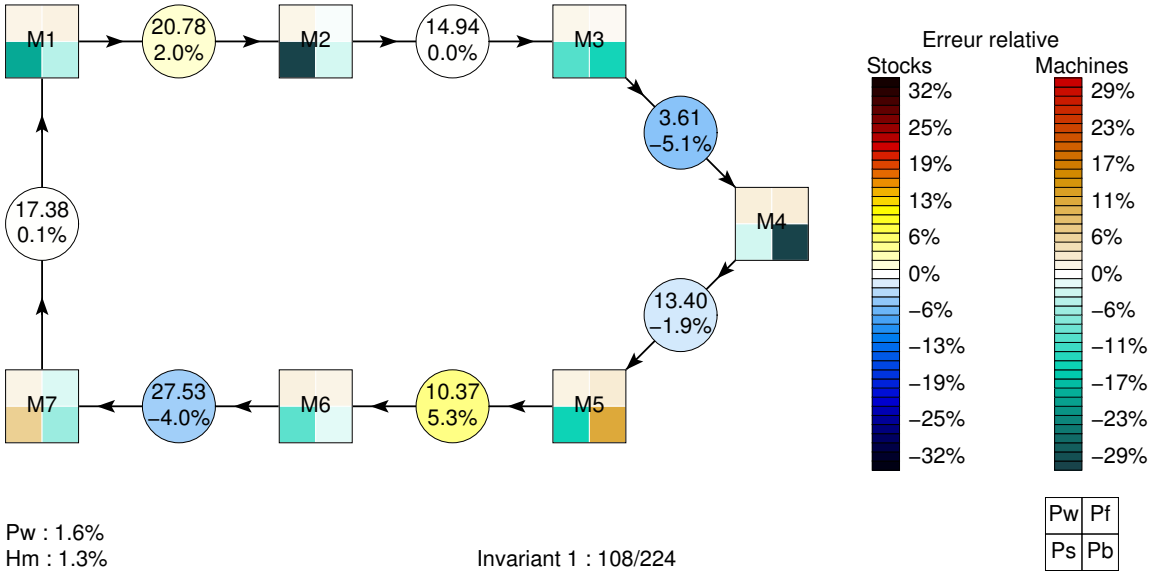

FIG. 6.3 – Synthèse comparative pour le premier exemple (version 1) avec $N = 108$ clients dans la boucle, avec des distributions exponentielles des temps de réparation.

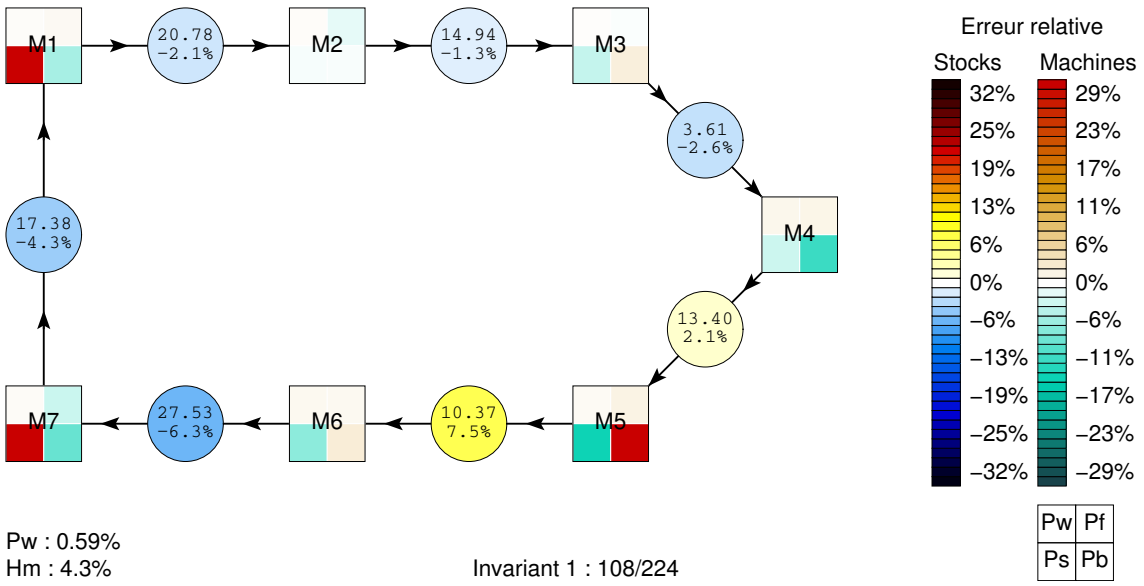

FIG. 6.4 – Synthèse comparative pour le premier exemple (version 1) avec $N = 108$ clients dans la boucle, avec des distributions générales exponentielles des temps de réparation.

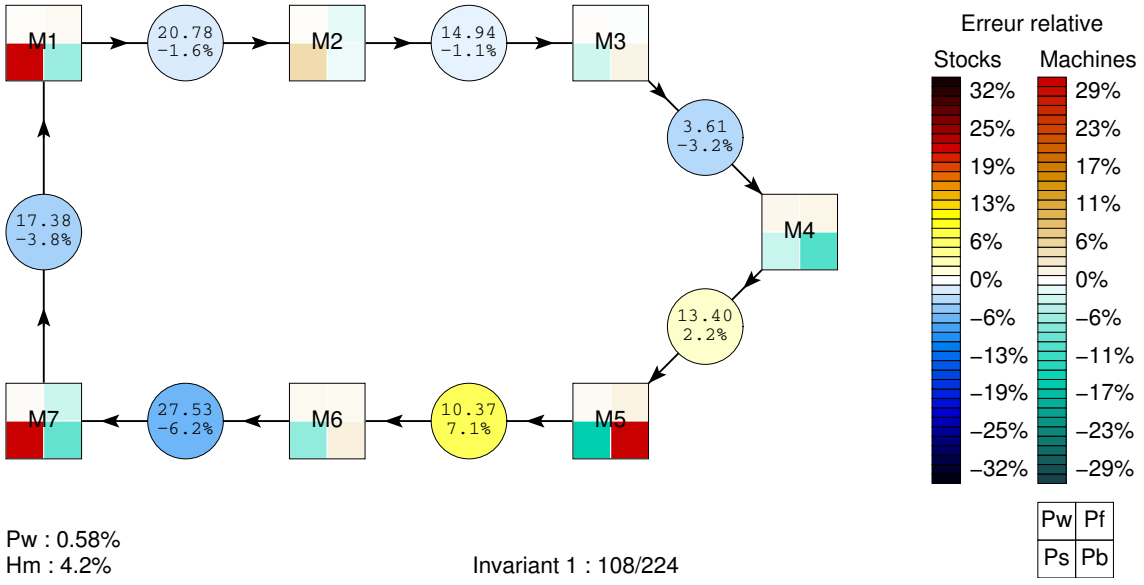

FIG. 6.5 – Synthèse comparative pour le premier exemple (version 1) avec $N = 108$ clients dans la boucle, avec des distributions HE_2 des temps de réparation.

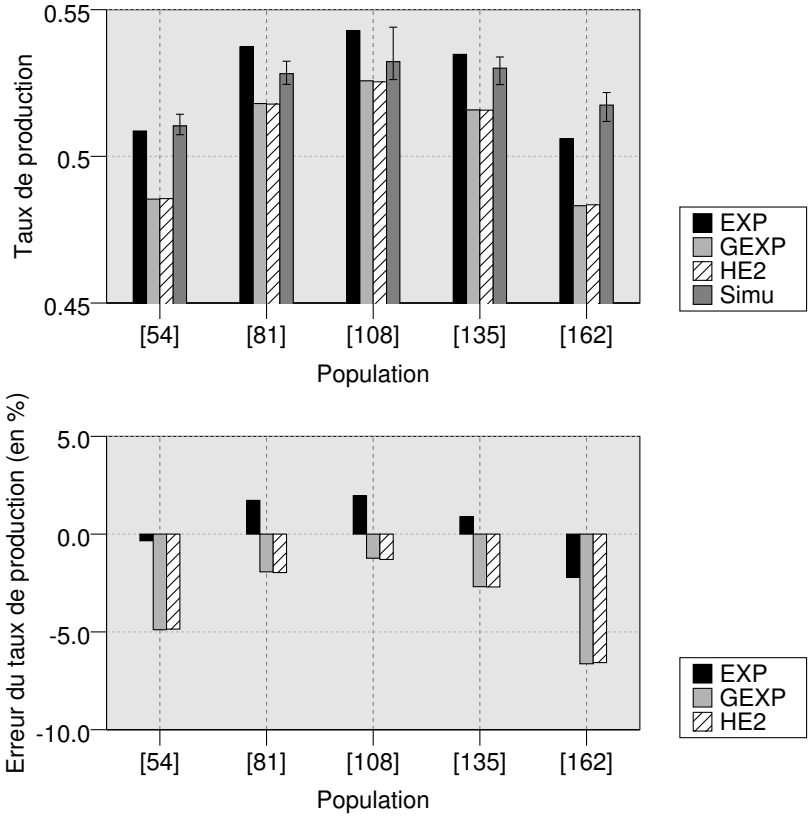

FIG. 6.6 – Comparaison de la productivité de la ligne donnée au tableau Table 6.1 page 102 (version 2).

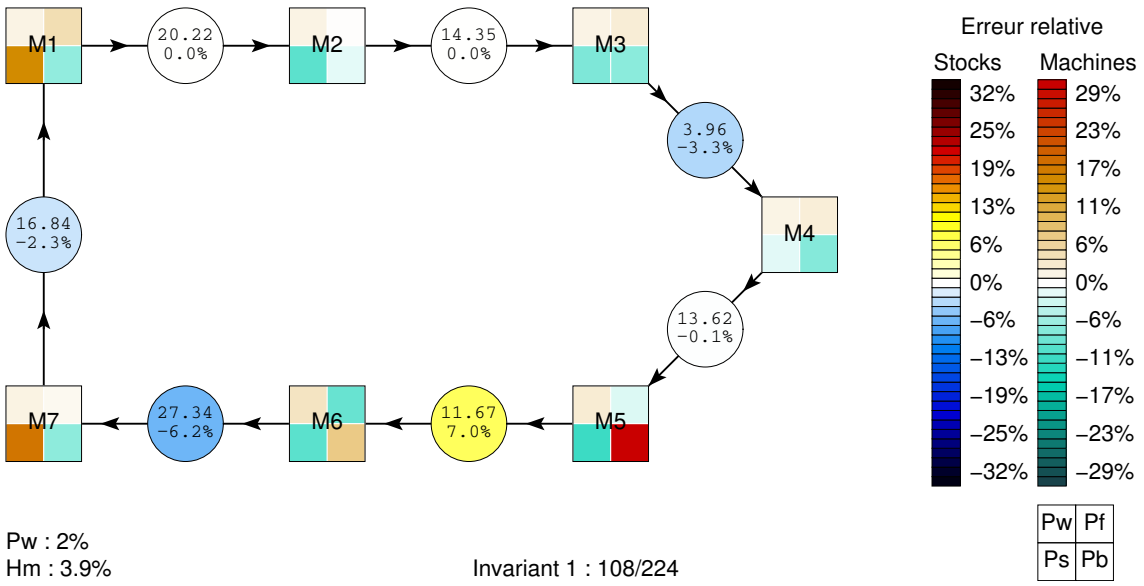

FIG. 6.7 – Synthèse comparative pour le premier exemple (version 2) avec $N = 108$ clients dans la boucle, avec des distributions exponentielles des temps de réparation.

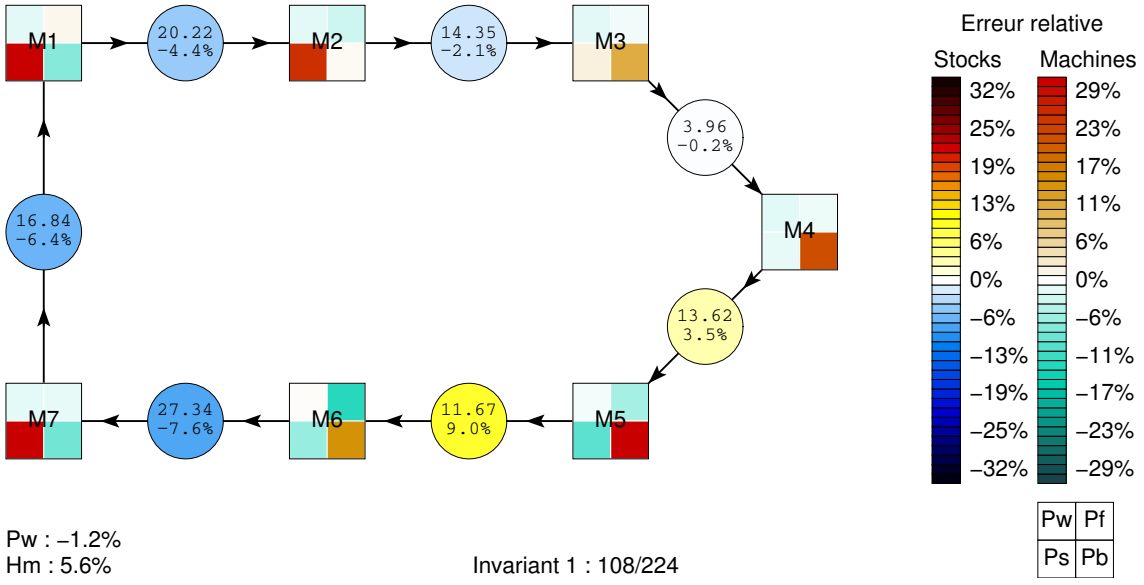

FIG. 6.8 – Synthèse comparative pour le premier exemple (version 2) avec $N = 108$ clients dans la boucle, avec des distributions générales exponentielles des temps de réparation.

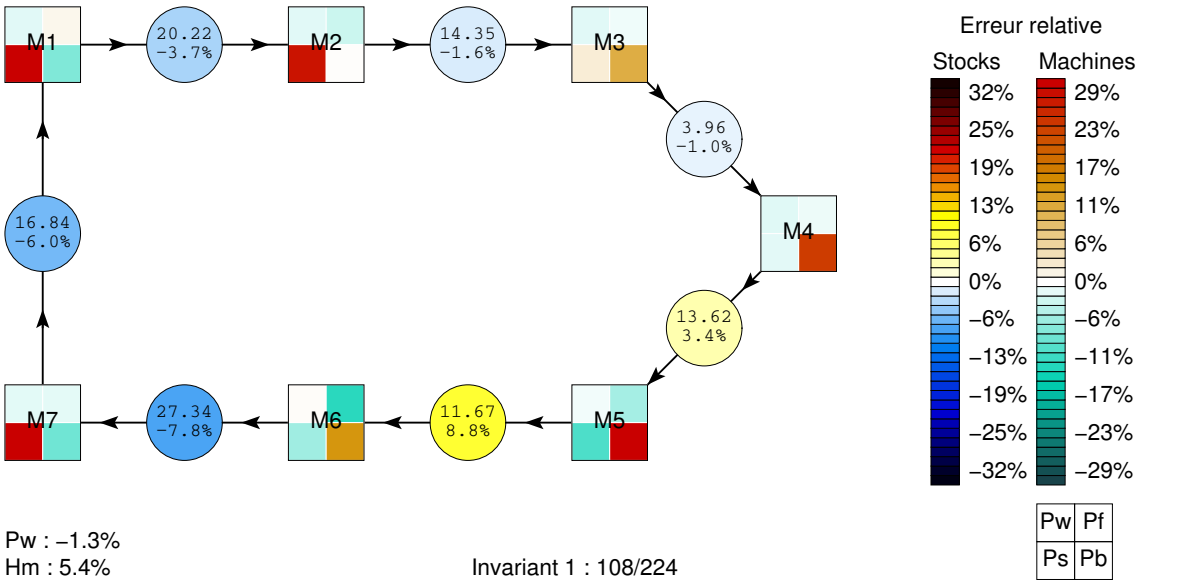

FIG. 6.9 – Synthèse comparative pour le premier exemple (version 2) avec $N = 108$ clients dans la boucle, avec des distributions HE_2 des temps de réparation.

production. De plus, pour les valeurs moyennes de N (autour de la moitié de la capacité de la ligne), les caractérisations GE ou HE_2 sont très proches et donnent des résultats sensiblement meilleurs que en exponentiel.

Si l'on s'intéresse aux estimations des en-cours moyens, c'est par contre la méthode avec des caractérisations exponentielles qui donne les meilleurs résultats.

6.3 Lignes à plusieurs boucles

6.3.1 Exemples de double-boucles symétriques

Nous proposons quelques exemples de lignes composées d'une branche principale qui se scinde en deux, puis se réassemblent. Cette topologie comprend donc deux invariants (deux boucles).

Cette topologie est utilisée dans l'une des unités de production du constructeur automobile français PSA.

On propose d'étudier le cas d'une ligne à double boucle, comportant $K = 12$ machines, dont le schéma est donné Figure 6.10 page suivante. Toutes les machines, et tous les stocks ont les mêmes paramètres, et on propose deux versions de cette ligne :

- $lambda = 0.001$
- $\mu = 0.1$
- $C = 10, 5$

Seuls sont présentés les résultats de la méthode de décomposition avec des distributions exponentielles des temps de réparation, car étant donné que la ligne originale est symétrique, tous les temps moyens de réparation sont les mêmes. Donc toutes les méthodes proposées sont strictement équivalentes. En effet, si l'on observe les équations 3.49c et 3.51c, et si tous les paramètres $m_u^{(n)}(i,j)$ (et $m_d^{(n)}(i,j)$) sont égaux, alors elles se simplifient toujours à la valeur initiale $m_i = \frac{1}{\mu_i}$ (ou $m_j = 1\mu_j$).

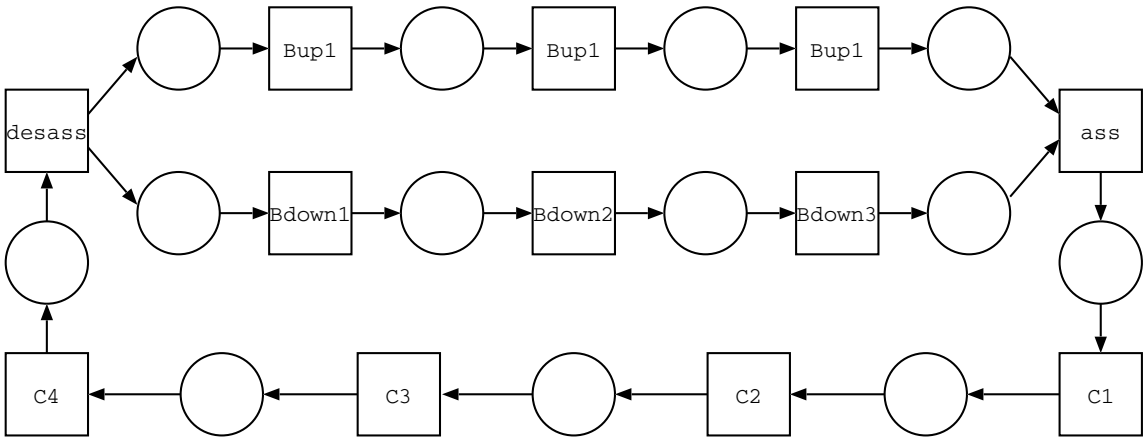

FIG. 6.10 – Premier exemple de ligne à double boucle.

Les résultats des productivités en fonction du nombre de clients dans chacun des deux invariants sont donnés Figure 6.11 page suivante et Figure 6.12 page suivante.

On donne aussi un exemple d'une comparaison de tous les paramètres de performance pour les valeurs des nombres de clients dans chacun des deux invariants de 24 et 37, Figure 6.13 page 111 et Figure 6.14 page 111.

Les résultats sont corrects pour la ligne dont les stocks sont plus importants (le taux de production, pour des valeurs moyennes des invariants, est souvent inférieur à 2 ou 3%). Par contre, ils sont moins bons quand les stocks sont plus petits ($C = 5$). Dans cette configuration, on sous-évalue nettement la productivité de la ligne (entre 5 et 10% d'erreur). On s'attendait à ce que les résultats soient meilleurs dans le cas de lignes dont les stocks sont plus importants, car on sait que ces méthodes fonctionnent par décorrélation le fonctionnement global de la ligne en chacun de ses stocks (par les dipôles). Or lorsque les stocks sont plus petits, les corrélations (négligées par la décomposition) se font plus importantes.

6.3.2 Exemples asymétriques

Les exemples suivants sont tirés de [28].

Exemple d'un système à deux boucles

Le système est représenté Figure 6.15 page 112. Les paramètres de la première version étudiée pour cette ligne sont donnés Table 6.2 page 111. Une synthèse des résultats comparés de la productivité sont disponibles Figure 6.16 page 112.

On constate que l'estimation des taux de production de la ligne est acceptable lorsque l'on se cantonne aux valeurs des invariants qui n'impliquent pas un déséquilibre trop important entre leurs niveaux de remplissage. A nouveau, cela se conçoit bien parce que nos méthodes sont incapables de rendre compte des phénomènes de corrélations liés à ces situations de fort déséquilibre entre les invariants. On constate aussi que toutes les méthodes proposées ont tendance à sous-évaluer les taux de production. C'est donc la méthode la plus simple (E) qui donne les meilleurs résultats.

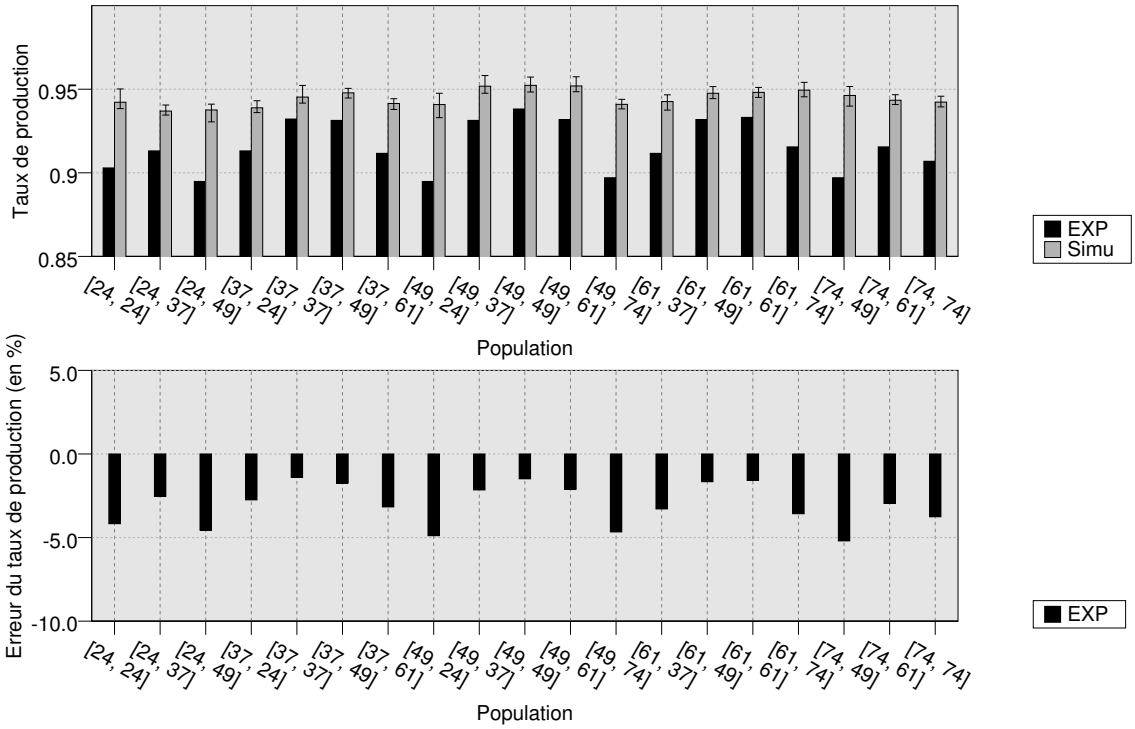

FIG. 6.11 – Productivité de la ligne pour $C = 10$.

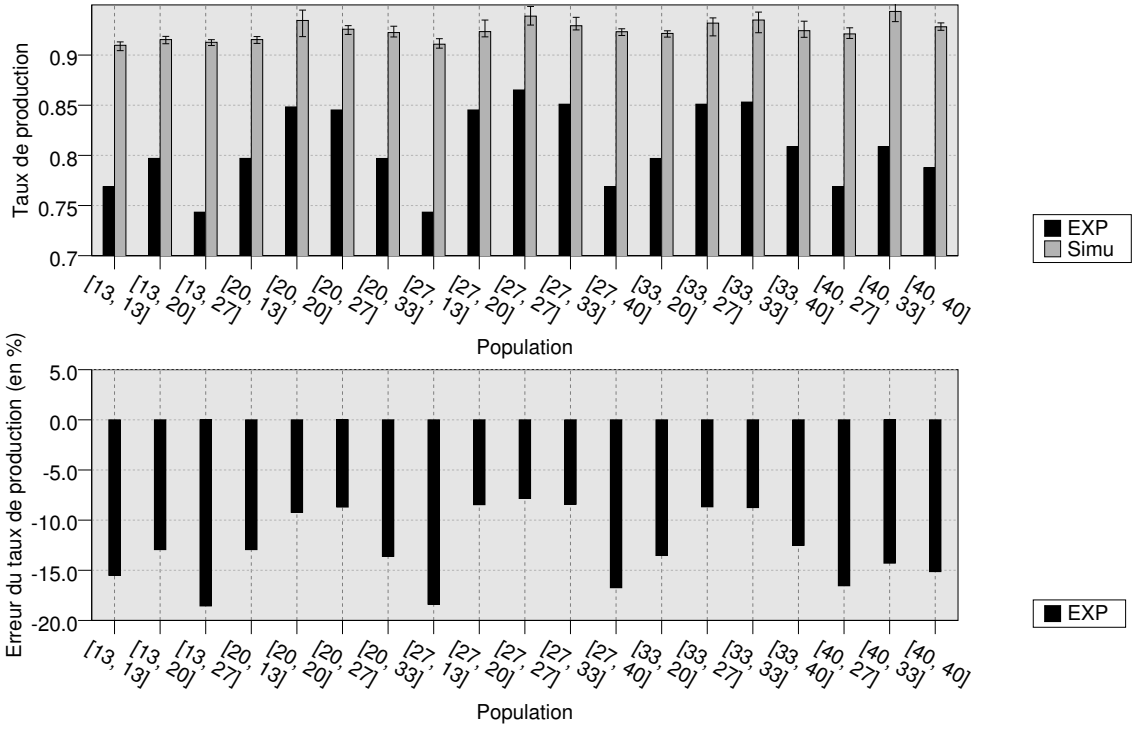

FIG. 6.12 – Productivité de la ligne pour $C = 5$.

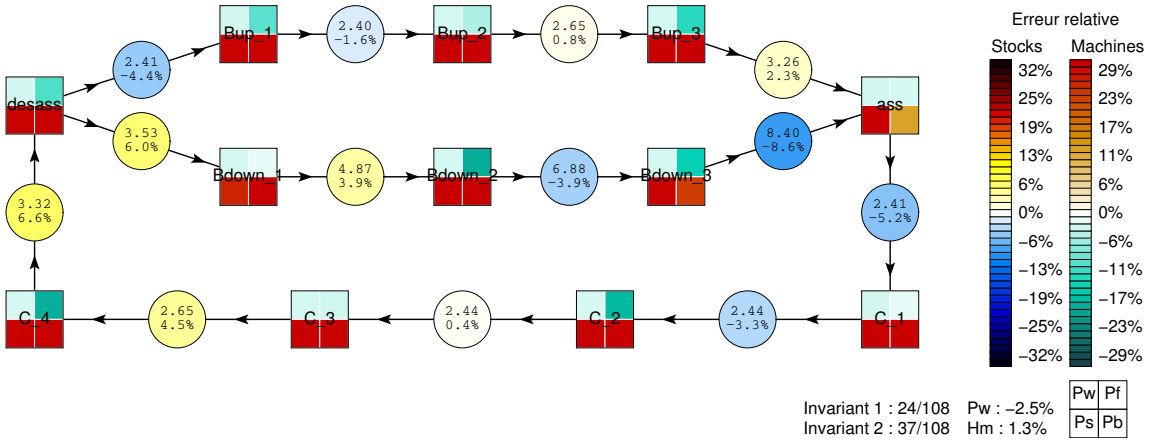

FIG. 6.13 – Comparaison de la méthode analytique pour la ligne présentée en Section 6.3.1 pour $C = 10$.

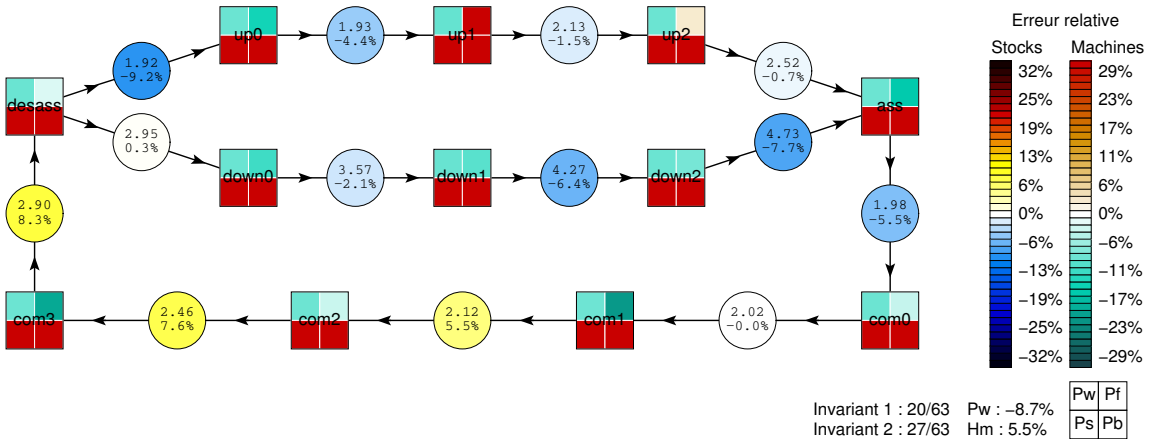

FIG. 6.14 – Comparaison de la méthode analytique pour la ligne présentée en Section 6.3.1 pour $C = 5$.

TAB. 6.2 – Paramètres utilisés pour l'exemple de ligne donné en Figure 6.15 page suivante.

i	1	2	3	4	5	6	7	8	9
λ_i	0.0046	0.0236	0.0184	0.0289	0.0107	0.0277	0.0450	0.0497	
μ_i	0.0568	0.0719	0.1090	0.1549	0.0544	0.1129	0.1493	0.1531	
e_i (en %)	92.5	75.3	85.5	84.3	83.6	80.3	76.4	75.5	
C_i	28	53	30	21	79	39	31	6	18

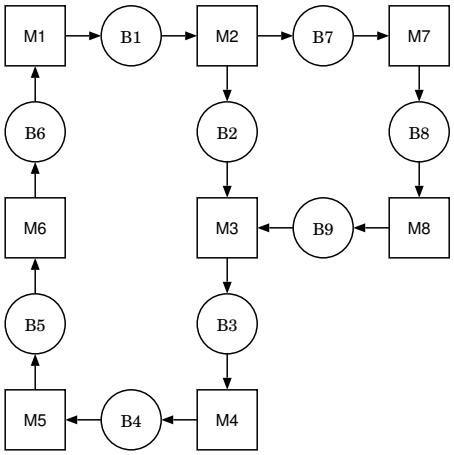

FIG. 6.15 – Exemple de ligne à deux boucles, tiré de [28] (2-Loop A/D System, Configuration I, Test Case 9, p.144).

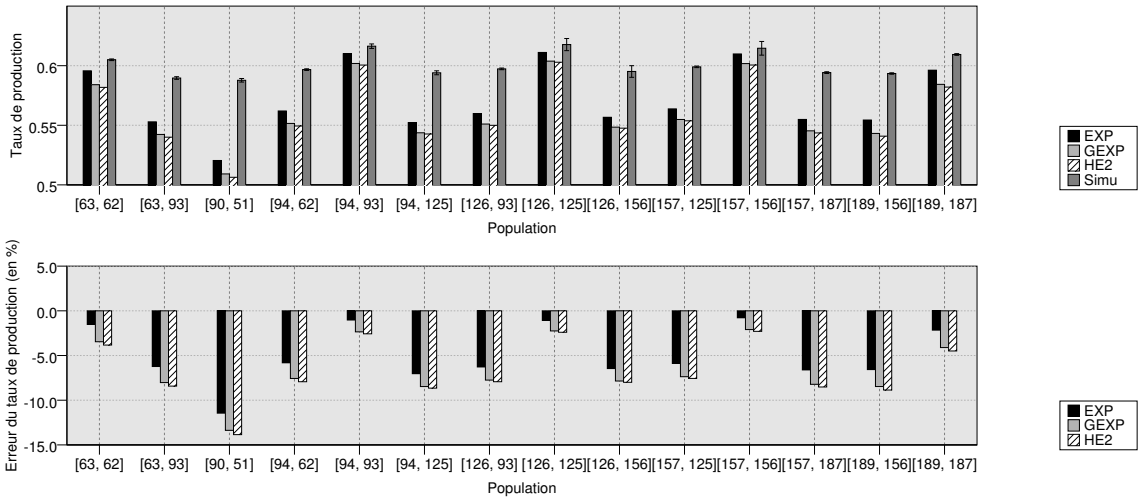

FIG. 6.16 – Résultats comparés de la productivité de la ligne donnée Figure 6.15.

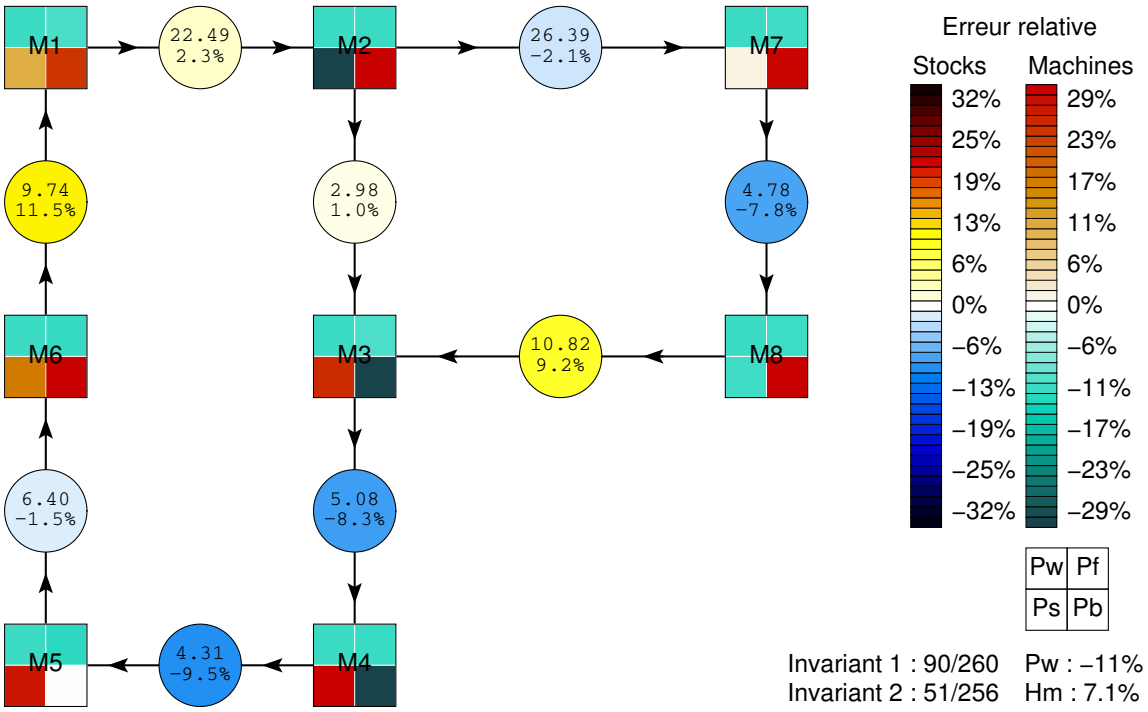

FIG. 6.17 – Résultats synthétiques avec des distributions exponentielles de la ligne donnée Figure 6.15 page ci-contre.

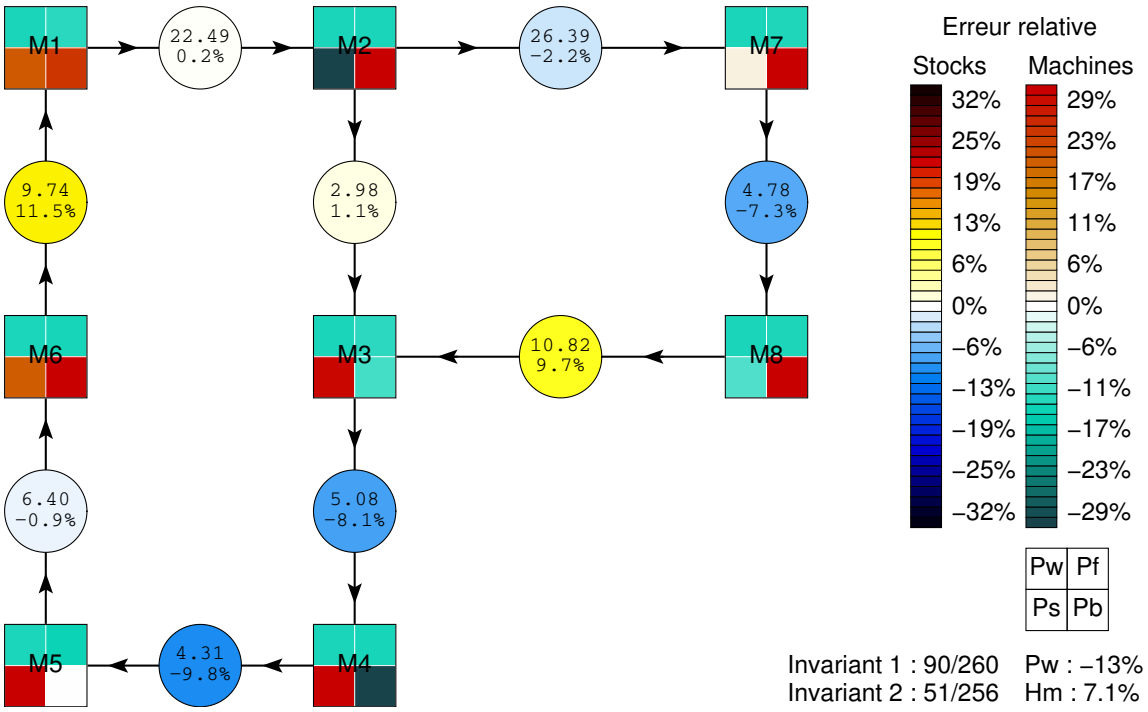

FIG. 6.18 – Résultats synthétiques avec des distributions exponentielles de la ligne donnée Figure 6.15 page ci-contre.

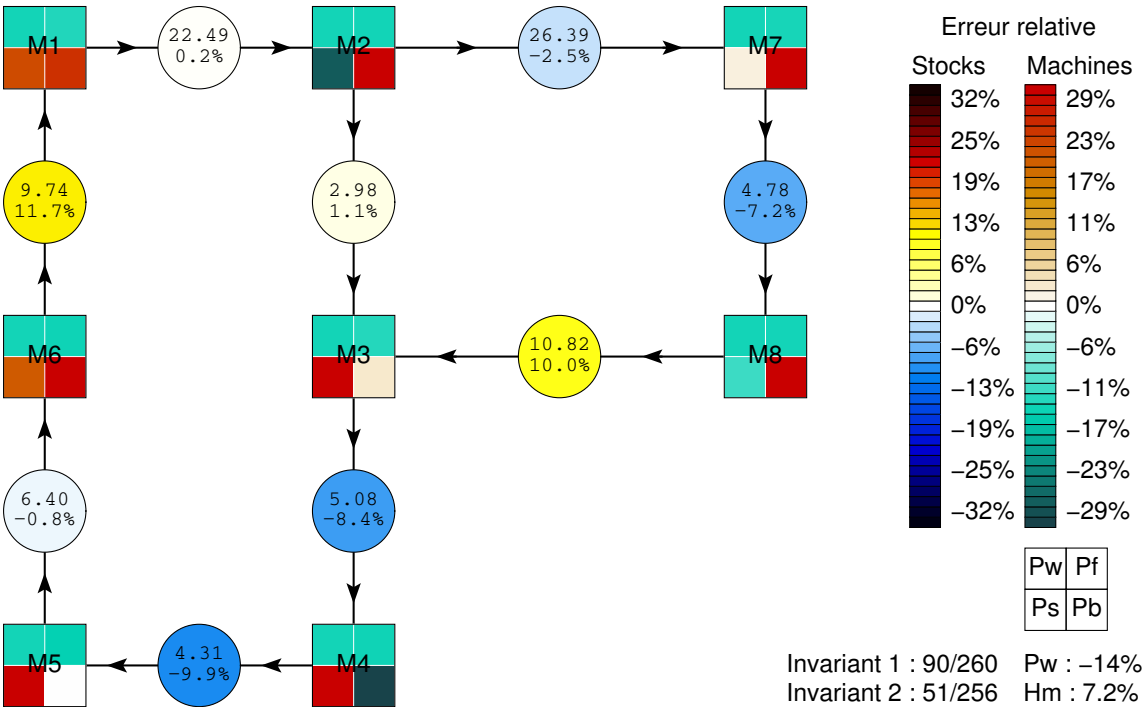

FIG. 6.19 – Résultats synthétiques avec des distributions exponentielles de la ligne donnée Figure 6.15 page 112.

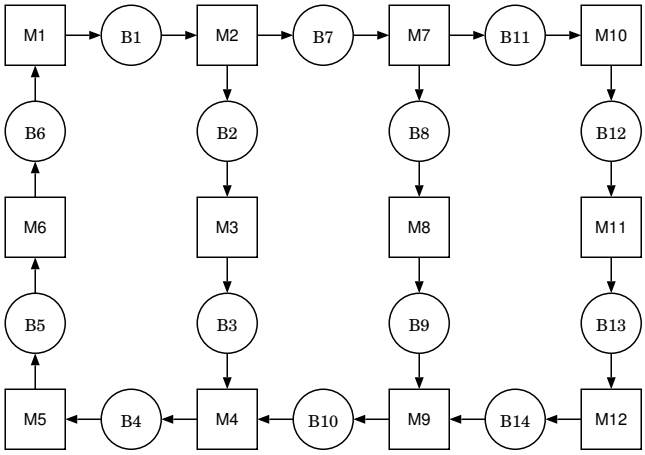

FIG. 6.20 – Exemple de ligne à trois boucles.

TAB. 6.3 – Paramètres des machines et des stocks de l'exemple de ligne donné en Figure 6.20.

i	1	2	3	4	5	6	7
λ_i	0.0023	0.0047	0.0196	0.0154	0.0314	0.0305	0.0180
μ_i	0.1475	0.1228	0.0677	0.1136	0.1994	0.1993	0.1258
e_i (en %)	98.5	96.3	77.5	88.1	86.4	86.7	87.5
C_i	28	21	44	44	26	16	8
i	8	9	10	11	12	13	14
λ_i	0.0493	0.0196	0.0154	0.0314	0.010		
μ_i	0.1613	0.0677	0.1136	0.1994	0.10		
e_i (en %)	76.6	77.5	88.1	86.4	90.0		
C_i	22	54	34	61	46	9	10

Les résultats détaillés, correspondant à l'exemple décrit dans [28], sont synthétisés Figure 6.17 à Figure 6.19 page précédente.

Exemple d'un système à trois boucles

On observe maintenant un exemple de ligne comportant trois boucles. Les paramètres de la ligne sont donnés Table 6.3. La comparaison des taux de production avec la simulation est synthétisée Figure 6.21 page suivante. Enfin, un cas particulier est détaillé figures 6.22 à 6.24

Encore une fois, pour les valeurs moyennes des nombres de clients dans les invariants, les résultats sont très corrects pour une ligne assez complexe et dont les boucles sont assez petites. En effet, les erreurs des estimations des taux de production sont en général inférieures à 5%. A nouveau, on sous-estime toujours les taux de production, c'est donc la méthode la plus simple qui donne les meilleurs résultats.

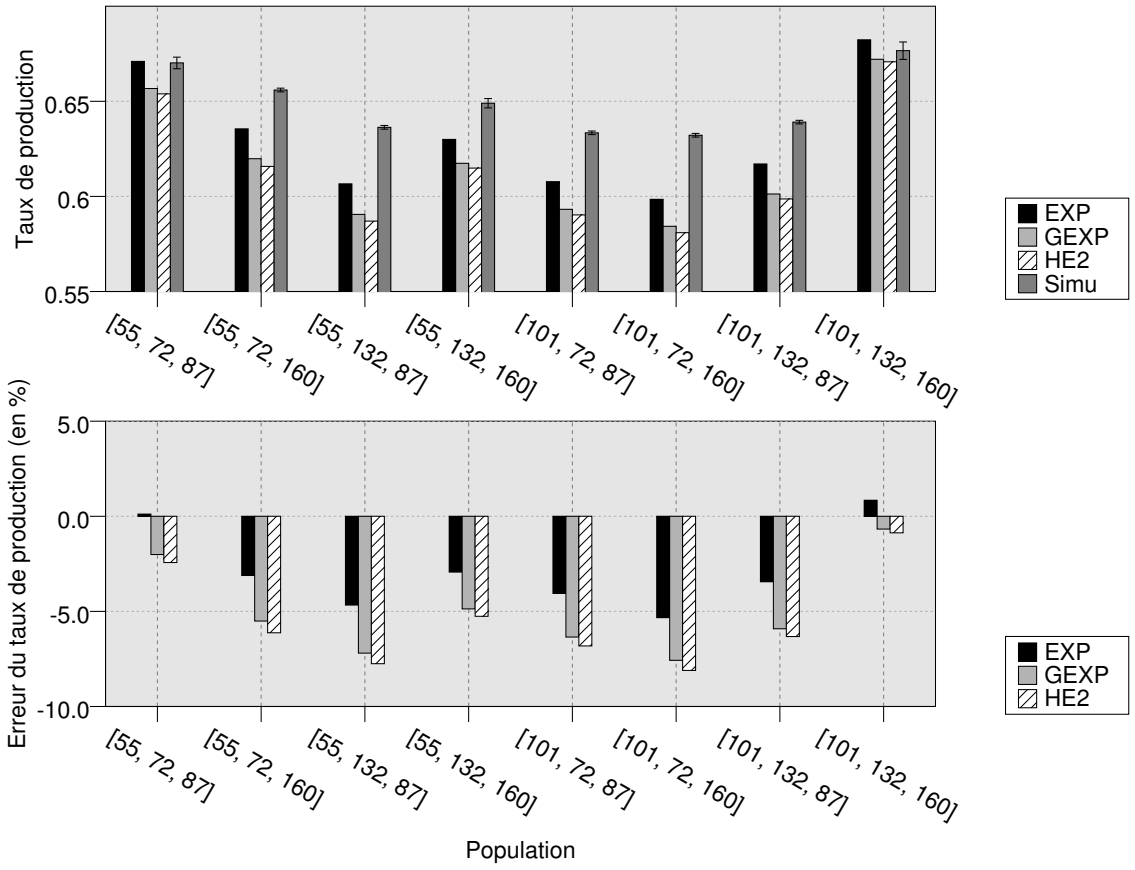

FIG. 6.21 – Résultats comparés de la productivité de la ligne donnée Figure 6.20 page 115.

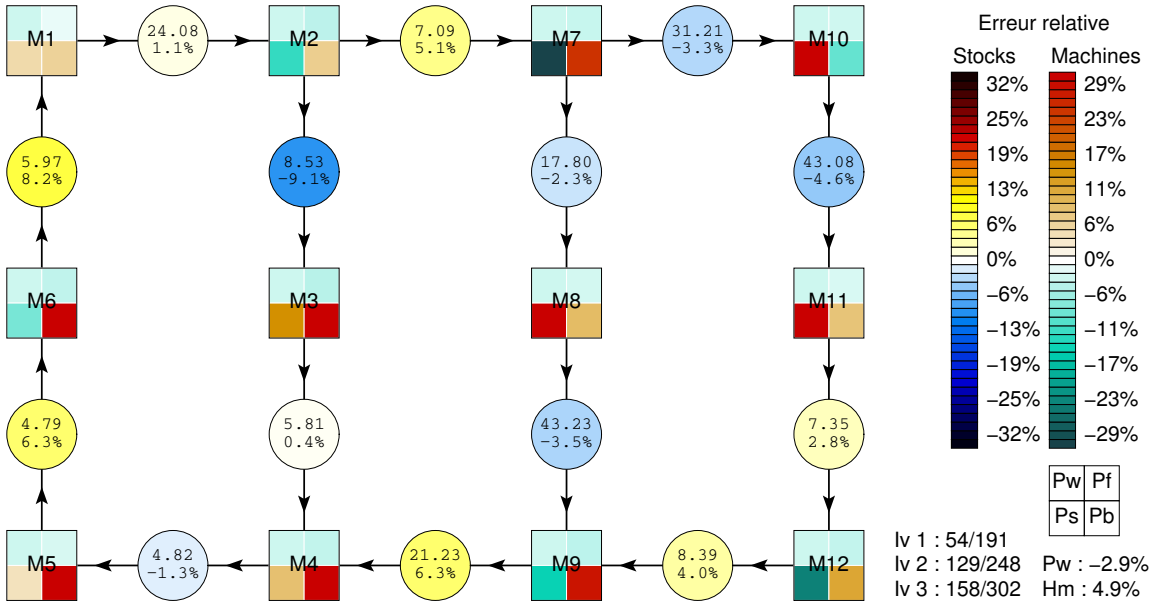

FIG. 6.22 – Résultats synthétique avec des distributions exponentielles de la ligne donnée Figure 6.20 page 115.

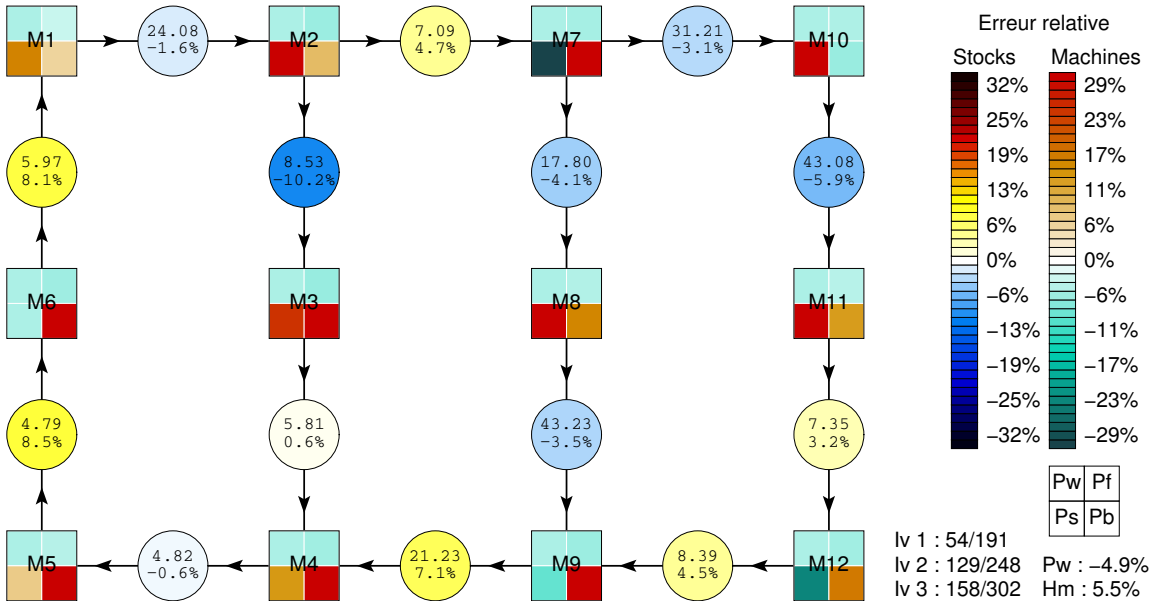

FIG. 6.23 – Résultats synthétique avec des distributions exponentielles de la ligne donnée Figure 6.20 page 115.

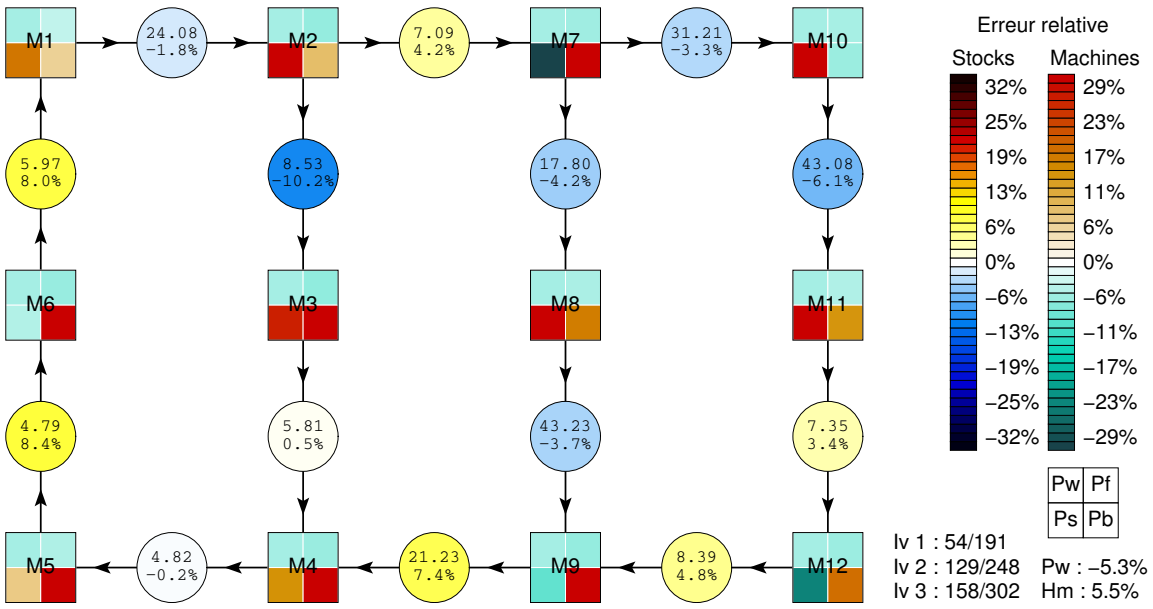

FIG. 6.24 – Résultats synthétique avec des distributions exponentielles de la ligne donnée Figure 6.20 page 115.

6.4 Synthèse des résultats

Nous n'avons présenté que quelques exemples, mais sur l'ensemble des tests que nous avons été amenés à faire, les résultats observés sont similaires.

Dans presque tous les cas (sauf pour quelques exemples de lignes simples comme la ligne présentée Section 6.2.2) les résultats obtenus en exponentiels sont meilleurs qu'avec les deux autres méthodes. D'une manière générale, les estimations des taux de production des lignes par la méthode de décomposition avec des temps de réparation exponentiels sont convenables. En effet on obtient des résultats avec souvent moins de 2% d'erreur pour un nombre de clients se situant autour de la zone optimale (environ la moitié de la capacité de chaque invariant de la ligne). Les résultats sont, par contre, souvent moins bons lorsqu'on utilise des distributions plus évoluées (générales exponentielles et HE_2). Cela se justifie par le fait que ces techniques de décomposition vont quasiment systématiquement sous-évaluer les taux de production des lignes comportant plusieurs boucles. Or la méthode avec des temps de réparation exponentiels donne toujours des valeurs plus élevées des taux de production que les méthodes plus sophistiquées. D'une manière générale, les résultats seront d'autant plus satisfaisants que les boucles seront "grandes", c'est-à-dire dont tous les invariants sont assez grands (en nombre de stockages et de machines).

Enfin, les estimations des en-cours moyens de stocks dépassent rarement les 6 à 7% de la capacité des stocks, ce qui est très correct pour ces méthodes de décomposition.

6.5 Discussion sur les résultats

Nous avons utilisé un système d'équations, qui, au niveau des machines d'assemblage ou de désassemblage, ne prend pas en compte les corrélations entre les événements qui ont lieu sur deux branches parallèles, et qui n'ont pas cours dans le cas de lignes ouvertes ne comportant pas de

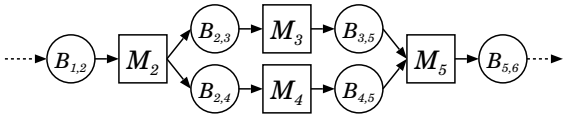

FIG. 6.25 – Exemple de portion de ligne où une branche se sépare en deux avant de se réassembler en une seule ligne. On montre des corrélations non prises en comptes dans un tel exemple.

boucle. Soit l'exemple simple d'une branche qui se sépare en deux sous-branches se regroupant un peu plus loin (Figure 6.25), et dont on considère que tous les stocks ont même capacité.

Si l'on suppose que la situation est la suivante :

- au temps $t = t_0$, les quatre buffers observés sont vides,
- au temps $t = t_1 (> t_0)$, la machines M_4 tombe en panne pour une durée d_1 telle que M_3 se retrouve non-alimentée. On a alors la situation au temps $t_2 = t_1 + d_1$:
  - M_2 est bloquée,
  - M_3 est non-alimentée,
  - M_4 est en panne,
  - M_5 est non-alimentée,
  - $B_{2,3}$ est vide,
  - $B_{3,5}$ est plein,
  - $B_{2,4}$ est plein,
  - $B_{4,5}$ est vide ;
- au temps $t_3 > t_2$, la machine M_4 est réparée,
- au temps $t_4 > t_3$, la machine M_3 tombe en panne. A ce moment, notre système (via la machine M_4) continue de fonctionner le temps d'inverser l'état des stocks $B_{2,3}$, $B_{3,5}$, $B_{2,4}$ et $B_{4,5}$. *Cela a lieu car une panne est survenue juste avant sur M_4, mais ce phénomène n'est pas du tout pris en compte dans notre approche.* Ce genre de corrélations sont ignorées (pour le cas de boucles simples, cela se résume à ignorer qu'un nombre important de clients présents dans une zone de la ligne (par exemple suite à une panne prolongée d'une machine) implique une faible quantité de clients dans la zone située juste après).

Ce type de phénomène n'est pas pris en compte par la technique de décomposition proposée.

Les résultats numériques obtenus restent malgré tout très corrects lorsqu'on utilise une caractérisation exponentielle des temps de panne des dipôles, et que la ligne fonctionne autour de son point de fonctionnement optimal (quand les invariants ne sont ni trop vides, ni trop pleins, et qu'il n'y a pas de gros déséquilibre entre les niveaux de remplissage de ces invariants).

L'utilisation de caractérisations plus fines de ces temps de réparation dégrade la qualité des résultats, ce qui peut paraître a priori surprenant. On sait en effet que l'utilisation de l'algorithme $DDX - HE_2$ donne de meilleurs résultats pour les lignes simples ouvertes [27]. Cette méthode donne en particulier de bien meilleures estimation des encours moyens des buffers. Or ces encours moyens sont un des critères essentiel qui détermine le point d'équilibre de la méthode appliquée aux lignes fermées. On aurait donc pu s'attendre à une amélioration sensible des résultats avec la caractérisation HE_2. Or, c'est le contraire qui s'est passé.

On savait que la technique (en exponentiel) appliquée aux lignes bouclées simples avait tendance à sous-estimer le taux de production de la ligne. Or, on sait que, lorsqu'on raffine ces caractérisations des temps de panne des buffers, on tend à diminuer l'estimation du taux de production (car on introduit de la variabilité). Donc l'utilisation de ces caractérisations plus fines sur les lignes fermées à accentué la sous-estimation des taux de production (sans que cela ne soit

compensé par le gain de précision de l'estimation des encours moyens des stocks).

Troisième partie

Autres améliorations

7

Lignes de production manufacturières ayant des zones de stockage non-fiables

On se propose d'étudier les stocks d'un système de production. Ils sont généralement considérés comme fiables dès lors que l'on développe des méthodes analytiques de lignes de production (voir entre autres [12]). Or, ils ne le sont pas en pratique (mais ils sont néanmoins nettement plus fiables que les machines qu'ils alimentent). Afin de proposer des méthodes pour analyser de telles lignes, on va d'abord détailler les mécanismes de panne de tels systèmes. On proposera, ensuite, des méthodes pour l'analyse de ces systèmes. Puis on les étudiera pour en estimer la pertinence sur des lignes simples en tandem.

Sommaire

7.1 Étude des systèmes réels

Les systèmes de stockage sont aussi en général des systèmes qui font transiter les pièces depuis la machine située en amont vers la machine située en aval. Traditionnellement, ils sont constitués d'une sorte de tapis roulant, ou de suite de chariots transportant les pièces. Aujourd'hui, de nouveaux types de stockages apparaissent sous forme d'"armoire".

7.1.1 Système traditionnel

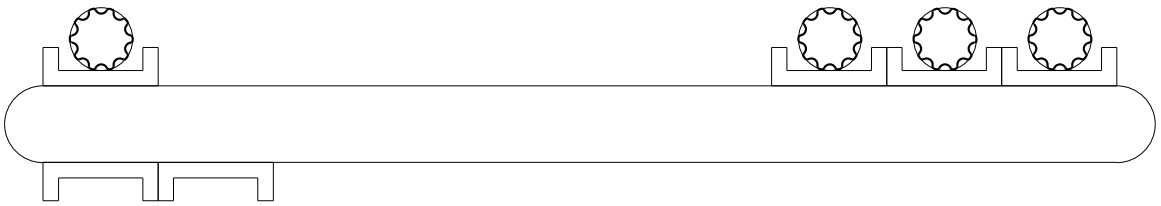

FIG. 7.1 – Modèle du stock traditionnel.

Le système de stockage couramment employé actuellement (appelé magasin) est constitué comme suit (voir Figure 7.1) :

- un système de déplacement des pièces se situe entre deux machines de production ;
- la machine amont, lorsqu'elle termine son traitement, s'occupe de placer la pièce sur le buffer ;
- de même, la machine en aval prend une pièce dans le stock des pièces accumulées sur le buffer lorsqu'elle va entamer un traitement.

Chaque pièce est prise en charge sur le buffer par un chariot. Le système tracteur avance en permanence, et chaque chariot s'accroche à ce dernier quand il a besoin d'avancer. Les chariots s'accumulent soit pleins devant la machine aval, soit vides devant la machine amont.

Lorsqu'une pièce est déposée ou accrochée sur un chariot, elle est acheminée par ce dernier vers la machine aval jusqu'à buter sur la queue des pièces déjà en place dans le stock. Enfin, un chariot vide vient se placer en attente d'un pièce devant la machine amont. De même, lorsque la machine aval retire une pièce du stock, son chariot vide va se replacer en attente vers la machine amont, et toutes les pièces accumulées sont décalées d'un cran vers la machine aval.

Les possibilités de panne d'un tel système sont donc :

- panne du système de traction. Les deux machines sont donc arrêtées dès la fin de leur traitement. Les pannes peuvent survenir à tout moment, puisque ce système tourne en permanence.
- pannes survenant au cours de l'évolution d'un chariot. Ces pannes sont typiquement liées à un problème du mécanisme d'accrochage du chariot au système tracteur, ou encore à une pièce mal placée sur la chariot provoquant une avarie. Elles peuvent survenir soit durant le transfert d'une pièce qui vient d'être déposée, soit au moment du décalage de toutes les pièces présentes dans le buffer juste après que la machine aval ait chargé une pièce.
- pannes liées aux capteurs. Elles surviennent au moment du processus de chargement/déchargement. Elles peuvent bloquer le buffer dans son ensemble.

7.1.2 Nouveau type de stock

On étudie un nouveau type de stock, dans lequel il n'y a pas de mécanisme de traction des pièces, mais uniquement une "étagère" à double accès. Cette étagère contient un nombre donné de places, et la machine amont pose des pièces dans les cases libres, tandis que la machine aval se sert dedans. Chaque case est munie de capteurs indiquant la présence d'une pièce.

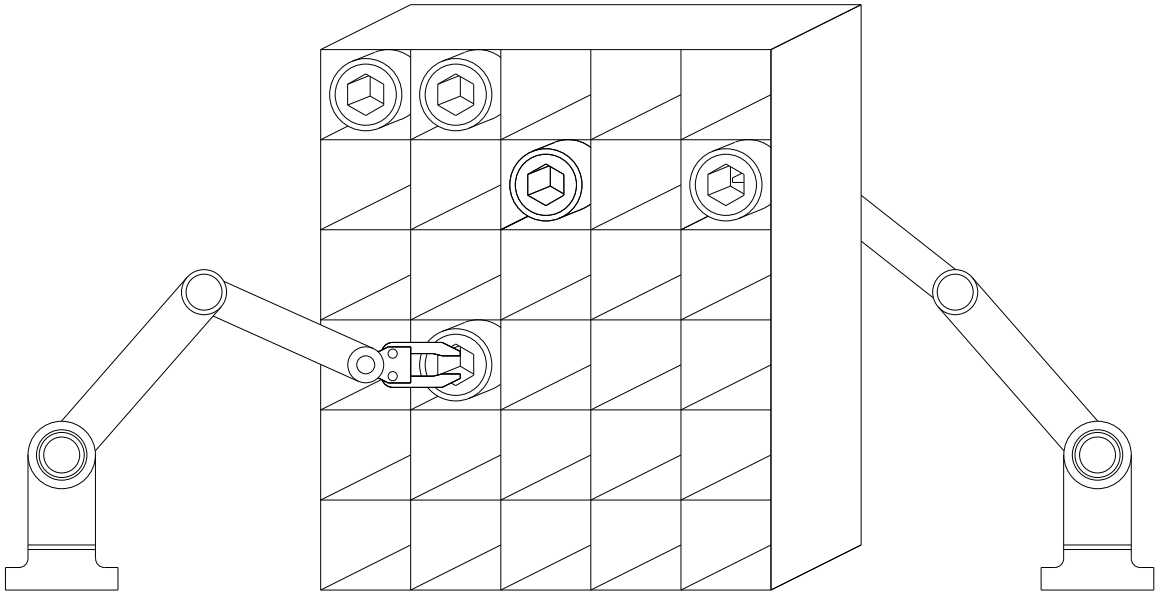

FIG. 7.2 – Nouveau système de stockage.

Ce sont les robots de traitement eux-mêmes qui exécutent les opérations de chargement et déchargement. Les pannes sont donc :

- panne d'un capteur. Cela peut arriver à tout moment et provoquer une erreur au moment d'une opération : le robot amont "voit" la case vide alors qu'elle est occupée, et y dépose une seconde pièce. De même le robot aval "voit" une case pleine alors qu'elle est vide.
- panne au moment d'une opération. L'opération se passe mal. Cela ne bloque en général que le robot en cause.

7.2 Modèle générique

On propose un modèle générique pour regrouper l'ensemble des pannes possibles (figure 7.3).

Chacune des pannes potentielles liée au buffer peut avoir deux impacts sur l'ensemble du système :

- elle n'entrave le fonctionnement que de l'élément en cause
- elle arrête tout le système.

De même, les occurrences de chaque panne peuvent être de deux types :

- aux instants (ponctuels) des opérations de chargement/déchargement. On appellera cela des pannes survenant pendant des opérations, ou Operation Occuring Failure (OOF).
- en permanence. On les nommera pannes survenant à tout instant, ou Permanent Occuring Failure (POF).

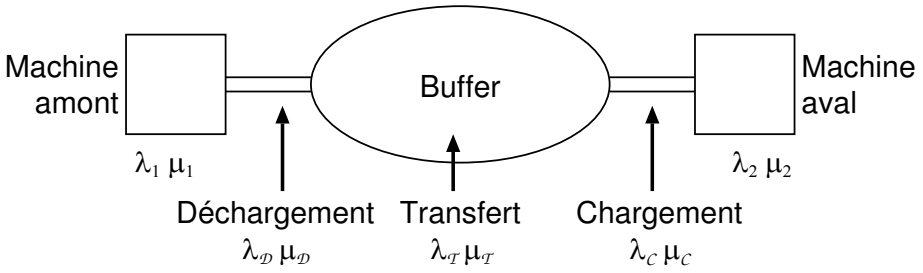

FIG. 7.3 – Modèle générique de stock.

On obtient la classification des pannes pour notre modèle générique :

	OOF	POF
Déchargement	uniquement au moment de l'opération	à tout moment
Chargement	uniquement au moment de l'opération	à tout moment
Transfert	uniquement quand un chariot est en cours de transfert	à tout moment

TAB. 7.1 – Occurrence d'une panne.

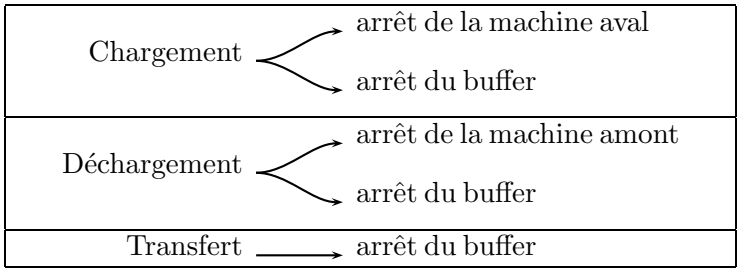

FIG. 7.4 – Impact d'une panne.

On notera que cette classification fait l'hypothèse implicite que les temps de décalage des chariots (pleins et vides), qui sont accumulés avant les machines amont et aval est négligeable devant les autres temps. On considère donc une panne liée à ces décalages comme pouvant survenir ponctuellement.

De plus, toute panne qui se produit pendant une opération de chargement ou de déchargement, mais dont l'impact est de ne bloquer que la machine qui est en train de réaliser cette opération, est normalement considérée comme une panne de la machine en cause. Par conséquent, elle n'est pas traitée ici. On se ramène au final à l'étude des pannes du buffer qui ne provoquent que l'arrêt du buffer (et qui ne provoquent l'arrêt des machines qu'aux instants de leur fin de service).

La schématisation de ces différentes pannes du buffer est présentée Figure 7.5.

7.2.1 Caractérisation des pannes

Les pannes de type POF sont naturellement caractérisées par les distributions des temps de réparation et de bon fonctionnement. En général, seules sont accessibles les moyennes de ces

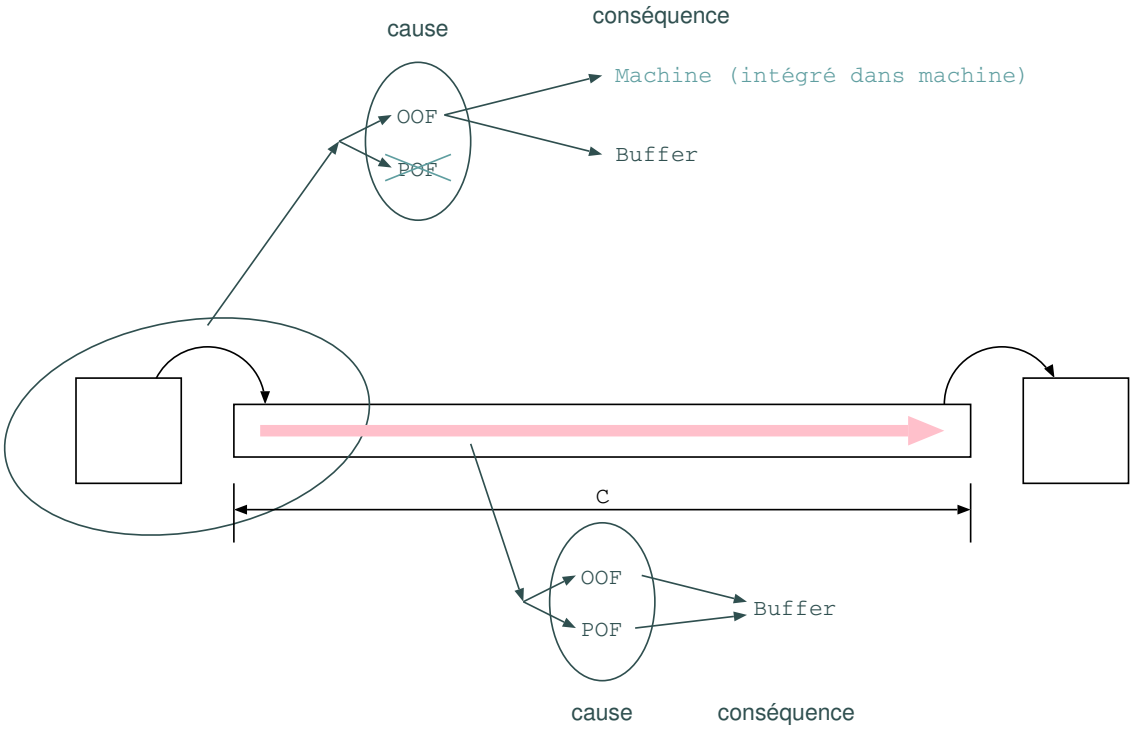

FIG. 7.5 – Mécanismes de panne du système du stock.

distributions (MTTF et MTTR). Cependant, on fait aussi l'hypothèse de pannes dépendant des opérations. Les différents éléments identifiés ne tombent alors en panne que s'ils sont effectivement en train de travailler. Ainsi, le système tracteur qui tourne tout le temps peut tomber en panne à chaque instant. Par contre, le transfert d'un chariot d'un bout à l'autre du buffer (s'il n'est pas ponctuel) ne peut engendrer de panne que lorsqu'un chariot est effectivement en train d'être emporté. Par la suite, on fera toujours l'hypothèse que ces pannes sont ponctuelles.

Les pannes de type OOF sont des pannes qui surviennent sur des actions dont le temps d'exécution est considéré négligeable par rapport aux autres temps (traitement, transfert complet, etc.). En conséquence, il est plus naturel de considérer leur occurrences comme un processus stochastique discret. On nommera le nombre moyen d'opérations avant de tomber en panne MNTF (Mean Number (of operations) To Failure). Le temps de réparation est quant à lui toujours caractérisé par une distribution dont on ne connaît en général que le premier moment (MTTR).

On peut alors déterminer des relations liant ces paramètres. En effet, si on considère le mécanisme de pannes OOF, on peut établir les relations suivante avec des pannes POF dépendant de l'état dans lequel on se trouve et de la vitesse de traitement des machines.

On observe un système homogène constitué de deux machines séparées d'un stock (fig. 7.3). La machine M_1 (resp. M_2) est caractérisée par son temps de traitement T, son taux de pannes λ_1 (resp. λ_2), et le taux de son temps de réparation μ_1 (resp. μ_2). De même, le buffer est de capacité $C_{1,2}$. Il est non fiable avec des pannes de deux types : POF avec un MNTF donné ($\nu_{1,2}$: c'est le taux de la distribution géométrique associée), et OOF avec un MTTF ($\lambda_{1,2}$). Le temps de réparation est connu et est le même pour les deux types de pannes ($\mu_{1,2}$).

Comparaison des mécanismes de panne

Les pannes de types OOF peuvent être vues comme des pannes POF dépendant de l'état du système (constitué des 2 machines encadrant le buffer et de ce dernier). En effet, si le temps de traitement d'une pièce par les machines est T, connu, on peut relier le temps de bon fonctionnement du buffer à son MNTF par les relations :

	OOF	POF
M_1 et M_2 fonctionnent	$MTTF_{eq}^{OOF} = \frac{2\,MNTF}{T} = \frac{2}{T\nu_{1,2}}$	$MTTF_{eq}^{POF} = MTTF$
M_1 fonctionne seule	$MTTF_{eq}^{OOF} = \frac{MNTF}{T} = \frac{1}{T\nu_{1,2}}$	$MTTF_{eq}^{POF} = MTTF$
M_2 fonctionne seule	$MTTF_{eq}^{OOF} = \frac{MNTF}{T} = \frac{1}{T\nu_{1,2}}$	$MTTF_{eq}^{POF} = MTTF$
aucune machine ne produit	$MTTF_{eq}^{OOF} = \infty$	$MTTF_{eq}^{POF} = MTTF = \infty$

Soit pour les taux relatifs des distributions exponentielles correspondantes :

	OOF	POF
M_1 et M_2 fonctionnent	$\lambda_{eq}^{OOF} = \frac{T\nu_{1,2}}{2}$	$\lambda_{eq}^{POF} = \lambda_{1,2}$
M_1 fonctionne seule	$\lambda_{eq}^{OOF} = T\nu_{1,2}$	$\lambda_{eq}^{POF} = \lambda_{1,2}$
M_2 fonctionne seule	$\lambda_{eq}^{OOF} = T\nu_{1,2}$	$\lambda_{eq}^{POF} = \lambda_{1,2}$
aucune machine ne produit	$\lambda_{eq}^{OOF} = 0$	$\lambda_{eq}^{POF} = 0$

7.3 Conséquences pour la simulation

Si on cherche à étudier par la simulation du modèle discret asynchrone, un système de production, dont on ne sait pas a priori si le mécanisme des pannes des buffers relève plutôt du type OOF ou du type POF, on doit choisir entre ces deux types de pannes. L'idéal serait de faire une combinaison des deux (car plus proche du système réel). Cependant, cela implique de connaître les paramètres de pannes des buffers avec précision, ce que qui n'est en général pas le cas.

On peut alors chercher à déterminer comment se comportent ces deux modèles de panne.

7.3.1 Première version

On observe un système dont on connaît le temps moyen de bon fonctionnement du buffer ($MTTF^{SYST}$, qui est normalement la seule information dont on dispose), et on fait des simulations du modèle OOF et POF avec les paramètres :

- $MTTF^{POF} = MTTF^{SYST}$
- $MNTF^{OOF} = \frac{MTTF^{SYST}}{T}$

On obtient typiquement des résultats qui ressemblent à la figure 7.6 (cas d'une ligne de 7 machines séparées de stocks de taille 7. Toutes les machines sont équivalentes avec $\lambda_i = 0.01$, $\mu_i = 0.1$. Les buffers ont des paramètres de pannes : $\mu_{i,i+1} = 0.1$, $I = \frac{MTTR}{MTTF}$ variable.)

C'est un résultat logique : le modèle OOF donne un taux de production plus faible que le modèle POF. En effet, lorsque les deux machines encadrant un buffer fonctionnent de concert, le taux de panne du modèle OOF est deux fois plus important que pour le modèle POF, puisqu'il y a durant cette période deux fois plus d'opérations (de chargement et de déchargement) par unité de temps.

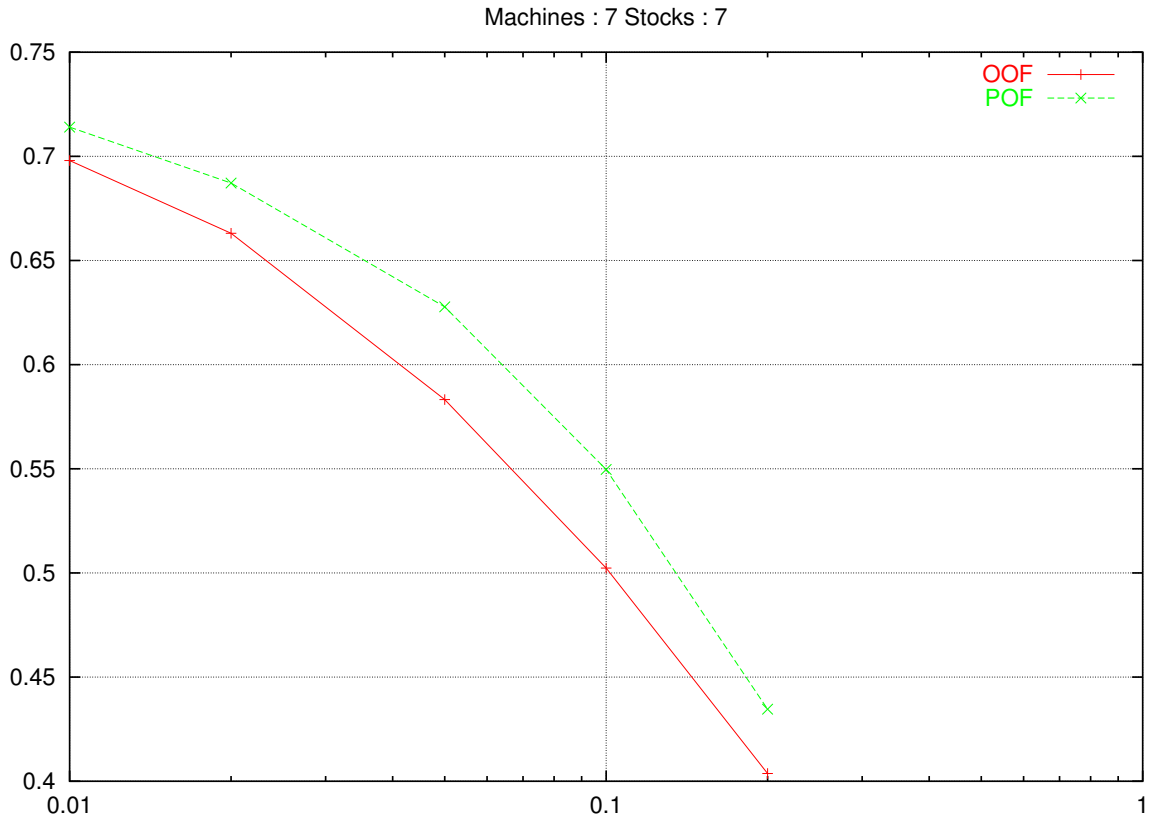

FIG. 7.6 – Résultats de simulation des deux modèles de panne de buffer.

7.3.2 Deuxième version

On aurait cependant tout aussi légitimement pu identifier les paramètres de la sorte :
- $MTTF^{POF} = MTTF^{SYST}$
- $MNTF^{OOF} = \frac{2\,MTTF^{SYST}}{U}$

Auquel cas, sur les périodes où les deux machines encadrant un buffer produisent ensembles, les deux modèles emploient le même mécanisme de panne. Dans ce cas, le MNTF est deux fois plus grand ; il n'y a pas de distinction entre les clients qui entrent dans le buffer et ceux qui sortent, — tous deux sont une opération — il y a donc deux fois plus d'opérations effectuées par unité de temps. A l'inverse, durant les périodes où seule une machine produit effectivement, il y a deux fois moins de pannes du buffer.

7.4 Conséquences pour l'analyse - Étude du dipôle

Plusieurs heuristiques sont envisageables pour résoudre simplement l'analyse un dipôle. Dans tous les cas, on considère un dipôle D_U constitué des machines M_u et M_d, et d'un buffer B_U. Les paramètres respectifs de ces éléments sont λ_u, μ_u, λ_d, μ_d, pour les taux de panne et réparation des machines, et λ_b, μ_b, C pour les taux de panne, de réparation et la capacité du buffer (figure 7.7).

Dans tous les cas, le principe est de décomposer le dipôle de manière à faire "ressortir" le

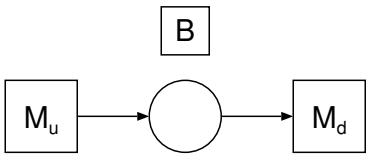

FIG. 7.7 – Représentation du dipôle étudié.

mécanisme de panne, en considérant le buffer comme fiable, mais en ajoutant une (ou deux) machine représentant les pannes du buffer.

7.4.1 Heuristique de Burman et dérivés

L'idée est de considérer le dipôle D_R équivalent à notre dipôle D_U, mais avec un buffer fiable. Ainsi sa résolution ne pose pas de problème (voir [18]). On considère ce système comme une machine unique dont on connaît l'efficacité. Le mécanisme de panne est alors étudiés comme un système constitué de deux machines : celle représentant le dipôle D_R, et celle représentant le mécanisme des pannes (machine dont les taux de panne et de réparation sont λ_b et μ_b, voir figure 7.8). Il s'agit en réalité d'une méthode d'agrégation [3].

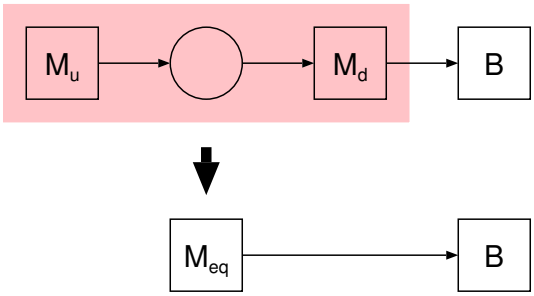

FIG. 7.8 – Heuristique de Burman.

Valeurs aux limites :
- $C = 0 : E = \frac{1}{1+I_u+I_d+I_b}$
- $C = \infty : E = \frac{1}{1+max(I_u,I_d)+I_b}$

Autres possibilités

Le cas présenté ci-dessus est directement issu du travail de la thèse de Burman [3]. Il peut être aussi intéressant d'envisager d'autres configurations de lignes équivalentes.

En plaçant la machine représentant les pannes du buffer juste avant M_d (figure 7.9).

Valeurs aux limites :
- $C = 0 : E = \frac{1}{1+I_u+I_d+I_b}$

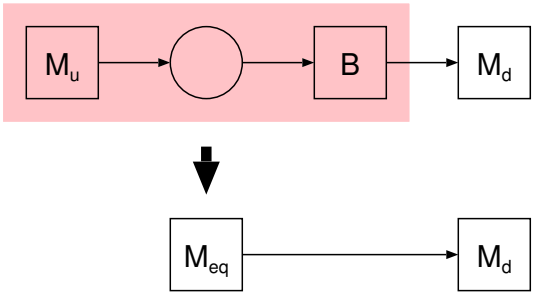

FIG. 7.9 – Variation de l'heuristique de Burman.

- $C = \infty : E = \frac{1}{1+max(I_u,I_b)+I_d}$

En plaçant la machine représentant les pannes du buffer juste après M_u (figure 7.10)

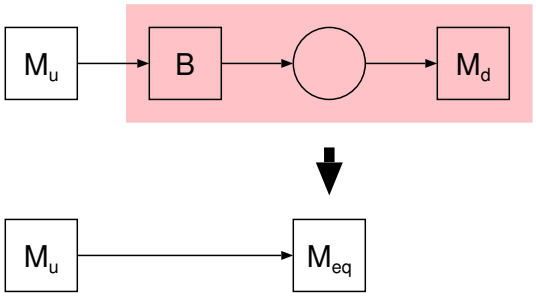

FIG. 7.10 – Autre variation de l'heuristique de Burman.

Valeurs aux limites :
- $C = 0 : E = \frac{1}{1+I_u+I_d+I_b}$
- $C = \infty : E = \frac{1}{1+max(I_b,I_d)+I_u}$

7.4.2 Agrégation des machines avant d'évaluer le dipôle

On ne change pas l'ordre dans lequel on fait les agrégations de machines. Lorsque deux machines se suivent, on les agrège avant d'évaluer les performances du dipôle.

On a donc plusieurs cas possibles. Si on ne rajoute qu'une seule machine pour représenter les pannes du buffer, on a deux possibilités : la mettre avant ou après après le buffer.

Après le buffer (figure 7.11)

Valeurs aux limites :
- $C = 0 : E = \frac{1}{1+I_u+I_d+I_b}$
- $C = \infty : E = \frac{1}{1+max(I_u,I_b+I_d)}$

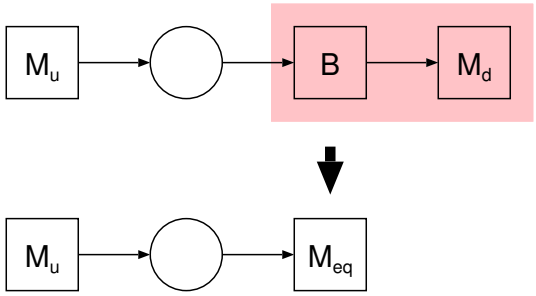

FIG. 7.11 – Agrégation de la machine aval avant évaluation du dipôle.

Avant le buffer (figure 7.12)

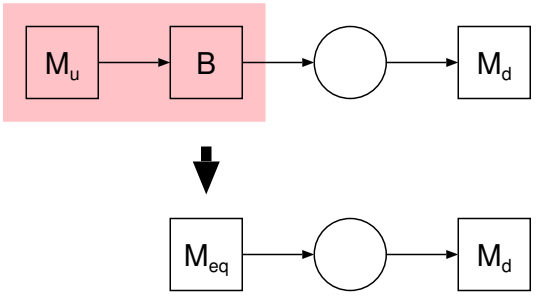

FIG. 7.12 – Agrégation de la machine amont avant évaluation du dipôle.

Valeurs aux limites :
- $C = 0 : E = \frac{1}{1+I_u+I_d+I_b}$
- $C = \infty : E = \frac{1}{1+max(I_u+I_b,I_d)}$

Autre choix Il peut cependant paraître insuffisant de modéliser les blocages du buffer uniquement sur la machine amont, ou uniquement sur la machine aval. On peut donc envisager de mettre deux machines représentant les pannes du stock : une machine de chaque côté du buffer. Cette transformation serait même exacte si les deux machines ajoutés étaient synchronisées (auquel cas, on aurait rien fait d'autre que de décomposer le comportement du buffer).

Valeurs aux limites : (figure 7.13)
- $C = 0 : E = \frac{1}{1+I_u+I_d+2.I_b}$
- $C = \infty : E = \frac{1}{1+I_b+max(I_u,I_d)}$

7.4.3 Schéma récapitulatif

Un schéma récapitulant les possibilités évoquées est donné figure 7.14.

7.4.4 Comparaisons et résultats numériques

On a comparé ces différentes méthodes pour les 6 cas suivants :

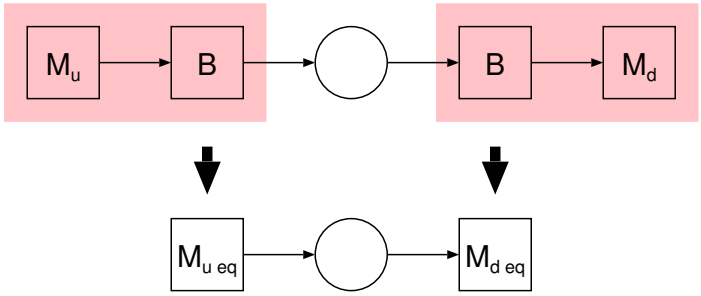

FIG. 7.13 – Agrégation des machines amont et aval avant évaluation du dipôle.

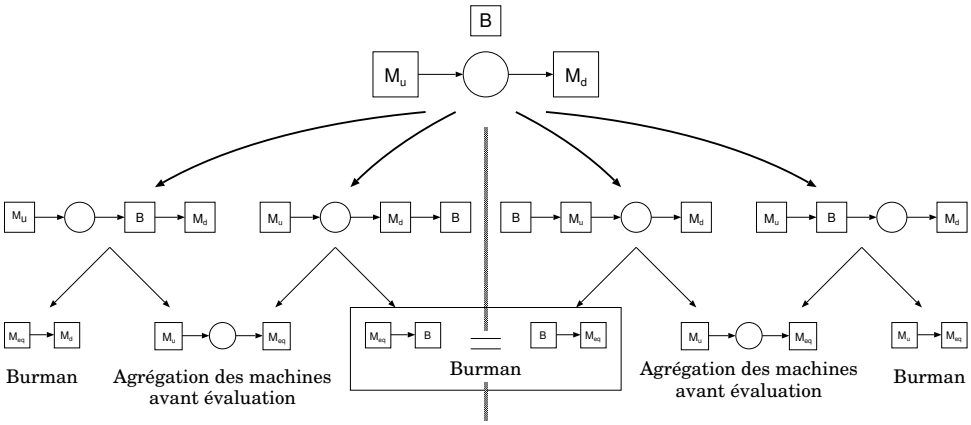

FIG. 7.14 – Récapitulation des choix possibles pour une seule machine (ajoutée au dipôle sans panne des stocks) qui représente les pannes du stock.

	λ_u	μ_u	λ_d	μ_s	λ_b	μ_b	I_u	I_d	I_b	e_u	e_d	e_b
Cas 1	0.01	0.1	0.02	0.2	0.005	0.1	0.1	0.1	0.05	0.9091	0.9091	0.9524
Cas 2	0.01	0.1	0.05	0.1	0.001	0.1	0.1	0.5	0.01	0.9091	0.6667	0.9901
Cas 3	0.01	0.1	0.01	0.1	0.01	0.1	0.1	0.1	0.1	0.9091	0.9091	0.9091
Cas 4	0.01	0.1	0.01	0.1	0.001	0.1	0.1	0.1	0.01	0.9091	0.9091	0.9901
Cas 5	0.07	0.1	0.03	0.0428	0.005	0.1	0.7	0.7	0.05	0.5882	0.5882	0.9524
Cas 6	0.07	0.1	0.03	0.0428	0.01	0.1	0.7	0.7	0.1	0.5882	0.5882	0.9091

Pour chacun de ces cas, on obtient les courbes Figure 7.20 page 137 à Figure **??** page ??.

7.5 Méthodes pour la ligne simple ouverte

Afin d'étudier ces systèmes à l'aide des méthodes analytiques, on cherche si une méthode simple de pré-traitement est quantitativement admissible. L'objectif est alors de trouver un moyen de se ramener à une ligne ouverte dont les buffers sont fiables.

7.5.1 Modélisation

Pour analyser plus aisément le système, on propose (figure 7.21) un modèle générique qui permette de dissocier les mécanismes de pannes des stocks. Pour cela, on éclate le mécanisme de

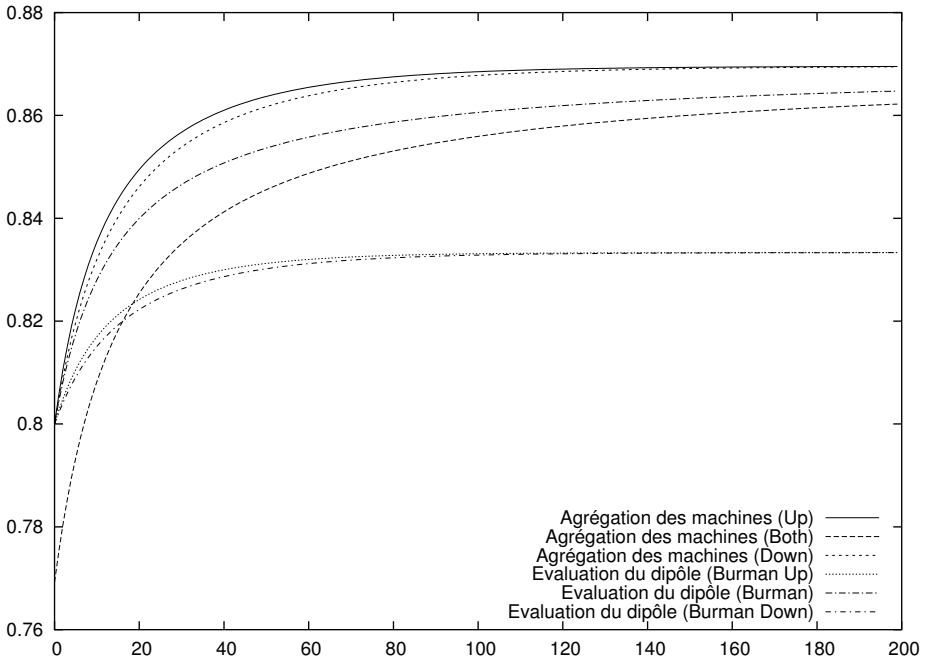

FIG. 7.15 – Cas 1.

panne de stocks en 3 machines, de temps de traitement nul mais pouvant tomber en panne.

On détermine les paramètres de ces machines selon le type de panne que l'on veut modéliser tel que décrit dans la section 7.2. En général, les machines M_{b1} et M_{b2} représentent les pannes de chargement ou de déchargement, qui sont pour l'essentiel de type OOF, et sont donc incluses dans le mécanisme de pannes du robot de la machine M_1 ou M_2.

Normalement, les machines M_c situées juste avant et juste après le buffer (fiable) représentent les pannes de transfert du buffer, qui conduisent à l'arrêt des deux machines encadrant le buffer. Ces deux pseudo-machines devraient donc être synchrones (c'est-à-dire tomber en panne et être réparées en même temps). Cela n'étant pas faisable simplement, on se borne, pour représenter ces pannes de transfert, à une approximation par une seule machine (située soit juste avant, soit juste après le buffer), ou bien encore par deux pseudo-machines indépendantes.

Dans le premier cas, on s'attend à une approximation correcte lorsque la capacité de stockage est faible. Dans le second, on s'attend à une meilleure approximation lorsque les capacités de stockage sont grandes.

7.5.2 Analyse par transformation de la ligne

On se propose donc d'ajouter avant et/ou après chaque machine de la ligne originale, une machine de même vitesse de traitement, représentant les mécanismes de panne du buffer. Le problème consiste à de choisir les paramètres de performance de ces machines.

On peut :

- placer une machine représentant le buffer **après** le buffer correspondant (fiable) de la ligne modifiée ;
- placer une machine représentant le buffer **avant** le buffer correspondant (fiable) de la ligne modifiée ;

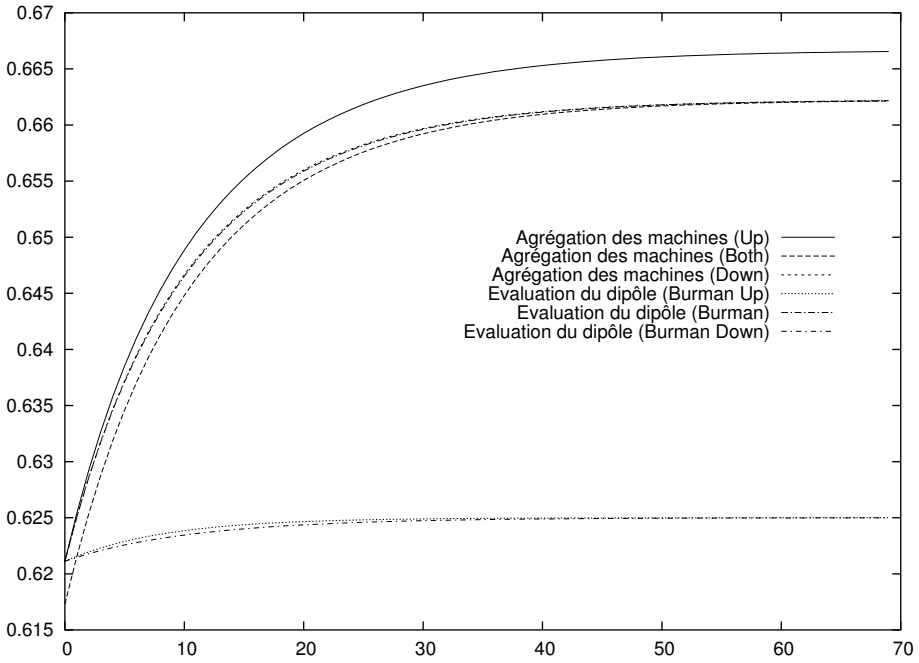

FIG. 7.16 – Cas 2.

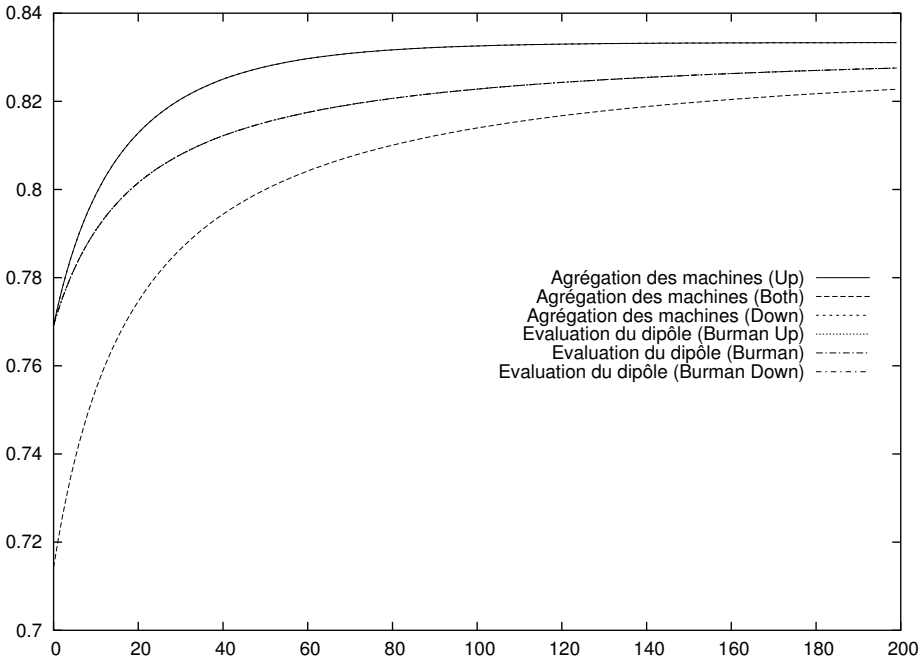

FIG. 7.17 – Cas 3.

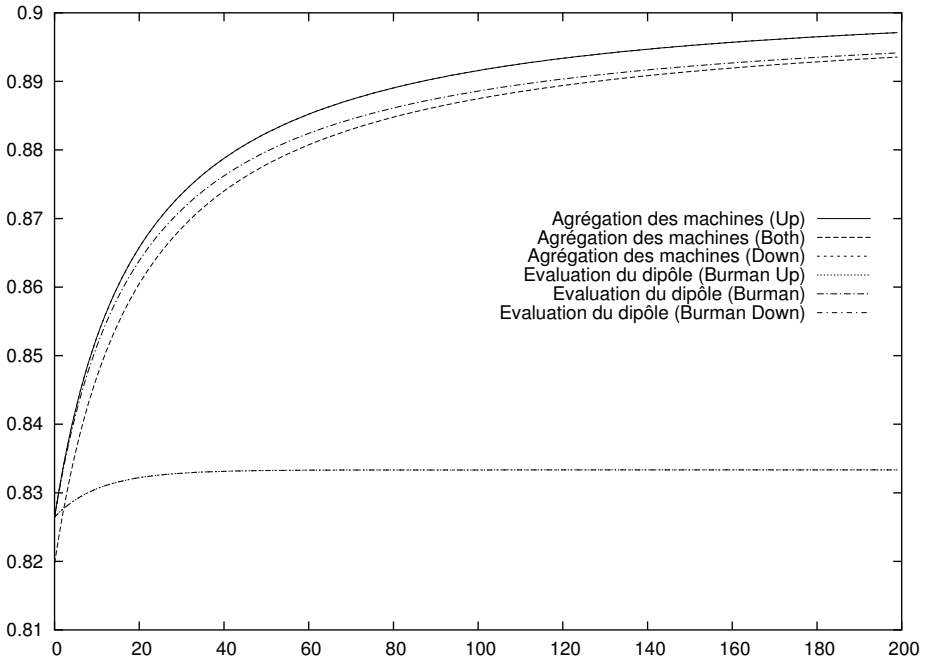

FIG. 7.18 – Cas 4.

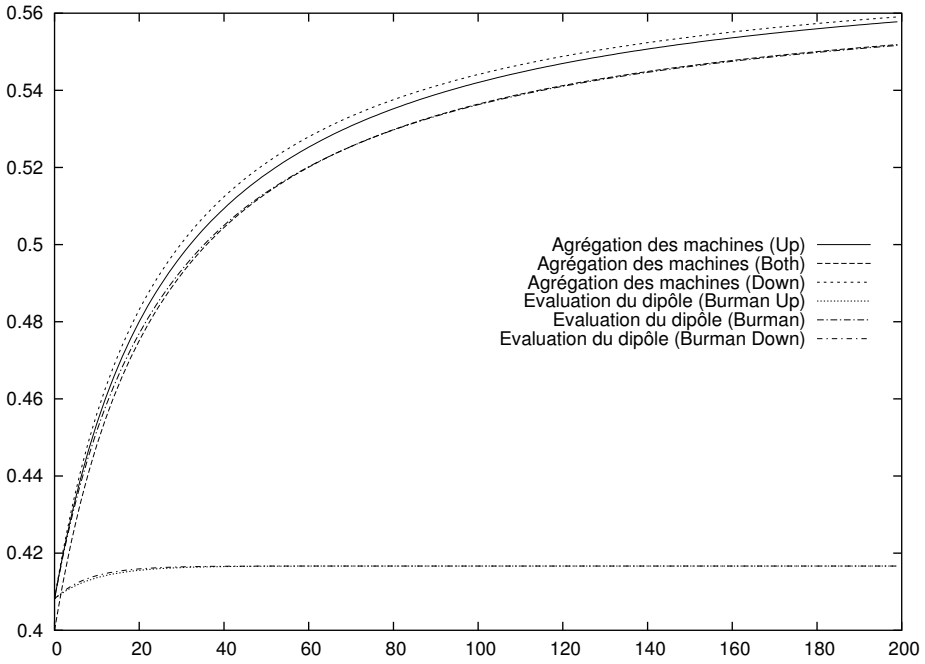

FIG. 7.19 – Cas 5.

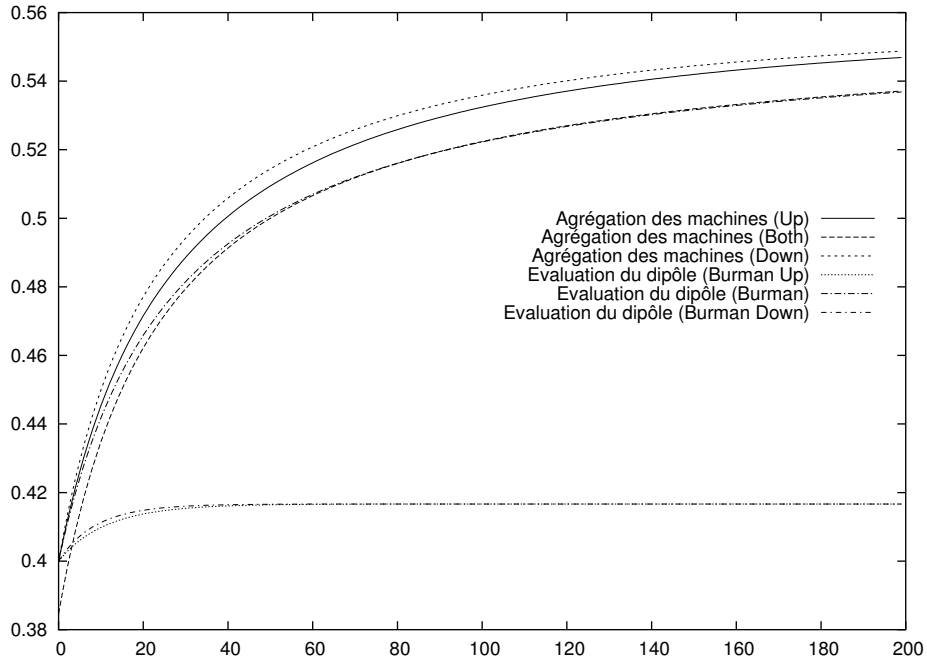

FIG. 7.20 – Cas 6.

– placer une machine représentant le buffer **avant et après** le buffer correspondant (fiable) de la ligne modifiée.

Pour le choix des paramètres de ces machines ajoutées, on peut prendre :

– **Machine avant le buffer** : $MTTF_{buf}^{eq} = MTTF_{buf}$.

– **Machine après le buffer** : $MTTF_{buf}^{eq} = MTTF_{buf}$.

– **Machine avant et après le buffer** : $MTTF_{buf}^{eq} = MTTF_{buf}$ ou $MTTF_{buf}^{eq} = \frac{MTTF_{buf}}{2}$.

Ce que l'on fait c'est donc de rajouter un mécanisme de panne avant et/ou après le buffer. Mais on fait alors des approximations : en effet, dans tous les cas, on perd la synchronisation des machines encadrant le buffer, même si l'on ajoute deux machines (avant et après).

Ainsi :

$C \to 0$ **:** cas limite où la taille du buffer tend vers 0. Dans ce cas, le système devient strictement équivalent à un système où les deux machines sont séparées d'une machine non fiable dont les paramètres de performance sont ceux du buffer. Auquel cas, le choix de prendre une machine représentant les pannes du buffer, placée avant ou après, est équivalent.

$C \to \infty$ **:** cas limite où la taille du buffer est infinie. Ici, il y a découplage complet effectué par le buffer. Les arrêts des machines issus des pannes du buffer, même s'ils sont synchrones en pratique, sont alors équivalents à des arrêts asynchrones de ces deux machines. Il semble donc plus naturel de choisir la configuration où il y a deux machines — avant et après le buffer — pour modéliser les pannes du buffer. Il convient même de choisir des paramètres égaux à ceux du buffer.

Une fois cette transformation effectuée, il suffit d'agréger en une machine unique chaque groupe de machines contiguës. On est alors en présence d'une ligne classique que l'algorithme DDX peut aisément résoudre [10, 11].

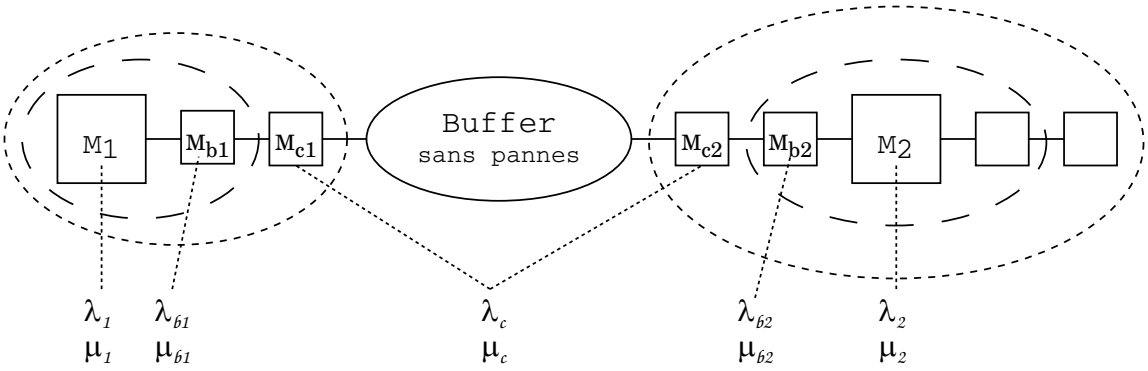

FIG. 7.21 – Modèle générique des mécanismes de panne de stock.

7.5.3 Par modification de l'algorithme DDX

Il convient préalablement de faire la remarque suivante : on sait (voir [3]) que l'on peut faire une approximation très correcte d'un dipôle dont le stock intermédiaire est susceptible de tomber en panne par un dipôle fiable suivi d'une machine représentant les pannes du buffer. L'analyse d'un tel système se fait alors en analysant d'abord le dipôle (dont le buffer est fiable, donc à l'aide de Dubois et Forrestier [18]), puis en utilisant la formule de Buzacott pour déterminer l'efficacité, et donc le taux de production, du système :

$$E_{eq} = \frac{1}{\frac{\lambda_b}{\mu_b} + \frac{1}{E_{fiab}}}$$

où E_{eq} est l'efficacité du système équivalent (approximation de celle du dipôle dont le buffer n'est pas fiable), et E_{fiab} est l'efficacité du dipôle issue du dipôle original, mais dont le buffer est fiable. Cette heuristique donne une très bonne approximation du taux de production, mais n'indique rien sur les autres paramètres de performance.

L'idée est alors d'utiliser cette heuristique dans une version modifiée du DDX. À chaque étape, au moment de déterminer les paramètres de performance d'un dipôle, on corrige, à l'aide de cette formule, l'efficacité obtenue par Dubois et Forrestier du dipôle correspondant et dont le buffer est fiable.

La simplicité de mise en œuvre de cette méthode lui donne un grand intérêt. Par contre on peut déjà prévoir certains problèmes : cette méthode implique que les paramètres de performance de chaque dipôle (agrégé) ne sont plus cohérents. En effet, l'efficacité n'est alors plus liée aux autres paramètres de performance du dipôle. Le système d'équations normalement résolu via le DDX est donc modifié, et l'on risque d'avoir des problèmes de convergence. De plus, les résultats obtenus ne seront pas "justifiables".

7.6 Résultats numériques

Pour étudier les méthodes proposées, on fait des comparaisons avec des simulations (type de pannes POF). On fait, dans un premier temps, une étude de ligne homogène (où tous les

paramètres sont identiques d'une machine à l'autre et d'un buffer à l'autre) ; on fait ensuite quelques tests sur quelques lignes dont les paramètres ont été tirés aléatoirement.

Remarque 16 : la méthode évoquée en 7.5.3 n'est pas étudiée ici. Elle pose beaucoup de problèmes de stabilité. L'algorithme est souvent divergent, ou au moins pas convergent en un temps raisonnable. Aussi, après l'avoir implémenté, nous avons choisi de ne pas le présenter plus avant.

7.6.1 Lignes régulières

Les tests proposés portent sur des lignes homogènes dont toutes les machines sont égales (mêmes paramètres), tout comme les buffers. L'objectif est de tester à chaque fois les différentes configurations proposées.

Pour chaque configuration étudiée, 4 méthodes ont été testées :

- *UP* : une seule machine modélisant les pannes de stock est rajoutée avant chaque buffer.
- *DOWN* : une seule machine modélisant les pannes de stock est rajoutée après chaque buffer.
- *BOTH* : deux machines modélisant les pannes de stock sont rajoutées avant et après chaque buffer. On peut choisir de mettre des taux de pannes égaux à ceux du buffer, ou égaux à la moitié de ceux-ci.

Les tests font varier la taille de la ligne étudiée (3, 5 et 10 machines), et la taille des stocks (3 et 10). Pour chaque cas d'étude, on a utilisé les paramètres suivants :

- $\lambda_i = 0.001$
- $\mu_i = 0.1$
- $T_i = 1.0$
- $\mu_{i,i+1} = 0.1$
- $I_{i,i+1} \in \{0.01, 0.1\}$

et :

- $\lambda_i = 0.001$
- $\mu_i = 0.1$
- $T_i = 1.0$
- $\mu_{i,i+1} = 0.01$
- $I_{i,i+1} \in \{0.01, 0.1\}$

Pour chaque cas de figure, on a fait deux séries de tests analytiques : une série où les capacités sont choisies égales à celles du modèle simulé, et l'autre où les capacités sont augmentées d'une unité (prise en compte de la place disponible sur les machines). On a, à chaque fois, obtenu les résultats bruts et fait les comparaisons entre les différentes méthodes utilisées et la simulation de type POF.

Productivité

On présente, pour chaque cas de figure, deux graphes (figures C.1 à **??**, disponibles en Annexe C). Le premier (graphe de gauche) donne les résultats des taux de production de la ligne pour la simulation (avec son estimation d'erreurs) et pour les différentes variantes de la méthode proposée. Le second (graphe de droite) donne les comparaisons des résultats des méthodes analytiques par rapport à la simulation (en pourcentage de la valeur obtenue par simulation).

On constate donc que les comportements ne sont pas les mêmes selon que l'on observe des buffers d'une fiabilité raisonnable ($I = 0.01$, soit $e = 0.99009$) ou des buffers moins fiables que les machines de la ligne. Ce sont principalement ces premiers résultats qui nous intéressent, car

ils sont plus proches de situations réalistes. Plus spécifiquement, les valeurs du paramètre $\mu_{i,i+1}$ sont les plus significatives.

Dans ces exemples, les solutions *UP*, *DOWN* et *BOTH* avec $\lambda_{analyse} = \lambda_{simulation}/2$ sont très proches avec des taux de sorties légèrement surévalués, mais en général de moins de 1% (moins de 3% dans le cas de la ligne de 10 machines avec des stock de taille 10, voir figure C.23). Ces résultats sont donc très satisfaisants.

Sur l'exemple de la figure C.23, il est intéressant de remarquer que pour $I = 0.1$ comme pour $I = 0.01$, la méthode *BOTH* où $\lambda_{analyse} = \lambda_{simulation}/2$ est la meilleure. Cela se comprend, car avec ces paramètres, les machines sont fortement découplées ; en effet les stocks y sont de taille assez importantes, et le nombre de machines dans la ligne est plus important. Ainsi, la perte de simultanéité de l'arrêt des deux machines lors d'une panne du stock par notre transformation est maintenant moins important.

7.6.2 En-cours moyens

On présente, ici aussi deux graphes pour chaque cas, l'un donnant les différences absolues entre les en-cours moyen issus de l'analyse et de la simulation, et l'autre les différences relatives, par rapport à la capacité de la zone de stock. Ces résultats sont donnés figures **??** à C.40 en Annexe C.

Les résultats montrent plusieurs aspects : d'une part, les méthodes sont bien meilleures pour l'estimation des en-cours moyens lorsque les capacités de stock $C_{ana} = C_{simu}$. La méthode *UP* tend à sous-estimer nettement les encours moyens, tandis que la méthode *DOWN* fait le contraire. La méthode *BOTH*, avec $\lambda_{analyse} = \lambda_{simulation}/2$ est en quelque sorte une moyenne des deux précédentes. Elle donne en moyenne de meilleurs résultats, avec des erreurs en général inférieures à 5% de la capacité du stock, et souvent bien moins.

Pour le cas de lignes homogènes, cette dernière méthode semble donc la mieux adaptée et fournit des résultats plus que satisfaisants.

7.6.3 Lignes homogènes quelconques

On présente ici des lignes dont les paramètres on été tirés aléatoirement, tel que :
- $\lambda_i \in [\frac{0.9}{200}, \frac{1.1}{200}]$
- $\mu_i \in [\frac{0.9}{10}, \frac{1.1}{10}]$
- $\lambda_{i,i+1} \in [\frac{0.9}{500}, \frac{1.1}{500}]$ ou
- $\lambda_{i,i+1} \in [\frac{0.9}{1000}, \frac{1.1}{1000}]$ ou
- $\lambda_{i,i+1} \in [\frac{0.9}{2000}, \frac{1.1}{2000}]$ ou
- $\mu_{i,i+1} \in [\frac{0.9}{10}, \frac{1.1}{10}]$

avec des lignes de 3 et 10 machines, pour des capacités de 3 et 10 zones de stockage. On a fait un tirage pour chaque combinaison de ces tailles de ligne et de stockage, soit 12 cas. L'ensemble des résultats sont disponibles Annexe C.

Ces résultats sont obtenus avec des paramètres proches de cas réels. Mais, étant donné la fiabilité importante des buffers, il est plus délicat d'interpréter ces résultats.

Cependant, on peut confirmer les conclusions obtenues avec les lignes symétriques. Si dans le cas de petites lignes, les solutions *UP*, *DOWN* ou *BOTH* avec $\lambda_{analyse} = \lambda_{simulation}/2$ sont meilleures, pour le lignes importantes (en nombre de machines et, éventuellement, en taille des stocks), il faut choisir la solution *BOTH* où $\lambda_{analyse} = \lambda_{simulation}$.

Concernant les encours, les résultats sont sensiblement moins précis. Cependant, on sait (voir [11, 27]) que les méthodes de décomposition sont souvent assez approximatives concernant

l'évaluation des encours.

On constate ainsi que, dans le cas où la capacité des stocks de la ligne étudiée par analyse est identique à celle des stocks de la ligne simulée, on a tendance à surévaluer les taux de production, mais à sous-évaluer les en-cours. Donc si on choisit maintenant de prendre des stocks plus grands, on va améliorer l'estimation des en-cours, mais au détriment des taux de production, ce qui n'est pas souhaitable.

8

Application pratique : le logiciel DispoPSA

On présente ici un exemple d'application pratique des méthodes présentées précédement. Il s'agit d'un logiciel que nous avons développé pour la société PSA. L'objectif est de proposer un outils d'analyse rapide et simple permettant d'évaluer les performances d'une ligne de production manufacturière dont la topologie est une double-ligne, telle que présentée dans l'introduction. Disposant d'une interface graphique simple, ce logiciel permet de définir l'ensemble des paramètres de la ligne. Nous disposons, pour cela, des paramètres des différents types de robot pouvant intervenir sur une telle ligne de montage, et de récupérer immédiatement une estimation des paramètres de performance de cette ligne.

Sommaire

8.1 Présentation et objectifs

8.1.1 Description de la ligne siamoise

La ligne siamoise standard (pour cet atelier de ferrage) est constituée de 76 postes de travail (qui peuvent être des postes robotisés ou des postes morts, auquel cas ils s'assimilent à une place de stockage. Une représentation de cette ligne siamoise est donnée Figure 8.1 page suivante. Pour représenter cette ligne, on découpe cette ligne en trois parties :

- la branche commune : c'est la partie de la ligne constituée des postes 29 à 2, où passent tous les chariots ;
- la branche supérieure : c'est la partie de la ligne constituée des postes 51 à 76 ;

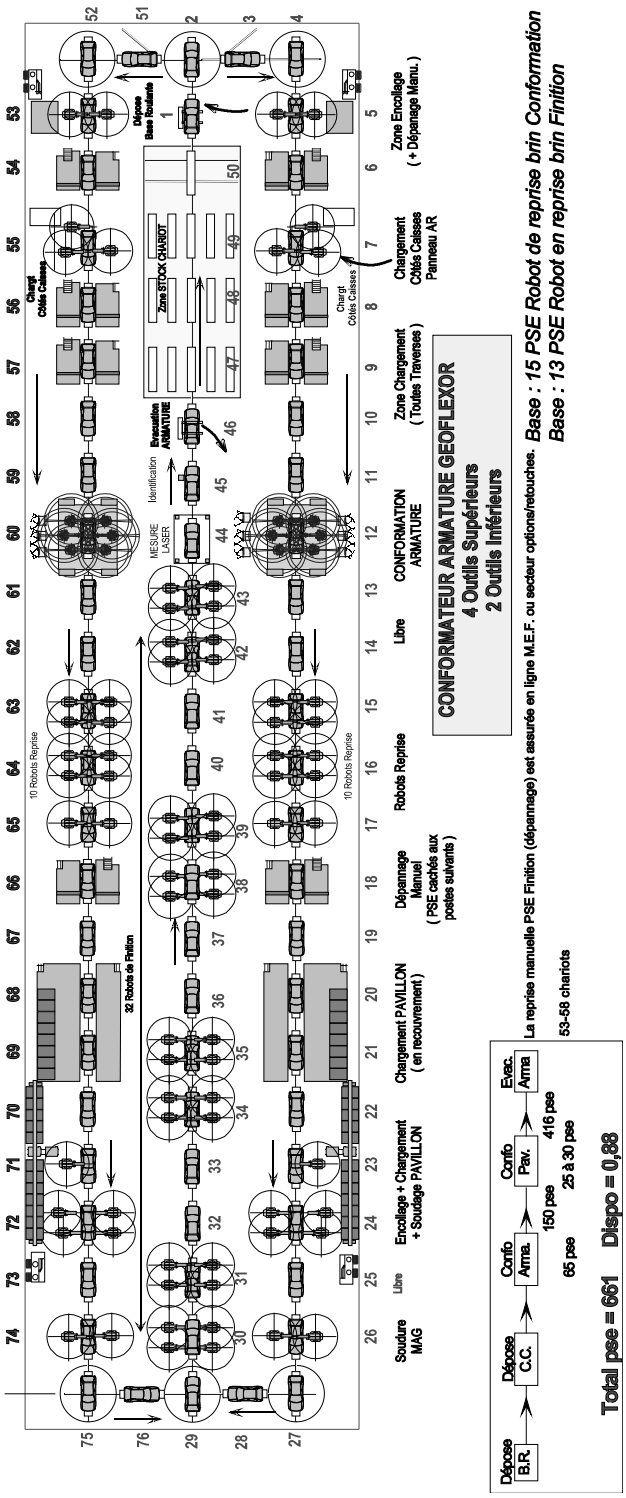

FIG. 8.1 – Vue schématique de la ligne siamoise d'un atelier de ferrage de PSA.

– la branche inférieure : c'est la partie de la ligne constituée des postes 3 à 28.

Les postes robotisés sont ceux sur lesquels il y a un ou plusieurs robots, symbolisés par des cercles avec un robot en leur centre.

On constate ainsi qu'un poste robotisé peut avoir entre un et six robots qui fonctionnent en parallèle sur la pièce. Le mode de fonctionnement est le suivant : en cas de panne de l'un des robots, tous les robots travaillant sur le poste sont arrêtés afin que l'opérateur puisse intervenir.

Les postes 2 et 29 sont des postes de routage. La politique de routage est décidée en fonction des objectifs de production.

La partie de la ligne consituée des postes 47 à 50 correspond à la zone où sont stockés les chariots transporteur. Il est ainsi facilement possible d'ajouter ou de retirer des chariots circulant dans la ligne.

Les châssis de voiture nus sont chargés sur des chariots au niveau du poste 1, qui est le point d'entrée de la ligne. De même, la caisse de la voiture, après usinage, est déchargée sur le poste 46, en sortie de ligne.

On fait l'hypothèse que le poste 1 n'est jamais en famine de châssis, et que le poste 46 n'est jamais bloqué par le processus de déchargement de la caisse.

On connaît les paramètres de performance de chaque robot que l'on attribue à un poste (l'ensemble des robots disponibles est stocké dans la *base de fiabilité*). Pour chaque poste robotisé, on peut définir un *temps de cycle*, qui n'est autre que le temps de traitement sur le poste.

Enfin, les temps de transfert des chariots, de poste en poste, ne sont pas nuls, mais définis par le *temps de transfert unitaire*, qui est une caractéristique de la ligne.

8.1.2 Cahier des charges

L'objectif est de proposer une application façile d'utilisation, avec une interface utilisateur graphique, permettant de créer, de paramétrer et de modifier une ligne siamoise (ou, accessoirement, une ligne fermée simple) d'un atelier de ferrage autombile chez PSA. Cet outil doit pouvoir être utilisé par le chef d'atelier pour vérifier et valider les modifications qu'il veut apporter à la configuration de la ligne en fonction des objectifs de production qui lui sont demandés.

L'application utilise comme moteur d'évalation des performances les méthodes analytique présentées dans les précédents chapitres.

On veut pouvoir :

– ajouter ou supprimer des postes ;
– définir un poste comme mort, robotisé ou comme un plateau tournant ;
– pour un poste robotisé, définir quels sont les robots qui travaillent dessus, en choisissant dans la base de fiabilité ;
– la base de fiabilité doit pouvoir être modifiée par l'utilisateur ;
– définir et modifier les caractéristiques globales de la ligne ;
– une fois la ligne parfaitement caractérisée, pourvoir effectuer une évaluation de productivité horaire, ainsi que sa disponibilité globale ;
– obtenir au besoin des résultats détaillés pour chaque poste robotisé et chaque stock défini par une suite continue de postes morts.

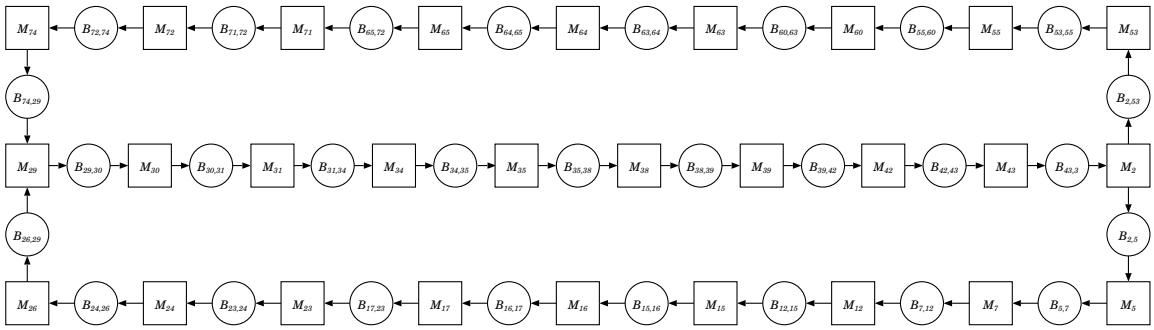

FIG. 8.2 – Modèle asynchrone discret de la ligne de production siamoise.

8.2 Méthode d'analyse de la ligne siamoise

8.2.1 Postes robotisés

Chaque poste robotisé peut avoir entre 1 et 6 robots qui travaillent en parallèle, et qui ne sont pas nécessairement des robots de même type (ayant des paramètres de fonctionnement différents). On fait donc une aggrégation de ces robots, pour n'avoir qu'une seule machines dont le fonctionnement doit être semblable aux robots fonctionant ensembles. Le temps de cycle (temps de traitement) est une valeur globale du poste, elle n'est donc pas affectée par cette transformation.

On suppose qu'il y a K robots fonctionnant en parallèle. Pour chaque robot R_i, on connaît ses paramètres de performance : $MTTF_i$ et $MTTR_i$. On cherche les taux (λ_{eq}, taux du temps de bon fonctionnement, et μ_{eq}, taux du temps de réparation) de la machine se comportant, en moyenne et en isolation, comme le poste correspondant. On rappelle que si l'un des robots tombe en panne, tous les robots s'arrêtent. Pendant la période de réparation, aucun autre robot ne peut tomber en panne, puisque'ils sont arrêtés. Une fois le robot réparé, tous les robots reprennent simultanément leur traitement à l'endroit où il avait été interrompu. Les paramètres agrégés sont donc :

$$\lambda_{eq} = \sum_{i=1}^{i \leq K} \lambda_i \mu = \frac{1}{\sum_{i=1}^{i \leq K} \frac{\lambda_i}{\lambda_{eq}} \frac{1}{\mu_i}}$$

Les taux des temps de bon fonctionnement s'aditionnent puisque les robots fonctionnent en parallèle, augmentant ainsi d'autant la probabilité d'en voir un tomber en panne. Le temps moyen de réparation de la machine équivalente est la moyenne pondérée par ces taux des temps moyens de réparation des robots.

8.2.2 Postes morts

Des postes morts consécutifs forment des zones de stockage. La capacité de chaque zone de stockage correspond simplement au nombre de postes morts consécutifs.

8.2.3 Modèle asynchrone discret

Finalement, le modèle asynchrone discret de cette ligne peut se représenter comme Figure 8.2. On note $C(i,j)^A$ la capacité du stock $B(i,j)$ situé entre les machines M_i (correspondant au poste robotisé i) et M_j (correspondant au poste robotisé j).

Délais

On connaît le temps de transfert unitaire T_{tu}. C'est le temps que met un chariot à aller d'un poste au poste suivant. Le délais total pour une zone de stockage du modèle asynchrone discret est donc :

$$d(i,j) = C(i,j)^A T_{tu}$$

8.2.4 Modèle continu

Le modèle continu que l'on va utiliser, pour pouvoir appliquer les méthodes de décomposition, est ainsi directement issu du modèle asynchrone discret évoqué ci-dessus. Les machines M_i^C ont les mêmes paramètres de performance que ceux du modèle discret, et les stocks ont une capacité qui prend en compte la place disponible sur les machines :

- $\lambda^C = \lambda^A$
- $\mu^C = \mu^A$
- $C(i,j)^C = C(i,j)^A + 1$
- $d(i,j)^C = C(i,j)^C T_{tu}$

Par contre, les mécanismes de routage ne se transposent pas aisément au modèle fluide.

Transformation du routage en assemblage/désassemblage

On se propose d'utiliser les travaux publiés dans [**?**]. Il s'agit de transformer le routage en un mécanisme d'assemblage/désassemblage. En effet, on ne connaît pas, a priori, la suite (déterministe, fonction des objectifs de production) des ordres de routage sur le poste 2. On le considère donc comme un routage probabiliste (ce que l'on peut toujours faire quand on connaît la suite d'ordres de routage). On définit ainsi par α_{sup} la probabilité que le routage se fasse vers la branche supérieure, et α_{inf} la probabilité que le routage se fasse vers la branche inférieure ($\alpha_{sup} + \alpha_{inf} = 1.0$). Si l'on considère ce routage comme probabiliste, on peut faire la transformation évoquée dans [**?**]. Ainsi, au lieu de considérer que $100\alpha_{sup}\%$ des chariots vont vers la branche supérieure, on considère que pour chaque chariot, une proportion α_{sup} de celui-ci va vers la branche supérieure, et le reste vers la branche inférieure. Il faut alors modifier les paramètres des deux branches (inférieure et supérieure) de la ligne en conséquence. Les transformations à apporter sont les suivantes :

- $C_{sup}^C(i,j) = \frac{C_{sup}^A(i,j)+1}{\alpha_{sup}}$
- $C_{inf}^C(i,j) = \frac{C_{inf}^A(i,j)+1}{\alpha_{inf}}$
- $U_{sup}^C(i) = \frac{1}{\alpha_{sup} T_i^A}$
- $U_{inf}^C(i) = \frac{1}{\alpha_{inf} T_i^A}$
- $d^C(i,j) = d^A(i,j)$

Délais dans les zones de stockage

On peut prendre en compte les délais de transport des pièces par les transporteurs grâce à la technique proposée dans [8]. On considère ces délais comme des réductions virtuelles des places disponibles dans les zones de stockage. En effet, si l'on observe un stock vide entre deux machines M_1 et M_2 qui produisent en un temps T, et dont le délais de transfert sont T_c, la machine M_1 va devoir produire $1 + \frac{T}{T_c}$ avant que M_2 ne commence à recevoir du matériel. Les $\frac{T}{T_c}$ unités de matériel supplémentaires ne sont donc nécessaires que pour faire la "connection" entre les deux machines. La zone de stockage apparaît donc virtuellement plus petite de la quantité $\frac{T}{T_c}$.

Il suffit alors de faire la transformation $C_{eq} = C - \frac{T}{T_c}$ à chaque zone de stockage, et de traiter la ligne résultante comme une ligne dont les transferts dans les stocks sont instantanés. Cette ligne correspond alors à un cas que l'on sait traiter.

Cependant, il faut faire attention, au moment de l'évaluation des encours moyens, à faire, après l'évaluation du dipôle (qui dépend des paramètres de performances du dipôles), la transformation inverse :

$$h_m^{corrigé} = h_m^{eval}(1 - ps)\frac{T}{T_c}C$$

Homogénéisation

Enfin, reste à traiter le problème lié à inhomogénéité de la ligne. En effet, rien n'impose que les temps de traitement soient tous les mêmes. Nous avons choisi d'utiliser la technique d'homogénéisation RSA [13] rappelée Section 2.6.3.

8.3 Présentation de l'application

Le logiciel a été développé en Python (pour toute la partie interface utilisateur) et en C++ (pour l'algorithme d'analyse). Une capture d'écran est visible Figure 8.3 page ci-contre.

Il est également possible de tracer la courbe de la capacité horaire de production de la ligne en fonction de plusieurs paramètres, comme par exemple le nombre total de chariots circulant dans la ligne (Figure 8.4 page 150).

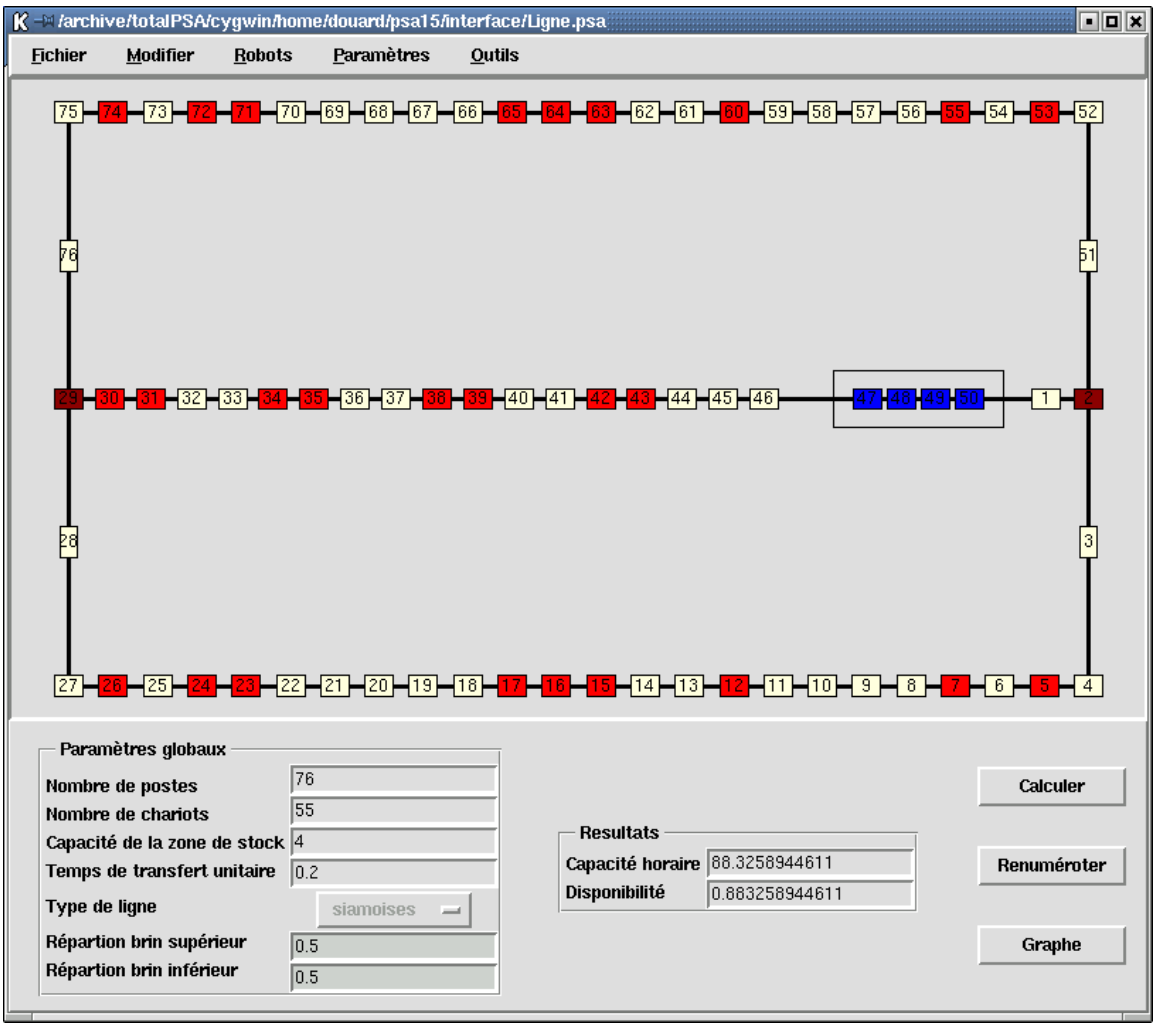

FIG. 8.3 – Capture d'écran de l'application DispoPSA. Exemple de ligne siamoise.

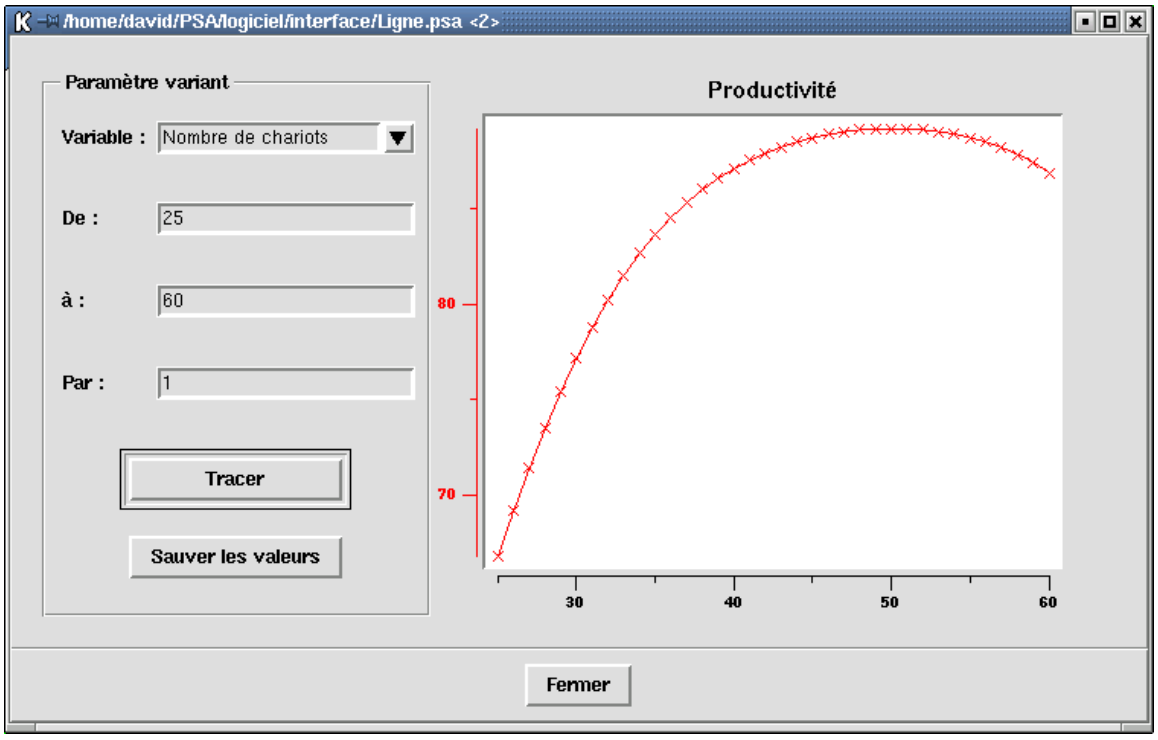

FIG. 8.4 – Capture de la fenêtre de l'application DispoPSA présentant la productivité horaire en fonction du nombre de chariots circulant dans la ligne.

Quatrième partie

Conclusion

9

Conclusion

Par l'observation des besoins spécifiques d'un industriel, PSA Peugeot Citroën, nous avons pu proposer un certain nombre d'améliorations aux techniques courantes d'analyse de lignes de production manufacturières basées sur l'algorithme DDX. Cela nous a conduit à développer un logiciel qui nous a permis de mettre en œuvre un certain nombre des résultats de nos travaux, et de rendre ces outils facilement utilisables dans un cadre bien précis : l'estimation rapide des performance d'un atelier de ferrage du constructeur automobile PSA.

D'un point de vue méthodologique, nous avons toujours privilégié des procédés simples pour aborder chaque nouvelle problématique. Ainsi, nous n'avons pas toujours cherché à réécrire l'ensemble des équations pour tenir compte des nouveaux phénomènes. Nous avons fait ce choix pour ne pas trop complexifier les méthodes de résolution. En effet, les méthodes analytiques perdent de leur intérêt lorsqu'elles sont lentes et complexes. De plus, les gains en précision s'avèrent souvent assez faibles. En outre, les paramètres du systèmes étant eux-mêmes approximatifs, des résultats très précis n'apportent pas beaucoup plus d'informations. Il est par exemple pratiquement impossible d'avoir une idée cv^2 du temps de bon fonctionnement d'une machine ; la moyenne de ce temps n'est souvent pas connue avec une très grande précision.

De ce fait, lorsque nous nous sommes intéressés aux mécanismes de panne dans les zones de stockage, nous avons présenté une étude fine de ces mécanismes de pannes, pour bien les comprendre. Puis, nous avons proposé des méthodes simples pour intégrer ces mécanismes de pannes à la méthode de décomposition de ligne DDX. Ces méthodes reposent sur une pré-transformation de la ligne afin de pouvoir l'analyser avec un DDX classique. Elles sont donc extrêmement fiables et simples à mettre en œuvre, et leurs résultats sont en général très satisfaisants. Le choix de la simplicité, au détriment de l'élaboration d'une nouvelle technique d'analyse intégrant de manière plus fine ces pannes de stocks, est ainsi justifié. Le gain eût été non significatif pour une complexité de mise en oeuvre sans mesure.

Une autre problématique importante que nous avons abordé nous est apparue lors de l'étude d'une unité particulière de production manufacturière (ligne siamoise). Nous avons alors été amenés à nous intéresser à la façon dont on peut adapter l'algorithme classique DDX à sa topologie.

L'algorithme DDX simple a déjà été transposé au cas des lignes fermées, mais la technique originale (voir [20, 17]) n'est pas du tout efficace en termes de temps de calcul et de robustesse. Ces défauts la rendent inutilisable en pratique. Nous nous sommes donc intéressé au fonctionnement fin de cet algorithme et nous nous sommes attaché à comprendre les raisons de son inefficacité algorithmique.

Nous avons ainsi pu proposer une nouvelle technique algorithmique de résolution des lignes

fermées simples. Cette nouvelle technique, en dehors de son efficacité informatique et de sa robustesse, offre l'avantage de pouvoir s'adapter naturellement aux cas de lignes plus complexes (ligne comportant plusieurs boucles).

Nous avons aussi étendu cette technique aux lignes fermées de topologies quelconques. Il a fallu, pour cela, résoudre d'autres problèmes liés à cette topologie complexe ; en particulier, il nous a fallu déterminer les invariants de population et comprendre leurs conséquences sur les équations traditionnelles (construites selon la même technique que pour le DDX simple).

Lorsque nous avons abordé ces situations de lignes complexes, nous avons utilisé les versions sophistiquées des équations du DDX où les dipôles ont des temps de réparation qui peuvent être exponentiels (DDX original), général-exponentiels, ou hyper-exponentiels à deux étages. En effet, les travaux antérieurs sur ces améliorations de l'algorithme DDX pour les lignes ouvertes avaient conclus que les résultats obtenus étaient sensiblement plus précis avec la technique la plus sophistiquée (HE_2). Nous espérions ainsi obtenir des résultats d'assez bonne qualité pour nos lignes fermées aux topologies complexes. Nous savions que les techniques de décomposition employées risquaient de poser des problèmes de précision des résultats ; fondamentalement, ces techniques de décomposition estiment les paramètres de performance de la ligne en décorrélant le fonctionnement local de la ligne (autour de chaque buffer, via les dipôles que l'on observe) du reste de la ligne. Or, quand les lignes comportent des boucles (parce qu'elles sont fermées ou du fait qu'un mécanismes de désassemblage est suivi par un assemblage, créant ainsi une boucle), les phénomènes de corrélations sont importants dans le fonctionnement global de la ligne.

Malgré cela, la technique donne de bons résultats lorsqu'on l'observe autour de son point de fonctionnement optimal. Elle est d'autant mieux adaptée que les lignes étudiées comportent de grands invariants (en nombre de machines comme en capacité).

9.1 Améliorations envisageables

La nouvelle méthode que nous proposons (pour analyser les lignes fermées comportant plusieurs boucles) donne des résultats satisfaisants dans l'ensemble. On pourrait cependant chercher à l'améliorer, particulièrement lorsqu'on s'éloigne du point de fonctionnement optimal de la ligne. Nous avons ainsi commencé à explorer quelques idées, mais elles restent, pour l'essentiel, à affiner. Il serait nécessaire de comprendre très finement les mécanismes sous-jacents qui induisent les corrélations pour en proposer une version efficace et justifiable.

A titre d'exemple, on peut obtenir de meilleurs résultats sur l'exemple exposé Figure 6.25 page 119, avec la situation évoquée Section 6.5. Dans le cas où les machines M_3 et M_4 sont identiques, elles vont en moyenne tomber en panne alternativement. On peut pousser encore la simplification du modèle, en considérant que les temps de bon fonctionnement de ces deux machines sont constants (et non plus aléatoires), les pannes de ces machines sont alors strictement alternées. Ces deux machines, encadrées de leurs buffers avec les buffers, vont se comporter virtuellement comme si leur temps de réparation était plus court ; ce délais correspond au temps nécessaire pour inverser l'état des 4 stocks qui encadrent ces machines. Ce raisonnement est bien sûr très simplificateur, mais des processus de ce type ont lieu, et on peut chercher des techniques de transformation qui vont tenter de les compenser.

Nous avons ainsi appliqué ce genre de raisonnement sur l'exemple présenté Section 6.3.1, avec $C = 5$ (Figure 6.12 page 110), et nous avons obtenu les résultats présentés Figure 9.1 page ci-contre. Ces résultats sont très encourageants. Nous avons appliqué une pré-transformation où les taux des temps de réparation sont corrigés d'un facteur dépendant du taux de remplissage des invariants de population. Cette méthode n'est cependant pas toujours concluante. Nous avons

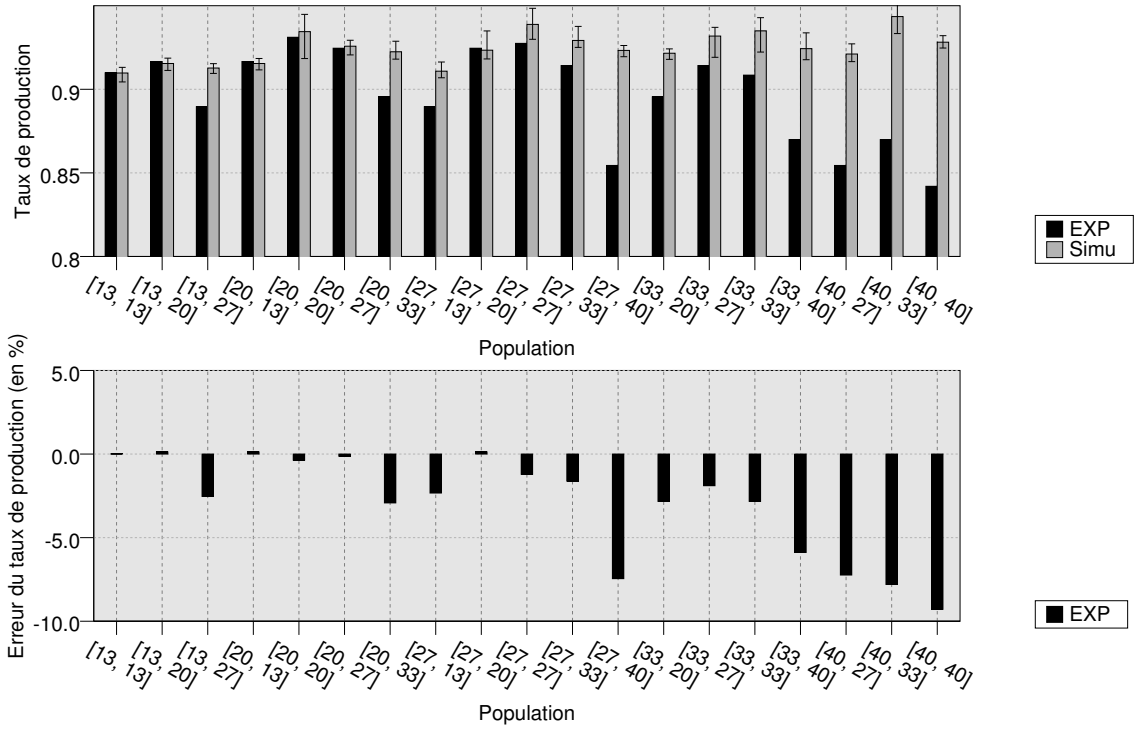

FIG. 9.1 – Productivité de la ligne présentée Section 6.3.1 pour $C = 5$, en appliquant une transformation des paramètres de la ligne.

tenté d'autres transformations, mais aucune d'entre elles n'étaient vraiment meilleure ni ne se justifiait véritablement. Aussi serait-il intéressant d'explorer cette voie, de bien comprendre les mécanismes en jeu et de trouver une heuristique donnant de meilleurs résultats, que l'on pourrait (au moins partiellement) justifier, et qui n'induirait aucune complexification de la méthode d'analyse.

La simplicité et l'efficacité des méthodes que l'on propose sont des atouts importants pour étudier les lignes fermées de topologies quelconques.

Bibliographie

[1] T. Altiok. On the phase-type approximations of general distributions. *IEE Transactions.*, 17(2) :110–116, 1985.

[2] F. Baskett, K.M. Chandy, R.R. Muntz, and F.G. Palacios. Open, closed and mixed networks of queues with different classes of customers. *J. ACM*, 22(2) :248–260, 1975.

[3] Mitchell H. Burman. *New Results in Flow Line Analysis.* PhD thesis, Massachussets Institute of Technology, June 1995.

[4] G. Buxey, N. Slack, and R. Wild. Production flow line systel design - a review. *AIIE Transactions*, 48 :37–48, 1973.

[5] J. A. Buzacott and L. E. Hanifin. Models of automatic transfer lines with inventory banks - A review and comparison. *AIIE Transactions*, 10(2) :197–207, 1978.

[6] J. A. Buzacott and J. G. Shanthikumar. *Stochastic Models of Manufacturing Systems.* Prentice Hall, Englewood Cliffs, NJ, 1993.

[7] M. Chaperon. *Calcul différentiel et calcul intégral.* Dunod, September 2003.

[8] C. Commault and A. Semeray. Taking into account delays in buffers for analytical performance evaluation of transfer lines. *IEE Transactions*, 22(2) :133–142, June 1990.

[9] Y. Dallery. On modeling failure and repair times in stochastic models of manufacturing systems using generalized exponential distributions. *Queuing Systems*, 15 :199–209, 1994.

[10] Y. Dallery, R. David, and X.-L. Xie. An efficient algorithm for analysis of transferlines with unreliable machines and finite buffers. *IIE Transactions*, 20 :280–283, 1988.

[11] Y. Dallery, R. David, and X.-L. Xie. Approximate analysis of transfer lines with unreliable machines and finite buffers. *IEEE Transactions on Automatic Control*, 34(9) :943–953, 1989.

[12] Y. Dallery and S. B. Gershwin. Manufacturing flow line systems : A revies of models and analytical results. *Queueing Systems*, 12 :3–94, 1992.

[13] Y. Dallery and H. Le Bihan. Homogeneisation techniques for the analysis of production lines with unreliable machines having different speeds. *European Journal of Control*, 3 :200–215, 1997.

[14] R. David, X. Xie, and Y. Dallery. Properties of continuous models of transfer lines with unreliable machines and finite buffers. *IMA J. of Mathematics in Business and Industry*, 6 :281–308, 1990.

[15] M. Di Mascolo. Méthode analytique d'évaluation des performances d'une ligne d'assemblage. Rapport de dea, Laboratoire d'Automatique de Grenoble, 1988.

[16] M. Di Mascolo, R. David, and Y. Dallery. Modeling and analysing of assembly systems with unreliable machines and finite buffers. *IIE Transactions*, 23(4) :315–330, 1991.

[17] D. Douard and B. Baynat. Improving efficiency of decomposition methods for closed-loop continuous-flow production models. In *Ninth international multi-conference on advanced computer systems*. Technical University of Szceczecin, October 2002.

[18] D. Dubois and J.-P. Forestier. Productivité et en-cours moyen d'un ensemble de deux machines séparées par une zone de stockage. *RAIRO Automatique*, 16(2) :105–132, 1982.

[19] G. Ewing, D. McNickle, and K. Pawlikowski. Spectral Analysis for Confidence Interval Estimation under Multiple Replications in Parallel. *Proc. European Simulation Symposium*, 2002.

[20] Y. Frein, C. Commault, and Y. Dallery. Modeling and analysis of closed-loop production lines with unreliable machines and finite buffers. *IIE Transactions*, 28 :545–554, 1996.

[21] S. B. Gershwin. An efficient decomposition method for approximate evaluation of tandem queues with finite storage space and blocking. *Opeartions Research*, 35 :291–305, 1987.

[22] S.B. Gershwin. Assembly/disassembly systems : An efficient decomposition algorithm for tree-structured nerworks. *IEE Transactions*, 23(4) :302–314, 1991.

[23] S.B. Gershwin and M.H. Burman. A decomposition method for analysing inhomogeneous assembly/disassembly systems. *Ann. Ope. Research*, 1998.

[24] S.B. Gershwin and I.C. Schick. Modelling and analysis of three-stage transfer lines with unreliable machines and finite buffers. *Operation Research*, 31(2) :354–380, 1983.

[25] S. Helber and H. Jusic. A new decomposition approach for non-cyclic continuous material flow lines with a merging flow of material. 2003.

[26] J.R. Jackson. Job-shop-like queueing systems. *Manag. Sci.*, 10(1) :131–142, 1963.

[27] H. Le Bihan. *De nouvelles méthodes analythiques pour l'évaluation des performances de lignes de production.* PhD thesis, Université Pierre et Marie Curie, 1998.

[28] R. Levantesi. *Analysis of Multiple Loop Assembly/Disassembly Networks*. PhD thesis, Politecnico di Milano, 2002.

[29] X.-G. Liu and J.A. Buzacott. Approximate models of assembly systems with finite banks. *European Journal of Operational Research*, 45 :143–154, 1990.

[30] N. Maggio. An analytical method for evaluating the performance of closed loop production lines with unreliable machines and finite buffer. Master's thesis, Politecnico di Milano, 2000.

[31] K Pawlikowski. The Akaroa Project. http ://www.cosc.canterbury.ac.nz/research/RG/net_sim/simulation_group

[32] K. Pawlikowski. Steady-state simulation of queueing processes : a survey of problems and solutions. *ACM Computing Surveys*, 22(2) :124–170, 1990.

[33] Proceedings of the European Simulation Multiconference, editor. *The OMNeT++ Discrete Event Simulation System*, June 2001.

[34] D. Sanchez. *Ordinary Differential Equations*. M.A.A., March 2003.

[35] A. Varga. OMNeT++. In *Software Tools for Networking*, volume 16. IEEE Network Interactive, June 2002.

[36] A. Vargas. OMNeT++ : a discrete event simulation environment.

Cinquième partie

Annexes

A

Résolution des systèmes d'équations différentielles pour l'analyse exacte du dipôle

B

Détermination d'une plus petite famille de semi-flots

On note ici C la matrice du réseau de Petri. Cette matrice est de la forme :

$$C = \begin{array}{c|cccccccc} & M_1 & M_2 & \dots & M_i & \dots & M_j & \dots & M_K \\ \hline B_{1,2} & +1 & -1 & \dots & 0 & \dots & 0 & \dots & 0 \\ \vdots & \multicolumn{8}{c}{\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots} \\ B_{i,j} & 0 & 0 & \dots & +1 & \dots & -1 & \dots & 0 \\ \vdots & \multicolumn{8}{c}{\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots} \end{array}$$

Chaque ligne de la matrice correspond à une liaison entre deux transitions (donc entre deux machines), et donc correspond à une place (un buffer). On met la valeur $+1$ dans la colonne correspondant à la transition source, et -1 dans la colonne de la transition destination, et 0 partout ailleurs.

Algorithme B.1 Recherche d'une plus petite famille génératrice de semi-flots

Requiert : la matrice C

1: **tant que** il existe une ligne et une colonne **faire**
2: choisir une colonne k
3: **pour** tout couple de lignes i et j tel que $C_{i,k} > 0$ et $C_{j,k} < 0$ **faire**
4: ajouter la ligne $((C_{i,k} \cdot j - C_{j,k} \cdot i)/pgcd(C_{i,k}, C_{j,k}))$
5: **fin pour**
6: **pour** toute ligne i telle que $C_{i,k} \neq 0$ **faire**
7: supprimer le ligne i
8: **fin pour**
9: **pour** tout couple de lignes i et j tel que $Support(i) \supset Support(j)$ **faire**
10: Supprimer le ligne i
11: **fin pour**
12: supprimer la colonne k
13: **fin tant que**
14: l'ensemble des indices de lignes est une plus petite famille génératrice de semi-flots

Le calcul d'une plus petite famille génératrice de semi-flots est obtenue en appliquant l'algorithme B.1. Le support d'une ligne i est défini comme :

Définition : Soit un vecteur v sur un ensemble E. Le support de v est défini par :
$Support(v) = \{e \in E | v_e \neq 0\}$.
Au cours de l'exécution de l'algorithme B.1, on fait des combinaisons linéaires de lignes. On peut par exemple arriver à la situation où on manipule une ligne définie comme par exemple $B_{1,2} + 2B_{5,4} + B_{5,9}$. Le support d'une telle ligne est alors $\{B_{1,2}, B_{5,4}, B_{5,9}\}$

On obtient donc au final des combinaisons linéaires de places, donc de buffers de la ligne d'origine. Chacune de ces combinaisons de buffers est un invariant de notre ligne.

C

Détails des résultats numériques pour l'étude des pannes de stocks au Chapitre 7

C.1 Lignes symétriques

Nous donnons ici toutes les courbes pour les exemple de lignes simples, homogènes et symétriques. Les tests font varier la taille de la ligne étudiée (3, 5 et 10 machines), et la taille des stocks (3 et 10). Pour chaque cas d'étude, on a utilisé les paramètres suivants :

- $\lambda_i = 0.001$
- $\mu_i = 0.1$
- $T_i = 1.0$
- $\mu_{i,i+1} = 0.1$
- $I_{i,i+1} \in \{0.01, 0.1\}$

et :

- $\lambda_i = 0.001$
- $\mu_i = 0.1$
- $T_i = 1.0$
- $\mu_{i,i+1} = 0.01$
- $I_{i,i+1} \in \{0.01, 0.1\}$

Pour chaque cas de figure, on a fait deux séries de tests analytiques : une où les capacité sont choisies égales à celles du modèle simulé, et l'autre où ces capacités sont augmentées d'un unité (prise en compte de la place disponible sur les machines). On a à chaque fois fait les résultats bruts et les comparaisons entre les différentes méthodes utilisées et la simulation proposés type POF.

C.2 Lignes homogènes quelconques

On présente ici des lignes dont les paramètres on été tirés aléatoirement, tel que :

- $\lambda_i \in [\frac{0.9}{200}, \frac{1.1}{200}]$
- $\mu_i \in [\frac{0.9}{10}, \frac{1.1}{10}]$
- $\lambda_{i,i+1} \in [\frac{0.9}{500}, \frac{1.1}{500}]$ ou
- $\lambda_{i,i+1} \in [\frac{0.9}{1000}, \frac{1.1}{1000}]$ ou
- $\lambda_{i,i+1} \in [\frac{0.9}{2000}, \frac{1.1}{2000}]$ ou

– $\mu_{i,i+1} \in [\frac{0.9}{10}, \frac{1.1}{10}]$

avec des lignes de 3 et 10 machines, pour des capacités de 3 et 10 zones de stockage. On a fait un tirage pour chaque combinaison de ces tailles de ligne et de stockage, soit 12 cas.

Dans chaque table de comparaison, les résultats sont donnés en pourcentage (de l'unité). Les en-cours moyen donnés avec la mention *optim* sont en pourcentage de la capacité totale du buffer, tandis que les autres sont en pourcentage de la valeur simulée.

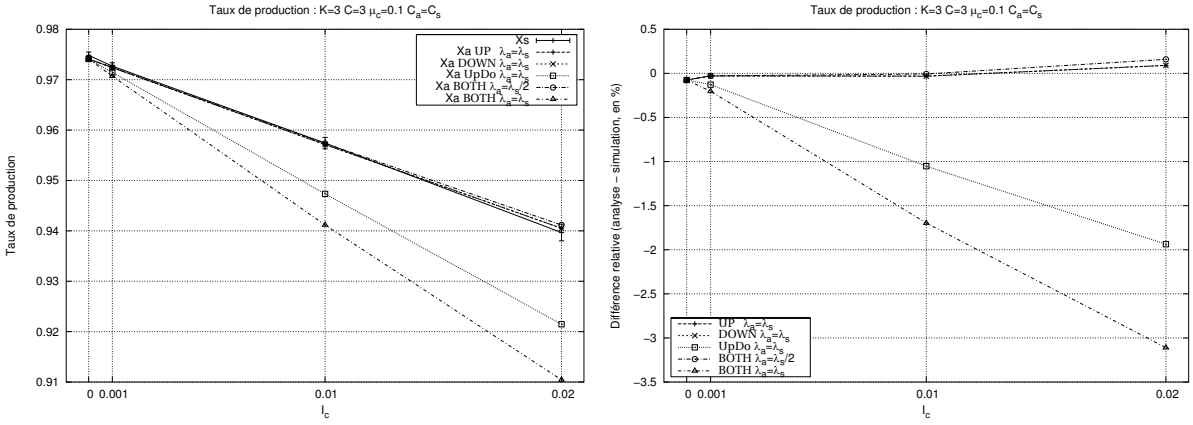

FIG. C.1 – Comparaison des taux de production. 3 machines, stocks = 3

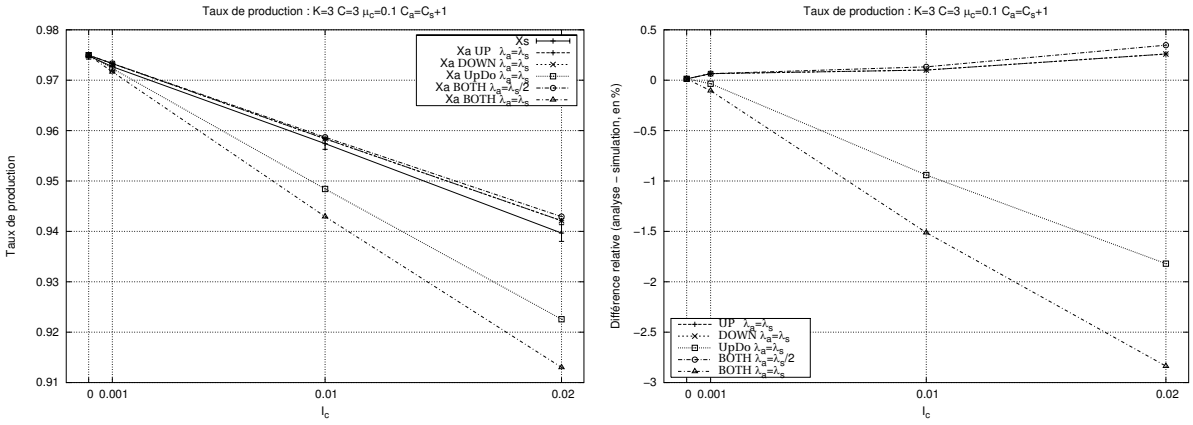

FIG. C.2 – Comparaison des taux de production. 3 machines, stocks = 3

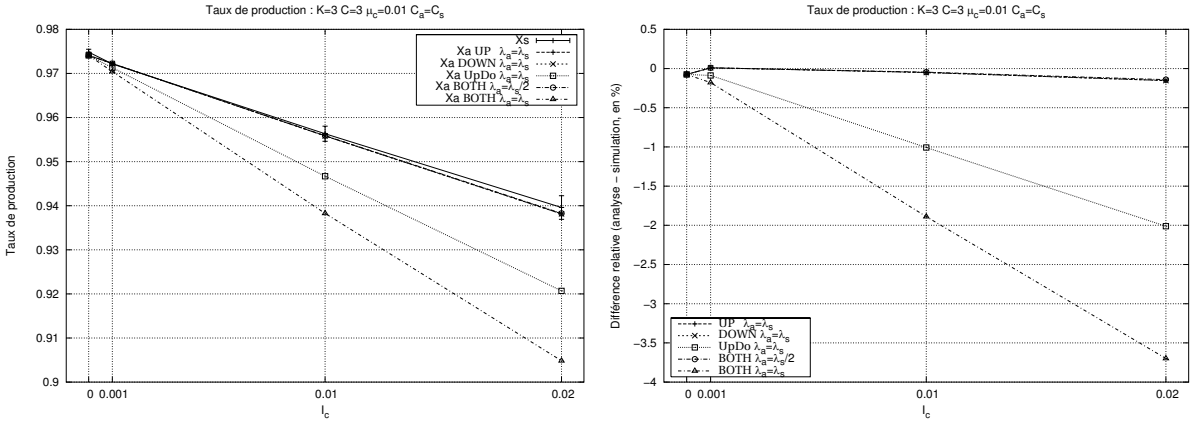

FIG. C.3 – Comparaison des taux de production. 3 machines, stocks = 3

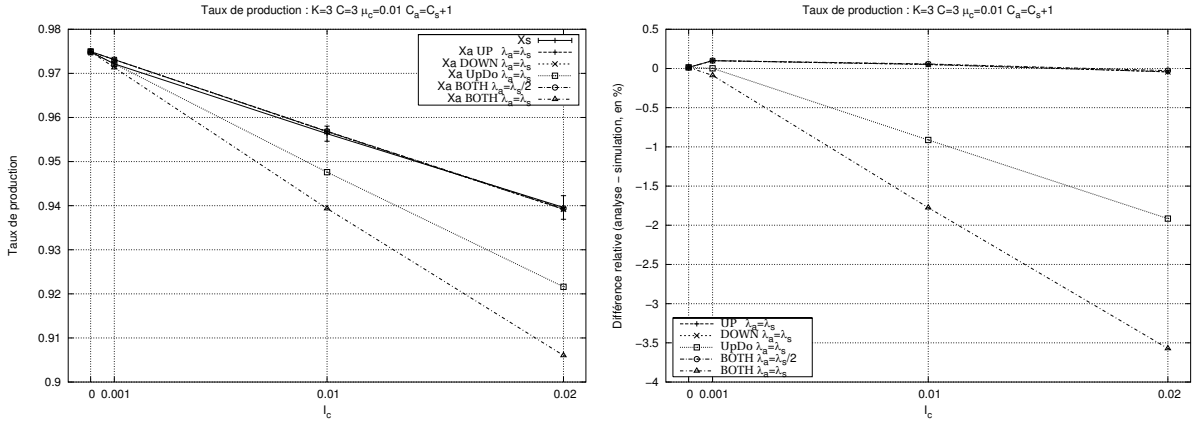

FIG. C.4 – Comparaison des taux de production. 3 machines, stocks = 3

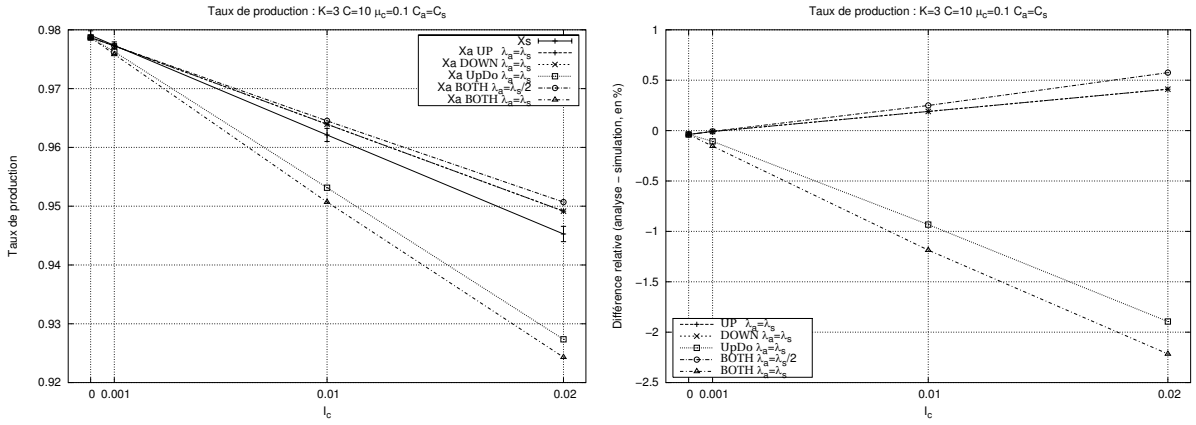

FIG. C.5 – Comparaison des taux de production. 3 machines, stocks = 10

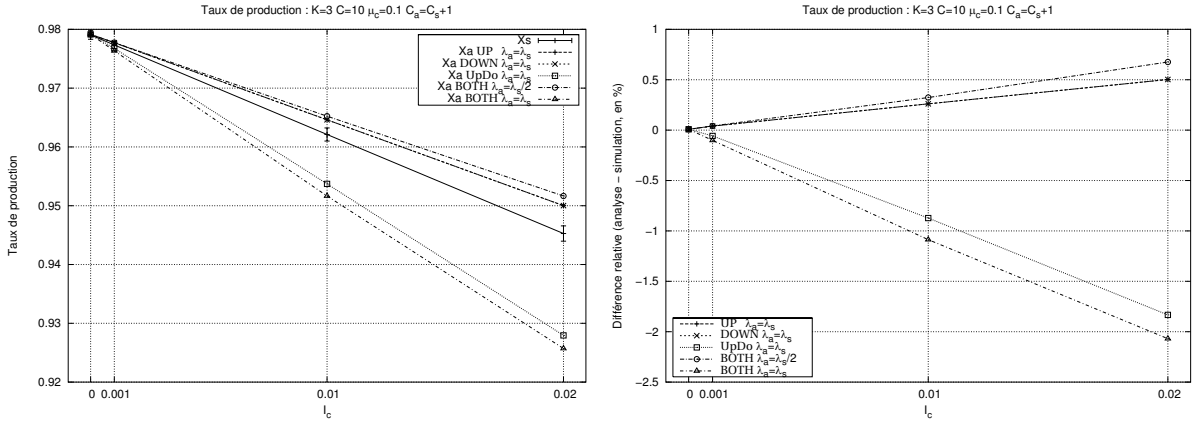

FIG. C.6 – Comparaison des taux de production. 3 machines, stocks = 10

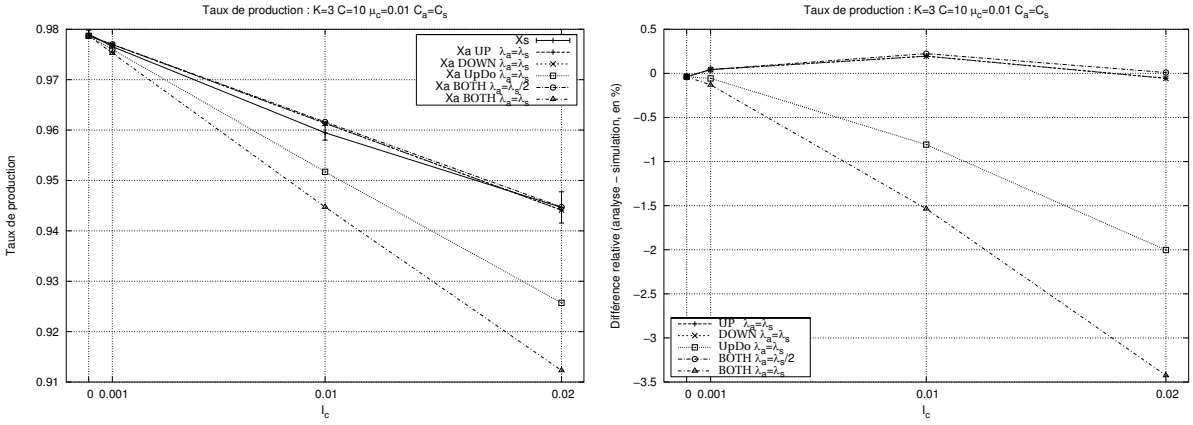

FIG. C.7 – Comparaison des taux de production. 3 machines, stocks = 10

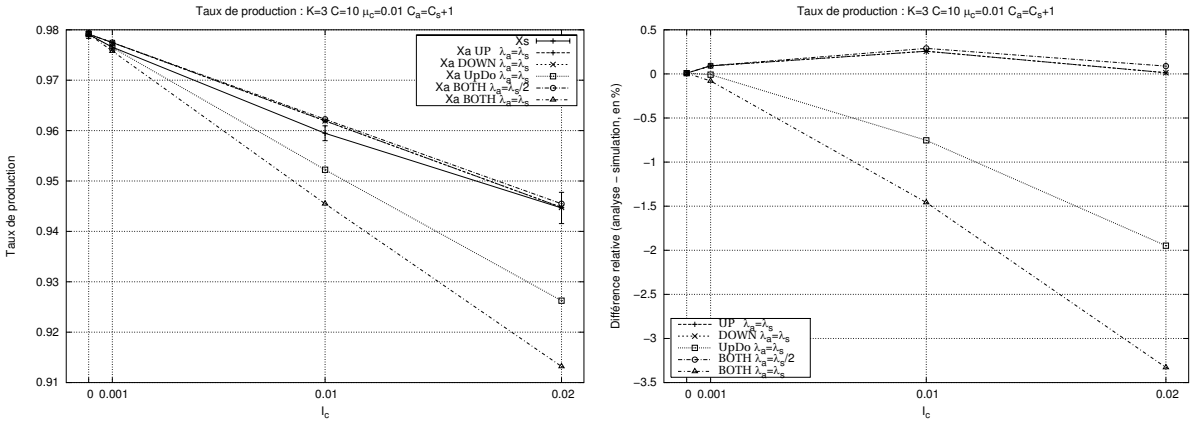

FIG. C.8 – Comparaison des taux de production. 3 machines, stocks = 10

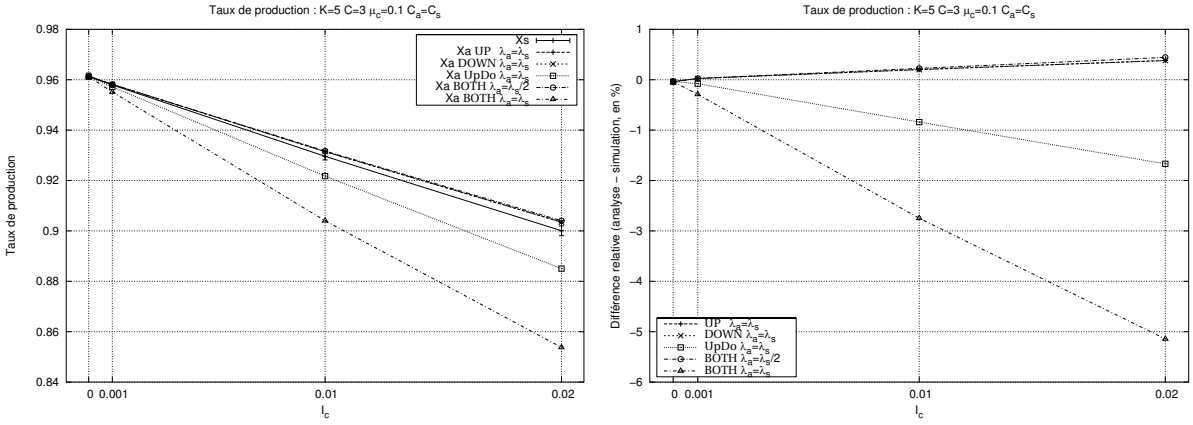

FIG. C.9 – Comparaison des taux de production. 5 machines, stocks = 3

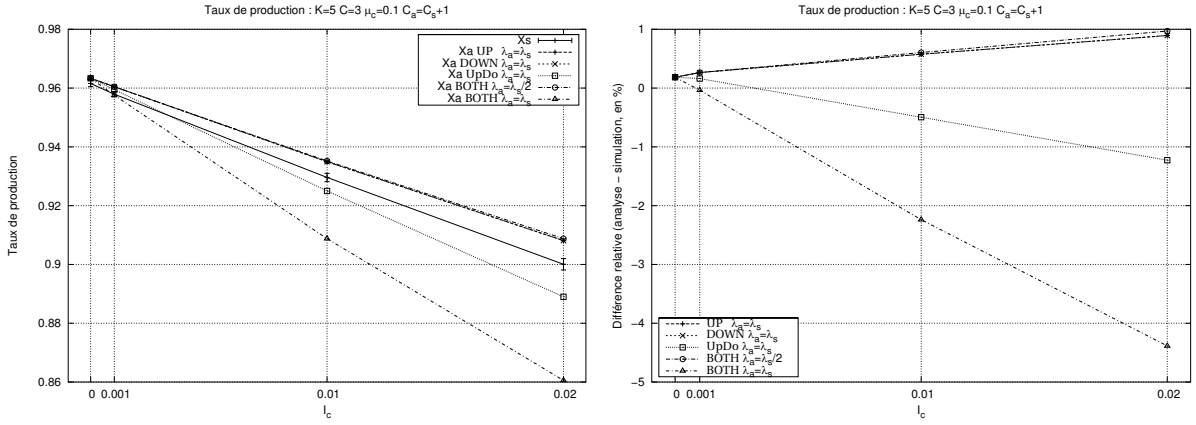

FIG. C.10 – Comparaison des taux de production. 5 machines, stocks = 3

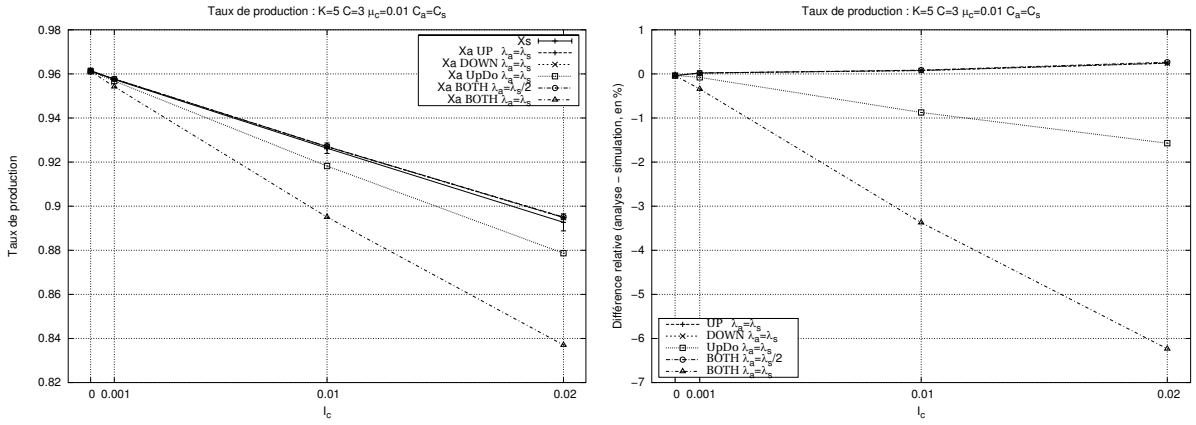

FIG. C.11 – Comparaison des taux de production. 5 machines, stocks = 3

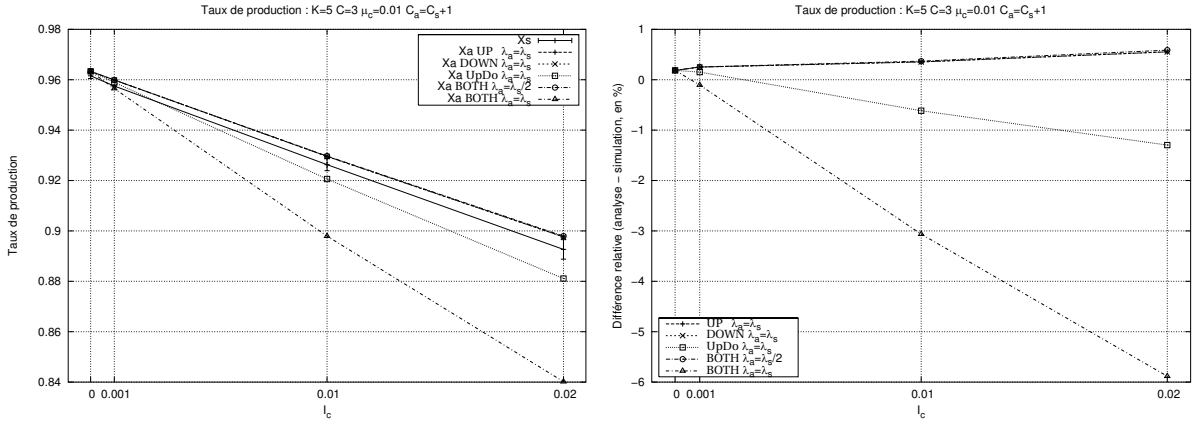

FIG. C.12 – Comparaison des taux de production. 5 machines, stocks = 3

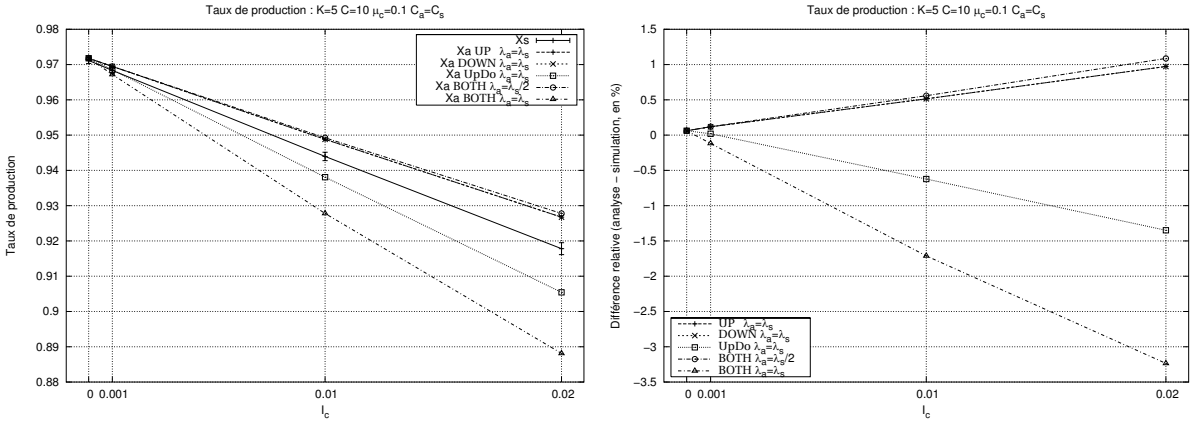

FIG. C.13 – Comparaison des taux de production. 5 machines, stocks = 10

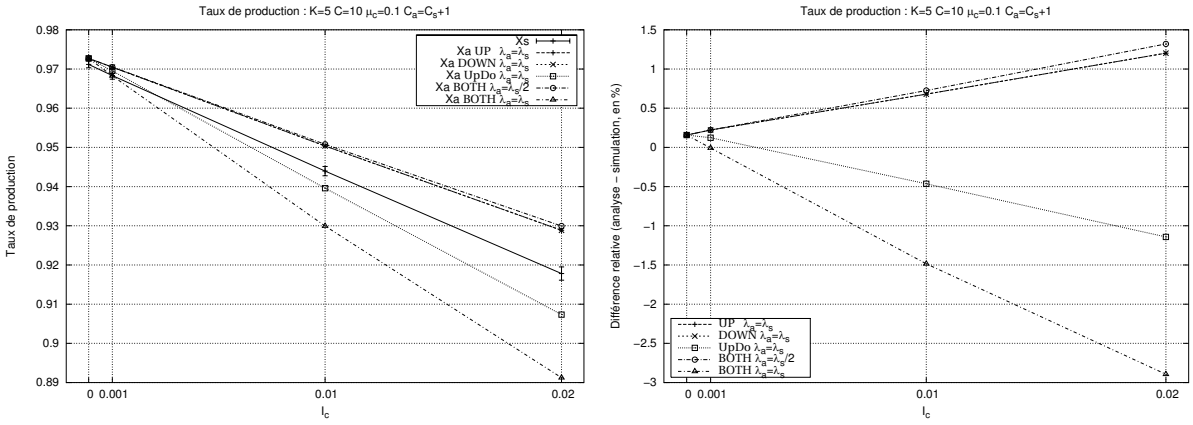

FIG. C.14 – Comparaison des taux de production. 5 machines, stocks = 10

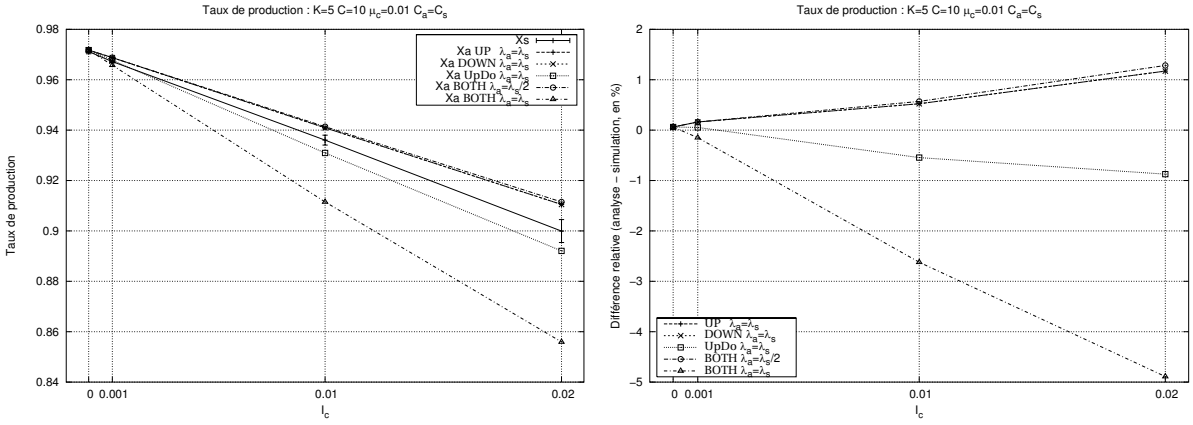

FIG. C.15 – Comparaison des taux de production. 5 machines, stocks = 10

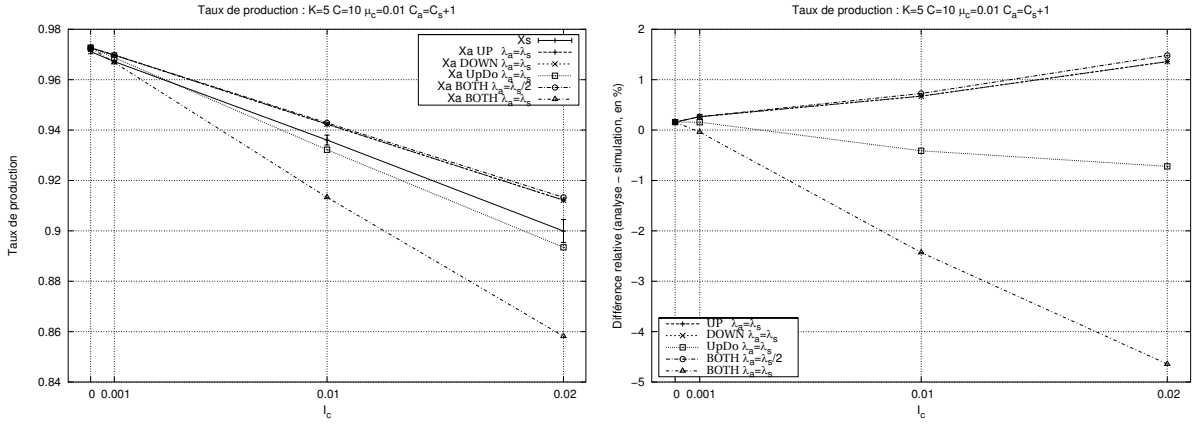

FIG. C.16 – Comparaison des taux de production. 5 machines, stocks = 10

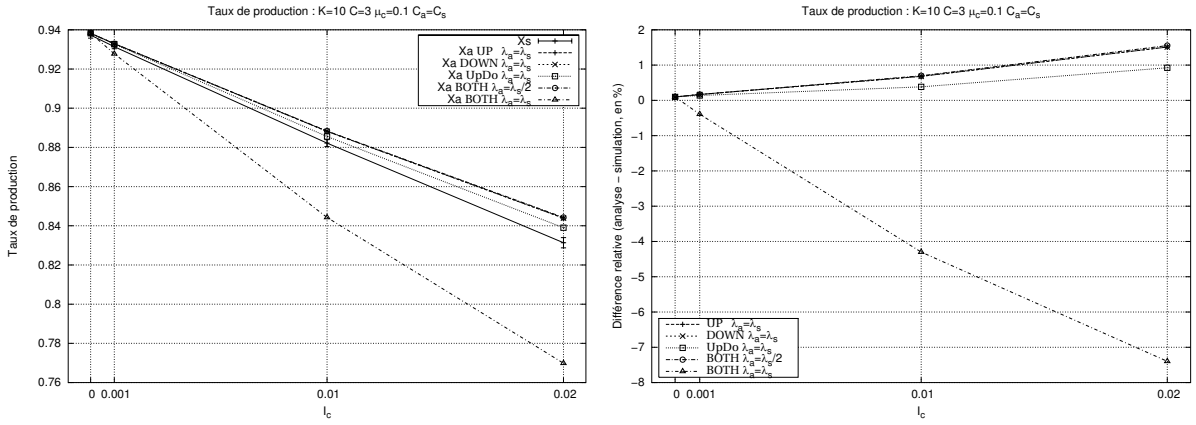

FIG. C.17 – Comparaison des taux de production. 10 machines, stocks = 3

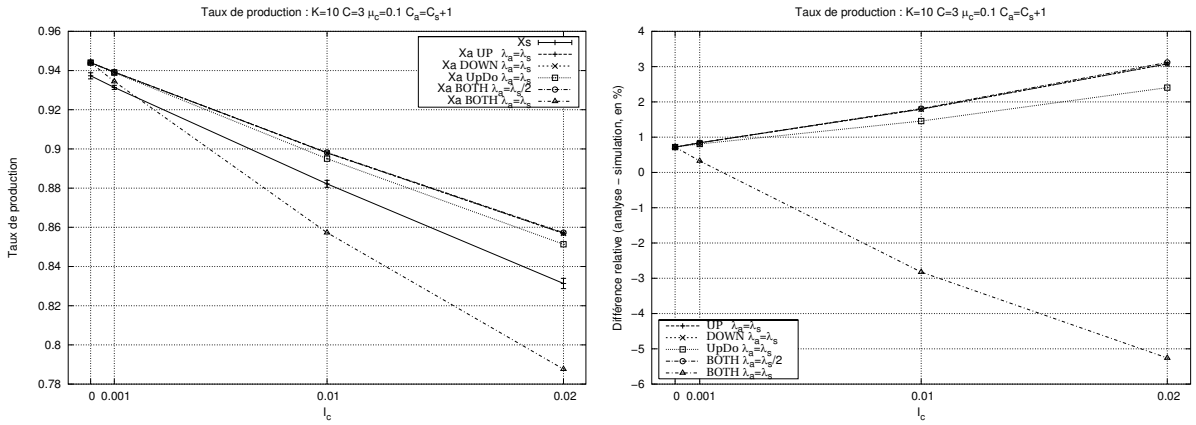

FIG. C.18 – Comparaison des taux de production. 10 machines, stocks = 3

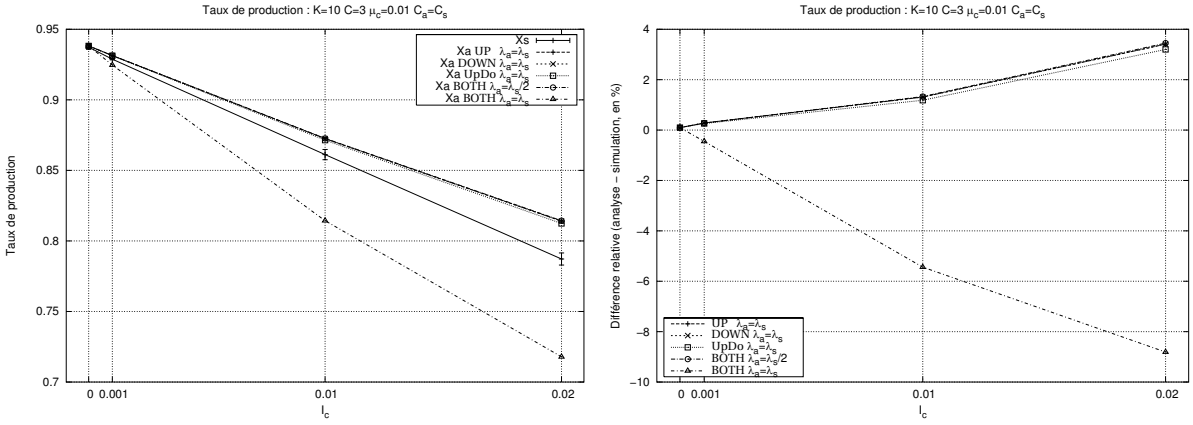

FIG. C.19 – Comparaison des taux de production. 10 machines, stocks = 3

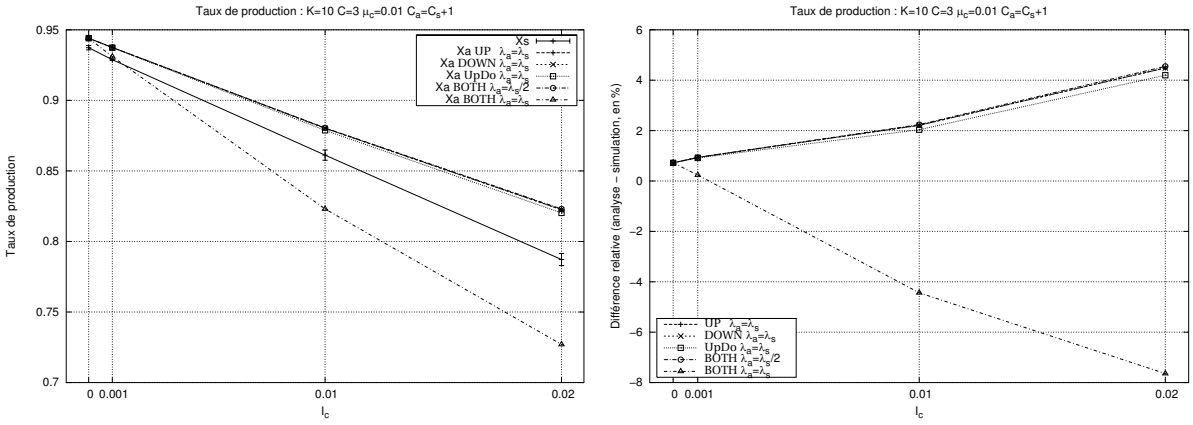

FIG. C.20 – Comparaison des taux de production. 10 machines, stocks = 3

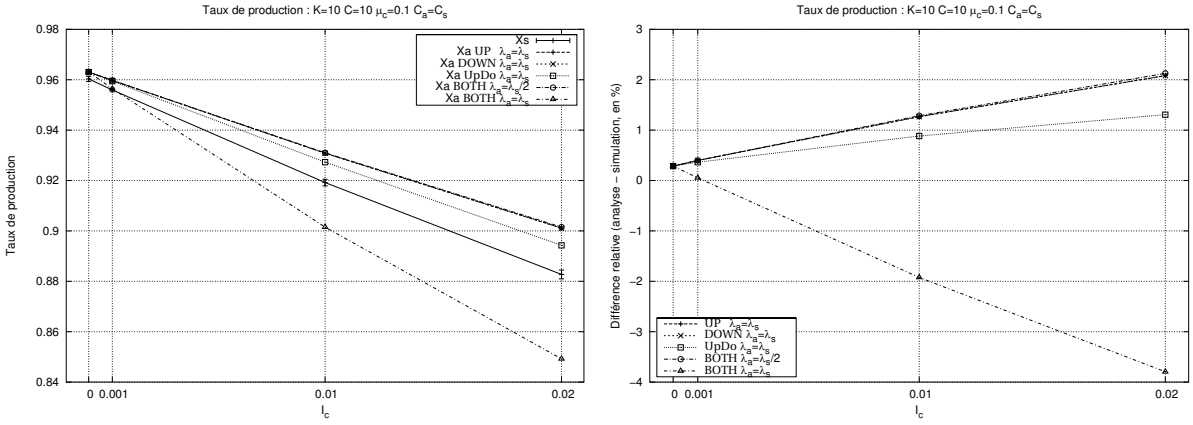

FIG. C.21 – Comparaison des taux de production. 10 machines, stocks = 10

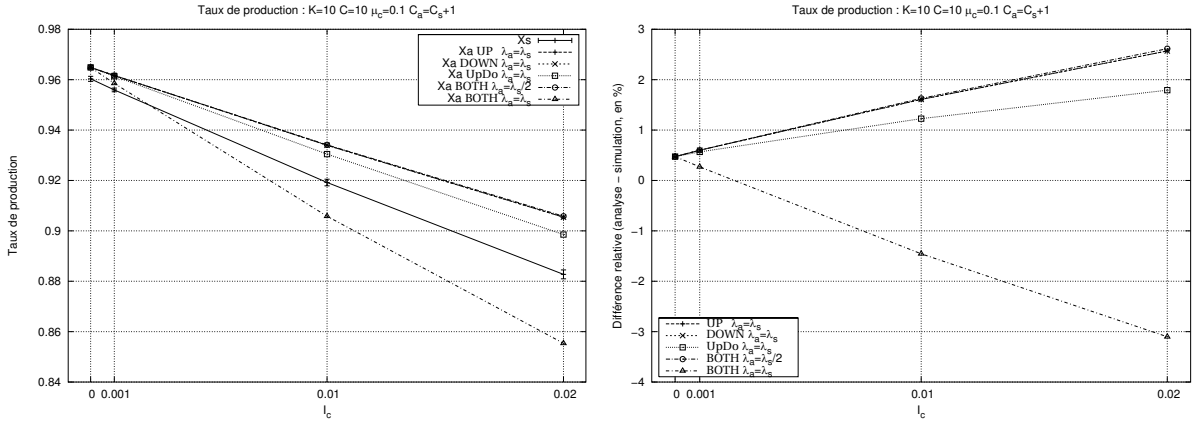

FIG. C.22 – Comparaison des taux de production. 10 machines, stocks = 10

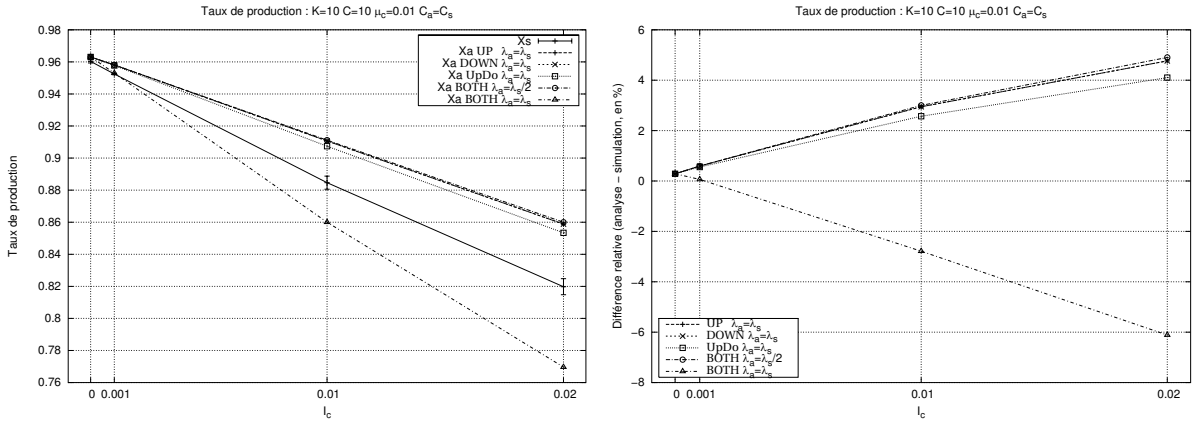

FIG. C.23 – Comparaison des taux de production. 10 machines, stocks = 10

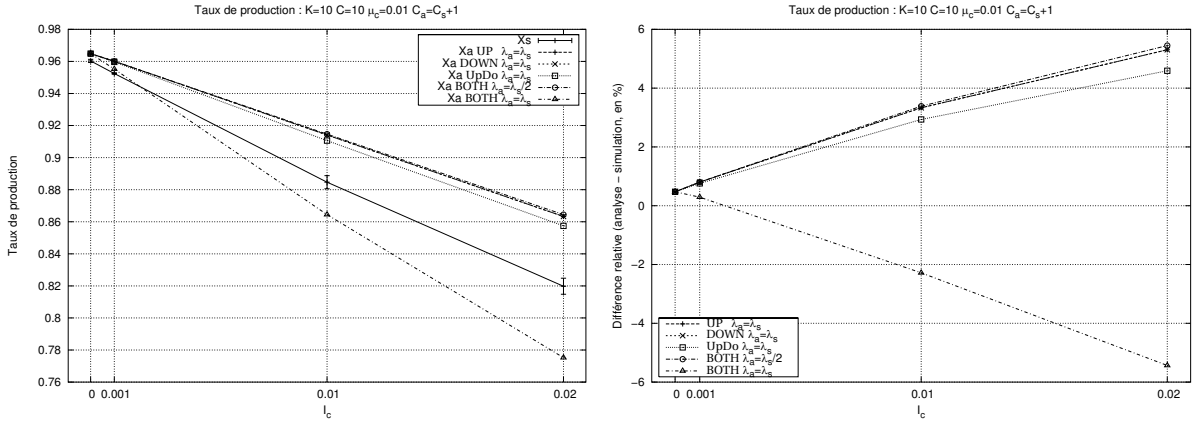

FIG. C.24 – Comparaison des taux de production. 10 machines, stocks = 10

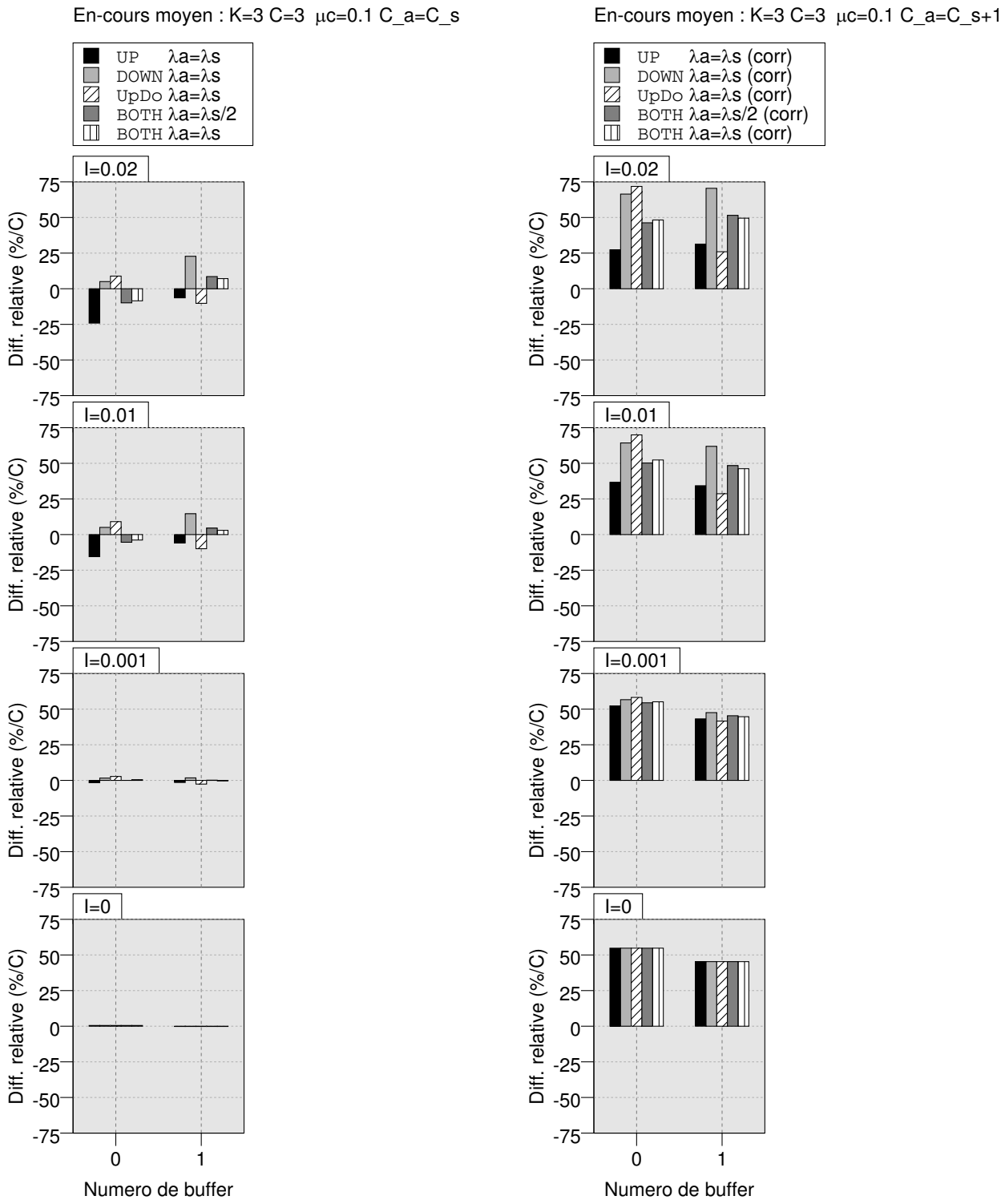

FIG. C.25 – Comparaison des encours des buffers. 3 machines, stocks = 3

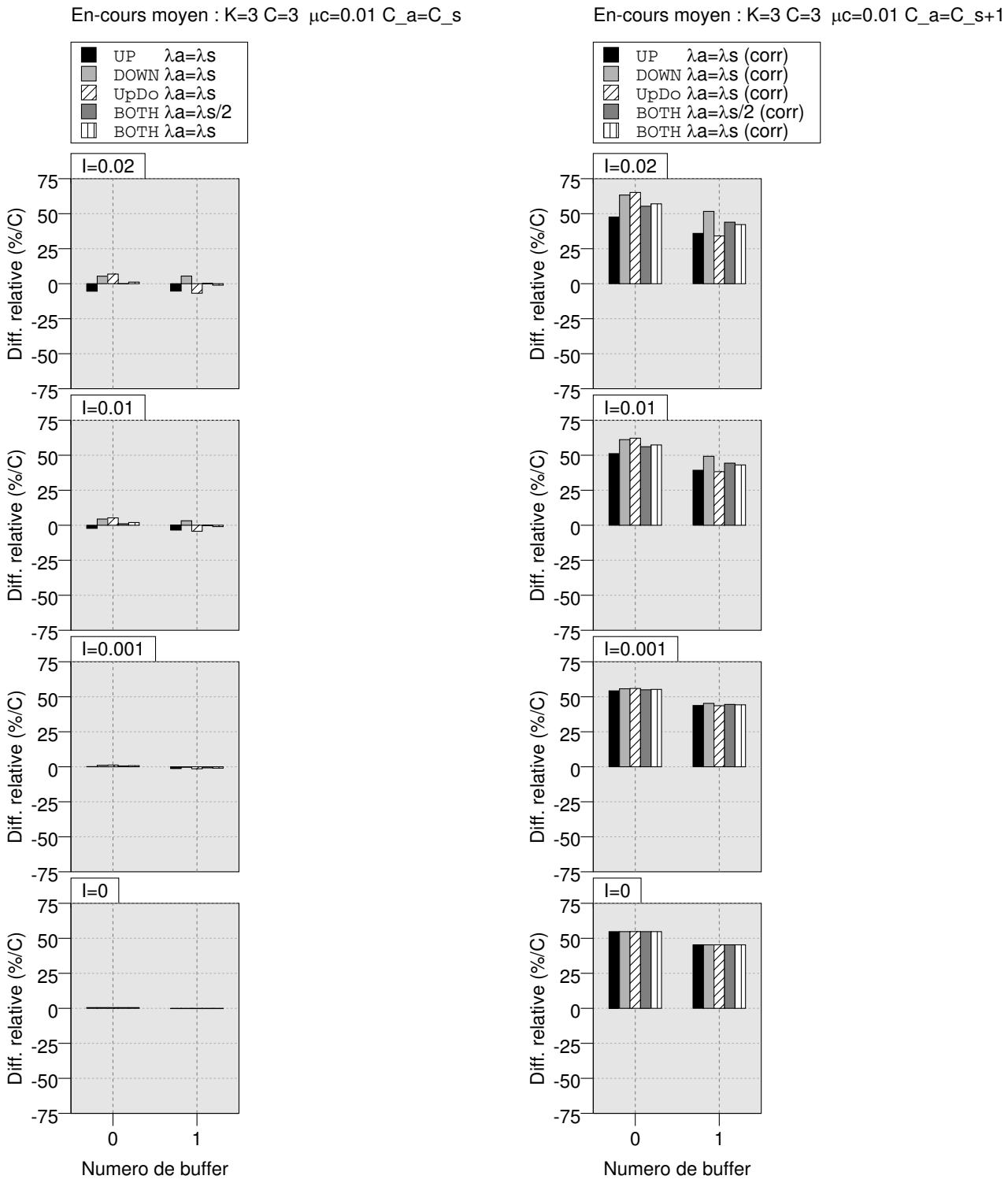

FIG. C.26 – Comparaison des encours des buffers. 3 machines, stocks = 3

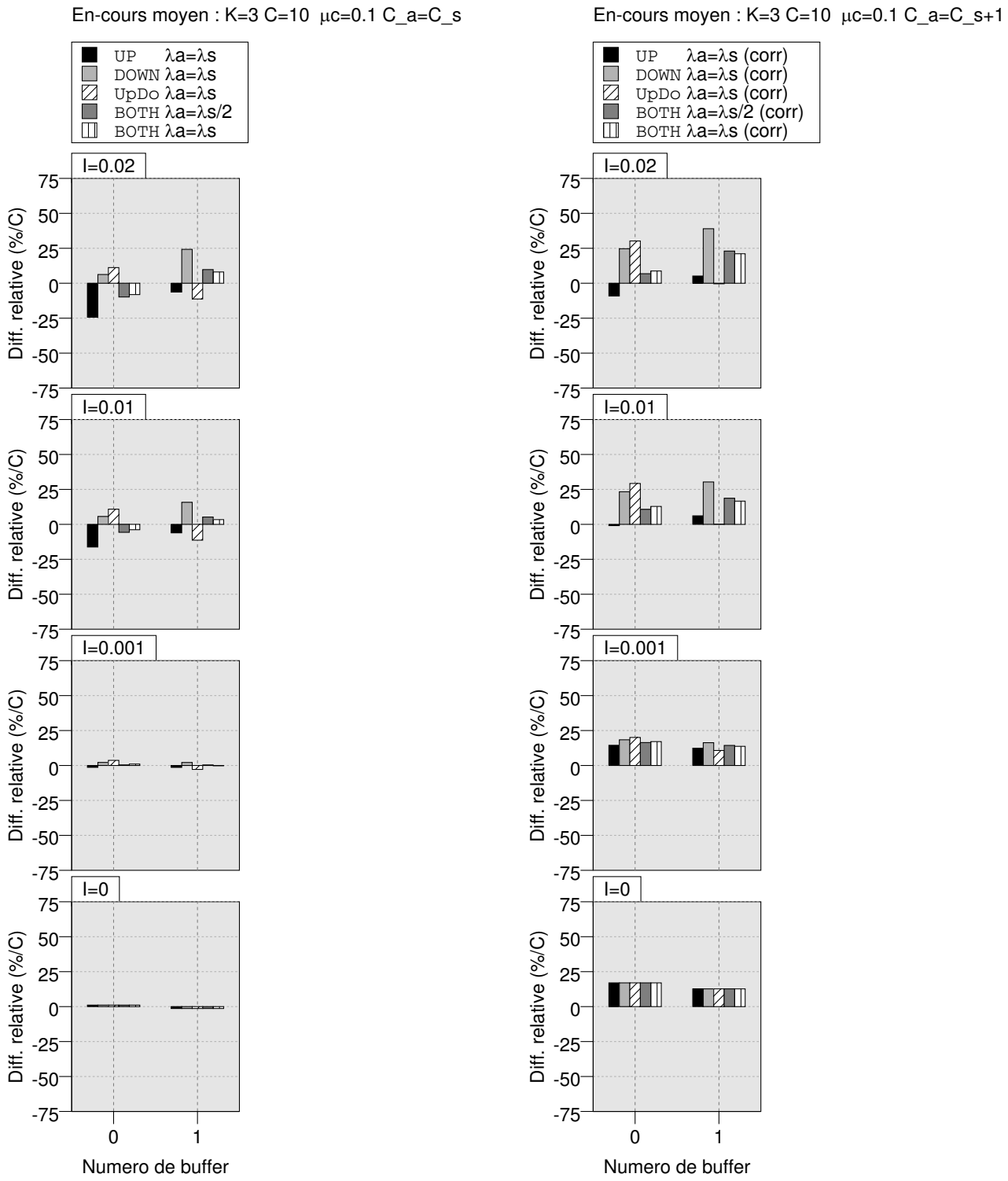

FIG. C.27 – Comparaison des encours des buffers. 3 machines, stocks = 10

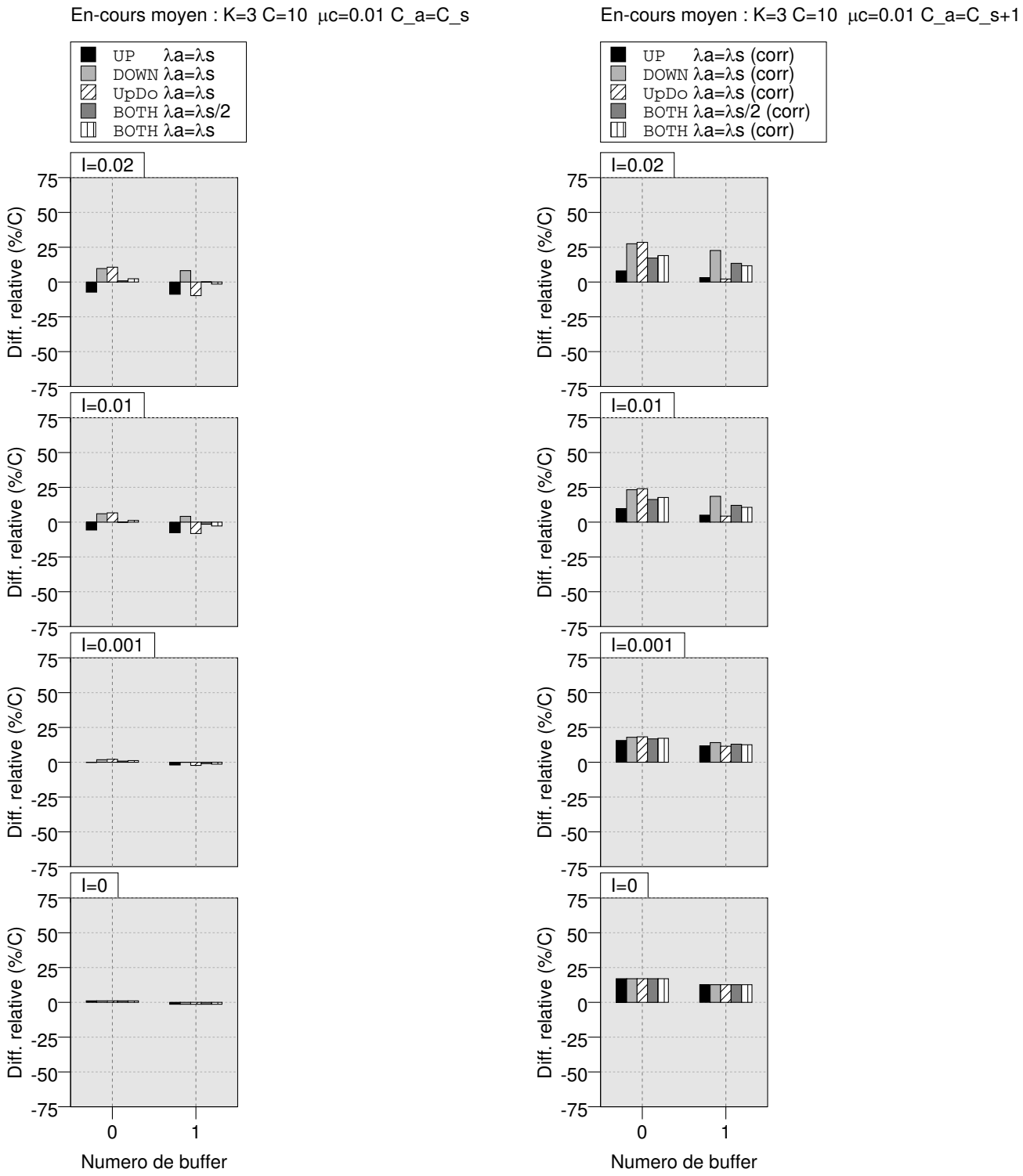

FIG. C.28 – Comparaison des encours des buffers. 3 machines, stocks = 10

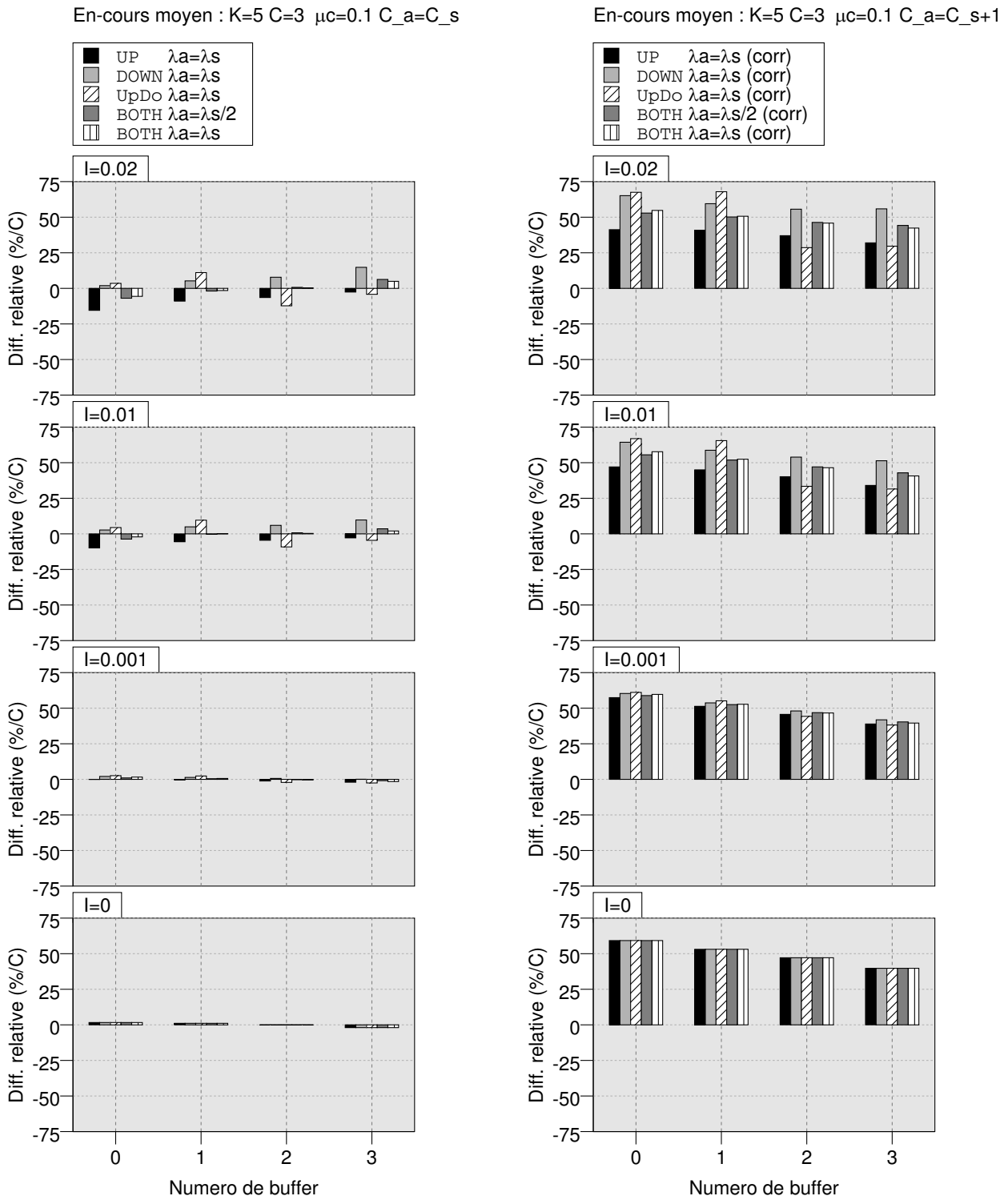

FIG. C.29 – Comparaison des encours des buffers. 5 machines, stocks = 3

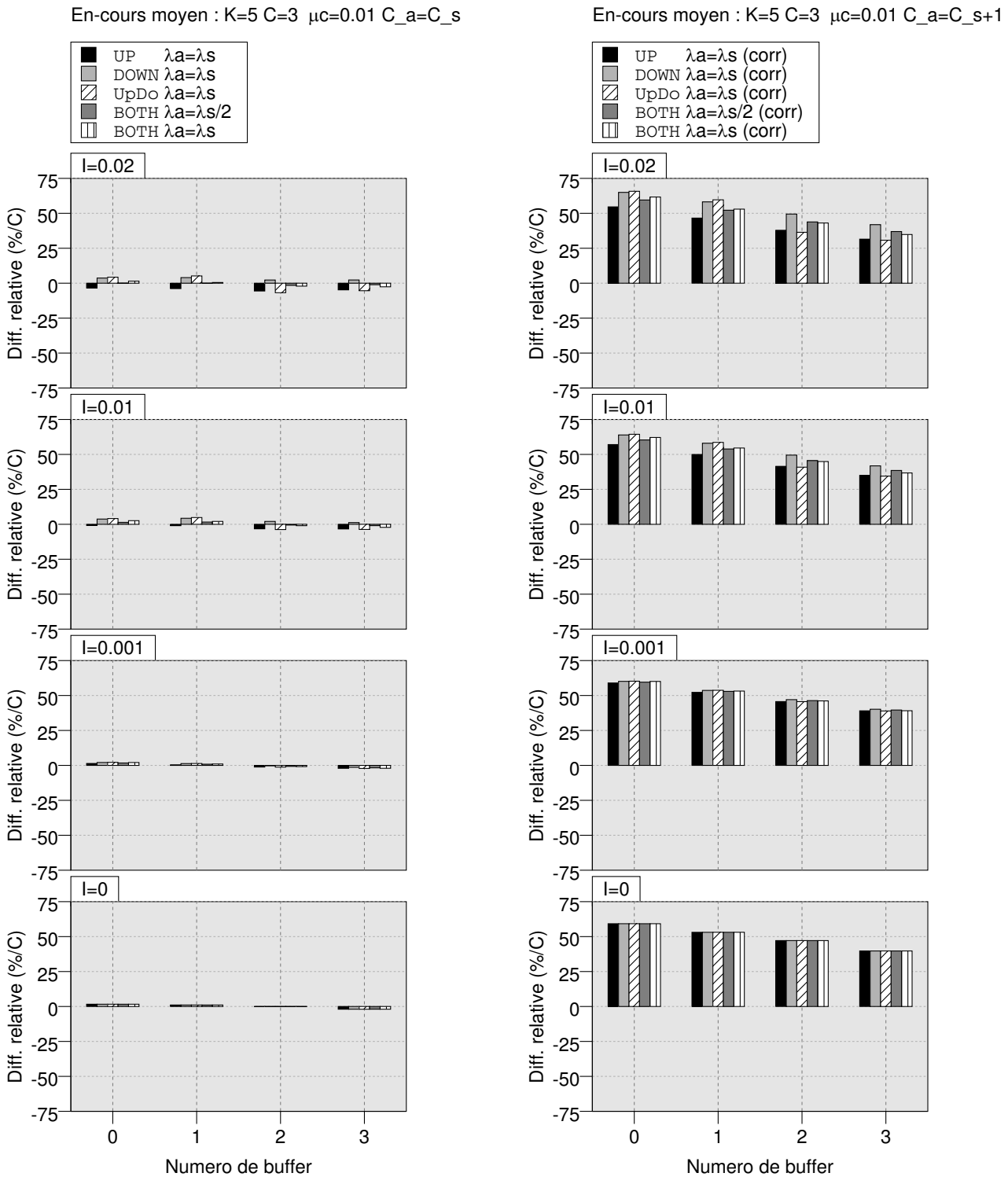

FIG. C.30 – Comparaison des encours des buffers. 5 machines, stocks = 3

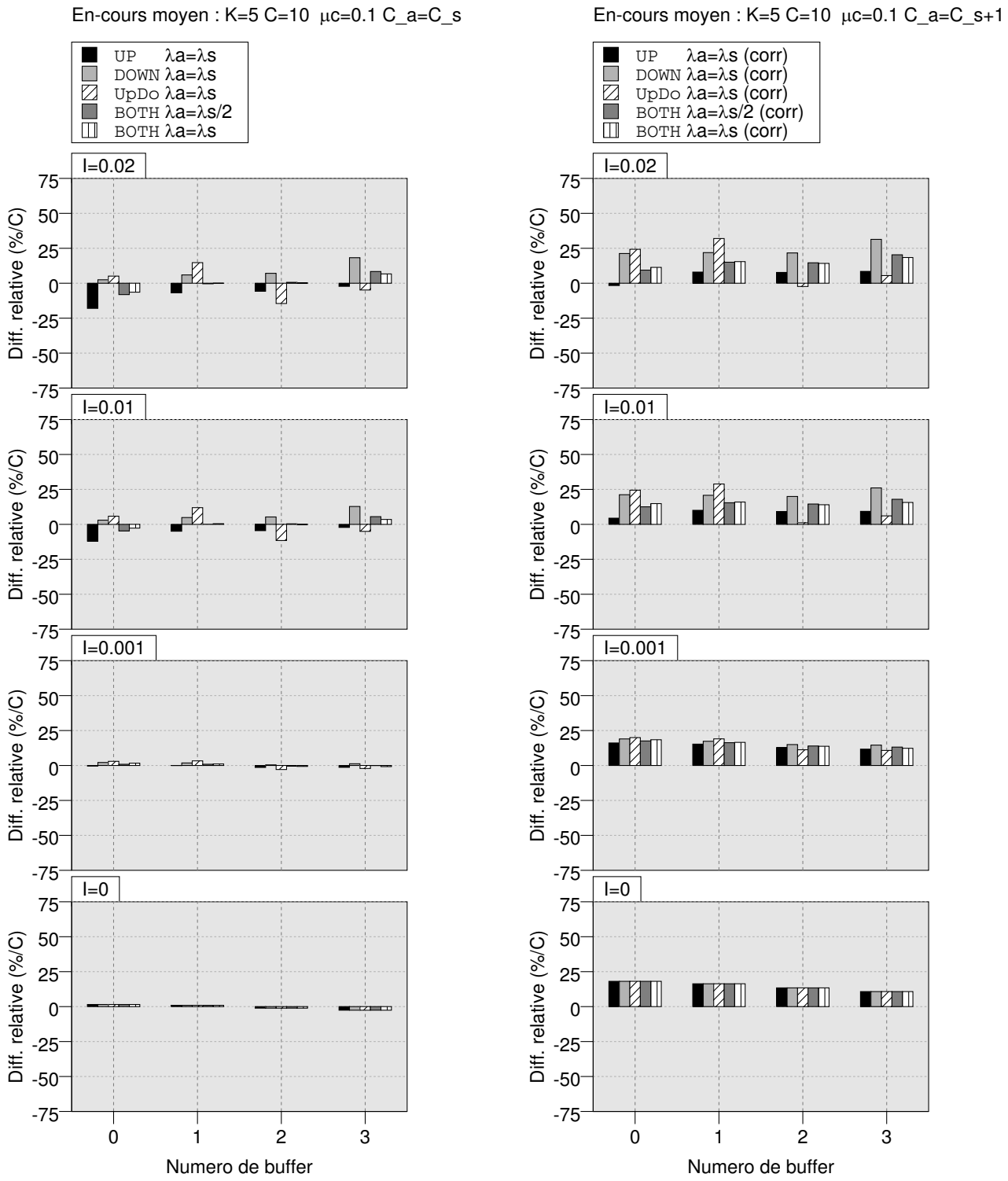

FIG. C.31 – Comparaison des encours des buffers. 5 machines, stocks = 10

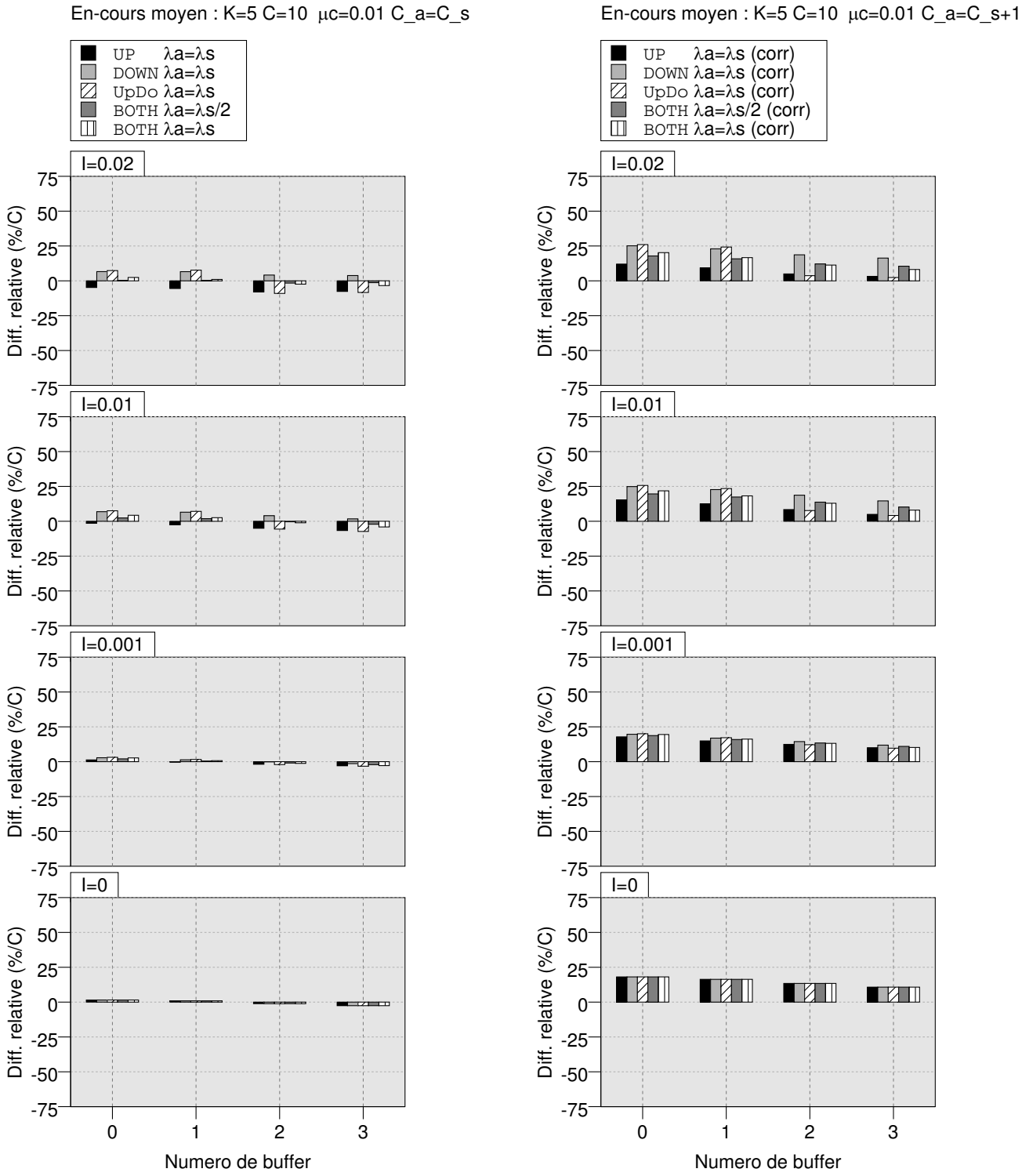

FIG. C.32 – Comparaison des encours des buffers. 5 machines, stocks = 10

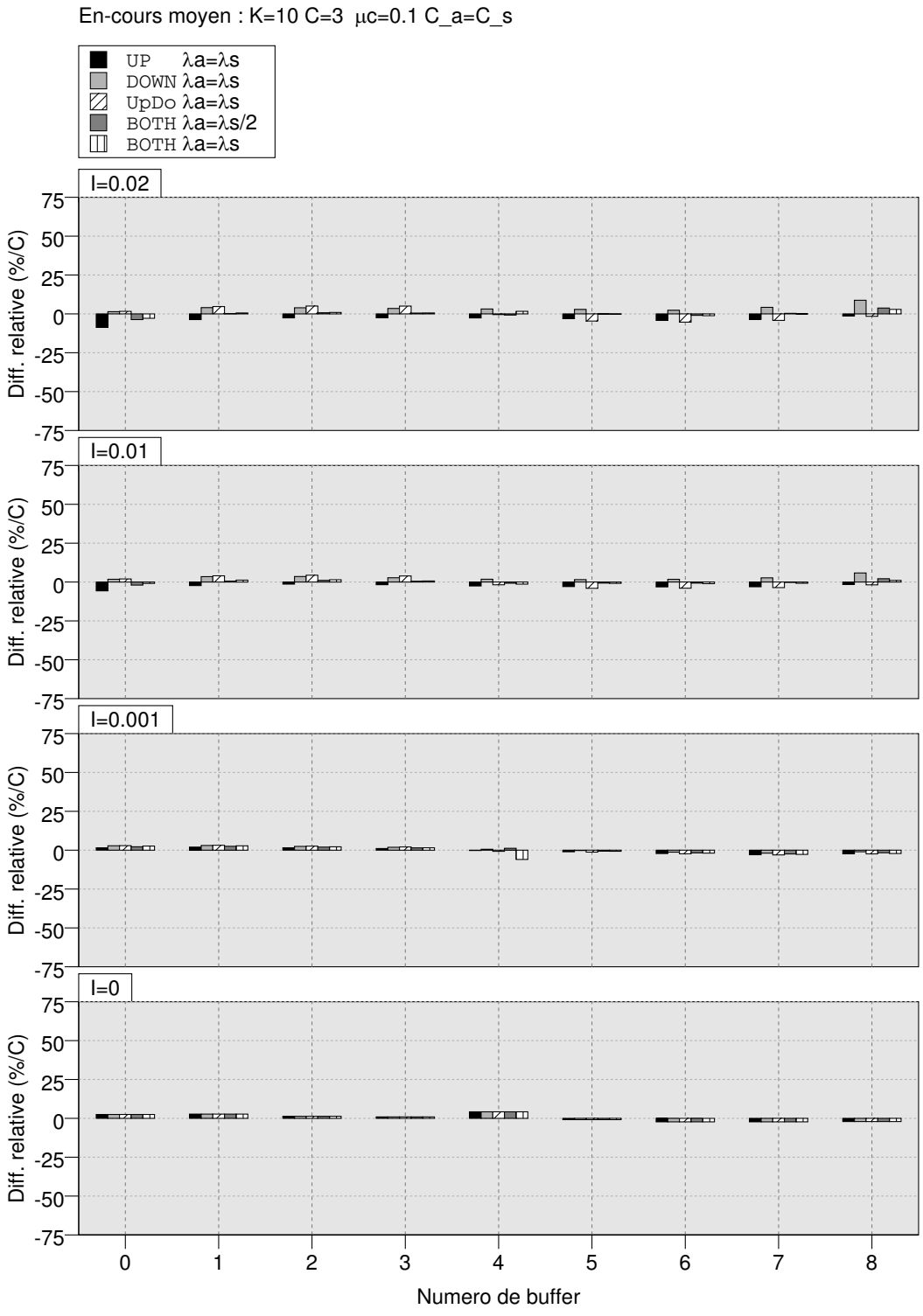

FIG. C.33 – Comparaison des encours des buffers. 10 machines, stocks = 3

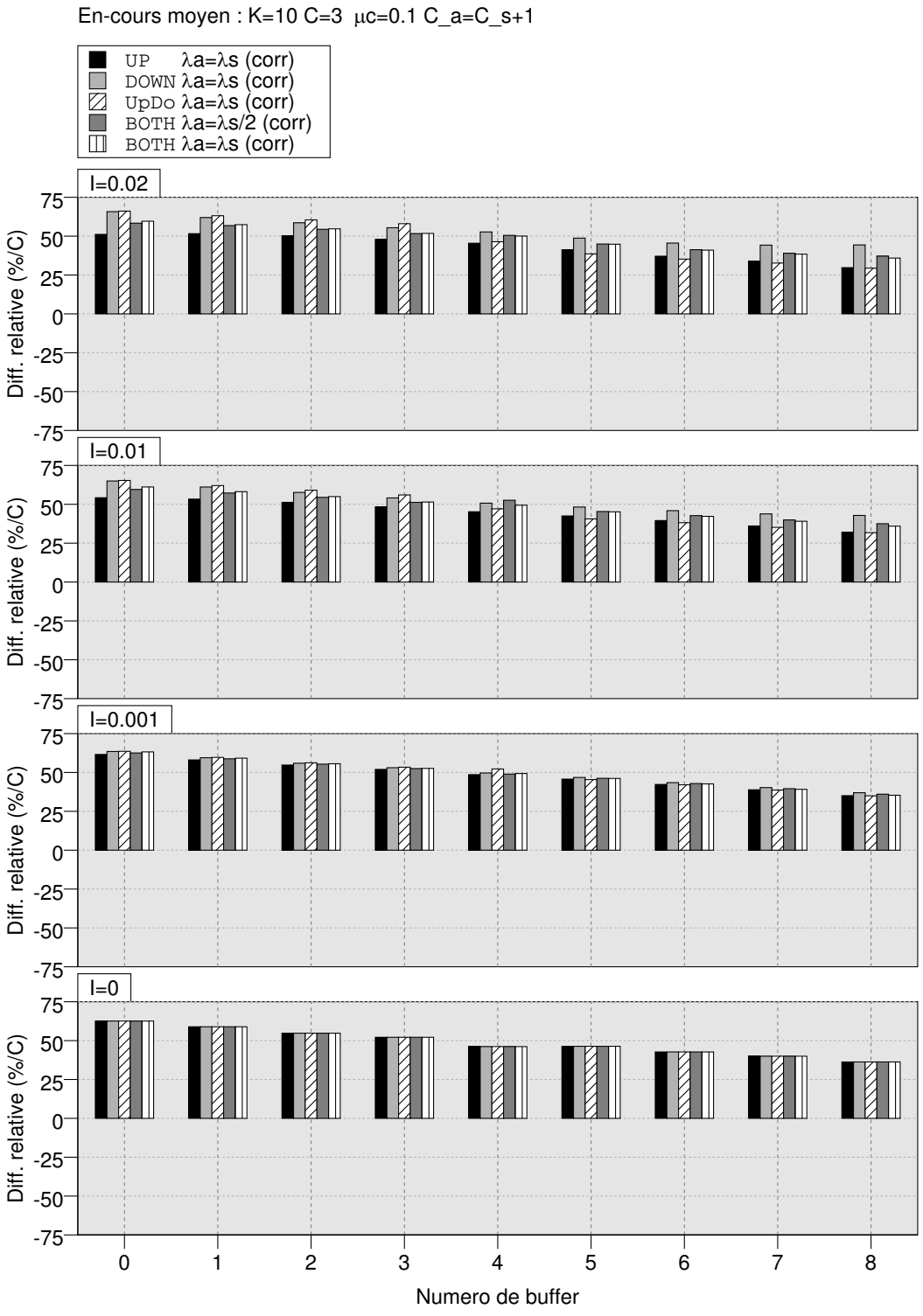

FIG. C.34 – Comparaison des encours des buffers. 10 machines, stocks = 3

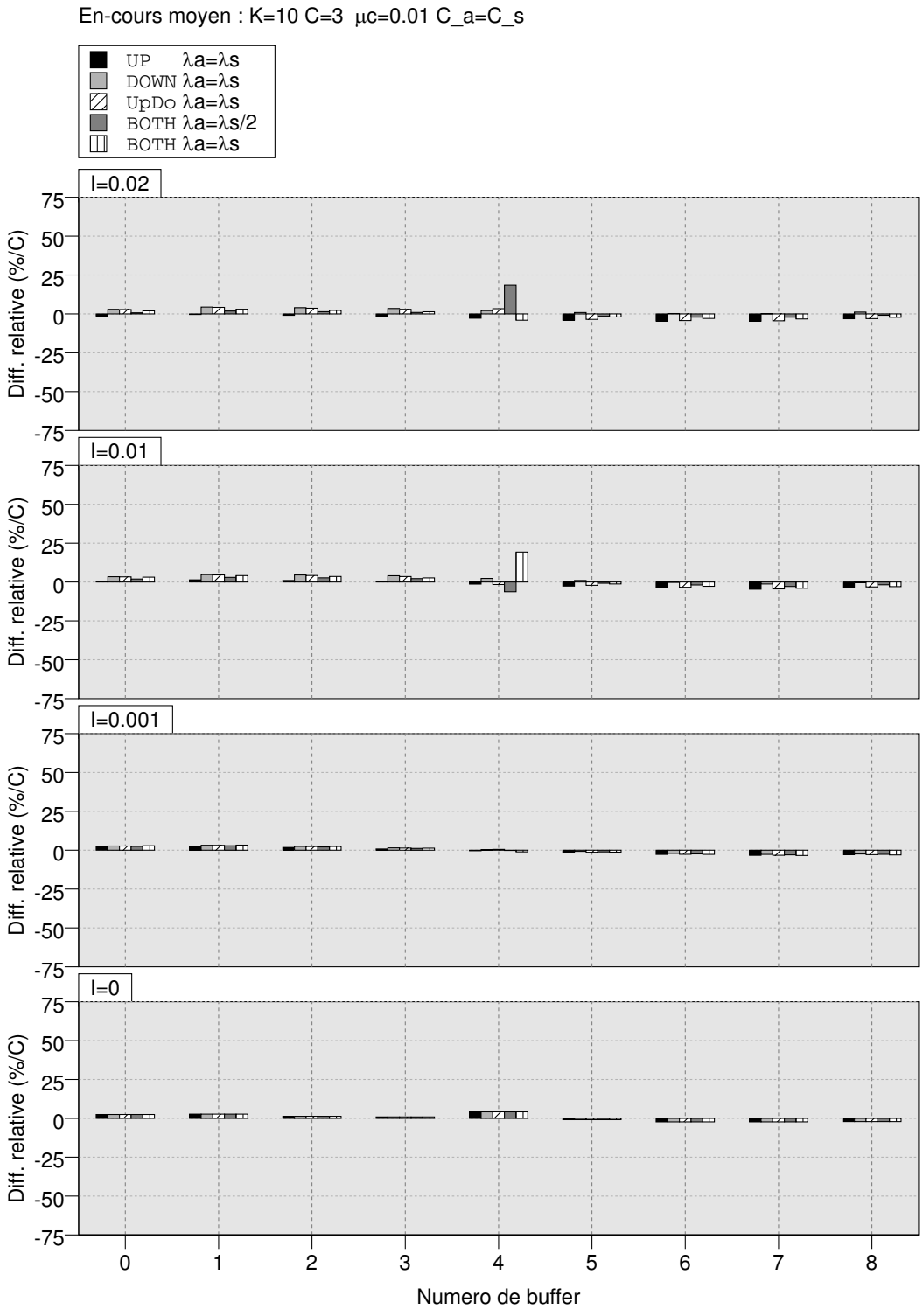

FIG. C.35 – Comparaison des encours des buffers. 10 machines, stocks = 3

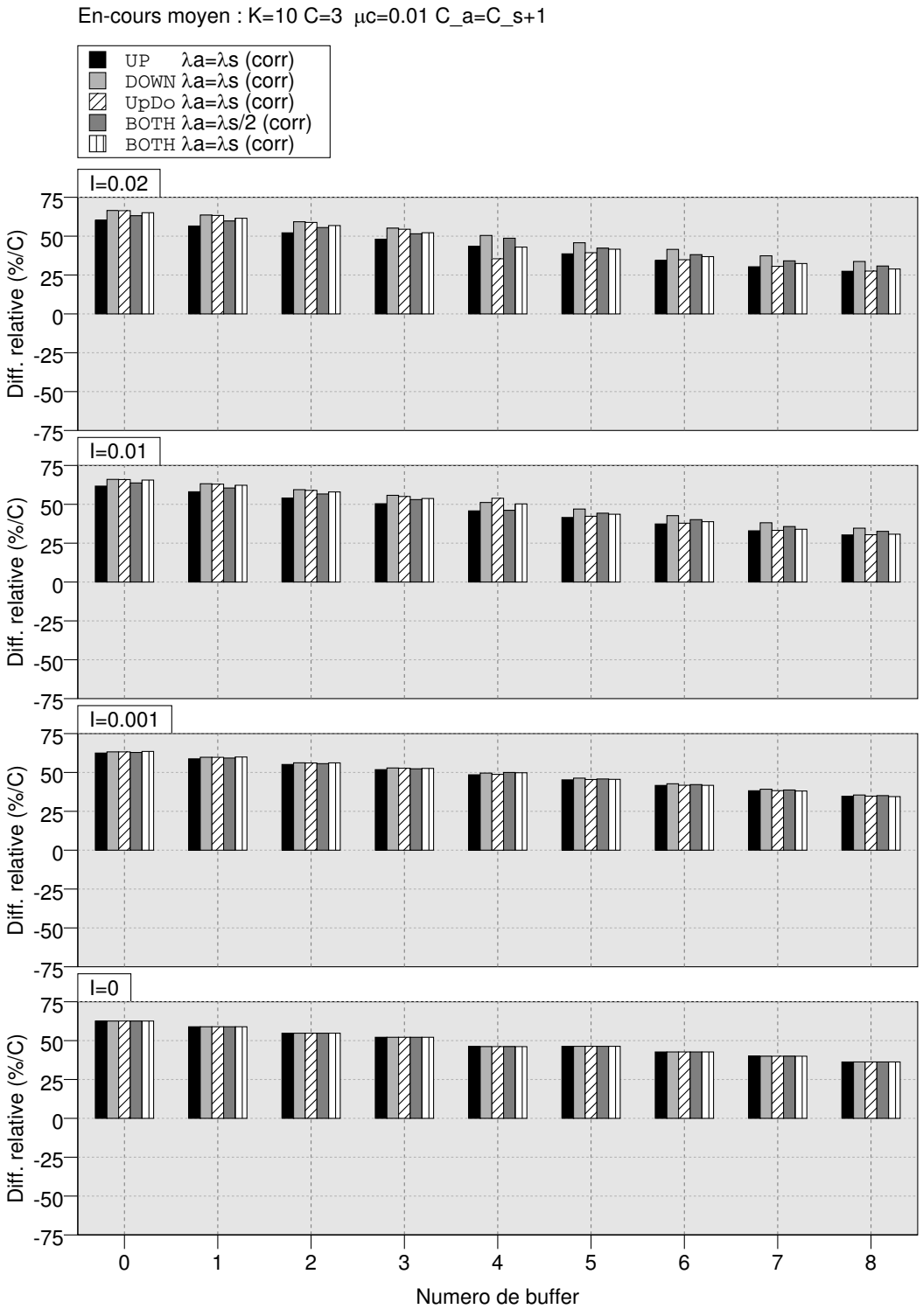

FIG. C.36 – Comparaison des encours des buffers. 10 machines, stocks = 3

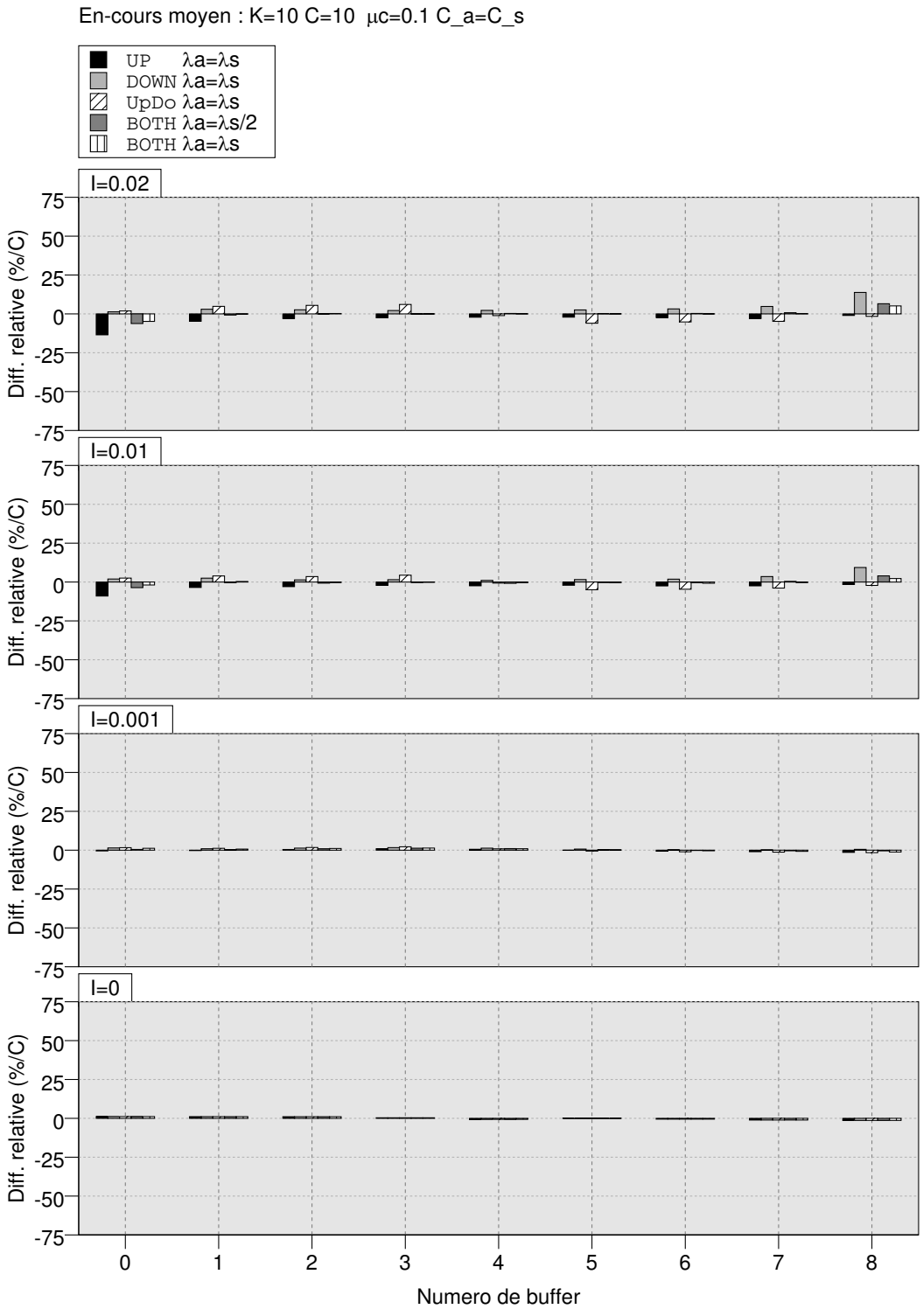

FIG. C.37 – Comparaison des encours des buffers. 10 machines, stocks = 10

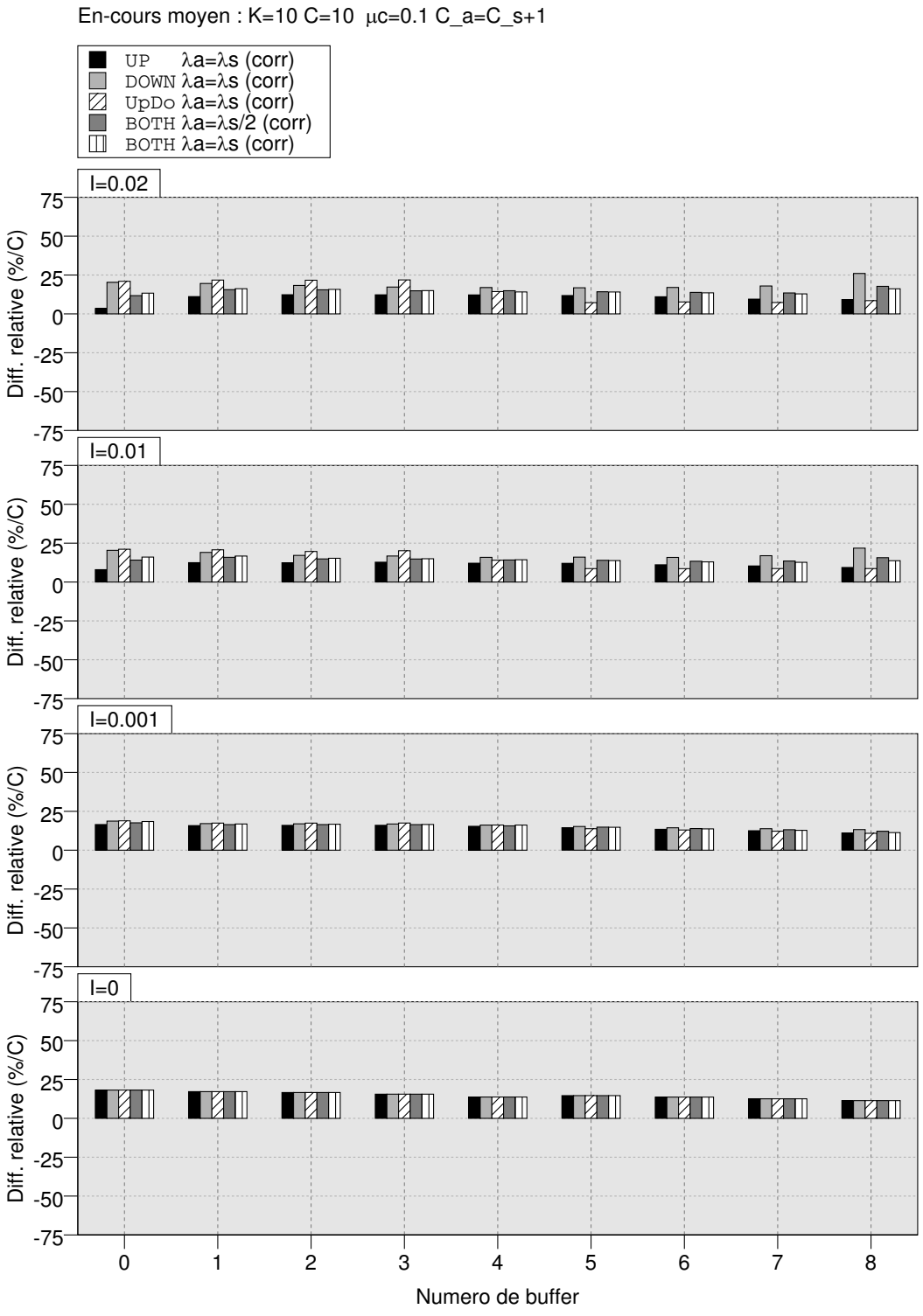

FIG. C.38 – Comparaison des encours des buffers. 10 machines, stocks = 10

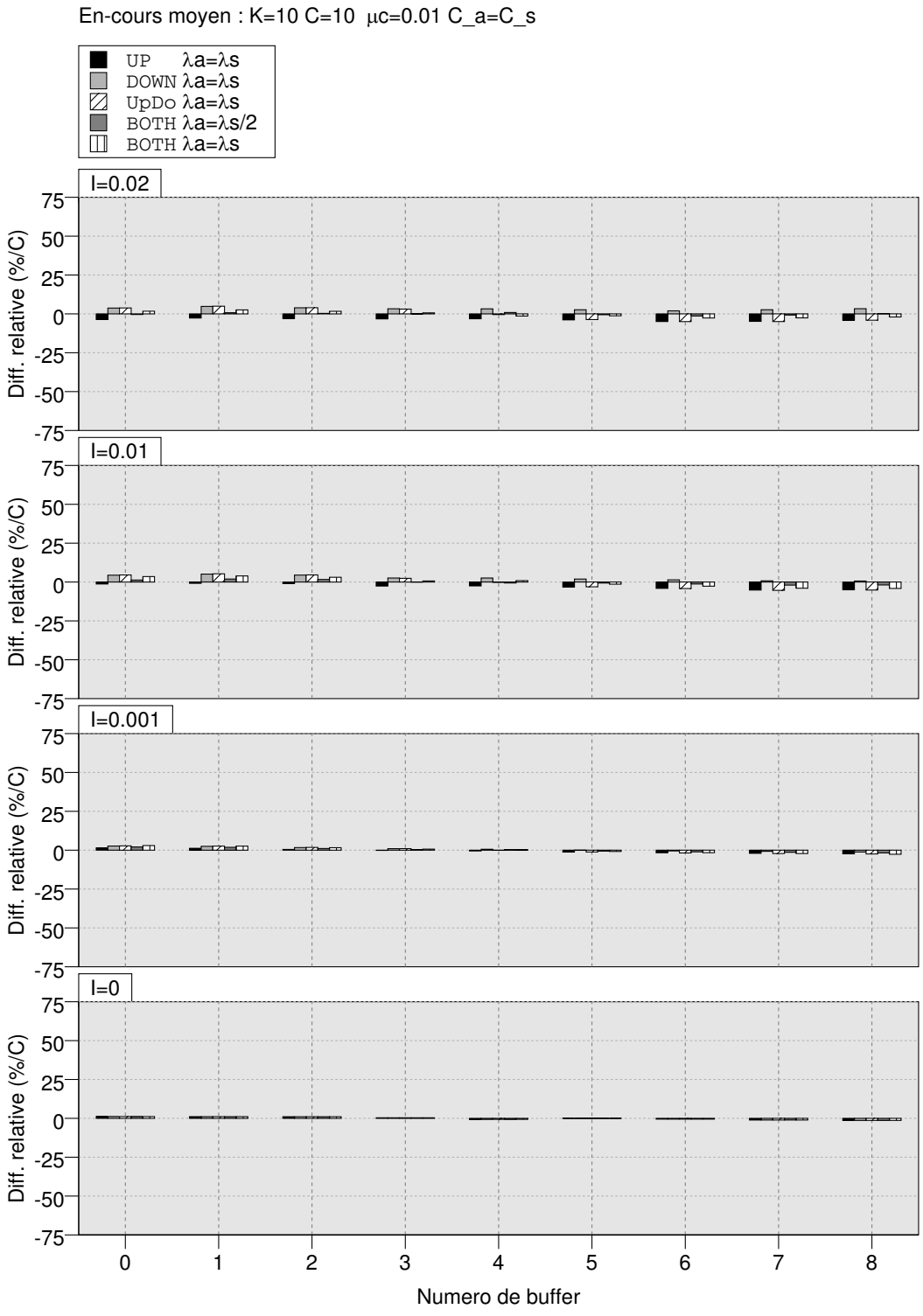

FIG. C.39 – Comparaison des encours des buffers. 10 machines, stocks = 10

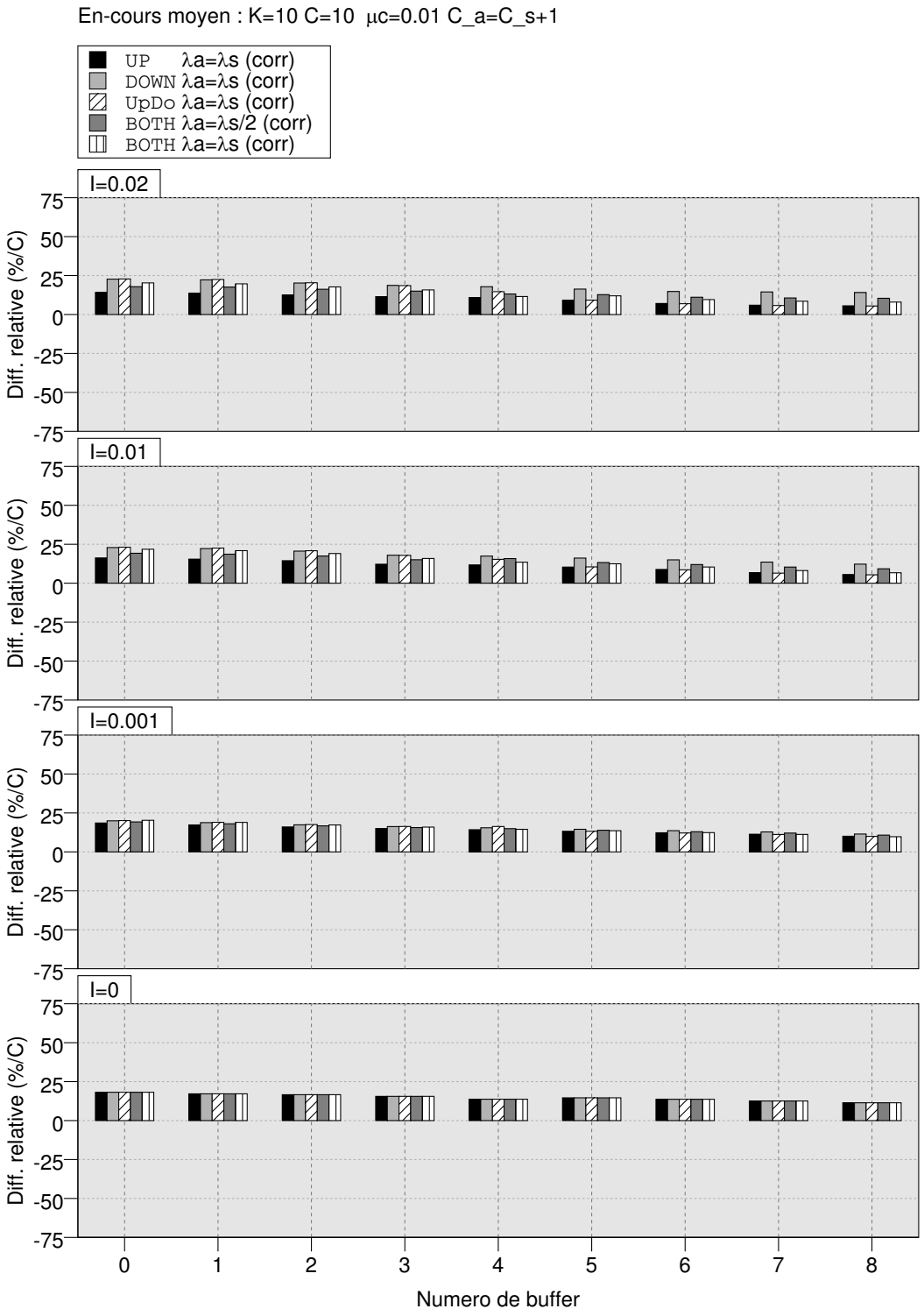

FIG. C.40 – Comparaison des encours des buffers. 10 machines, stocks = 10

	M_1	M_2	M_3
U	1.000000	1.000000	1.000000
λ	0.005351	0.005480	0.005176
μ	0.101471	0.090094	0.099718
I	0.0527	0.0608	0.0519
e	0.9499	0.9426	0.9506

	$C_{1,2}$	$C_{2,3}$
C	3	3
λ	0.000904	0.001038
μ	0.093903	0.101806
I	0.0096	0.0101
e	0.9904	0.9899

TAB. C.1 – Paramètres de la ligne n°1.

Machine	P_w	P_s	P_b	P_f
M_1	0.85918	0	0.095327	0.045488
M_2	0.85917	0.045205	0.044237	0.051388
M_3	0.85917	0.097369	0	0.043465

Queue	En-cours	En-cours (corrigé)
$C_{1,2}$	1.98241	2.98241
$C_{2,3}$	0.970194	1.92499
Total	2.9526	4.90739

TAB. C.2 – Résultats de simulation à évènements discrets (ligne n°1).

$\lambda_c(ana) = 0.5 * \lambda_c(sim) \star C(ana) = C(sim) \star$ BOTH

	M_1	M_2	M_3
p_w	-0.0087%	-0.0073%	-0.0068%
p_s	0%	-0.49%	-0.54%
p_b	-0.39%	-0.44%	0%

	$C_{1,2}$	$C_{2,3}$	
H_m	-33%	-48%	
H_m(C)	-33%	-31%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) \star$ UP

	M_1	M_2	M_3
p_w	-0.019%	-0.017%	-0.017%
p_s	0%	-0.12%	-0.095%
p_b	-0.79%	-0.83%	0%

	$C_{1,2}$	$C_{2,3}$	
H_m	-36%	-53%	
H_m(C)	-36%	-34%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) \star$ BOTH

	M_1	M_2	M_3
p_w	-1.3%	-1.3%	-1.3%
p_s	0%	-0.24%	0.38%
p_b	0.56%	-0.16%	0%

	$C_{1,2}$	$C_{2,3}$	
H_m	-32%	-49%	
H_m(C)	-32%	-32%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) \star$ DOWN

	M_1	M_2	M_3
p_w	-0.0093%	-0.0079%	-0.0074%
p_s	0%	-0.87%	-0.98%
p_b	0.028%	-0.037%	0%

	$C_{1,2}$	$C_{2,3}$	
H_m	-31%	-44%	
H_m(C)	-30%	-28%	

TAB. C.3 – Comparaison des résultats analytiques avec la simulation (ligne n°1).

$\lambda_c(ana) = 0.5 * \lambda_c(sim) \star C(ana) = C(sim) + 1 \star$ BOTH

	M_1	M_2	M_3
p_w	0.4%	0.41%	0.41%
p_s	0%	-0.72%	-0.98%
p_b	-0.82%	-0.66%	0%
	$C_{1,2}$	$C_{2,3}$	
H_m	-12%	-30%	
H_m(C)	-12%	-19%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) + 1 \star$ UP

	M_1	M_2	M_3
p_w	0.39%	0.39%	0.39%
p_s	0%	-0.34%	-0.53%
p_b	-1.2%	-1%	0%
	$C_{1,2}$	$C_{2,3}$	
H_m	-15%	-36%	
H_m(C)	-15%	-23%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) + 1 \star$ BOTH

	M_1	M_2	M_3
p_w	-0.85%	-0.84%	-0.84%
p_s	0%	-0.48%	-0.093%
p_b	0.089%	-0.4%	0%
	$C_{1,2}$	$C_{2,3}$	
H_m	-10%	-31%	
H_m(C)	-10%	-20%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) + 1 \star$ DOWN

	M_1	M_2	M_3
p_w	0.4%	0.4%	0.41%
p_s	0%	-1.1%	-1.4%
p_b	-0.41%	-0.27%	0%
	$C_{1,2}$	$C_{2,3}$	
H_m	-8.1%	-24%	
H_m(C)	-8%	-15%	

TAB. C.4 – Comparaison des résultats analytiques avec la simulation (ligne n°1).

	M_1	M_2	M_3
U	1.000000	1.000000	1.000000
λ	0.005351	0.005480	0.005176
μ	0.101471	0.090094	0.099718
I	0.0527	0.0608	0.0519
e	0.9499	0.9426	0.9506

	$C_{1,2}$	$C_{2,3}$
C	10	10
λ	0.000904	0.001038
μ	0.093903	0.101806
I	0.0096	0.0101
e	0.9904	0.9899

TAB. C.5 – Paramètres de la ligne n°2.

Machine	P_w	P_s	P_b	P_f
M_1	0.88015	0	0.074591	0.04526
M_2	0.88015	0.032595	0.034278	0.052974
M_3	0.88018	0.074278	0	0.045544

Queue	En-cours	En-cours (corrigé)
$C_{1,2}$	6.51353	7.51353
$C_{2,3}$	3.44139	4.40879
Total	9.95492	11.9223

TAB. C.6 – Résultats de simulation à évènements discrets (ligne n°2).

$\lambda_c(ana) = 0.5 * \lambda_c(sim) \star C(ana) = C(sim) \star$ BOTH

	M_1	M_2	M_3
p_w	0.12%	0.12%	0.11%
p_s	0%	-0.43%	-0.58%
p_b	-0.66%	-0.63%	0%

	$C_{1,2}$	$C_{2,3}$	
H_m	-15%	-20%	
H_m(C)	-11%	-8.7%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) \star$ UP

	M_1	M_2	M_3
p_w	0.096%	0.095%	0.093%
p_s	0%	-0.087%	-0.11%
p_b	-1.1%	-0.97%	0%

	$C_{1,2}$	$C_{2,3}$	
H_m	-18%	-27%	
H_m(C)	-14%	-12%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) \star$ BOTH

	M_1	M_2	M_3
p_w	-1%	-1%	-1%
p_s	0%	-0.27%	0.15%
p_b	0.1%	-0.45%	0%

	$C_{1,2}$	$C_{2,3}$	
H_m	-13%	-21%	
H_m(C)	-10%	-9.4%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) \star$ DOWN

	M_1	M_2	M_3
p_w	0.11%	0.11%	0.11%
p_s	0%	-0.76%	-1%
p_b	-0.23%	-0.27%	0%

	$C_{1,2}$	$C_{2,3}$	
H_m	-11%	-12%	
H_m(C)	-8.1%	-5.5%	

TAB. C.7 – Comparaison des résultats analytiques avec la simulation (ligne n°2).

$\lambda_c(ana) = 0.5 * \lambda_c(sim) \star C(ana) = C(sim) + 1 \star$ BOTH

	M_1	M_2	M_3
p_w	0.34%	0.34%	0.34%
p_s	0%	-0.55%	-0.82%
p_b	-0.9%	-0.74%	0%
	$C_{1,2}$	$C_{2,3}$	
H_m	-6.3%	-11%	
H_m(C)	-4.8%	-5%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) + 1 \star$ UP

	M_1	M_2	M_3
p_w	0.32%	0.32%	0.31%
p_s	0%	-0.21%	-0.35%
p_b	-1.3%	-1.1%	0%
	$C_{1,2}$	$C_{2,3}$	
H_m	-11%	-20%	
H_m(C)	-7.9%	-8.6%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) + 1 \star$ BOTH

	M_1	M_2	M_3
p_w	-0.76%	-0.76%	-0.76%
p_s	0%	-0.4%	-0.1%
p_b	-0.15%	-0.58%	0%
	$C_{1,2}$	$C_{2,3}$	
H_m	-4.8%	-13%	
H_m(C)	-3.6%	-5.8%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) + 1 \star$ DOWN

	M_1	M_2	M_3
p_w	0.33%	0.33%	0.33%
p_s	0%	-0.87%	-1.3%
p_b	-0.47%	-0.39%	0%
	$C_{1,2}$	$C_{2,3}$	
H_m	-2%	-3.1%	
H_m(C)	-1.5%	-1.4%	

TAB. C.8 – Comparaison des résultats analytiques avec la simulation (ligne n°2).

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
U	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
λ	0.005493	0.005339	0.004957	0.004779	0.005159	0.005382	0.004583	0.004859	0.005161	0.005019
μ	0.095074	0.092096	0.105425	0.107107	0.096769	0.099232	0.097342	0.093320	0.109071	0.091466
I	0.0577	0.0579	0.0470	0.0446	0.0533	0.0542	0.0470	0.0520	0.0473	0.0548
e	0.9453	0.9452	0.9550	0.9572	0.9493	0.9485	0.9550	0.9505	0.9548	0.9479

	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$
C	3	3	3	3	3	3	3	3	3
λ	0.000983	0.000926	0.001043	0.001041	0.001090	0.001070	0.001044	0.001060	0.000944
μ	0.108538	0.094235	0.101584	0.099541	0.099100	0.101466	0.092900	0.109325	0.104954
I	0.0090	0.0098	0.0102	0.0104	0.0109	0.0105	0.0112	0.0096	0.0089
e	0.9910	0.9902	0.9898	0.9896	0.9891	0.9895	0.9888	0.9903	0.9910

TAB. C.9 – Paramètres de la ligne n°3.

Machine	P_w	P_s	P_b	P_f
M_1	0.71491	0	0.24491	0.040177
M_2	0.71491	0.036423	0.2076	0.041071
M_3	0.71491	0.068733	0.18255	0.033813
M_4	0.71491	0.09056	0.162	0.032534
M_5	0.71491	0.1098	0.13606	0.03923
M_6	0.71491	0.13623	0.10934	0.039515
M_7	0.71492	0.16316	0.088255	0.033662
M_8	0.71491	0.18644	0.061253	0.037398
M_9	0.71491	0.21551	0.036235	0.033345
M_{10}	0.71491	0.24616	0	0.038932

Queue	En-cours	En-cours (corrigé)
$C_{1,2}$	2.54354	3.54354
$C_{2,3}$	2.16885	3.13242
$C_{3,4}$	1.92671	2.85798
$C_{4,5}$	1.73622	2.64566
$C_{5,6}$	1.50391	2.39411
$C_{6,7}$	1.24626	2.11003
$C_{7,8}$	1.04884	1.88568
$C_{8,9}$	0.776068	1.58962
$C_{9,10}$	0.430358	1.21484
Total	13.3807	21.3739

TAB. C.10 – Résultats de simulation à évènements discrets (ligne n°3).

$\lambda_c(ana) = 0.5 * \lambda_c(sim) \star C(ana) = C(sim) \star$ BOTH

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
p_w	0.91%	0.91%	0.91%	0.91%	0.91%	0.91%	0.91%	0.91%	0.91%	0.91%
p_s	0%	-0.25%	-0.36%	-0.48%	-0.64%	-0.91%	-1.1%	-1.2%	-1.3%	-1.3%
p_b	-1.4%	-1.4%	-1.3%	-1.2%	-0.99%	-0.76%	-0.65%	-0.46%	-0.38%	0%

	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$
H_m	-28%	-30%	-32%	-35%	-38%	-42%	-46%	-53%	-66%
H_m(C)	-33%	-31%	-31%	-30%	-30%	-29%	-29%	-28%	-27%

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) \star$ UP

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
p_w	0.9%	0.9%	0.9%	0.9%	0.9%	0.9%	0.9%	0.9%	0.9%	0.9%
p_s	0%	0.00092%	-0.12%	-0.24%	-0.41%	-0.66%	-0.84%	-0.94%	-1%	-0.98%
p_b	-1.7%	-1.7%	-1.6%	-1.4%	-1.2%	-0.98%	-0.92%	-0.69%	-0.63%	0%

	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$
H_m	-29%	-31%	-33%	-35%	-39%	-43%	-47%	-55%	-68%
H_m(C)	-34%	-32%	-31%	-31%	-31%	-30%	-30%	-29%	-28%

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) \star$ BOTH

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
p_w	-1.9%	-1.9%	-1.9%	-1.9%	-1.9%	-1.9%	-1.9%	-1.9%	-1.9%	-1.9%
p_s	0%	-0.19%	-0.012%	0.17%	0.3%	0.3%	0.41%	0.61%	0.91%	1.3%
p_b	1.2%	0.81%	0.59%	0.4%	0.29%	0.24%	0.027%	-0.084%	-0.3%	0%

	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$
H_m	-27%	-29%	-31%	-34%	-38%	-42%	-47%	-54%	-68%
H_m(C)	-32%	-30%	-30%	-30%	-30%	-29%	-29%	-29%	-27%

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) \star$ DOWN

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
p_w	0.92%	0.92%	0.92%	0.92%	0.92%	0.92%	0.91%	0.92%	0.92%	0.92%
p_s	0%	-0.5%	-0.6%	-0.71%	-0.86%	-1.2%	-1.3%	-1.5%	-1.6%	-1.6%
p_b	-1.1%	-1.2%	-1.1%	-0.93%	-0.75%	-0.53%	-0.39%	-0.23%	-0.12%	0%

	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$
H_m	-27%	-29%	-31%	-34%	-37%	-40%	-45%	-51%	-63%
H_m(C)	-32%	-30%	-30%	-30%	-29%	-28%	-28%	-27%	-25%

TAB. C.11 – Comparaison des résultats analytiques avec la simulation (ligne n°3).

$\lambda_c(ana) = 0.5 * \lambda_c(sim) \star C(ana) = C(sim) + 1 \star$ BOTH

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
p_w	3%	3%	3%	3%	3%	3%	3%	3%	3%	3%
p_s	0%	-0.42%	-0.77%	-1.2%	-1.6%	-2.2%	-2.6%	-3.1%	-3.4%	-3.6%
p_b	-3.7%	-3.5%	-3.1%	-2.7%	-2.2%	-1.7%	-1.3%	-0.87%	-0.53%	0%
	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$	
H_m	-5.5%	-8.6%	-11%	-13%	-17%	-21%	-26%	-34%	-50%	
H_m(C)	-6.5%	-8.9%	-10%	-12%	-14%	-15%	-16%	-18%	-20%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) + 1 \star$ UP

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
p_w	3%	3%	3%	3%	3%	3%	3%	3%	3%	3%
p_s	0%	-0.17%	-0.54%	-0.96%	-1.4%	-2%	-2.4%	-2.8%	-3.1%	-3.2%
p_b	-4%	-3.8%	-3.4%	-2.9%	-2.5%	-1.9%	-1.6%	-1.1%	-0.77%	0%
	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$	
H_m	-6.9%	-9.7%	-12%	-15%	-18%	-23%	-28%	-36%	-54%	
H_m(C)	-8.2%	-10%	-12%	-13%	-15%	-16%	-18%	-19%	-22%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) + 1 \star$ BOTH

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
p_w	0.38%	0.38%	0.38%	0.38%	0.38%	0.38%	0.38%	0.38%	0.38%	0.38%
p_s	0%	-0.36%	-0.45%	-0.58%	-0.78%	-1.1%	-1.3%	-1.4%	-1.3%	-1.1%
p_b	-1.2%	-1.4%	-1.4%	-1.2%	-1.1%	-0.79%	-0.7%	-0.52%	-0.47%	0%
	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$	
H_m	-4.3%	-7.4%	-10%	-13%	-17%	-21%	-27%	-35%	-53%	
H_m(C)	-5.1%	-7.7%	-9.5%	-11%	-13%	-15%	-17%	-19%	-21%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) + 1 \star$ DOWN

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
p_w	3%	3%	3%	3%	3%	3%	3%	3%	3%	3%
p_s	0%	-0.66%	-1%	-1.4%	-1.9%	-2.4%	-2.9%	-3.4%	-3.7%	-3.9%
p_b	-3.3%	-3.3%	-2.9%	-2.5%	-2%	-1.5%	-1.1%	-0.66%	-0.29%	0%
	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$	
H_m	-4.1%	-7.4%	-9.7%	-12%	-16%	-20%	-24%	-31%	-46%	
H_m(C)	-4.8%	-7.7%	-9.2%	-11%	-12%	-14%	-15%	-17%	-19%	

TAB. C.12 – Comparaison des résultats analytiques avec la simulation (ligne n°3).

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M
U	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.0000
λ	0.005493	0.005339	0.004957	0.004779	0.005159	0.005382	0.004583	0.004859	0.005161	0.0050
μ	0.095074	0.092096	0.105425	0.107107	0.096769	0.099232	0.097342	0.093320	0.109071	0.0914
I	0.0577	0.0579	0.0470	0.0446	0.0533	0.0542	0.0470	0.0520	0.0473	0.05
e	0.9453	0.9452	0.9550	0.9572	0.9493	0.9485	0.9550	0.9505	0.9548	0.94

	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$
C	10	10	10	10	10	10	10	10	10
λ	0.000983	0.000926	0.001043	0.001041	0.001090	0.001070	0.001044	0.001060	0.000944
μ	0.108538	0.094235	0.101584	0.099541	0.099100	0.101466	0.092900	0.109325	0.104954
I	0.0090	0.0098	0.0102	0.0104	0.0109	0.0105	0.0112	0.0096	0.0089
e	0.9910	0.9902	0.9898	0.9896	0.9891	0.9895	0.9888	0.9903	0.9910

TAB. C.13 – Paramètres de la ligne n°4.

Machine	P_w	P_s	P_b	P_f
M_1	0.80103	0	0.1525	0.04647
M_2	0.80103	0.029015	0.12407	0.045884
M_3	0.80103	0.05001	0.11074	0.03822
M_4	0.80103	0.062193	0.10113	0.035642
M_5	0.80103	0.069758	0.087616	0.041592
M_6	0.80103	0.084185	0.070795	0.043987
M_7	0.80104	0.10145	0.060444	0.03707
M_8	0.80104	0.1138	0.043372	0.041784
M_9	0.80104	0.1325	0.02784	0.038627
M_{10}	0.80104	0.15588	0	0.043084

Queue	En-cours	En-cours (corrigé)
$C_{1,2}$	7.7826	8.7826
$C_{2,3}$	6.56458	7.53557
$C_{3,4}$	6.03269	6.98268
$C_{4,5}$	5.68366	6.62147
$C_{5,6}$	5.08494	6.01518
$C_{6,7}$	4.28413	5.19994
$C_{7,8}$	3.88708	4.78564
$C_{8,9}$	3.12913	4.01533
$C_{9,10}$	2.03412	2.90162
Total	44.4829	52.84

TAB. C.14 – Résultats de simulation à évènements discrets (ligne n°4).

$\lambda_c(ana) = 0.5 * \lambda_c(sim) \star C(ana) = C(sim) \star$ BOTH

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
p_w	1.8%	1.8%	1.8%	1.8%	1.8%	1.8%	1.8%	1.8%	1.8%	1.8%
p_s	0%	-0.48%	-0.61%	-1%	-1.2%	-1.4%	-1.7%	-1.9%	-2.1%	-2.3%
p_b	-2.2%	-2.2%	-2%	-1.7%	-1.6%	-1.3%	-1.1%	-0.78%	-0.46%	0%
	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$	
H_m	-12%	-13%	-14%	-15%	-17%	-17%	-19%	-22%	-28%	
H_m(C)	-10%	-9.8%	-10%	-10%	-10%	-8.9%	-9.3%	-8.8%	-8%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) \star$ UP

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
p_w	1.7%	1.7%	1.7%	1.7%	1.7%	1.7%	1.7%	1.7%	1.7%	1.7%
p_s	0%	-0.26%	-0.42%	-0.83%	-1%	-1.2%	-1.5%	-1.7%	-1.8%	-1.9%
p_b	-2.6%	-2.4%	-2.2%	-1.9%	-1.8%	-1.4%	-1.3%	-0.94%	-0.67%	0%
	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$	
H_m	-14%	-14%	-16%	-17%	-18%	-19%	-21%	-25%	-33%	
H_m(C)	-12%	-11%	-11%	-11%	-11%	-9.8%	-10%	-9.9%	-9.7%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) \star$ BOTH

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
p_w	-0.4%	-0.4%	-0.4%	-0.4%	-0.4%	-0.4%	-0.4%	-0.4%	-0.4%	-0.4%
p_s	0%	-0.46%	-0.39%	-0.59%	-0.61%	-0.62%	-0.72%	-0.7%	-0.56%	-0.38%
p_b	-0.28%	-0.68%	-0.73%	-0.65%	-0.79%	-0.62%	-0.66%	-0.54%	-0.44%	0%
	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$	
H_m	-9.9%	-11%	-13%	-15%	-16%	-17%	-20%	-24%	-33%	
H_m(C)	-8.7%	-8.5%	-9.1%	-9.8%	-9.9%	-8.9%	-9.8%	-9.7%	-9.4%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) \star$ DOWN

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
p_w	1.8%	1.8%	1.8%	1.8%	1.8%	1.8%	1.8%	1.8%	1.8%	1.8%
p_s	0%	-0.69%	-0.8%	-1.2%	-1.4%	-1.6%	-1.9%	-2.2%	-2.3%	-2.7%
p_b	-1.8%	-2%	-1.8%	-1.5%	-1.5%	-1.1%	-0.9%	-0.61%	-0.25%	0%
	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$	
H_m	-9.8%	-12%	-13%	-14%	-15%	-15%	-17%	-19%	-22%	
H_m(C)	-8.6%	-8.8%	-9%	-9.3%	-9.2%	-8.1%	-8.3%	-7.6%	-6.4%	

TAB. C.15 – Comparaison des résultats analytiques avec la simulation (ligne n°4).

$\lambda_c(ana) = 0.5 * \lambda_c(sim) \star C(ana) = C(sim) + 1 \star$ BOTH

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
p_w	2.5%	2.5%	2.5%	2.5%	2.5%	2.5%	2.5%	2.5%	2.5%	2.5%
p_s	0%	-0.58%	-0.8%	-1.3%	-1.6%	-1.8%	-2.2%	-2.5%	-2.7%	-3.1%
p_b	-3%	-2.9%	-2.6%	-2.2%	-2%	-1.6%	-1.4%	-0.97%	-0.56%	0%
	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$	
H_m	-3.8%	-5.3%	-6.4%	-7%	-8.3%	-8.5%	-10%	-13%	-18%	
H_m(C)	-3.4%	-4%	-4.4%	-4.6%	-5%	-4.4%	-5%	-5%	-5.1%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) + 1 \star$ UP

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
p_w	2.5%	2.5%	2.5%	2.5%	2.5%	2.5%	2.5%	2.5%	2.5%	2.5%
p_s	0%	-0.36%	-0.61%	-1.1%	-1.4%	-1.6%	-2%	-2.3%	-2.5%	-2.7%
p_b	-3.3%	-3.1%	-2.8%	-2.3%	-2.2%	-1.8%	-1.6%	-1.1%	-0.76%	0%
	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$	
H_m	-6%	-6.9%	-7.9%	-8.4%	-10%	-10%	-13%	-16%	-24%	
H_m(C)	-5.3%	-5.2%	-5.5%	-5.6%	-6%	-5.4%	-6.1%	-6.3%	-7.1%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) + 1 \star$ BOTH

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
p_w	0.39%	0.39%	0.39%	0.39%	0.39%	0.39%	0.39%	0.39%	0.39%	0.39%
p_s	0%	-0.57%	-0.6%	-0.9%	-1%	-1.1%	-1.3%	-1.3%	-1.3%	-1.2%
p_b	-1.1%	-1.4%	-1.4%	-1.2%	-1.2%	-1%	-0.95%	-0.74%	-0.54%	0%
	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$	
H_m	-1.8%	-3.4%	-4.9%	-6.4%	-8%	-8.4%	-12%	-15%	-23%	
H_m(C)	-1.5%	-2.5%	-3.4%	-4.2%	-4.8%	-4.4%	-5.6%	-6.1%	-6.8%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) + 1 \star$ DOWN

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
p_w	2.5%	2.5%	2.5%	2.5%	2.5%	2.5%	2.5%	2.5%	2.5%	2.5%
p_s	0%	-0.78%	-0.99%	-1.5%	-1.7%	-2%	-2.4%	-2.7%	-3%	-3.5%
p_b	-2.6%	-2.7%	-2.4%	-2%	-1.9%	-1.5%	-1.2%	-0.8%	-0.35%	0%
	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$	
H_m	-1.6%	-3.8%	-4.8%	-5.6%	-6.7%	-6.7%	-8.1%	-9.3%	-11%	
H_m(C)	-1.4%	-2.9%	-3.4%	-3.7%	-4%	-3.5%	-3.9%	-3.7%	-3.3%	

TAB. C.16 – Comparaison des résultats analytiques avec la simulation (ligne n°4).

	M_1	M_2	M_3
U	1.000000	1.000000	1.000000
λ	0.004596	0.005158	0.005387
μ	0.095341	0.098162	0.096949
I	0.0482	0.0525	0.0555
e	0.9540	0.9500	0.9473

	$C_{1,2}$	$C_{2,3}$
C	3	3
λ	0.000461	0.000506
μ	0.099729	0.103677
I	0.0046	0.0048
e	0.9953	0.9951

TAB. C.17 – Paramètres de la ligne n°5.

Machine	P_w	P_s	P_b	P_f
M_1	0.87236	0	0.086608	0.041033
M_2	0.87234	0.037282	0.044191	0.046184
M_3	0.87234	0.07926	0	0.048396

Queue	En-cours	En-cours (corrigé)
$C_{1,2}$	2.07114	3.07114
$C_{2,3}$	1.11602	2.07874
Total	3.18716	5.14988

TAB. C.18 – Résultats de simulation à évènements discrets (ligne n°5).

$\lambda_c(ana) = 0.5 * \lambda_c(sim) \star C(ana) = C(sim) \star$ BOTH

	M_1	M_2	M_3
p_w	-0.029%	-0.028%	-0.028%
p_s	0%	-0.12%	-0.19%
p_b	-0.27%	-0.23%	0%
	$C_{1,2}$	$C_{2,3}$	
H_m	-33%	-47%	
H_m(C)	-34%	-32%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) \star$ UP

	M_1	M_2	M_3
p_w	-0.027%	-0.026%	-0.026%
p_s	0%	0.057%	0.02%
p_b	-0.48%	-0.42%	0%
	$C_{1,2}$	$C_{2,3}$	
H_m	-35%	-49%	
H_m(C)	-36%	-34%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) \star$ BOTH

	M_1	M_2	M_3
p_w	-0.66%	-0.66%	-0.66%
p_s	0%	0.0039%	0.27%
p_b	0.19%	-0.094%	0%
	$C_{1,2}$	$C_{2,3}$	
H_m	-33%	-48%	
H_m(C)	-34%	-33%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) \star$ DOWN

	M_1	M_2	M_3
p_w	-0.034%	-0.033%	-0.033%
p_s	0%	-0.3%	-0.4%
p_b	-0.066%	-0.029%	0%
	$C_{1,2}$	$C_{2,3}$	
H_m	-32%	-45%	
H_m(C)	-33%	-31%	

TAB. C.19 – Comparaison des résultats analytiques avec la simulation (ligne n°5).

$\lambda_c(ana) = 0.5 * \lambda_c(sim) \star C(ana) = C(sim) + 1 \star$ BOTH

	M_1	M_2	M_3
p_w	0.35%	0.36%	0.36%
p_s	0%	-0.32%	-0.6%
p_b	-0.68%	-0.44%	0%

	$C_{1,2}$	$C_{2,3}$	
H_m	-12%	-28%	
H_m(C)	-12%	-20%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) + 1 \star$ UP

	M_1	M_2	M_3
p_w	0.36%	0.36%	0.36%
p_s	0%	-0.14%	-0.39%
p_b	-0.88%	-0.63%	0%

	$C_{1,2}$	$C_{2,3}$	
H_m	-14%	-31%	
H_m(C)	-14%	-22%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) + 1 \star$ BOTH

	M_1	M_2	M_3
p_w	-0.26%	-0.26%	-0.26%
p_s	0%	-0.2%	-0.15%
p_b	-0.23%	-0.31%	0%

	$C_{1,2}$	$C_{2,3}$	
H_m	-11%	-29%	
H_m(C)	-11%	-20%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) + 1 \star$ DOWN

	M_1	M_2	M_3
p_w	0.35%	0.35%	0.35%
p_s	0%	-0.5%	-0.8%
p_b	-0.47%	-0.24%	0%

	$C_{1,2}$	$C_{2,3}$	
H_m	-9.8%	-25%	
H_m(C)	-10%	-17%	

TAB. C.20 – Comparaison des résultats analytiques avec la simulation (ligne n°5).

	M_1	M_2	M_3
U	1.000000	1.000000	1.000000
λ	0.004596	0.005158	0.005387
μ	0.095341	0.098162	0.096949
I	0.0482	0.0525	0.0555
e	0.9540	0.9500	0.9473

	$C_{1,2}$	$C_{2,3}$
C	10	10
λ	0.000461	0.000506
μ	0.099729	0.103677
I	0.0046	0.0048
e	0.9953	0.9951

TAB. C.21 – Paramètres de la ligne n°6.

Machine	P_w	P_s	P_b	P_f
M_1	0.89193	0	0.065072	0.042994
M_2	0.89193	0.02781	0.033402	0.046856
M_3	0.89193	0.058677	0	0.049393

Queue	En-cours	En-cours (corrigé)
$C_{1,2}$	6.61142	7.61142
$C_{2,3}$	3.98367	4.95586
Total	10.5951	12.5673

TAB. C.22 – Résultats de simulation à évènements discrets (ligne n°6).

$\lambda_c(ana) = 0.5 * \lambda_c(sim) \star C(ana) = C(sim) \star$ BOTH

	M_1	M_2	M_3
p_w	0.086%	0.086%	0.087%
p_s	0%	-0.24%	-0.33%
p_b	-0.3%	-0.27%	0%

	$C_{1,2}$	$C_{2,3}$	
H_m	-14%	-20%	
H_m(C)	-10%	-10%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) \star$ UP

	M_1	M_2	M_3
p_w	0.092%	0.092%	0.092%
p_s	0%	-0.081%	-0.11%
p_b	-0.51%	-0.45%	0%

	$C_{1,2}$	$C_{2,3}$	
H_m	-16%	-24%	
H_m(C)	-12%	-12%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) \star$ BOTH

	M_1	M_2	M_3
p_w	-0.46%	-0.46%	-0.46%
p_s	0%	-0.16%	0.038%
p_b	0.075%	-0.2%	0%

	$C_{1,2}$	$C_{2,3}$	
H_m	-13%	-21%	
H_m(C)	-9.8%	-11%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) \star$ DOWN

	M_1	M_2	M_3
p_w	0.073%	0.073%	0.073%
p_s	0%	-0.4%	-0.53%
p_b	-0.077%	-0.093%	0%

	$C_{1,2}$	$C_{2,3}$	
H_m	-11%	-17%	
H_m(C)	-8.7%	-8.3%	

TAB. C.23 – Comparaison des résultats analytiques avec la simulation (ligne n°6).

$\lambda_c(ana) = 0.5 * \lambda_c(sim) \star C(ana) = C(sim) + 1 \star$ BOTH

	M_1	M_2	M_3
p_w	0.29%	0.29%	0.29%
p_s	0%	-0.35%	-0.55%
p_b	-0.52%	-0.39%	0%
	$C_{1,2}$	$C_{2,3}$	
H_m	-5.2%	-12%	
H_m(C)	-4%	-5.7%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) + 1 \star$ UP

	M_1	M_2	M_3
p_w	0.3%	0.3%	0.3%
p_s	0%	-0.19%	-0.33%
p_b	-0.73%	-0.56%	0%
	$C_{1,2}$	$C_{2,3}$	
H_m	-7.6%	-15%	
H_m(C)	-5.8%	-7.6%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) + 1 \star$ BOTH

	M_1	M_2	M_3
p_w	-0.25%	-0.25%	-0.25%
p_s	0%	-0.27%	-0.19%
p_b	-0.15%	-0.31%	0%
	$C_{1,2}$	$C_{2,3}$	
H_m	-4.5%	-13%	
H_m(C)	-3.5%	-6.4%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) + 1 \star$ DOWN

	M_1	M_2	M_3
p_w	0.28%	0.28%	0.28%
p_s	0%	-0.5%	-0.75%
p_b	-0.29%	-0.21%	0%
	$C_{1,2}$	$C_{2,3}$	
H_m	-2.8%	-7.7%	
H_m(C)	-2.1%	-3.8%	

TAB. C.24 – Comparaison des résultats analytiques avec la simulation (ligne n°6).

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M
U	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.0000
λ	0.004567	0.004948	0.004554	0.005492	0.005334	0.005404	0.004518	0.005355	0.004997	0.0054
μ	0.108338	0.107041	0.090132	0.109559	0.092046	0.109199	0.097935	0.103334	0.108508	0.0909
I	0.0421	0.0462	0.0505	0.0501	0.0579	0.0494	0.0461	0.0518	0.0460	0.06
e	0.9595	0.9558	0.9519	0.9522	0.9452	0.9528	0.9559	0.9507	0.9559	0.94

	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$
C	3	3	3	3	3	3	3	3	3
λ	0.000542	0.000503	0.000499	0.000460	0.000518	0.000474	0.000487	0.000474	0.000457
μ	0.099689	0.102197	0.104540	0.108585	0.097354	0.108489	0.100245	0.098655	0.097928
I	0.0054	0.0049	0.0047	0.0042	0.0053	0.0043	0.0048	0.0048	0.0046
e	0.9945	0.9951	0.9952	0.9957	0.9947	0.9956	0.9951	0.9952	0.9953

TAB. C.25 – Paramètres de la ligne n°7.

Machine	P_w	P_s	P_b	P_f
M_1	0.74061	0	0.22795	0.031436
M_2	0.74061	0.027364	0.19695	0.035075
M_3	0.74061	0.051433	0.17069	0.03727
M_4	0.74061	0.074725	0.14737	0.037299
M_5	0.74061	0.095804	0.12089	0.042699
M_6	0.74061	0.12299	0.099954	0.036449
M_7	0.74061	0.14315	0.082406	0.033836
M_8	0.74061	0.16252	0.05901	0.037862
M_9	0.74061	0.18886	0.036871	0.03366
M_{10}	0.74061	0.21577	0	0.043621

Queue	En-cours	En-cours (corrigé)
$C_{1,2}$	2.58683	3.58683
$C_{2,3}$	2.22768	3.20032
$C_{3,4}$	1.99821	2.94677
$C_{4,5}$	1.75793	2.6832
$C_{5,6}$	1.50836	2.41256
$C_{6,7}$	1.28236	2.15937
$C_{7,8}$	1.1202	1.97706
$C_{8,9}$	0.832754	1.67024
$C_{9,10}$	0.518368	1.32951
Total	13.8327	21.9659

TAB. C.26 – Résultats de simulation à évènements discrets (ligne n°7).

$\lambda_c(ana) = 0.5 * \lambda_c(sim) \star C(ana) = C(sim) \star$ BOTH

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
p_w	0.45%	0.45%	0.45%	0.45%	0.45%	0.45%	0.45%	0.45%	0.45%	0.45%
p_s	0%	-0.31%	-0.39%	-0.41%	-0.47%	-0.62%	-0.64%	-0.72%	-0.75%	-0.74%
p_b	-0.65%	-0.46%	-0.46%	-0.39%	-0.39%	-0.24%	-0.21%	-0.16%	-0.12%	0%

	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$
H_m	-27%	-28%	-31%	-34%	-37%	-41%	-45%	-53%	-64%
H_m(C)	-32%	-30%	-30%	-30%	-30%	-30%	-30%	-29%	-28%

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) \star$ UP

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
p_w	0.46%	0.46%	0.46%	0.46%	0.46%	0.46%	0.46%	0.46%	0.46%	0.46%
p_s	0%	-0.15%	-0.27%	-0.31%	-0.4%	-0.5%	-0.55%	-0.61%	-0.61%	-0.58%
p_b	-0.86%	-0.61%	-0.58%	-0.48%	-0.51%	-0.33%	-0.32%	-0.29%	-0.26%	0%

	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$
H_m	-28%	-29%	-31%	-34%	-38%	-42%	-46%	-54%	-65%
H_m(C)	-33%	-31%	-30%	-30%	-30%	-30%	-30%	-30%	-29%

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) \star$ BOTH

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
p_w	-0.98%	-0.98%	-0.98%	-0.98%	-0.98%	-0.98%	-0.98%	-0.98%	-0.98%	-0.98%
p_s	0%	-0.22%	-0.1%	0.029%	0.071%	0.039%	0.16%	0.21%	0.37%	0.61%
p_b	0.64%	0.57%	0.4%	0.36%	0.24%	0.26%	0.16%	0.057%	-0.08%	0%

	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$
H_m	-27%	-28%	-31%	-34%	-37%	-41%	-46%	-53%	-65%
H_m(C)	-32%	-30%	-30%	-30%	-30%	-30%	-30%	-30%	-29%

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) \star$ DOWN

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
p_w	0.44%	0.44%	0.44%	0.44%	0.44%	0.44%	0.44%	0.44%	0.44%	0.44%
p_s	0%	-0.47%	-0.51%	-0.5%	-0.55%	-0.73%	-0.72%	-0.84%	-0.88%	-0.91%
p_b	-0.44%	-0.32%	-0.34%	-0.3%	-0.26%	-0.15%	-0.1%	-0.042%	0.017%	0%

	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$
H_m	-26%	-28%	-30%	-33%	-37%	-41%	-45%	-52%	-63%
H_m(C)	-31%	-30%	-30%	-30%	-29%	-29%	-30%	-29%	-28%

TAB. C.27 – Comparaison des résultats analytiques avec la simulation (ligne n°7).

$\lambda_c(ana) = 0.5 * \lambda_c(sim) \star C(ana) = C(sim) + 1 \star$ BOTH

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
p_w	2.5%	2.5%	2.5%	2.5%	2.5%	2.5%	2.5%	2.5%	2.5%	2.5%
p_s	0%	-0.46%	-0.77%	-1%	-1.4%	-1.8%	-2.1%	-2.4%	-2.7%	-2.9%
p_b	-2.7%	-2.4%	-2.2%	-1.9%	-1.6%	-1.2%	-0.9%	-0.58%	-0.28%	0%
	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$	
H_m	-3.8%	-5.7%	-8.5%	-12%	-16%	-21%	-25%	-33%	-47%	
H_m(C)	-4.6%	-6.1%	-8.4%	-11%	-13%	-15%	-17%	-19%	-21%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) + 1 \star$ UP

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
p_w	2.5%	2.5%	2.5%	2.5%	2.5%	2.5%	2.5%	2.5%	2.5%	2.5%
p_s	0%	-0.31%	-0.66%	-0.94%	-1.3%	-1.7%	-2%	-2.3%	-2.6%	-2.7%
p_b	-3%	-2.6%	-2.3%	-2%	-1.7%	-1.3%	-1%	-0.7%	-0.42%	0%
	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$	
H_m	-4.7%	-6.4%	-9.1%	-13%	-17%	-21%	-26%	-35%	-49%	
H_m(C)	-5.6%	-6.8%	-8.9%	-11%	-14%	-15%	-17%	-19%	-22%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) + 1 \star$ BOTH

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
p_w	1.1%	1.1%	1.1%	1.1%	1.1%	1.1%	1.1%	1.1%	1.1%	1.1%
p_s	0%	-0.38%	-0.49%	-0.62%	-0.86%	-1.2%	-1.3%	-1.6%	-1.7%	-1.6%
p_b	-1.5%	-1.5%	-1.4%	-1.2%	-1%	-0.74%	-0.55%	-0.37%	-0.25%	0%
	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$	
H_m	-3.5%	-5.8%	-8.8%	-13%	-17%	-21%	-26%	-34%	-49%	
H_m(C)	-4.2%	-6.2%	-8.7%	-11%	-13%	-15%	-17%	-19%	-22%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) + 1 \star$ DOWN

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
p_w	2.4%	2.4%	2.4%	2.4%	2.4%	2.4%	2.4%	2.4%	2.4%	2.4%
p_s	0%	-0.62%	-0.88%	-1.1%	-1.4%	-1.9%	-2.1%	-2.5%	-2.8%	-3%
p_b	-2.5%	-2.3%	-2.1%	-1.8%	-1.5%	-1.1%	-0.79%	-0.47%	-0.15%	0%
	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$	
H_m	-2.9%	-5.1%	-8%	-12%	-16%	-20%	-24%	-32%	-45%	
H_m(C)	-3.5%	-5.4%	-7.8%	-11%	-13%	-14%	-16%	-18%	-20%	

TAB. C.28 – Comparaison des résultats analytiques avec la simulation (ligne n°7).

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
U	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000
λ	0.004567	0.004948	0.004554	0.005492	0.005334	0.005404	0.004518	0.005355	0.004997	0.005467
μ	0.108338	0.107041	0.090132	0.109559	0.092046	0.109199	0.097935	0.103334	0.108508	0.090985
I	0.0421	0.0462	0.0505	0.0501	0.0579	0.0494	0.0461	0.0518	0.0460	0.0600
e	0.9595	0.9558	0.9519	0.9522	0.9452	0.9528	0.9559	0.9507	0.9559	0.9433

	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$
C	10	10	10	10	10	10	10	10	10
λ	0.000542	0.000503	0.000499	0.000460	0.000518	0.000474	0.000487	0.000474	0.000457
μ	0.099689	0.102197	0.104540	0.108585	0.097354	0.108489	0.100245	0.098655	0.097928
I	0.0054	0.0049	0.0047	0.0042	0.0053	0.0043	0.0048	0.0048	0.0046
e	0.9945	0.9951	0.9952	0.9957	0.9947	0.9956	0.9951	0.9952	0.9953

TAB. C.29 – Paramètres de la ligne n°8.

Machine	P_w	P_s	P_b	P_f
M_1	0.82119	0	0.14461	0.034195
M_2	0.82119	0.018778	0.1218	0.038229
M_3	0.82119	0.032632	0.1045	0.041684
M_4	0.82119	0.047353	0.090856	0.040605
M_5	0.82119	0.057803	0.073839	0.047171
M_6	0.82119	0.075227	0.063491	0.040092
M_7	0.8212	0.085315	0.055144	0.038344
M_8	0.8212	0.094965	0.041499	0.042337
M_9	0.8212	0.11156	0.028858	0.038374
M_{10}	0.8212	0.1293	0	0.049494

Queue	En-cours	En-cours (corrigé)
$C_{1,2}$	8.17843	9.17843
$C_{2,3}$	7.04505	8.02627
$C_{3,4}$	6.3725	7.33987
$C_{4,5}$	5.67669	6.62934
$C_{5,6}$	4.99934	5.94153
$C_{6,7}$	4.47042	5.3952
$C_{7,8}$	4.17803	5.09272
$C_{8,9}$	3.36952	4.27455
$C_{9,10}$	2.50017	3.38861
Total	46.7902	55.2665

TAB. C.30 – Résultats de simulation à évènements discrets (ligne n°8).

$\lambda_c(ana) = 0.5 * \lambda_c(sim) \star C(ana) = C(sim) \star$ BOTH

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
p_w	1.3%	1.3%	1.3%	1.3%	1.3%	1.3%	1.3%	1.3%	1.3%	1.3%
p_s	0%	-0.31%	-0.39%	-0.58%	-0.66%	-0.84%	-0.94%	-1.2%	-1.4%	-1.5%
p_b	-1.6%	-1.4%	-1.3%	-1.2%	-1.1%	-0.94%	-0.71%	-0.53%	-0.27%	0%

	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$	
H_m	-11%	-12%	-12%	-14%	-16%	-19%	-20%	-23%	-29%	
H_m(C)	-10%	-9.3%	-9.1%	-9.3%	-9.7%	-10%	-10%	-9.7%	-9.7%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) \star$ UP

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
p_w	1.3%	1.3%	1.3%	1.3%	1.3%	1.3%	1.3%	1.3%	1.3%	1.3%
p_s	0%	-0.18%	-0.32%	-0.53%	-0.63%	-0.76%	-0.89%	-1.1%	-1.3%	-1.3%
p_b	-1.8%	-1.5%	-1.4%	-1.2%	-1.2%	-0.99%	-0.79%	-0.62%	-0.39%	0%

	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$	
H_m	-12%	-12%	-13%	-14%	-17%	-19%	-21%	-24%	-31%	
H_m(C)	-11%	-9.8%	-9.5%	-9.5%	-10%	-10%	-11%	-10%	-11%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) \star$ BOTH

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
p_w	0.2%	0.2%	0.2%	0.2%	0.2%	0.2%	0.2%	0.2%	0.19%	0.19%
p_s	0%	-0.24%	-0.18%	-0.28%	-0.32%	-0.44%	-0.45%	-0.65%	-0.68%	-0.58%
p_b	-0.69%	-0.79%	-0.8%	-0.72%	-0.72%	-0.62%	-0.46%	-0.37%	-0.25%	0%

	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$	
H_m	-11%	-12%	-13%	-14%	-16%	-19%	-21%	-24%	-31%	
H_m(C)	-9.9%	-9.4%	-9.4%	-9.4%	-9.8%	-10%	-11%	-10%	-11%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) \star$ DOWN

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
p_w	1.2%	1.2%	1.2%	1.2%	1.2%	1.2%	1.2%	1.2%	1.2%	1.2%
p_s	0%	-0.43%	-0.46%	-0.63%	-0.68%	-0.92%	-0.99%	-1.3%	-1.5%	-1.7%
p_b	-1.3%	-1.3%	-1.2%	-1.1%	-1%	-0.89%	-0.63%	-0.43%	-0.15%	0%

	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$	
H_m	-10%	-11%	-12%	-14%	-16%	-18%	-19%	-21%	-26%	
H_m(C)	-9.2%	-8.8%	-8.7%	-9.1%	-9.3%	-9.8%	-9.8%	-9.1%	-8.8%	

TAB. C.31 – Comparaison des résultats analytiques avec la simulation (ligne n°8).

$\lambda_c(ana) = 0.5 * \lambda_c(sim) \star C(ana) = C(sim) + 1 \star$ BOTH

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
p_w	1.9%	1.9%	1.9%	1.9%	1.9%	1.9%	1.9%	1.9%	1.9%	1.9%
p_s	0%	-0.39%	-0.56%	-0.82%	-0.97%	-1.2%	-1.4%	-1.7%	-2%	-2.2%
p_b	-2.3%	-2%	-1.9%	-1.7%	-1.5%	-1.3%	-0.98%	-0.72%	-0.37%	0%
	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$	
H_m	-3%	-3.3%	-4.1%	-5.7%	-8.3%	-11%	-11%	-13%	-19%	
H_m(C)	-2.8%	-2.7%	-3%	-3.8%	-4.9%	-5.7%	-5.7%	-5.7%	-6.3%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) + 1 \star$ UP

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
p_w	1.9%	1.9%	1.9%	1.9%	1.9%	1.9%	1.9%	1.9%	1.9%	1.9%
p_s	0%	-0.27%	-0.49%	-0.77%	-0.94%	-1.1%	-1.3%	-1.7%	-1.9%	-2%
p_b	-2.5%	-2.2%	-1.9%	-1.7%	-1.6%	-1.3%	-1.1%	-0.81%	-0.49%	0%
	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$	
H_m	-4.4%	-4.1%	-4.7%	-6.1%	-9%	-11%	-12%	-15%	-22%	
H_m(C)	-4%	-3.3%	-3.4%	-4%	-5.4%	-6%	-6.2%	-6.4%	-7.4%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) + 1 \star$ BOTH

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
p_w	0.91%	0.91%	0.91%	0.91%	0.91%	0.91%	0.91%	0.91%	0.9%	0.9%
p_s	0%	-0.33%	-0.36%	-0.54%	-0.64%	-0.83%	-0.92%	-1.2%	-1.3%	-1.3%
p_b	-1.4%	-1.4%	-1.4%	-1.2%	-1.2%	-0.98%	-0.74%	-0.57%	-0.35%	0%
	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$	
H_m	-2.5%	-3.5%	-4.6%	-5.9%	-8.3%	-11%	-12%	-15%	-22%	
H_m(C)	-2.3%	-2.8%	-3.4%	-3.9%	-5%	-5.8%	-6.1%	-6.3%	-7.4%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) + 1 \star$ DOWN

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
p_w	1.9%	1.9%	1.9%	1.9%	1.9%	1.9%	1.9%	1.9%	1.9%	1.9%
p_s	0%	-0.51%	-0.63%	-0.86%	-0.99%	-1.3%	-1.4%	-1.8%	-2.1%	-2.4%
p_b	-2%	-1.9%	-1.8%	-1.6%	-1.4%	-1.2%	-0.9%	-0.63%	-0.25%	0%
	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$	
H_m	-1.7%	-2.6%	-3.6%	-5.4%	-7.5%	-9.9%	-10%	-12%	-16%	
H_m(C)	-1.6%	-2.1%	-2.6%	-3.6%	-4.5%	-5.3%	-5.3%	-5.1%	-5.3%	

TAB. C.32 – Comparaison des résultats analytiques avec la simulation (ligne n°8).

	M_1	M_2	M_3
U	1.000000	1.000000	1.000000
λ	0.004993	0.005178	0.005385
μ	0.099766	0.109760	0.092894
I	0.0500	0.0471	0.0579
e	0.9523	0.9549	0.9452

	$C_{1,2}$	$C_{2,3}$
C	3	3
λ	0.001985	0.002065
μ	0.095306	0.101412
I	0.0208	0.0203
e	0.9795	0.9800

TAB. C.33 – Paramètres de la ligne n°9.

Machine	P_w	P_s	P_b	P_f
M_1	0.84975	0	0.10919	0.04106
M_2	0.84975	0.050941	0.059012	0.040294
M_3	0.84975	0.10034	0	0.049908

Queue	En-cours	En-cours (corrigé)
$C_{1,2}$	2.09978	3.09978
$C_{2,3}$	1.00096	1.95002
Total	3.10073	5.04979

TAB. C.34 – Résultats de simulation à évènements discrets (ligne n°9).

$\lambda_c(ana) = 0.5 * \lambda_c(sim) \star C(ana) = C(sim) \star$ BOTH

	M_1	M_2	M_3
p_w	0.27%	0.27%	0.27%
p_s	0%	-0.91%	-1.1%
p_b	-1.3%	-1.1%	0%

	$C_{1,2}$	$C_{2,3}$	
H_m	-34%	-46%	
H_m(C)	-36%	-30%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) \star$ UP

	M_1	M_2	M_3
p_w	0.25%	0.25%	0.25%
p_s	0%	-0.1%	-0.2%
p_b	-2.2%	-1.9%	0%

	$C_{1,2}$	$C_{2,3}$	
H_m	-39%	-54%	
H_m(C)	-41%	-35%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) \star$ BOTH

	M_1	M_2	M_3
p_w	-2.3%	-2.3%	-2.3%
p_s	0%	-0.37%	0.82%
p_b	0.55%	-0.6%	0%

	$C_{1,2}$	$C_{2,3}$	
H_m	-33%	-48%	
H_m(C)	-34%	-31%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) \star$ DOWN

	M_1	M_2	M_3
p_w	0.24%	0.24%	0.24%
p_s	0%	-1.7%	-1.9%
p_b	-0.4%	-0.3%	0%

	$C_{1,2}$	$C_{2,3}$	
H_m	-29%	-38%	
H_m(C)	-30%	-25%	

TAB. C.35 – Comparaison des résultats analytiques avec la simulation (ligne n°9).

$\lambda_c(ana) = 0.5 * \lambda_c(sim) \star C(ana) = C(sim) + 1 \star$ BOTH

	M_1	M_2	M_3
p_w	0.71%	0.71%	0.71%
p_s	0%	-1.1%	-1.6%
p_b	-1.8%	-1.3%	0%
	$C_{1,2}$	$C_{2,3}$	
H_m	-13%	-27%	
H_m(C)	-14%	-17%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) + 1 \star$ UP

	M_1	M_2	M_3
p_w	0.7%	0.7%	0.7%
p_s	0%	-0.35%	-0.67%
p_b	-2.7%	-2.1%	0%
	$C_{1,2}$	$C_{2,3}$	
H_m	-20%	-38%	
H_m(C)	-21%	-25%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) + 1 \star$ BOTH

	M_1	M_2	M_3
p_w	-1.8%	-1.8%	-1.8%
p_s	0%	-0.64%	0.28%
p_b	0.014%	-0.88%	0%
	$C_{1,2}$	$C_{2,3}$	
H_m	-12%	-30%	
H_m(C)	-12%	-20%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) + 1 \star$ DOWN

	M_1	M_2	M_3
p_w	0.67%	0.67%	0.67%
p_s	0%	-1.9%	-2.4%
p_b	-0.85%	-0.55%	0%
	$C_{1,2}$	$C_{2,3}$	
H_m	-6%	-16%	
H_m(C)	-6.2%	-10%	

TAB. C.36 – Comparaison des résultats analytiques avec la simulation (ligne n°9).

	M_1	M_2	M_3
U	1.000000	1.000000	1.000000
λ	0.004993	0.005178	0.005385
μ	0.099766	0.109760	0.092894
I	0.0500	0.0471	0.0579
e	0.9523	0.9549	0.9452

	$C_{1,2}$	$C_{2,3}$
C	10	10
λ	0.001985	0.002065
μ	0.095306	0.101412
I	0.0208	0.0203
e	0.9795	0.9800

TAB. C.37 – Paramètres de la ligne n°10.

Machine	P_w	P_s	P_b	P_f
M_1	0.87178	0	0.084295	0.043922
M_2	0.87178	0.040868	0.046968	0.040385
M_3	0.87177	0.077696	0	0.050533

Queue	En-cours	En-cours (corrigé)
$C_{1,2}$	6.77122	7.77122
$C_{2,3}$	3.66381	4.62294
Total	10.435	12.3942

TAB. C.38 – Résultats de simulation à évènements discrets (ligne n°10).

$\lambda_c(ana) = 0.5 * \lambda_c(sim) \star C(ana) = C(sim) \star$ BOTH

	M_1	M_2	M_3
p_w	0.46%	0.46%	0.46%
p_s	0%	-1.2%	-1.4%
p_b	-1.4%	-1.2%	0%

	$C_{1,2}$	$C_{2,3}$	
H_m	-16%	-18%	
H_m(C)	-12%	-8.1%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) \star$ UP

	M_1	M_2	M_3
p_w	0.43%	0.43%	0.44%
p_s	0%	-0.43%	-0.46%
p_b	-2.3%	-1.9%	0%

	$C_{1,2}$	$C_{2,3}$	
H_m	-23%	-31%	
H_m(C)	-18%	-14%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) \star$ BOTH

	M_1	M_2	M_3
p_w	-1.8%	-1.8%	-1.8%
p_s	0%	-0.81%	0.16%
p_b	0.14%	-0.9%	0%

	$C_{1,2}$	$C_{2,3}$	
H_m	-14%	-22%	
H_m(C)	-11%	-10%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) \star$ DOWN

	M_1	M_2	M_3
p_w	0.37%	0.37%	0.37%
p_s	0%	-1.8%	-2.2%
p_b	-0.36%	-0.45%	0%

	$C_{1,2}$	$C_{2,3}$	
H_m	-7.9%	-5%	
H_m(C)	-6.2%	-2.3%	

TAB. C.39 – Comparaison des résultats analytiques avec la simulation (ligne n°10).

$\lambda_c(ana) = 0.5 * \lambda_c(sim) \star C(ana) = C(sim) + 1 \star$ BOTH

	M_1	M_2	M_3
p_w	0.7%	0.7%	0.7%
p_s	0%	-1.3%	-1.6%
p_b	-1.6%	-1.3%	0%
	$C_{1,2}$	$C_{2,3}$	
H_m	-7.6%	-8.5%	
H_m(C)	-5.9%	-3.9%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) + 1 \star$ UP

	M_1	M_2	M_3
p_w	0.67%	0.67%	0.67%
p_s	0%	-0.56%	-0.71%
p_b	-2.5%	-2%	0%
	$C_{1,2}$	$C_{2,3}$	
H_m	-16%	-23%	
H_m(C)	-12%	-11%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) + 1 \star$ BOTH

	M_1	M_2	M_3
p_w	-1.5%	-1.5%	-1.5%
p_s	0%	-0.96%	-0.13%
p_b	-0.15%	-1%	0%
	$C_{1,2}$	$C_{2,3}$	
H_m	-5.7%	-13%	
H_m(C)	-4.4%	-6.1%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) + 1 \star$ DOWN

	M_1	M_2	M_3
p_w	0.6%	0.61%	0.61%
p_s	0%	-2%	-2.4%
p_b	-0.61%	-0.58%	0%
	$C_{1,2}$	$C_{2,3}$	
H_m	1.1%	5.4%	
H_m(C)	0.84%	2.5%	

TAB. C.40 – Comparaison des résultats analytiques avec la simulation (ligne n°10).

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
U	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000
λ	0.004809	0.004697	0.004746	0.005451	0.004951	0.005163	0.005284	0.004968	0.004882	0.004919
μ	0.105925	0.090056	0.090084	0.109406	0.100332	0.091955	0.104865	0.106432	0.102087	0.092959
I	0.0454	0.0521	0.0526	0.0498	0.0493	0.0561	0.0503	0.0466	0.0478	0.0529
e	0.9565	0.9504	0.9499	0.9525	0.9529	0.9468	0.9520	0.9554	0.9543	0.9497

	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$
C	3	3	3	3	3	3	3	3	3
λ	0.001937	0.001954	0.001867	0.001968	0.002016	0.002126	0.002092	0.001977	0.002052
μ	0.094522	0.100577	0.105601	0.102805	0.096432	0.093302	0.105043	0.103915	0.103557
I	0.0204	0.0194	0.0176	0.0191	0.0209	0.0227	0.0199	0.0190	0.0198
e	0.9799	0.9809	0.9826	0.9812	0.9795	0.9777	0.9804	0.9813	0.9805

TAB. C.41 – Paramètres de la ligne n°11.

Machine	P_w	P_s	P_b	P_f
M_1	0.68383	0	0.28534	0.030827
M_2	0.68383	0.037638	0.24353	0.035001
M_3	0.68383	0.070522	0.20972	0.035931
M_4	0.68382	0.098924	0.18311	0.034147
M_5	0.68382	0.12398	0.15728	0.034926
M_6	0.68382	0.15099	0.12654	0.038654
M_7	0.68382	0.18425	0.097908	0.034021
M_8	0.68382	0.21271	0.07121	0.032258
M_9	0.68382	0.24197	0.041276	0.032937
M_{10}	0.68382	0.28024	0	0.035942

Queue	En-cours	En-cours (corrigé)
$C_{1,2}$	2.65464	3.65464
$C_{2,3}$	2.27213	3.23449
$C_{3,4}$	1.98826	2.91774
$C_{4,5}$	1.74777	2.64885
$C_{5,6}$	1.53234	2.40836
$C_{6,7}$	1.27803	2.12705
$C_{7,8}$	1.01973	1.83548
$C_{8,9}$	0.750433	1.53772
$C_{9,10}$	0.376245	1.13428
Total	13.6196	21.4986

TAB. C.42 – Résultats de simulation à évènements discrets (ligne n°11).

$\lambda_c(ana) = 0.5 * \lambda_c(sim) \star C(ana) = C(sim) \star$ BOTH

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
p_w	1.6%	1.6%	1.6%	1.6%	1.6%	1.6%	1.6%	1.6%	1.6%	1.6%
p_s	0%	-0.94%	-0.98%	-1%	-1.2%	-1.6%	-1.9%	-2%	-2.2%	-2.4%
p_b	-2.4%	-2.2%	-2%	-1.9%	-1.7%	-1.6%	-1.3%	-0.99%	-0.85%	0%

	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$	
H_m	-28%	-29%	-31%	-34%	-37%	-41%	-46%	-53%	-65%	
H_m(C)	-34%	-32%	-30%	-30%	-30%	-29%	-28%	-27%	-25%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) \star$ UP

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
p_w	1.6%	1.6%	1.6%	1.6%	1.6%	1.6%	1.6%	1.6%	1.6%	1.6%
p_s	0%	-0.38%	-0.54%	-0.71%	-0.87%	-1.2%	-1.4%	-1.6%	-1.7%	-1.7%
p_b	-3.1%	-2.7%	-2.4%	-2.3%	-2.1%	-2.1%	-1.7%	-1.4%	-1.4%	0%

	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$	
H_m	-30%	-31%	-32%	-35%	-38%	-43%	-48%	-56%	-71%	
H_m(C)	-36%	-33%	-31%	-31%	-31%	-30%	-29%	-28%	-27%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) \star$ BOTH

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
p_w	-3.2%	-3.2%	-3.2%	-3.2%	-3.2%	-3.2%	-3.2%	-3.2%	-3.2%	-3.2%
p_s	0%	-0.7%	-0.14%	0.19%	0.36%	0.42%	0.61%	1.1%	1.5%	2.1%
p_b	2%	1.4%	1.1%	0.8%	0.53%	0.16%	-0.046%	-0.23%	-0.67%	0%

	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$	
H_m	-27%	-29%	-31%	-34%	-37%	-41%	-46%	-54%	-68%	
H_m(C)	-33%	-31%	-30%	-30%	-29%	-29%	-28%	-27%	-26%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) \star$ DOWN

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
p_w	1.6%	1.6%	1.6%	1.6%	1.6%	1.6%	1.6%	1.6%	1.6%	1.6%
p_s	0%	-1.5%	-1.4%	-1.4%	-1.6%	-2%	-2.4%	-2.5%	-2.7%	-3.1%
p_b	-1.7%	-1.7%	-1.6%	-1.5%	-1.3%	-1.1%	-0.87%	-0.57%	-0.32%	0%

	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$	
H_m	-26%	-28%	-30%	-33%	-35%	-39%	-43%	-50%	-60%	
H_m(C)	-32%	-30%	-29%	-29%	-28%	-27%	-26%	-25%	-23%	

TAB. C.43 – Comparaison des résultats analytiques avec la simulation (ligne n°11).

$\lambda_c(ana) = 0.5 * \lambda_c(sim) \star C(ana) = C(sim) + 1 \star$ BOTH

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
p_w	3.8%	3.8%	3.8%	3.8%	3.8%	3.8%	3.8%	3.8%	3.8%	3.8%
p_s	0%	-1.1%	-1.4%	-1.7%	-2.2%	-2.9%	-3.6%	-3.9%	-4.3%	-4.7%
p_b	-4.7%	-4.4%	-4%	-3.6%	-3.1%	-2.6%	-2%	-1.4%	-1%	0%
	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$	
H_m	-5.3%	-7.3%	-9.5%	-13%	-16%	-20%	-26%	-33%	-50%	
H_m(C)	-6.4%	-7.9%	-9.3%	-11%	-13%	-14%	-16%	-17%	-19%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) + 1 \star$ UP

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
p_w	3.8%	3.8%	3.8%	3.8%	3.8%	3.8%	3.8%	3.8%	3.8%	3.8%
p_s	0%	-0.56%	-0.98%	-1.4%	-1.9%	-2.5%	-3.1%	-3.5%	-3.9%	-4%
p_b	-5.5%	-4.9%	-4.3%	-3.9%	-3.5%	-3.1%	-2.4%	-1.8%	-1.5%	0%
	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$	
H_m	-7.8%	-9.3%	-11%	-15%	-18%	-23%	-29%	-38%	-58%	
H_m(C)	-9.5%	-10%	-11%	-13%	-14%	-16%	-18%	-19%	-22%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) + 1 \star$ BOTH

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
p_w	-0.82%	-0.82%	-0.82%	-0.82%	-0.82%	-0.82%	-0.82%	-0.82%	-0.82%	-0.82%
p_s	0%	-0.87%	-0.58%	-0.57%	-0.77%	-1.1%	-1.2%	-1%	-0.89%	-0.5%
p_b	-0.54%	-1%	-1.1%	-1%	-0.96%	-1%	-0.86%	-0.7%	-0.84%	0%
	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$	
H_m	-4.2%	-6.9%	-9.3%	-12%	-15%	-20%	-26%	-35%	-53%	
H_m(C)	-5.2%	-7.4%	-9%	-11%	-12%	-14%	-16%	-18%	-20%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) + 1 \star$ DOWN

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
p_w	3.8%	3.8%	3.8%	3.8%	3.8%	3.8%	3.8%	3.8%	3.8%	3.8%
p_s	0%	-1.6%	-1.8%	-2%	-2.6%	-3.3%	-4%	-4.3%	-4.8%	-5.5%
p_b	-4%	-3.9%	-3.6%	-3.2%	-2.7%	-2.2%	-1.6%	-1%	-0.5%	0%
	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$	
H_m	-2.7%	-5.3%	-8%	-11%	-14%	-17%	-23%	-29%	-41%	
H_m(C)	-3.3%	-5.7%	-7.7%	-9.8%	-11%	-12%	-14%	-15%	-16%	

TAB. C.44 – Comparaison des résultats analytiques avec la simulation (ligne n°11).

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M
U	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.0000
λ	0.004809	0.004697	0.004746	0.005451	0.004951	0.005163	0.005284	0.004968	0.004882	0.0049
μ	0.105925	0.090056	0.090084	0.109406	0.100332	0.091955	0.104865	0.106432	0.102087	0.0929
I	0.0454	0.0521	0.0526	0.0498	0.0493	0.0561	0.0503	0.0466	0.0478	0.05
e	0.9565	0.9504	0.9499	0.9525	0.9529	0.9468	0.9520	0.9554	0.9543	0.94

	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$
C	10	10	10	10	10	10	10	10	10
λ	0.001937	0.001954	0.001867	0.001968	0.002016	0.002126	0.002092	0.001977	0.002052
μ	0.094522	0.100577	0.105601	0.102805	0.096432	0.093302	0.105043	0.103915	0.103557
I	0.0204	0.0194	0.0176	0.0191	0.0209	0.0227	0.0199	0.0190	0.0198
e	0.9799	0.9809	0.9826	0.9812	0.9795	0.9777	0.9804	0.9813	0.9805

TAB. C.45 – Paramètres de la ligne n°12.

Machine	P_w	P_s	P_b	P_f
M_1	0.7735	0	0.19143	0.035072
M_2	0.7735	0.030984	0.15518	0.040335
M_3	0.77351	0.05358	0.13209	0.04082
M_4	0.7735	0.069795	0.1181	0.038604
M_5	0.7735	0.083383	0.10435	0.038759
M_6	0.7735	0.098496	0.083815	0.044187
M_7	0.7735	0.1209	0.065977	0.039622
M_8	0.7735	0.13791	0.052233	0.036354
M_9	0.7735	0.15497	0.034472	0.037066
M_{10}	0.7735	0.18571	0	0.040796

Queue	En-cours	En-cours (corrigé)
$C_{1,2}$	8.42683	9.42683
$C_{2,3}$	7.04316	8.01218
$C_{3,4}$	6.28457	7.23099
$C_{4,5}$	5.6492	6.5794
$C_{5,6}$	5.09733	6.01395
$C_{6,7}$	4.35433	5.25583
$C_{7,8}$	3.64904	4.52814
$C_{8,9}$	3.00663	3.86872
$C_{9,10}$	1.73556	2.58059
Total	45.2466	53.4966

TAB. C.46 – Résultats de simulation à évènements discrets (ligne n°12).

$\lambda_c(ana) = 0.5 * \lambda_c(sim) \star C(ana) = C(sim) \star$ BOTH

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
p_w	2.5%	2.5%	2.5%	2.5%	2.5%	2.5%	2.5%	2.5%	2.5%	2.5%
p_s	0%	-1.2%	-1.4%	-1.5%	-1.8%	-2.3%	-2.7%	-2.9%	-3.1%	-3.5%
p_b	-3.5%	-3.1%	-2.8%	-2.7%	-2.4%	-2.1%	-1.6%	-1.3%	-1.2%	0%
	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$	
H_m	-13%	-13%	-14%	-15%	-16%	-17%	-18%	-22%	-24%	
H_m(C)	-12%	-10%	-10%	-10%	-9.5%	-8.8%	-8.3%	-8.4%	-6.3%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) \star$ UP

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
p_w	2.5%	2.5%	2.5%	2.5%	2.5%	2.5%	2.5%	2.5%	2.5%	2.5%
p_s	0%	-0.75%	-1.1%	-1.3%	-1.6%	-2%	-2.3%	-2.5%	-2.7%	-2.7%
p_b	-4.3%	-3.5%	-3%	-2.9%	-2.7%	-2.4%	-1.9%	-1.6%	-1.6%	0%
	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$	
H_m	-17%	-15%	-15%	-17%	-18%	-20%	-21%	-26%	-37%	
H_m(C)	-16%	-12%	-11%	-11%	-11%	-10%	-9.7%	-10%	-9.5%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) \star$ BOTH

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
p_w	-1.3%	-1.3%	-1.3%	-1.3%	-1.3%	-1.3%	-1.3%	-1.3%	-1.3%	-1.3%
p_s	0%	-1%	-0.76%	-0.64%	-0.81%	-0.99%	-1%	-0.79%	-0.55%	-0.17%
p_b	-0.22%	-0.64%	-0.7%	-0.8%	-0.82%	-0.89%	-0.79%	-0.8%	-1%	0%
	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$	
H_m	-12%	-12%	-13%	-14%	-15%	-17%	-19%	-23%	-31%	
H_m(C)	-11%	-9.8%	-9.7%	-9.4%	-9%	-8.7%	-8.6%	-9.1%	-7.9%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) \star$ DOWN

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
p_w	2.5%	2.5%	2.5%	2.5%	2.5%	2.5%	2.5%	2.5%	2.5%	2.5%
p_s	0%	-1.7%	-1.6%	-1.6%	-2%	-2.5%	-3.1%	-3.2%	-3.5%	-4.3%
p_b	-2.7%	-2.6%	-2.6%	-2.4%	-2.1%	-1.8%	-1.4%	-1%	-0.7%	0%
	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$	
H_m	-9.8%	-11%	-13%	-14%	-14%	-14%	-15%	-17%	-12%	
H_m(C)	-9.2%	-8.6%	-9.2%	-9.2%	-8.3%	-7.4%	-6.9%	-6.7%	-3.2%	

TAB. C.47 – Comparaison des résultats analytiques avec la simulation (ligne n°12).

$\lambda_c(ana) = 0.5 * \lambda_c(sim) \star C(ana) = C(sim) + 1 \star$ BOTH

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
p_w	3.3%	3.3%	3.3%	3.3%	3.3%	3.3%	3.3%	3.3%	3.3%	3.3%
p_s	0%	-1.3%	-1.6%	-1.8%	-2.2%	-2.7%	-3.2%	-3.5%	-3.8%	-4.3%
p_b	-4.3%	-3.8%	-3.4%	-3.2%	-2.9%	-2.5%	-1.9%	-1.5%	-1.3%	0%
	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$	
H_m	-5.2%	-4.8%	-6%	-7.3%	-7.3%	-8.3%	-9.7%	-13%	-14%	
H_m(C)	-4.9%	-3.9%	-4.3%	-4.8%	-4.4%	-4.4%	-4.4%	-4.8%	-3.7%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) + 1 \star$ UP

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
p_w	3.3%	3.3%	3.3%	3.3%	3.3%	3.3%	3.3%	3.3%	3.3%	3.3%
p_s	0%	-0.86%	-1.3%	-1.6%	-2%	-2.5%	-2.9%	-3.2%	-3.4%	-3.5%
p_b	-5.1%	-4.2%	-3.6%	-3.4%	-3.1%	-2.8%	-2.2%	-1.8%	-1.7%	0%
	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$	
H_m	-9%	-7.2%	-7.4%	-8.9%	-9.3%	-11%	-13%	-17%	-28%	
H_m(C)	-8.5%	-5.8%	-5.4%	-5.8%	-5.6%	-6%	-5.9%	-6.6%	-7.3%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) + 1 \star$ BOTH

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
p_w	-0.42%	-0.42%	-0.42%	-0.42%	-0.42%	-0.42%	-0.42%	-0.42%	-0.42%	-0.42%
p_s	0%	-1.2%	-0.98%	-0.97%	-1.2%	-1.5%	-1.7%	-1.5%	-1.4%	-1.1%
p_b	-1.1%	-1.5%	-1.4%	-1.4%	-1.3%	-1.3%	-1.1%	-1%	-1.2%	0%
	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$	
H_m	-3.5%	-4.2%	-5.4%	-6%	-6.4%	-8%	-10%	-14%	-21%	
H_m(C)	-3.3%	-3.3%	-3.9%	-4%	-3.9%	-4.2%	-4.7%	-5.6%	-5.5%	

$\lambda_c(ana) = 1 * \lambda_c(sim) \star C(ana) = C(sim) + 1 \star$ DOWN

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
p_w	3.3%	3.3%	3.3%	3.3%	3.3%	3.3%	3.3%	3.3%	3.3%	3.3%
p_s	0%	-1.7%	-1.8%	-1.9%	-2.4%	-3%	-3.6%	-3.8%	-4.2%	-5.1%
p_b	-3.5%	-3.4%	-3.2%	-3%	-2.6%	-2.1%	-1.7%	-1.3%	-0.82%	0%
	$C_{1,2}$	$C_{2,3}$	$C_{3,4}$	$C_{4,5}$	$C_{5,6}$	$C_{6,7}$	$C_{7,8}$	$C_{8,9}$	$C_{9,10}$	
H_m	-1.3%	-2.5%	-4.7%	-5.7%	-5.2%	-5.2%	-6.3%	-7.7%	-0.39%	
H_m(C)	-1.2%	-2%	-3.4%	-3.7%	-3.1%	-2.7%	-2.8%	-3%	-0.1%	

TAB. C.48 – Comparaison des résultats analytiques avec la simulation (ligne n°12).

D

Détails sur le simulateur à évènements discrets

Résumé

L'observation d'un cas réel de ligne de production manufacturière de nouvelle génération qui se met en place chez PSA-Peugeot Citroën permet de mettre à jour un certain nombre de points que l'on ne sait pas traiter par des méthodes analytiques telles quelles. Dans ce travail, nous avons étudié ces points. En particulier, nous avons proposé une nouvelle méthode de résolution de l'algorithme DDX appliqué aux lignes fermées, puis nous l'avons étendue aux lignes fermées dont la topologie peut être quelconque. Nous obtenons au final un algorithme très rapide, très robuste, et qui reste raisonablement précis malgré la complexité des modèles analysés.

Nous avons aussi étudié et proposé des méthodes simples pour traiter les mécanismes de pannes qui peuvent survenir dans les zones de stockage qui se trouvent entre les machines de ces lignes de production. Ces mécanismes sont traditionellement négligé dans ce genre d'analyse, et nous proposons des méthodes qui permettent de les prendre en compte sans pour autant augmenter la complexité des analyses.

Enfin, nous avons proposé un logiciel simple et complet, développé pour PSA, pour permettre une analyse approximative mais très rapide de la ligne de production d'un atelier de ferrage automobile.

Mots-clés: décomposition, système de production, multi-boucle, fermé, pannes de stock

Abstract

The observation of a real exmaple of production line from the french car manufacturer PSA Peugeot Citroën, allows to reveal some points that cannot be treatened by existing analytical methods as is.

We thus proposed a new method to solve the DDX algorithm applied to simple closed-loop production lines. We then extended this new method to closed-loop production lines with complex topology (consisting in several loops lines). We propose a very efficient and robust algorithm, which results in very acceptable estimation of the performance parameters of the line.

We also studied, and proposed simple solutions, to deal with unreliable buffers. The failures of thoses buffers are generally less important than the failures of the machines, and thus are often neglicted in analytical studies of the production lines. We thus propose simple methods for taking those buffer failures into account in the analysis of the production line without increasing the complexity of the analyse.

Last, we designed and implemented a graphical software for PSA, so they may design and analyse one of their flexible production line, called "atelier de ferrage", which gives immediate estimations of the performance of the line.

Keywords: decomposition method, production line, closed-loop, mutli-loop, unreliable buffers.