UNIVERSITE SAAD DAHLAB DE BLIDA

Faculté des Sciences

Département de Mathématiques

MEMOIRE DE MAGISTER

Spécialité : Recherche Opérationnelle

LE FLOW SHOP STOCHASTIQUE

A 2-MACHINES

Par

MEHDI Ouafia

Devant le jury composé de :

M. Blidia	Professeur, U. de Blida	Président
S. Bouroubi	Maître de conférences, U.S.T.H.B, Alger	Examinateur
M. Boudhar	Maître de conférences, U.S.T.H.B, Alger	Examinateur
H. Ould Rouis	Maître de conférences, U. de Blida	Examinateur
N. Grangeon	Maître de conférences, U. de Clermont-Ferrand	Invitée
A. Derbala	Maître de conférences, U. de Blida	Rapporteur

Blida, Février 2008.

RESUME

Nous considérons le problème d'ordonnancement de type flowshop stochastique à 2-machines afin de minimiser l'espérance mathématique de la longueur d'ordonnancement appelée makespan et notée E (C_{max}). Les règles de Johnson [01] et Talwar [02] établissent que dans un ordonnancement optimal, une tâche i précède une tâche j si $E(\min(A_i , B_j)) \leq E (\min (A_j , B_i))$, où A_i , B_i et « E » représentent respectivement des variables aléatoires associées aux temps d'exécution de la tâche i sur la première machine, sur la seconde machine et leurs espérances mathématiques. De notre recherche bibliographique, les principaux résultats établis dans la littérature sont présentés. Nous reprenons, dans leur grande ligne et leurs principales originalités, les travaux qui, selon nous, marquent des jalons importants dans la résolution du flowshop stochastique à 2-machines par les ordres stochastiques. Pour évaluer E(C_{max}), une heuristique a été étudiée, implémentée et testée sur un nombre de données élevé des temps d'exécution de tâches supposés aléatoires de distribution exponentielle. Nous avons vérifié expérimentalement qu'elle est asymptotiquement convergente vers la solution optimale. Son évaluation constituera une borne inférieure pour notre objectif.

MOTS-CLES : flow shop, longueur d'ordonnancement, ordre stochastique, Simulation.

ABSTRACT

We consider a 2-machine stochastic flowshop scheduling problem to minimize the expected schedule length or makespan denoted E (C_{max}). Johnson [01] and Talwar [02] rules claimed that in an optimal scheduling, a task i precede task j if $E (\min (A_i, B_j)) \leq E (\min(A_j , B_i))$ where A_i , B_i and « E » are associated respectively to the stochastic variables of the processing times on the first machine, the second machine and their expectations. From our bibliography research, the established results in the literature are presented. We outline from their spirit and their originality, the works carried out using stochastic orders to solve stochastic flowshop. For evaluating E (C_{max}), an heuristic is studied, implemented and tested on a huge number of tasks processing times supposed to fill an exponential distribution. We experimentally verify the asymptotic convergence to the optimal solution.

KEYWORDS: flowshop, makespan, stochastic order , simulation.

ملخص

نعتبر إشكالية الترتيب من نوع فلوشوب لآلتين ، من أجل إعطاء حد أدنى للمتوسط الحسابي لطول الترتيب المسمى ماكسبان و الممثلة ب$E(C_{max})$.

إن القواعد المتبعة من طرف جونسون [01] و تالور [02] تنص على أنه في الترتيب الأمثل، العمل i يسبق العمل إذا تحقق $E(\min(A_i, B_j)) \leq E(\min(A_j, B_i))$

أين A_i ، B_i و E تمثل على الترتيب متغيرات عشوائية ملحقة لأزمنة الانجاز للعمل i على الآلة الأولى ، الآلة الثانية و المتوسط الحسابي لهذه المتغيرات.

من بحثنا في المرا جع ، النتائج الأساسية مدونة ،ممثلة في البحوث للقواعد الأساسية ، و التي تعتبر معالم مهمة في حل الفلوشوب لآلتين بواسطة الترتيبات العرضية. لتقدير قيمة $E(C_{max})$ منهج كشف اتبع، درس و اختبر على عدد معتبر من المعطيات ذوي أزمنة تنفيذ عشوائية للأعمال المقترحة بتوزيع أسي، لقد تحققنا تجريبيا على أن منهج الكشف متقارب نحو الحل الأمثل، تقديره يمثل حدا أدنى بالنسبة لهدفنا.

كلمات المفتاح: فلوشوب، الترتيب العرضي ، التمثيل الناظمي.

REMERCIEMENTS

Je remercie tout d'abord, notre vénéré Allah,
Le tout puissant, à qui nous devons le tout.

Je tiens à exprimer ma gratitude à mon responsable de recherche Monsieur Derbala Ali**,** maître de conférences à l'université Saad Dahlab de Blida, pour avoir diriger mes travaux de cette recherche. Je le remercie également pour ces précieux conseils, et surtout sa disponibilité.

Je remercie Monsieur Blidia Mostafa, Professeur à l'université Saad Dahlab de Blida, pour avoir accepté d'examiner ce travail, et qui m'a fait l'honneur de présider ce jury. Qu'il trouve ici l'expression de ma profonde reconnaissance.

J'adresse mes sincères remerciements à Monsieur Ould Rouis Hamid, maître de conférences à l'université Saad Dahlab de Blida, à Messieurs Boudhar Mourad et Bouroubi Sadek, Maîtres de conférences à l'université Houari Boumediene d'Alger, de Madame Nathalie Grangeon, maître de conférences à l'université de Clermont-Ferrand, LIMOS de France, d'avoir accepté d'être membre de jury et de m'avoir fait l'honneur d'examiner mon travail.

Je tiens à remercier Monsieur Tami Omar, chef de département de Mathématiques de l'université Saad Dahlab de Blida, pour ses encouragements.

Je ne saurais oublier de remercier mes chers parents, qui étaient toujours à mes cotés et m'ont tant aidé et soutenu. Qu'ils trouvent ici l'expression de ma sincère gratitude et ma profonde reconnaissance.

Mes remerciements s’adressent également à mon frère Merouane de m’avoir aidé pour la réalisation de ce mémoire, à ma sœur Nadia et ma belle sœur Ibtissem pour leurs encouragements.

Je remercie enfin mes chères amies Amina, Selma et Nadia, tous mes camarades et compagnons de recherche du département de mathématiques pour leur éternelle bonne humeur et leur sympathie.

TABLE DES MATIERES

LISTE DES ILLUSTRATIONS, GRAPHIQUES ET TABLEAUX

INTRODUCTION

Dans un atelier manufacturier, un ensemble de " n " tâches sont à exécuter sur 2-machines disposées en série ou en ligne appelée flowshop. Les tâches sont supposées être dans l'atelier à l'instant zéro. Il n'est pas requis que chaque tâche doit s'exécuter sur chaque machine. Dans ce cas, son temps d'exécution sur la dite machine est nul. A_i et B_i représentent les temps d'exécution de la tâche « i » respectivement sur la première et la seconde machine. Deux modèles interviennent, avec une aire et sans aire de stockage entre les deux machines. Un espace ou une aire de stockage ou une zone tampon sont synonymes. Quand une tâche a terminé son exécution sur la première machine, elle est stockée dans une zone tampon avant la seconde machine, lorsque celle-ci est occupée dans l'exécution d'une tâche. Dans le cas où la zone n'existe pas, la tâche sera bloquée sur la première machine si la seconde machine n'a pas terminé son exécution. Les actions de charger, décharger ou mettre une pièce dans la zone tampon sont accomplies par un robot ou un chariot ou une navette. Ce chargeur n'est pas considéré comme une machine. Les contraintes classiques de l'atelier sont imposées. Une machine ne peut exécuter plus d'une tâche à la fois et chaque tâche n'est exécutée que sur une seule machine. Un ordonnancement est formé d'une permutation de « n » tâches, associée à une liste. Dés qu'une machine est libre, la tâche suivante de la liste est exécutée. Cette liste reste inchangée durant l'évolution du processus. Nous considérons que l'ordonnancement est non-préemptif.

Dans le cas où les temps d'exécutions sont connus ou déterministes, et si l'objectif est de déterminer une séquence de tâches optimale qui minimise la longueur d'ordonnancement C_{max}, l'algorithme de Johnson [01] suffit à produire un tel ordonnancement. Il est résoluble en O(n log n). Il consiste à partitionner l'ensemble des tâches en deux sous ensembles, dans le premier, les tâches qui ont moins de

temps d'exécution sur la première machine sont ordonnancées dans l'ordre croissant de leur temps d'exécution, dans le second le reste des tâches le sont par ordre décroissant. On exécute dans l'ordre les deux sous ensembles de tâches. Il se note SPT(1)-LPT(2) qui est optimal pour $F2//C_{max}$. L'algorithme peut être formulé par l'assertion suivante : Dans un ordonnancement optimal, qui peut ne pas être unique, une tâche i précède une tâche j si Min (A_i, B_j) < Min (A_j, B_i).

Les problèmes réels du monde incluent quelques paramètres inconnus, notamment dans le système industriel, la non disponibilité des tâches, la panne sur les machines etc. Ils impliquent une certaine incertitude. Cependant considérer le système dans ce contexte est plus réaliste que le cas déterministe. L'aléatoire peut représenter, l'incertitude de l'ordonnanceur sur les temps d'exécution, des erreurs sur les mesures des temps d'exécution, les fluctuations aléatoires dans la fonction objectif ou la vitesse de l'opérateur et / ou de la machine, la non homogénéité des tâches nécessitant de différents temps d'exécution et / ou l'aléatoire en temps requis pour le réglage de la machine pour les différentes tâches à exécuter. Chaque type de tâche répond à une demande spécifique qui possède généralement des fluctuations aléatoires autour d'une valeur moyenne. La présence de ces fluctuations ne permet pas de connaître précisément les demandes futures et impose donc une grande réactivité du gestionnaire de l'atelier afin d'assurer la satisfaction de sa clientèle.

Dans la suite de ce mémoire, les temps d'exécution des tâches sont supposés aléatoires de lois connues. Les problèmes où les temps sont incertains de lois connues sont dits stochastiques. La résolution des problèmes de flowshop stochastiques se fait en général par l'utilisation d'au moins trois classes de méthodes stochastiques, parmi ces méthodes on cite le recuit simulé, les algorithmes génétiques et les processus bandits, des processus à 2-décisions markovien etc.

La comparaison stochastique est utile chaque fois que le système comporte des éléments incertains mais qu'ils sont supposés suivre des distributions de probabilité connues. Elle détermine et estime des mesures de performance tel la moyenne ou la distribution du critère à optimiser ...etc.

Dans notre cas d'étude, l'outil de résolution des problèmes sera « les ordres stochastiques ». Cette approche consiste à comparer, des variables aléatoires selon un ordre stochastique et par suite comparer les ordonnancements qui sont formés d'une suite finie de variables aléatoires. On définit des formes de dominances stochastiques entre les tâches à qui on leurs associe un temps de traitement aléatoire sur la machine correspondante, afin de minimiser la longueur d'ordonnancement.

Le mémoire est organisé en cinq chapitres:
Dans le premier, on présente et on formule le problème étudié. Nous donnons des notions sur les ordres stochastiques.
La résolution du flow shop stochastique est l'objet de considération dans le second et le troisième chapitre où nous présentons les différents travaux à deux machines, avec et sans zone tampon. Une recherche bibliographique a été faite et quelques résultats obtenus ont été exposés et démontrés.

Si les *temps d'exécutions* suivent une loi de distribution *exponentielle* et sont indépendants entre eux, de moyenne égale à l'inverse du paramètre de la loi, le problème de minimiser l'espérance du makespan est résolu exactement par la règle de Talwar [02]. Elle stipule qu'une tâche i précède une tâche j si et seulement si

$$1/ E(A_i) - 1/ E(B_i) \geq 1/E(A_j)-1/ E(B_j).$$

L'optimalité de cette règle a été prouvé par Cunningham et Dutta [03]. Nous proposons une méthodologie de recherche de la solution optimale. Pour évaluer l'espérance du makespan $E(C_{max})$, ces auteurs ont proposé une formule récurrente utilisant les chaînes de Markov et les équations de Chapman-Kolmogorov. Dans le quatrième chapitre, la règle de Talwar ainsi qu'un schéma itératif pour le flowshop à deux machines sont exposés. Une heuristique appelée « Pseudo Déterministe de Johnson » notée PDJ tirée de la littérature [04] est présentée. C'est la version stochastique de l'algorithme de Johnson quand les données sont les espérances des temps d'exécutions. Elle a été étudiée, implémentée et testée sur un nombre élevé de données. Nous proposons sa comparaison avec la solution optimale. Nous vérifions expérimentalement qu'elle est asymptotiquement convergente en espérance vers la solution optimale avec une probabilité égale à « un » dite aussi « presque sûrement ». Son évaluation est

une borne inférieure pour l'espérance du makespan. Les résultats expérimentaux et la performance de cette heuristique ont été discutés. L'heuristique de résolution PDJ, la recherche de la séquence de Talwar et l'évaluation de l'espérance du makespan ont été implémentés en utilisant la procédure rand() du logiciel scilab-3.1.1 de calcul numérique, copyright (c) 1989-2005 Consortium Scilab (INRIA, ENPC). Elle est utilisée pour générer des variables aléatoires. Les tests ont été réalisés sur un ordinateur Pentium 4 cadencé à 2.1 GHz et 128 Mo de Ram sous windows XP.

Déterminer une séquence optimale de tâches quand les temps d'exécution sont des variables aléatoires de distribution arbitraires est difficile. Kalczynski et Kamburowski (2006) [05] ont présenté récemment une nouvelle règle d'ordonnancement qui inclut les deux règles de Johnson et Talwar comme des cas particuliers. Une façon de mesurer la performance de cette règle approchée consiste à faire une analyse expérimentale. Dans le chapitre cinq, un exemple est donné où un test a été effectué dans le cas d'une distribution de temps d'exécution des tâches est de Weibull.

Une conclusion générale commente les résultats où quelques perspectives de recherches sont proposées.

Les travaux de ce mémoire [06, 07, 08] ont fait l'objet d'un exposé dans un séminaire hebdomadaire de notre département, d'une acceptation d'un poster dans une conférence nationale au « Séminaire International De Mathématiques Appliquées et Simulations », 22-25 Avril 2007, Oum El Bouaghi, Algérie, et d'un article publié à CPI07, « 5^{ième} conférence Internationale sur la conception et la production intégrées », 22, 23 & 24 Octobre 2007, Rabat, Maroc.

CHAPITRE 1
LES ORDRES STOCHASTIQUES DANS L'ORDONNANCEMENT

L'approche des ordres stochastiques consiste à comparer des variables aléatoires, des vecteurs aléatoires ou même des processus aléatoires. Dans le cas où les temps d'exécution des tâches sont aléatoires, les longueurs des ordonnancements, fonctions des temps d'exécution le sont aussi. Des définitions, des notations et une formulation du problème sont fournies. Pour pouvoir comparer ces longueurs, des ordres stochastiques, l'outil de résolution des ordonnancements stochastiques, sont exposés.

1.1. Généralités sur les ordonnancements :

Le problème d'ordonnancement de tâches dans un atelier manufacturier consiste à programmer dans le temps, leurs exécutions en leur attribuant les ressources nécessaires, de manière à satisfaire un critère, tout en respectant les contraintes de réalisation. Pour une introduction plus détaillée nous renvoyons le lecteur à [09], [10], [11] et [12].

Une *tâche* J_i est un travail dont sa réalisation nécessite un certain nombre d'unités de temps (durée d'exécution) et d'unités de chaque ressource. Elle peut être constituée de plusieurs opérations élémentaires.

Une *gamme opératoire* est la succession de machines sur laquelle passe une tâche donnée. Une *ressource* est un moyen technique ou humain dont la disponibilité limitée ou non et connu a priori.

Un problème est dit *préemptif* si la préemption est autorisée, c'est à dire l'exécution d'une tâche peut être interrompue et reprise ultérieurement, avec ou *sans mémoire*. Une tâche est exécutée une seconde fois à partir du point où elle a été interrompue ou elle est reprise du début de son exécution.

Il est dit *statique* si la priorité des tâches est fixée et ne doit pas changer durant l'évolution du processus. Si les priorités des tâches changent en tout instant, il est *dynamique*. Les priorités dépendent de toute l'information disponible tels le temps courant, les durées d'attente des tâches pour exécution, les tâches en exécution sur les autres machines, les temps d'exécution que les tâches ont reçus sur ces machines etc.

Lorsque les données du problème sont connues de façon certaine, on parle de problème *" déterministe"*, par opposition à un problème *"stochastique"* où certaines données sont des variables aléatoires de distributions de probabilité connues.

Une affectation de tâches qui respecte les contraintes liées à l'environnement des machines et aux caractéristiques des tâches est dite *ordonnancement réalisable* ou *ordonnancement*. Si l'ordonnancement optimise le critère d'optimalité, il est dit *optimal*.

1.1.1 Les caractéristiques des machines

Les machines peuvent être *en parallèles*, faisant la même fonction (chaque tâche nécessite qu'une seule opération), ou *spécialisées* dans l'exécution de certaines opérations (une tâche est constituée de plusieurs opérations). Dans ce dernier cas, les machines sont disposées en général en série.

On distingue trois types de machines spécialisées, selon le mode de passage des opérations sur les différentes machines. Si les opérations élémentaires d'une tâche sont liées par un ordre identique pour toutes les tâches (gamme identique), on parle d'un problème du *flow shop*. Si elles sont liées par un ordre d'exécution qui n'est pas nécessairement le même pour toutes les tâches et dans ce cas une tâche peut repasser une seconde fois sur la même machine pour exécution, on parle d'un *Job shop.* Si elles ne sont pas liées par un ordre particulier, le problème est dit *Open shop.*

Un problème de flow shop est dit de *permutation* si la permutation de tâches à exécuter sur les machines reste inchangée.

En fonction de la vitesse d'exécutions on distingue trois types de machines parallèles: identiques, uniformes et quelconques. Pour plus de détail voir [12].

1.1.2. Les caractéristiques des tâches

A chaque tâche $J_i \in T$, on associe les paramètres suivants :

1. La *durée d'exécution* p_{ij} de la tâche J_i sur la machine M_j pour tout $i = 1,\ldots, n$, $j = 1,\ldots, m$. Dans le cas du flow shop, chaque tâche J_i est divisée en un nombre fini d'opérations élémentaires $O_{i1}, O_{i2},\ldots, O_{im}$, au nombre de machines.
2. La *date d'arrivée* ou *date de disponibilité* (*release date*) r_i de la tâche, la date où la tâche J_i est prête pour exécution. Si les dates d'arrivées sont supposées les mêmes pour toutes les tâches, on pose $r_i = 0$, pour tout $i = 1,..., n$.
3. La *date de fin d'exécution* C_i de la tâche qui est une variable à déterminer.

1.1.3. Les critères d'optimalité

Les critères d'optimalité les plus utilisés font intervenir la durée totale de l'ordonnancement, le délai d'exécution, les retards de l'ordonnancement, etc... Dans notre étude on s'intéresse à la durée totale d'ordonnancement, notée C_{max} qui est égale à la date de fin d'exécution de la dernière tâche noté $C_{max} = \max_i \{C_i\}$, c'est la *longueur d'ordonnancement* (de l'anglais *schedule length*) appelée aussi *makespan.*

1.1.4. Classification des problèmes d'ordonnancement

Il existe un grand nombre de problèmes d'ordonnancement, une notation et une classification s'imposent. En 1979, GRAHAM et al. [13] ont classifié ces problèmes en utilisant une notation à trois champs $\alpha/\beta/\gamma$. Le champ α décrit l'environnement machine, β donne les caractéristiques des tâches et des contraintes et γ correspond au critère à optimiser.

Exemple 1.1: F2 / / C_{max} signifie un ordonnancement de tâches sur 2-machines en flow shop, les tâches sont disponibles à l'instant 0, le mode d'exécution est non préemptif. L'objectif est de minimiser la longueur d'ordonnancement, soit le makespan.

1.1.5. Représentation d'un ordonnancement :

On représente souvent un ordonnancement par un *diagramme de GANTT*, un repère orthogonal dans un plan. L'axe des abscisses représente le temps et l'axe des ordonnées représente l'ensemble des machines. Pour chaque machine, on représente la séquence de tâches effectuées dans le temps. Un tel diagramme met en évidence l'occupation des machines par les différents travaux ainsi que le temps d'oisiveté. Dans la figure 1.1, les parties en gris représentent les périodes d'oisiveté d'une machine, dues aux éventuelles indisponibilités des tâches.

Exemple 1.2 : Trois tâches J_1, J_2, J_3 sont à exécutées sur deux machines flowshop. Les durées d'exécutions A_i et B_i de la tâche « J_i » respectivement sur la première et la seconde machine sont données dans le tableau suivant :

Tableau 1.1 : Durées d'exécutions

Tâches J_i	A_i	B_i
J_1	3	2
J_2	4	3
J_3	5	2

Une solution générée selon la liste L = (J_1, J_2, J_3) de longueur d'ordonnancement C_{max} de 14 unités de temps est représentée par le diagramme ci-dessous.

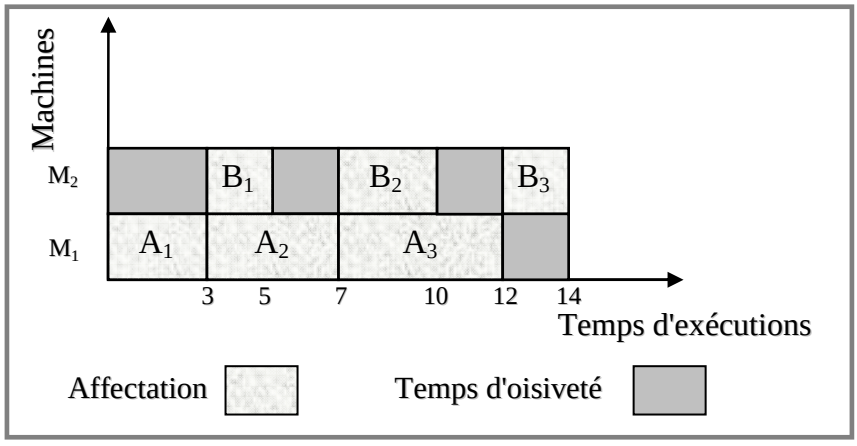

Figure1.1: Diagramme de Gantt

Cet ordre est optimal même s' il ne suit pas la règle de Johnson, ce qui explique que cette règle est seulement suffisante pour la minimisation de la longueur d'ordonnnacement.

Une autre représentation possible de la longueur d'une séquence de tâches dans un flow shop à 2-machines peut être faite par le temps de fin d'exécution du *réseau PERT* qu'on indiquera dans le chapitre trois.

Nous nous intéressons aux problèmes d'ordonnancement stochastique de type flow shop à 2-machines. L'objectif est de déterminer une séquence optimale qui minimise l'espérance du makespan par les ordres stochastiques. Cette approche repose sur la comparaison stochastique. On est emmené à comparer des variables aléatoires, des vecteurs aléatoires ou même des processus aléatoires. Une variable aléatoire peut être définit par sa densité, sa fonction de répartition, son taux d'hasards ou d'autres fonctions.

1.2. Notions de probabilité

Cette notion est indispensable à l'étude de la comparaison stochastique.
En théorie des probabilités, une variable aléatoire est une fonction mesurable définie sur un espace de probabilités. La mesure image correspondante est appelée loi de la variable aléatoire. Ce type de fonction permet de modéliser un phénomène aléatoire.
Soit $(\Omega, \mathcal{T}, P)$ *un espace de probabilité.*

Une *variable aléatoire* réelle *X* dite également variable stochastique, définit sur $(\Omega, \mathcal{T}, P)$ est une application de Ω dans $\mathbb{R}$ tel que:

$\forall I \subset \mathbb{R},\ X^{-1}(I) = \{\omega \in \Omega / X(\omega) \in I\} \in \mathcal{T}$..

Elle est dite discrète si X (Ω) est un ensemble fini ou dénombrable. Dans le cas continu, X (Ω) est un intervalle de $\mathbb{R}$.
Une variable aléatoire X est une fonction dont les valeurs ou les intervalles de valeurs sont associés à des probabilités. Pour la définir, il faut spécifier l'ensemble des valeurs possibles, appelé domaine des événements et sa distribution de probabilité. L'ensemble des valeurs possibles peut être discret, ou continu sur un intervalle. X représente la variable aléatoire, et ω sa réalisation.

La *loi de probabilité* d'une variable aléatoire discrète *X* est donnée par une fonction *f* non négative vérifiant :i) $f(x) \geq 0$, pour tout $x \in \mathbb{R}$.

ii) $\{x, f(x) \neq 0\}$ est un ensemble fini ou dénombrable.

iii) $\sum_{x \in X} f(x) = 1$.

La fonction *f* caractérise la loi de probabilité ou encore sa densité.

On associe naturellement une distribution à une variable aléatoire pour décrire la répartition des valeurs qu'elle peut prendre. Une *fonction de répartition* F_X d'une variable aléatoire X définit de Ω à valeur dans [0,1] est donnée en cas discret par $F_X(t) = P(X \leq t)$, et se note $F_X(t) = \int_{-\infty}^{t} f(x)dx$ en cas continu.

$\overline{F_X}$ représente l'expression $\overline{F_X}(t) = 1 - F_X(t)$.

L'espérance mathématique d'une variable aléatoire X est définie par $E(X) = \sum_x xP(X = x)$, et $E(X) = \int_{-\infty}^{+\infty} xf(x)dx$ respectivement dans le cas discret et continu.

Définition 1.1 [14]. Le **«** *support d'une variable aléatoire* » est l'ensemble des réalisations qui a une densité strictement positive. Il se note par

$$Supp(X) = \{x, f(x) > 0\}.$$

1.3. Les ordres stochastiques

Nous rappelons qu'un ordre partiel sur un ensemble E est une relation binaire notée ″< " réflexive, antisymétrique et transitive.

Soit E un ensemble pré-ordonné par une relation $\mathcal{P}$, qui est réflexive et transitive.

Proposition 1.1 : On sait construire une relation d'ordre O sur l'ensemble quotient $E/\mathcal{R}$ modulo une relation d'équivalence $\mathcal{R}$ à partir d'un pré-ordre $\mathcal{P}$ définit sur un ensemble E.

En ordonnancement stochastique, l'espace des événements est discret et dénombrable. Il doit être muni d'une relation d'ordre ayant au moins les propriétés d'un préordre (réflexif, transitif), notée $\prec \preceq \prec$. On définit des formes de dominance stochastique parmi les opérations élémentaires à qui on leurs associe le temps de traitement aléatoire sur la machine correspondante, afin de minimiser la longueur suivant cet ordre.
Dans la suite de ce chapitre, *X, Y* représentent des variables aléatoires de fonctions de répartition respectivement F_1 et F_2 et de densités f_1 et f_2.

On appelle « ordre stochastique » un ordre partiel définit sur un espace de probabilité. Pour comparer deux variables aléatoires réelles, l'ordre stochastique le plus naturel est la comparaison de leurs distributions.

Définition 1.2 [14]: Une variable aléatoire *X* est dite *stochastiquement* plus grande que la variable aléatoire *Y*, et que l'on note $X \geq_{st} Y$ si

$$P(X > t) \geq P(Y > t), \ \forall t \in \mathbb{R}$$

Cette inégalité est équivalente à $F_1(t) \leq F_2(t), \ \forall t \in \mathbb{R}$.

Donnons quelques propriétés sur les ordres stochastiques.

Proposition 1.2: si $X \geq_{st} Y$ *alors* $(-Y) \geq_{st} (-X)$

D'une façon élémentaire, des notions sur les ordres « presque sûrement », des ordres en « espérance » et en « variance » peuvent être définies.

Définition 1.3 [14]: i) Une variable aléatoire *X* est dite supérieure *presque sûrement* à une autre variable aléatoire *Y* et se note $X \geq_{as} Y$ (almost surely) si $P(X \geq Y) = 1$.
ii) Une variable aléatoire *X* est dite plus grande en *espérance* que la variable aléatoire *Y* si $E(X) \geq E(Y)$.
iii) Une variable aléatoire *X* est dite plus grande en *variance* que la variable aléatoire *Y* si $Var(X) \geq Var(Y)$.

En utilisant des notions de temps restant à l'exécution d'une tâche à l'instant t définit par une variable aléatoire $X_t = [X-t / X > t]$, de distribution F_t, on peut définir d'autres ordres stochastiques.

Définition 1.4 [14]: Deux variables aléatoires X et Y sont dites ordonnées par *ordre croissant du facteur d'utilisation* dit aussi *taux de hasard* (Increasing failure rate ordered), que l'on note par $X \geq_{hr} Y$ si $\overline{F}_1(t) / \overline{F}_2(t)$ croit pour tout t.
De manière équivalente, pour tout t, $X_t \geq_{st} Y_t$

En ordonnancement, une variable aléatoire X peut représenter le temps d'exécution d'une tâche donnée. Notons par $X_{t,}$ le temps restant à l'exécution d'une tâche à l'instant ″ t ″ sachant qu'elle ne termine son exécution qu'après l'instant t. A partir de ces notions, on peut définir et même caractériser certaines lois. Par un exemple, la variable exponentielle est l'unique loi sans mémoire. $F_{t,}$ la distribution associée à la variable X_t ne dépend pas de t.
En effet, soit X_t une variable aléatoire exponentielle de paramètre λ.

$$P(X_t > s) = P(X - t > s / X > t) = \frac{P(X > t+s)}{P(X > t)} = e^{-\lambda s} = P(X > s).$$

Définition 1.5 [14]: Deux variables aléatoires continues X et Y sont dites ordonnées selon le *rapport de vraisemblance croissant monotone* (Increasing monotone likelihood ratio ordered), que l'on note par $X \geq_{lr} Y$ si $f_1(t) / f_2(t)$ croit pour tout t.

Les notions suivantes sont généralisables au cas de variables à valeurs dans un espace muni d'un ordre partiel.

Théorème 1.1 [14]**:** Soient X, Y, Θ trois variables aléatoires.
i) Soient $\{X_i\}$, $\{Y_i\}$ deux séquences de variables aléatoires indépendantes.
Si $X_i \geq_{st} Y_i$, pour tout $i = 1, \ldots, n$ alors pour toute fonction croissante f de $\mathbb{R}^n$ *dans* $\mathbb{R}$, $f(X_1, \ldots, X_n) \geq_{st} f(Y_1, \ldots, Y_n)$.
ii) $[X / \Theta = \theta] \geq_{st} [Y / \Theta = \theta]$, pour tout θ du support de Θ, alors $X \geq_{st} Y$.

Le même résultat s'énonce dans le cas de i) pour les ordres $\geq_{hr}$ et $\geq_{lr}$ lorsque f est une fonction croissante.
Ce théorème verra son application dans le chapitre suivant.

Pour l'étude des systèmes d'ordonnancement, les ordres $\geq_{st}$ et $\geq_{hr}$ s'imposent naturellement car bien souvent les variables définissant la performance de ces systèmes peuvent être construites par une succession d'opérations menant à des fonctions croissantes.

1.4. Propriétés des ordres stochastiques :

Ce paragraphe est consacré à la relation qui existe entre les différents ordres dont nous rappelons avec démonstration les principales propriétés.
X et Y représentent des variables aléatoires de densité f et g, de fonctions de répartition F et G et de taux de hasard τ_1 et τ_2 respectivement.

Lemme1.1 [15]:

Si $X \geq_{st} Y$ alors $E(X) \geq E(Y)$.

Preuve: On suppose que X et Y sont deux variables aléatoire discrètes non négatives alors:

$$E(X) = \sum_0^\infty xP(X = x) = \sum_0^\infty P(X > x) \geq \sum_0^\infty P(Y > x) = E(Y).$$

Une variable aléatoire Z quelconque peut s'écrire comme la différence de deux variables aléatoires non négatives : $Z = Z^+ - Z^-$

$$\text{Où } Z^+ = \begin{cases} Z \text{ si } Z \geq 0 \\ 0 \quad \text{sinon} \end{cases} \quad \text{et} \quad Z^- = \begin{cases} -Z \text{ si } Z < 0 \\ 0 \quad \text{sinon} \end{cases}$$

Si $X \geq_{st} Y$ alors, $X^+ \geq_{st} Y^+$ et $X^- \leq_{st} Y^-$.

En effet, $P(X^+ > t) = P(+X > t) \geq P(+Y > t) = P(Y^+ > t)$

$P(X^- > t) = P(-X > t) = P(X < -t) = 1 - P(X \geq -t) \leq 1 - P(Y \geq -t) = P(Y < -t)$

$= P(-Y > t) = P(Y^- > t)$.

Par conséquent: $E(X) = E(X^+) - E(X^-) \geq E(Y^+) - E(Y^-) = E(Y)$.

Ce résultat est utile, si deux longueurs d'ordonnancement sont minimisées stochastiquement alors elles le seront en espérance mathématique.

Proposition 1.3 [15]**:**

Le facteur d'utilisation ou taux de hasard est définit par $\tau_i(t) = f_i(t)/\overline{F_i}(t)$ pour tout i =1 et 2.

i) X et Y sont dits ordonnées par ordre croissant du facteur d'utilisation si et seulement si $\forall t \in \mathbb{R}$, $\tau_1(t) \leq \tau_2(t)$.

ii) X et Y sont dites ordonnées selon le rapport de vraisemblance croissant monotone si $\forall t \in$ R, $\tau_1(t) \leq \tau_2(t)$.

Preuve: *i)* Si $X \geq_{hr} Y$, alors $\overline{F}(t)/\overline{G}(t) \geq \overline{F}(x)/\overline{G}(x)$ (1), pour tout $x \leq t$.

$\tau_1(t) = f(t)/\overline{F}(t) = (-\overline{F}(t))'/\overline{F}(t)$, en substituant $\overline{F}(t)$ de l'inégalité (1) et en dérivant par rapport à t, on trouve : $\tau_1(t) = (-\overline{F}(t))'/\overline{F}(t) \leq \overline{F}(x)(-\overline{G}(t))'/\overline{G}(x)\overline{F}(t)$

En utilisant l'inégalité (1) on en déduit que: $\tau_1(t) \leq g(t)/\overline{G}(t) = \tau_2(t)$.

Notons que (F(t)) ' = f (t) .

ii) Si $X \geq_{lr} Y$, alors $f(x) \geq g(x) f(t)/g(t)$, pour tout $x \geq t$.

$$\tau_1(t) = f(t)/\overline{F}(t) = f(t)/\int_t^{+\infty} f(x)\,dx \leq f(t)/\int_t^{+\infty} \frac{f(t)g(x)}{g(t)}\,dx = g(t)/\int_t^{+\infty} g(x) = g(t)/\overline{G}(t) = \tau_2(t)$$

Proposition 1.3 [15]: i) $X \geq_{hr} Y$ si et seulement si $X \geq_{st} Y$.

ii) Si $X \geq_{lr} Y$ alors $X \geq_{hr} Y$.

Preuve: i) si $X \geq_{hr} Y$ alors leur taux de hasard sont inversement ordonnés par l'inégalité, $\tau_1 \leq \tau_2$ pour tout *t,* qui est équivalent à ce que $\overline{F}$ (t) $\geq \overline{G}$ (t). D'où $X \geq_{st} Y$.

ii) Si $X \geq_{lr} Y$ alors en utilisant la définition de l'ordre au sens du rapport de vraisemblance, le rapport $f(t)/g(t)$ croit pour tout t, par conséquent $\overline{F}$ (t)/$\overline{G}$ (t) l'est aussi. D'où $X \geq_{hr} Y$.

Des relations de dépendances peuvent être représentées par le diagramme ci-dessous.

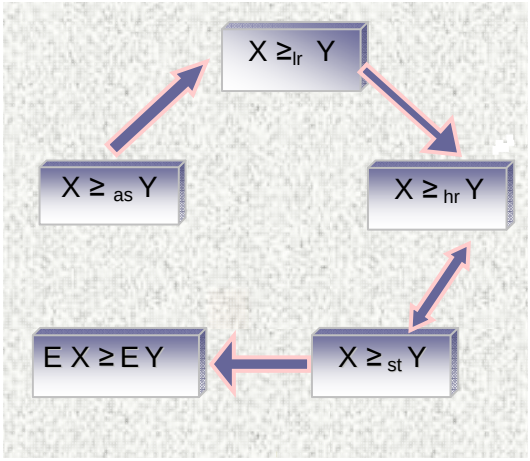

Figure 1.2 : Relations en chaîne des ordres stochastiques.

Nous avons répertorié les principales définitions et propriétés sur les ordres stochastiques qui seront utilisés ultérieurement. Pour plus de détails, Le lecteur consultera les ouvrages de Barlow [14], Marshall [15], Ross [16], Shaked et Shantikumar [17] et Oukid [18].

CHAPITRE 2
LE FLOWSHOP STOCHASTIQUE
A 2-MACHINES

Travaux précurseurs et continuateurs

" N " tâches sont à exécuter sur 2-machines disposées en ligne. Les temps d'exécution des tâches sont supposés aléatoires de lois connues. A l'entrée de l'atelier, les pièces sont disponibles pour être charger sur la première machine. Chaque fois une pièce est traitée, un chariot qui n'est pas considéré comme une machine, la transporte et la déplace pour la mettre sur la seconde machine ou dans la zone tampon qui de nouveau sera exécutée. Une fois son exécution est terminée sur la seconde machine, la dite pièce est acheminée vers la sortie de l'atelier. La situation est représentée par la figure ci-dessous.

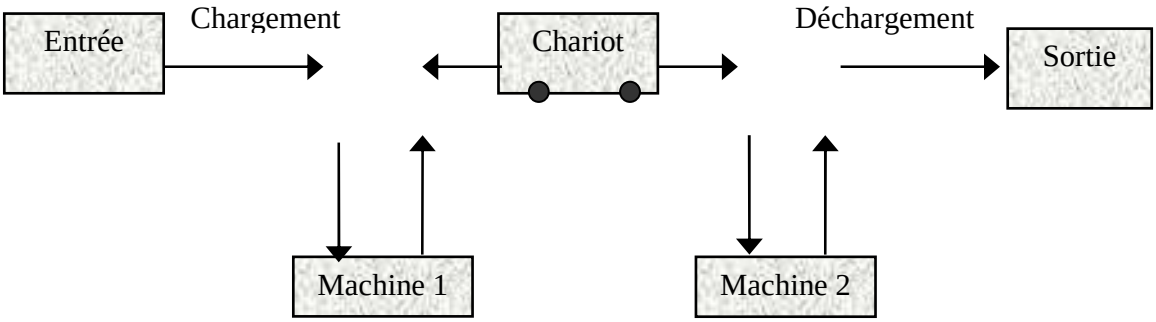

Figure 2.1: Actions de charger et décharger dans un atelier à deux machines en série.

Nous supposons que le temps de transport des pièces par le chariot est négligeable. Il est inclus dans le temps d'exécution des pièces. L'objectif est de déterminer une séquence optimale de tâches qui minimise l'espérance de la longueur d'ordonnancement.

Dans le cas déterministe, le problème d'ordonnancement d'un nombre de tâches sur un ensemble de machines en série a fait l'objet d'un grand nombre de publications d'articles et de considérations de plusieurs auteurs [09], [19], [12], etc…
Dans des problèmes réels, le temps d'exécution d'une tâche est en général incertain. Ses fluctuations et variations stochastiques ont fait l'objet de notre étude.

2.1. Flow shop stochastique à 2-machines avec une capacité de stockage illimitée

Une aire de stockage existe et est supposée de capacité infinie entre les deux machines. Quand une tâche a terminé son exécution sur la première machine, elle est stockée dans une aire appelée aussi zone tampon avant la seconde machine, lorsque celle ci est occupée dans l'exécution d'une tâche. Si la seconde machine est libre, la pièce est directement mise sur celle-ci pour exécution.

2.1.1. Etat de la connaissance et des recherches

Nous supposons que les temps d'exécution des tâches sur les deux machines sont identiques, indépendants et distribués exponentiellement sauf indications. Une variable aléatoire se notera dans la suite par l'abréviation v.a. De notre recherche bibliographique, nous reprenons, dans leur grande ligne et leurs principales originalités, les travaux qui, selon nous, marquent des jalons importants dans la résolution du flowshop stochastique à 2- machines.

a. Les précurseurs

- *Johnson* [01] a résolu le problème déterministe à 2-machines.

- *Banerjee*[20] est peut être un des premiers auteurs qui se sont intéressés au cas stochastique du flowshop. Il a résolu le problème d'ordonnancement à une seule machine en utilisant le critère de la minimisation de la probabilité maximale du retard de n'importe quelle tâche.

- *Makino* [21] a considéré le problème de 2-tâches sur 2-machines, 2-tâches sur 3-machines dont l'objectif est de minimiser l'espérance du makespan.

- Après avoir résolu le problème de 3-tâches sur 2-machines, *Talwar* [02] a considéré le problème de n tâches sur 2-machines et conjectura que si les temps d'exécution sont des v.a indépendantes alors la règle optimale est qu'une tâche i doit précéder une tâche j si et seulement si $1/E(A_i) - 1/E(B_i) \geq 1/E(A_j) - 1/E(B_j)$. Cette règle donne un ordonnancement avec une espérance du makespan minimale.

- *Bagga* [22] a résolu le cas de 4-tâches sur 2-machines. Il a étendu l'étude au cas de 2-tâches sur 3-machines, si les temps d'exécution de tâches sont de distribution quelconque et a donné une preuve incomplète de la conjecture de Talwar.

- Dans *Cunningham et Dutta* [03], une règle d'ordonnancement optimale est fournie, antérieurement conjecturée par Talwar [02]. La preuve complète de cette conjecture est aussi faite. Les règles de Johnson [01] et Talwar [02] établissent qu'une tâche i précède une tâche j si et seulement si $E(\min(A_i, B_j)) \leq E(\min(A_j, B_j))$. Une formule déduite d'une analyse des chaînes de markov, qui évalue l'espérance du temps de fin d'exécution d'une suite quelconque de tâches est établie.

- *Garey* et al [23] ont étudié la complexité des problèmes de type flow shop. La détermination d'un ordonnancement de longueur minimale dans un flow shop à m-machines ($m \geq 3$) est NP-complet. Il se formule comme un problème de 3-partition. Le cas stochastique l'est aussi. La détermination d'un flow time pondéré minimale dans un flow shop à m-machines est NP-complet pour $m \geq 2$.

Les auteurs *Levner* [24], *Weber* [25] et *Muth* [26] ont été très actifs par leurs publications.

b. Les continuateurs

L'étude analytique de nombreux modèles d'ordonnancement est très complexe et les résultats obtenus sont difficilement interprétables. Dans ce cas, on tente d'éluder cette difficulté par l'obtention d'approximation ou d'encadrement (majorations ou minorations) des principales mesure de performance pour la longueur d'ordonnancement. De notre recherche bibliographique, plus de dix articles sont en notre possession et on fait l'objet d'une étude où les résultats sont énoncés et démontrés rigoureusement. Nous en sélectionnerons quelques uns que nous considérons comme illustratif des « ordres stochastiques ».

● *Pinedo* [27] utilise la dualité entre la file d'attente et le flow shop, il exhibe des résultats dans le cas où les temps d'exécutions des tâches sont identiquement distribués de moyenne connue, il indique la position de chaque tâche dans la séquence.

On définit X_{ij}, le temps d'exécution de la tâche i sur la machine j, de distribution F_i qui sont supposés indépendants identiquement distribués. Le problème est de déterminer une séquence de « n » tâches qui minimise l'espérance de la longueur d'ordonnancement.
Les notations suivantes sont utilisées dans la preuve du lemme et du théorème qui le succède.

On note par $E(X_i)$, espérance de la v.a X_i de distribution F_i.
Sous une séquence de tâches (J_1, J_2,..., J_n) on suppose que :

$$E(X_k) = \max_{i=1,\ldots,n}(E(X_i)), \quad S_1 = \sum_{i=1}^{k-1} E(X_i) \text{ et } \quad S_2 = \sum_{i=k+1}^{n} E(X_i)$$

Une borne inférieure de l'espérance du makespan est obtenue.

Lemme 2.1 [27]: Sous n'importe qu'elle séquence de tâches (J_1, J_2,..., J_n), on a

$$E(C\max) \geq \sum_{i=1}^{n} E(X_i) + E(X_k).$$

Preuve : L'espérance du temps nécessaire pour que la tâche « k » termine son exécution sur la seconde machine est égal à au moins 2 $E(X_k)$. Son espérance pour

commencer son exécution sur la première machine est égal à S_1, et l'espérance de temps d'exécutions des tâches qui succèdent sur la seconde machine après la tâche « k » est égale à au moins S_2. D'où le résultat.

Rappelons qu'une v.a X_1 est dite supérieure presque sûrement à une v.a X_2 et se note $X_1 \geq_{as} X_2$ (almost surely) si $P(X_1 \geq X_2) = 1$
Par la notion des ordres presque sûrement, on peut définir celle de non chevauchement.

Définition 2.2 [27] : Si $X_1 \leq_{as} X_2 \leq_{as} \ldots \leq_{as} X_n$ alors les temps d'exécutions sont dits non chevauchants (de l'anglais nonoverlapping).

Cet ordre est utilisé par Tembe et Wolff [28] pour déterminer un ordre optimal dans une station en série. Ils trouvent que quand les temps de services des clients sont non chevauchant alors l'instant du départ du dernier client pour vider le système est minimisé stochastiquement, si la séquence est ordonnancée du plus grand au plus petit temps de service. En appliquant ces résultats pour le flow shop on trouve que le makespan est stochastiquement minimisé si les tâches sont ordonnés par ordre décroissant en espérance du temps d'exécution (LEPT). Du fait que la distribution du makespan ne change pas lorsque la séquence de tâche est inversée [26], alors l'ordre plus petit en espérance du temps d'exécution (SEPT) est aussi optimale stochastiquement.

Définition 2.3 : Une suite de tâches (J_1 , ..., J_n) est appelée séquence SEPT- LEPT s'il y a une tâche J_k dans la séquence tel que

$$E(X_{J_1}) < E(X_{J_2}) < \ldots < E(X_{J_k})$$

$$E(X_{J_k}) > E(X_{J_{k+1}}) > \ldots > E(X_{J_n})$$

Les politiques SEPT et LEPT sont aussi SEPT-LEPT.

Sous la condition de non chevauchement, la borne inférieure est atteinte pour la minimisation de l'espérance du makespan.

Théorème 2.1 [27] **:** Si les temps d'exécution sont non chevauchant alors n'importe qu'elle séquence SEPT-LEPT minimise l'espérance du makespan de valeur

$$E(C_{\max}) = \sum_{i=1}^{n} \mathrm{E}(\mathrm{X_i}) + \mathrm{E}(\mathrm{X_k})$$

Preuve : Les tâches successeurs et prédécesseurs de la tâche « k » de plus grande espérance, sont respectivement exécutées par ordre croissant et décroissant de leurs espérances. Par conséquent, l'espérance du temps écoulé depuis le début de la première tâche sur la première machine jusqu'à la fin d'exécution de la dernière tâche sur la seconde machine est égale à exactement $S_1 + 2E(X_k) + S_2$. D'où le résultat.

● *Ku et Niu* [29] donnent une condition suffisante sur les distributions de temps d'exécution pour que le makespan devient stochastiquement plus petit quand deux tâches adjacentes dans une suite donnée sont transposées. Cette condition contient les deux règles de Johnson (1954) et de Talwar (1967) comme un cas particulier.

On étudie l'effet de transposer dans un ordonnancement de permutation deux tâches adjacentes.

Soient $S_1 : J_1, J_2,..., J_k, J_{k+1}, ..., J_n$ et $S_2 : J_1, J_2, ..., J_{k+1}, J_k, ... , J_n$

T : le temps requis pour que les tâches $J_1 J_2 ....J_{k-1}$ terminent leur exécution sur la machine1

$$T = \sum_{i=1}^{k-1} A_i$$

L : le temps nécessaire après T pour la machine 2 pour terminer l'exécution de $J_1 , J_2, ...,J_{k-1}$

A : Le temps total pour exécuter les tâches k et $k+1$ sur la machine 1.

$$A = A_k + A_{k+1}.$$

$R_1 (R_2)$: Temps additionnel nécessaire pour vider le système après $T + A$ sous $S_1 (S_2)$ et

$\delta_1 (\delta_2)$: Temps additionnel après $t + A$ nécessaire pour la machine 2 pour terminer l'exécution des tâches $J_1 , J_2 ,...,J_k , J_{k+1}$ sous la séquence $S_1 (S_2)$.

Les makespan M_1 et M_2 sous les séquences S_1 et S_2 sont respectivement

$$M_i = T + A + R_i, \quad i = 1, 2. \qquad (2.1)$$

La figure 2.2 illustre la représentation de la séquence des tâches S_i

Notons que les v.a T et A sont indépendantes l'une de l'autre, R_i dépend de T et A, pour $i = 1$ et 2.

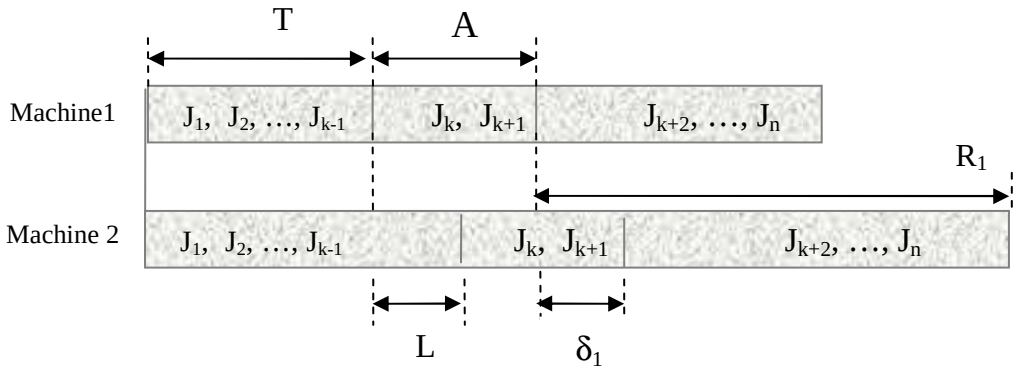

Figure 2.2. Représentation de la séquence S_1.

Des résultats sont établis et sont donnés sous forme de lemmes et de théorèmes. Deux lemmes sont indispensables pour justifier et démontrer le théoréme 2.2.

Lemme 2.2 [29] **:** Il existe une fonction r croissante définie de l'ensemble R^+ tel que $R_i = r(\delta_i)$, pour tout $i = 1$ et 2.

En effet, la minimisation du temps additionnel δ suite à une permutation de deux tâches consécutives diminue le temps additionnel R nécessaire pour vider le système après $T + A$.

Rappelons que B_i représente le temps d'exécution de la tâche « i » sur la seconde machine.

Lemme 2.3 [29] **:** Deux formules de caractérisation des temps additionnel sont établies.

$\delta_1 = B_k + B_{k+1} - Min (A_{k+1} , B_k , A - L).$

$\delta_2 = B_k + B_{k+1} - Min (A_k , B_{k+1} , A - L).$

Preuve: 1) Considérons la séquence S_1

Soient I_k et I_{k+1} les temps d'oisiveté sur la seconde machine avant l'exécution de la tâche J_k, et J_{k+1} respectivement. Cette durée est représentée en gris dans les deux figures suivantes.

La figure 2.3.illustre la présentation du Diagramme de Gantt sous la séquence S_1.

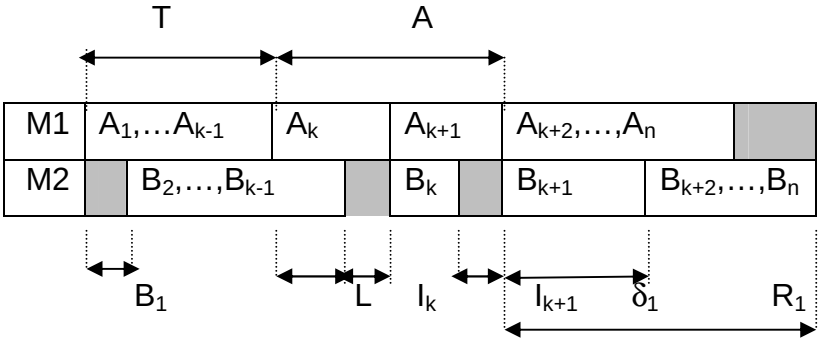

Figure 2.3. Diagramme de Gantt sous la séquence S_1

Soit $I_k = max (A_k – L , 0)$

$I_{k+1} = max (A_k + A_{k+1} - L - I_k - B_k , 0)$

De la figure 2.3. On peut remarquer que $\delta_1 = L+I_k + B_k + I_{k+1} + B_{k+1} - A$ (2.2)

en substituant la formule de I_{k+1} dans inégalité (2.2), on obtient :

$\delta_1 = L + B_k + B_{k+1} – A + max (A_k + A_{k+1} - L – B_k , I_k)$ (2.3) ,en remplaçant la formule de I_k dans celle de (2.3),on trouve

$$\delta_1 = B_k + B_{k+1} + max(- B_k , - A_{k+1}, L - A)$$

Quand on applique que $max (X, Y)= - min (-X, -Y),$ on en déduit que :

$\delta_1 = B_k + B_{k+1} + min (B_k , A_{k+1}, A –L)$

2) Considérons la séquence S_2

La figure 2.3 illustre la présentation du Diagramme de Gantt sous la séquence S_2

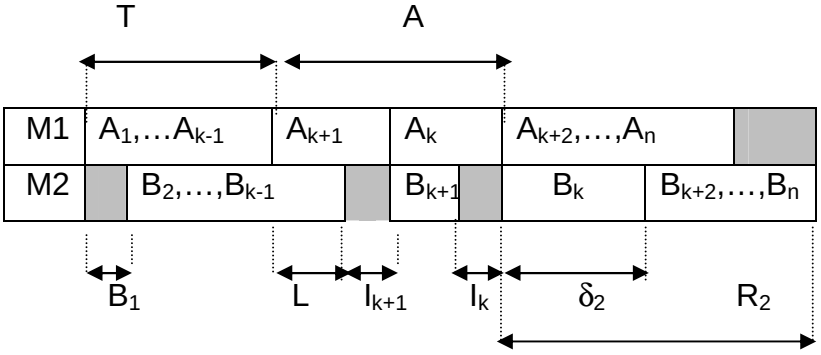

Figure 2.4. Diagramme de Gantt sous la séquence S_2.

Soit $I_k = max (A_k + A_{k+1} - L - I_{k+1} - B_{k+1} , 0)$

$I_{k+1} = max (A_{k+1} - L, 0).$

$\delta_2 = L + I_{k+1} + B_{k+1} + I_k + B_k - A$ (2.4), en substituant la formule de I_k dans la formule (2.4) , on obtient la formule suivante:

$\delta_2 = L - A + B_k + B_{k+1} + max (A_k + A_{k+1} - L - B_{k+1} , I_{k+1})$, on remplace I_{k+1} dans cette dernière on obtient, $\delta_2 = B_k + B_{k+1} + max (- B_{k+1,} - A_k, L - A)$.

Comme $max (X, Y) = - min (-X, -Y)$, on en déduit que :

$\delta_2 = B_k + B_{k+1} + min (B_{k+1,} , A_k, A - L)$

Pour tout i =1 et 2, on note par (X / .), une variable aléatoire conditionnelle.

Théorème 2. 2 [29] : Une condition suffisante pour que $M_1 \leq_{st} M_2$ est que :

$[min (A_k , B_{k+1}) / A_k + A_{k+1} = a \text{ et } B_k + B_{k+1} = b]$

$\leq_{st} [min (A_{k+1} , B_k) / A_k + A_{k+1} = a \text{ et } B_k + B_{k+1} = b]$ (2.5)

Cette dominance doit être vérifiée pour tout a, b≥ 0 et où les distributions des variables aléatoires conditionnelles soient définis.

Preuve : Il suffit de montrer la dominance stochastique suivante :

$(M_1 / L = l, A_k + A_{k+1} = a, B_k + B_{k+1} = b) \leq_{st} (M_2 / L = l, A_k + A_{k+1} = a, B_k + B_{k+1} = b)$

pour tout l, a, b ≥ 0.

A partir de la formule (2.1), on a pour tout i = 1 et 2

$(M_1 / L=l, A_k + A_{k+1}=a, B_k + B_{k+1}=b) = (T / L=l) + a + (r (\delta_i) / L=l, A_k+A_{k+1}=a, B_k+B_{k+1}=b)$

Du lemme 2.2 et du fait que l'ordre stochastique est préservé pour toute fonction croissante, il est suffisant de montrer que:

$(\delta_1 / L = l, A_k + A_{k+1} = a, B_k + B_{k+1} = b) \leq_{st} (\delta_2 / L = l, A_k + A_{k+1} = a, B_k + B_{k+1} = b)$.

Rappelons que $[X / \Theta = \theta] \geq_{st} [Y / \Theta = \theta]$, pour tout θ du support de Θ, alors $X \geq_{st} Y$ .En effet si $[min (A_k , B_{k+1}) / A_k + A_{k+1} = a \text{ et } B_k + B_{k+1} = b] \leq_{st}$

$[min (A_{k+1} , B_k) / A_k + A_{k+1} = a \text{ et } B_k + B_{k+1} = b]$, pour tout a ,b dans le support de $A_k+A_{k+1} , B_k+B_{k+1}$ respectivement alors

$min(A_k , B_{k+1}) \leq_{st} min(A_{k+1}, B_k)$ (2.6).

Soit A- L, une v.a indépendante des variables A_k et B_{k+1} (A_{k+1} et B_k) sous la séquence S_1 (S_2) alors, il est simple de montrer que la formule (2.6) implique que

$$\min (A_k , B_{k+1}, A - L) \leq_{st} (\min (A_{k+1} , B_k , A - L) \qquad (2.7)$$

Du fait que $X \leq_{st} Y$ implique que $- Y \leq_{st} - X$ alors la formule (2.7) implique que

$$- \min (A_{k+1} , B_k , A- L) \leq_{st} - \min (A_k , B_{k+1}, A- L).$$

Comme $X \leq_{st} Y$ entraîne $X+Z \leq_{st} Y +Z$, pour tout Z une v.a indépendante respectivement de X et Y, alors :

$$B_k + B_{k+1} - \min (B_k , A_{k+1}, A -L) \leq_{st} B_k + B_{k+1} - \min (B_{k+1}, A_k, A -L).$$

D'où $\delta_1 \leq_{st} \delta_2$.

La proposition suivante contient la règle de Johnson [01] comme un cas particulier.

Proposition 2. 1 [29]: Si l'une des deux conditions suivantes est satisfaite :

i) $(A_k / A_k + A_{k+1} = a) \leq_{st} (A_{k+1} / A_k + A_{k+1} = a)$ et

$(B_k / B_k + B_{k+1} = b) \geq_{st} (B_{k+1} / B_k + B_{k+1} = b), \forall (a , b) \geq 0$.

ii) $P[\min (A_k , B_{k+1}) \leq \min (A_{k+1} , B_k)] = 1$.

alors la condition (2.5) est vérifiée**.**

Preuve: i) Notons que lorsque les temps d'exécution sont indépendants, la probabilité conditionnelle peut s'écrire sous la forme suivante:

$P(A_k > t / A_k + A_{k+1} = a) = P(A_k > t / A_k + A_{k+1} = a, B_k + B_{k+1} = b)$. Et du fait que $P(\min(A_k , B_{k+1}) > t / A_k + A_{k+1} = a$ et $B_k + B_{k+1} = b)$

$= P (A_k > t, B_{k+1} > t / A_k + A_{k+1} = a$ et $B_k + B_{k+1} = b).$ On en déduit le résultat.

La proposition suivante montre que la règle de Talwar [02] minimise le makespan stochastiquement.

Proposition 2.2 [29]: Quand les temps d'exécutions sont exponentiellement distribués de moyenne $E(A_i)$ et $E(B_i)$ respectivement, pour $i = 1, 2,.., n$ alors, la condition (2.5) est vérifiée si on a la condition de Talwar qui est définie par : $1/E(A_k) - 1/E(B_k) \geq 1/E(A_{k+1}) - 1/E(B_{k+1})$ **.**

Preuve : En se basant sur la formule de probabilité conditionnelle, et du fait que A_k et B_{k+1} respectivement A_{k+1} et B_k sont des variables aléatoires indépendantes alors:

$P(min(A_k , B_{k+1}) > t \ / A_k + A_{k+1} = a \ et \ B_k + B_{k+1} = b)$

$= P (A_k > t , B_{k+1} > t / A_k + A_{k+1} = a , B_k + B_{k+1} = b)$

$$= \frac{P(A_k > t, B_{k+1} > t, A_k + A_{k+1} = a, B_k + B_{k+1} = b)}{\mathrm{P}(\mathrm{A_k} + A_{k+1} = a, B_k + B_{k+1} = b)}$$

Par conséquent, il est élémentaire de vérifier que la condition (2.5) est équivalente à : $P (A_k > t , B_{k+1} > t, A_k + A_{k+1} = a , B_k + B_{k+1} = b)$

$\leq P (A_{k+1} > t, B_k > t ,A_k + A_{k+1} = a , B_k + B_{k+1} = b)$

Rappelons que *p(.)* désigne la probabilité.

Pour tout $0 \leq t \leq min [a, b]$ et $a, b \geq 0$, on a l'égalité suivante :

$P (A_k > t, B_{k+1} > t, A_k + A_{k+1} = a, B_k + B_{k+1} = b)$

$= P (A_k > t, A_k + A_{k+1} = a) P (B_{k+1} > t, B_k + B_{k+1} = b)$

$= P (A_k + A_{k+1} = a / A_k > t) P(A_k > t) P(B_k + B_{k+1} = b / B_{k+1} > t) P(B_{k+1} > t)$ (2.8)

En utilisant la particularité de la loi exponentielle par la propriété de « *sans mémoire* ». Rappelons que $P (X + Y = a \ / X > t) = P (X + Y = a - t + t \ / \ X > t)$

$= P (X + Y = a- t)$, *alors*

La formule (2.8) peut s'écrire :

$$\exp\left\{-(\frac{1}{E(A_k)} + \frac{1}{E(B_{k+1})})t\right\} P(A_k + A_{k+1} = a / A_k > t) P(B_k + B_{k+1} = b / B_{k+1} > t)$$

De l'hypothèse on peut en déduire que cette dernière est inférieure à la formule suivante :

$$\exp\left\{-(\frac{1}{E(A_{k+1})} + \frac{1}{E(B_k)})t\right\} P(A_k + A_{k+1} = a / A_{k+1} > t) P(B_k + B_{k+1} = b / B_k > t) .$$

Ce qui termine la preuve.

Remarque 2.1 : La condition ii) de la proposition (2.1) est une extension de la règle de Johnson au cas stochastique et la proposition (2.2) montre que la règle de Talwar minimise le makespan stochastiquement. La condition (2.5) explique la relation entre la règle de Johnson et celle de Talwar.

Rappelons qu'une v.a X de densité f_X est dite « *plus petite dans le sens du rapport de vraisemblance* » qu'une v.a Y de densité f_Y et on note $X \leq_{lr} Y$ si $\frac{f_Y(x)}{f_X(x)} \leq \frac{f_Y(y)}{f_X(y)}$, pour tout $0 \leq x \leq y$.

Proposition 2.3 [29]: Soient X et Y deux v.a indépendantes, positives de densités f_X et f_Y si $X \leq_{lr} Y$ alors $(X / X + Y = s) \leq_{lr} (Y / X + Y = s), \forall s \geq 0$.

Preuve : En utilisant la définition de l'ordre au sens du rapport de vraisemblance, il suffit de montrer que

$\frac{f_{Y/X+Y=s}(x)}{f_{X/X+Y=s}(x)} \leq \frac{f_{Y/X+Y=s}(y)}{f_{X/X+Y=s}(y)}$, pour tout $x \leq y$, en effet

$\frac{f_{Y/X+Y=s}(x)}{f_{X/X+Y=s}(x)} = \frac{f_y(x) f_X(s-x)}{f_X(x) f_Y(s-x)}$, or comme $X \leq_{lr} Y$ alors $\frac{f_Y(x)}{f_X(x)} \leq \frac{f_Y(y)}{f_X(y)}$ (2.9)

pour tout $0 \leq x \leq y$

Si $x \leq y$ alors $s - y \leq s - x$, ce qui donne $\frac{f_X(s-x)}{f_Y(s-x)} \leq \frac{f_X(s-y)}{f_Y(s-y)}$ (2.10)

pour tout $s - y \leq s - x$

De la formule (2.9) et (2.10) on obtient $\frac{f_Y(x) f_X(s-x)}{f_X(x) f_Y(s-x)} \leq \frac{f_Y(y) f_X(s-y)}{f_X(y) f_Y(s-y)}$

D'autre part $\frac{f_Y(y) f_X(s-y)}{f_X(y) f_Y(s-y)} = \frac{f_{Y/X+Y=s}(y)}{f_{X/X+Y=s}(y)}$.

D'où le résultat.

Corollaire 2. 1 [29]: Si $A_k \leq_{lr} A_{k+1}$ et $B_k \geq_{lr} B_{k+1}$ alors la condition (2.5) est établie.

Preuve: Le résultat s'obtient en combinant la proposition (2.3) et la proposition (2.1 i)

En effet si $A_k \leq_{lr} A_{k+1}$ alors $(A_k / A_k + A_{k+1} = a) \leq_{lr} (A_{k+1} / A_k + A_{k+1} = a)$ et si $B_k \geq_{lr} B_{k+1}$ alors $(B_k / B_k + B_{k+1} = b) \geq_{lr} (B_{k+1} / B_k + B_{k+1} = b)$. Du fait que $X \leq_{lr} Y$ entraîne $X \leq_{st} Y$, alors le résultat se déduit de la proposition 2.1 i).

- Dans le livre de *Shaked et Shanthikumar* [17], au paragraphe scheduling, *Righter* [17] a donné des résultats sur l'ordonnancement stochastique par des ordres.

- *Ramudhin et al.* [30] ont étudié le comportement asymptotique et en espérance du flowshop sous des ordonnancements optimaux et ceux fournis par des heuristiques.

CHAPITRE 3
RECHERCHES ET TRAVAUX CONTEMPORAINS

Par travaux contemporains, nous sous-entendons les travaux effectués dans la dernière décennie.

3.1. Flow shop stochastique à 2-machines avec une capacité de stockage illimitée

• *Dans Elmaghrabi et Thoney* [31], le flow shop à 2-machines stochastique est considéré. Ils proposent une approche exacte de résolution avec une complexité du plus mauvais cas exponentielle et des approximations qui ne sont pas gourmands en calcul. Des résultats expérimentaux montrent que la procédure est à moins de 1% de la valeur de la solution optimale.

• *Kamburowski* [32] présente une nouvelle condition sur les distributions de temps d'exécution qui implique que la longueur d'ordonnancement diminue stochastiquement lorsqu'on permute deux tâches quelconques. Cette condition est plus faible que celle donnée par Ku et Niu [29], et ne se restreint pas au cas de deux tâches adjacentes.

Le Makespan d'une séquence de tâches peut être représenté par le temps de fin d'exécution d'un réseau PERT.

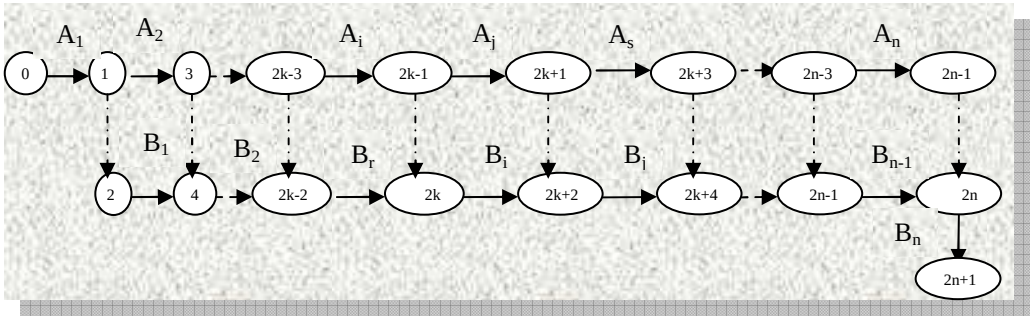

Figure 3.1. Réseau pour la séquence des tâches (1, 2,..., k-2, r, i, j, s, k+3,..., n)

Les noeuds (0) et (2n+1) représentent le début et la fin de l'exécution du projet. Supposons que la tâche *i* est à la $k^{ième}$ position dans la séquence d'ordonnancement (k ≠1, n).

Les arcs *(2k-3, 2k-1)* et *(2k, 2k+2)* décrivent l'exécution de la tâche *i* sur la machine *A* et *B* respectivement. Les arcs verticaux représentent des activités de durée nulle.
Le Makespan *M*, est la longueur du plus long chemin du noeud (0) à (2n+1)
Le temps le plus tôt, T_l que l'événement (noeud) *l* survienne est la longueur du plus long chemin de *(0)* à *l*.

$\underline{T_l}$ la longueur du plus long chemin de *l* à *(2n+1).*

Si on inverse tous les arcs du réseau et on suppose que les tâches sont exécutées en premier sur la machine *B* et après sur la machine *A*, le makespan de la séquence inversée est le même, et, par exemple $\underline{T_{2k+2}}$ et $\underline{T_{2k-1}}$ représentent les débuts de la tâche « *i* » sur les machines *B* et *A* respectivement. Puisque le réseau contient « *n chemins* » qui partagent des arcs en commun, le makespan est le plus grand des *n* dépendants variables aléatoires. Cette dépendance, parmi tant d'autres, fait aussi le problème du calcul l'espérance du makespan très difficile. La seule méthode exacte connue est quand les temps d'exécution de tâches sont de distribution exponentielle.

Pour une suite de tâches données, on peut chercher des réductions sur ces makespans qui résulte de la transposition de deux tâches.
Sans perte de généralité, considérons deux tâches *i* et *j* intermédiaires et adjacentes.

Soient $\pi_1 = (\rho, r, i, j, s, \varpi)$ et $\pi_2 = (\rho, r, j, i, s, \varpi)$ avec ρ et ϖ des sous suites de tâches n'ayant pas d'éléments *r, i, j* et *s*. Supposons que le tâche *i* est à la $k^{ième}$ position dans π_1 .

Les makespans correspondants peuvent être représentés comme suit:

$M_1 = max\ (X, Y, Z_1)$ et $M_2 = max\ (X, Y, Z_2)$ où

$$\left.\begin{aligned} X &= T_{2k-2} + B_r + B_i + B_j + \underline{T_{2k+4}} \\ Y &= T_{2k-3} + A_i + A_j + A_s + \underline{T_{2k+3}} \\ Z_1 &= T_{2k-3} + A_i + B_j + max\ (A_j, B_i) + \underline{T_{2k+4}} \\ Z_2 &= T_{2k-3} + A_j + B_i + max\ (A_i, B_j) + \underline{T_{2k+4}} \end{aligned}\right\} \quad (3.1)$$

La v.a X représente la longueur du plus long chemin de (0) à (2n+1) qui conduit B_r, soit , passe par l'arc $(2k-2, 2k)$ de longueur B_r. Identiquement, Y représente la longueur du plus long chemin qui contient A_s.

Autant que $Z_1(Z_2)$ représente la longueur du plus long chemin qui contient A_i et A_j (A_j et B_i) sous la suite $\pi_1\ (\pi_2)$.

Définissons les v.a Q_r et $\underline{Q_S}$ comme suit :

$$Q_r = T_{2k-2} + B_r - T_{2k-3} \quad \text{et} \quad \underline{Q_S} = \underline{T_{2k+3}} + A_s - \underline{T_{2k+4}} \qquad (3.2)$$

Q_r est le temps qui sépare la fin d'exécution de la tâche r sur la machine A et B et $\underline{Q_S}$ a une interprétation analogue pour la séquence inversée.

En substituant la formule (3.2) dans (3.1), on obtient la représentation convenable suivante :

$$M_1 = T_{2k-3} + \underline{T_{2k+4}} + \max\ (U, V, W_1)$$

$$M_2 = T_{2k-3} + \underline{T_{2k+4}} + \max\ (U, V, W_2) \text{ où} \qquad (3.3)$$

$$U = Q_r + B_{ij} \quad \text{et} \quad V = \underline{Q_S} + A_{ij} \qquad (3.4)$$

$$W_1 = A_i + B_j + \max\ (A_j, B_i) \quad \text{et} \quad W_2 = A_j + B_i + \max\ (A_i, B_j) \text{ et}$$

$$A_{ij} = A_i + A_j \quad \text{et} \quad B_{ij} = B_i + B_j$$

W_l représente la longueur du plus long chemin commençant de l'arc du noeud $A_i\ (A_j)$ et se terminant à l'arc du noeud $B_j\ (B_i)$ sous π_l, pour tout l = 1 et 2.

A_{ij} (B_{ij}) représentent la longueur de l'unique chemin qui commence de l'arc A_i (B_i) et se termine avec l'arc A_j (B_j) sous π_1 (π_2).

T_{2k-3}, le temps de fin d'exécution du job r sur la machine *A* et $\underline{T_{2k+4}}$, le temps de fin d'exécution de la tâche *s* sur la machine *B* sous la séquence inversée.

Ce résultat généralise celui de *Ku* et *Niu*[29] aux cas de tâches non adjacentes.

On suppose que les temps d'exécution des tâches sont des v.a.

Théorème 3.1 [32]*:* $\forall q_r$ et $\underline{q_S}$ dans le support Q_r et $\underline{Q_S}$

$$R_1(q_r , \underline{q_S}) = Max (q_r + B_{ij} , \underline{q_S} + A_{ij} , W_1)$$

$$\leq_{st} \; Max (q_r + B_{ij} , \underline{q_S} + A_{ij} , W_2) = R_2(q_r , \underline{q_S}) \qquad (3.5)$$

alors $M_1 \leq_{st} M_2$.

Preuve : La v.a conditionnelle $(X / Y = y)$ est notée par (X / y) **.**

La preuve est établie du fait que, Pour tout $l = 1$ et 2 on a :

$(M_1/q_r, \underline{q_S}) = (T_{2k-3}/q_r) + (\underline{T_{2k+4}} / \underline{q_S}) + R_1 (q_r , \underline{q_S})$ et que les variables aléatoires (T_{2k-3}/q_r), $(\underline{T_{2k+4}} / \underline{q_S})$ et $R_1 (q_r , \underline{q_S})$ sont indépendantes. □

Pour deux tâches adjacentes *i* et *j*, rappelons la condition proposée par *Ku* et *Niu* [29].

Si $(min [A_k , B_{k+1}] / a_{ij}, b_{ij}) \leq_{st} (min [A_{k+1} , B_k] / a_{ij}, b_{ij})$ (3.6)

$\forall a_{ij} , b_{ij}$ dans les supports de A_{ij} et B_{ij} alors $M_1 \leq_{st} M_2$.

La condition (3.6) semble plus simple à vérifier que celle de (3. 5), mais ce n'est pas le cas car les variables aléatoires conditionnelles évoquées dans la condition (3.6) sont typiquement difficiles à manier.

Le théorème suivant montre que la condition (3.5) est une condition plus faible.

Théorème 3.2 [32] : Pour deux tâches adjacentes *i* et *j*, la condition (3.6) implique la condition (3.5) mais pas l'inverse.

Preuve : Du fait que $X \leq_{st} Y$ implique que $(-Y) \leq_{st} (-X)$ et que l'ordre stochastique est préservé pour toute fonction croissante en particulier la fonction maximum, alors la condition (3.5) peut être réécrite comme:

$$Max\,(q_r - a_{ij},\ \underline{q_S} - b_{ij},\ -min\,(A_j, B_i)\ /\ a_{ij},\ b_{ij},\ q_r,\ \underline{q_S})$$

$$\leq_{st} Max\,(q_r - a_{ij},\ \underline{q_S} - b_{ij},\ -min\,(A_i, B_j)\ /\ a_{ij}, b_{ij}, q_r,\ \underline{q_S}) \qquad (3.7)$$

En ajoutant $a_{ij} + b_{ij}$ aux deux cotés de la formule (3.7), et en conditionnant sur A_{ij} et B_{ij} on obtient :

$$Max\,(q_r + B_{ij},\ \underline{q_S} + A_{ij},\ A_{ij} + B_{ij} - min\,(A_j, B_i)\ /\ q_r,\ \underline{q_S}) \leq_{st}$$

$$Max\,(q_r + B_{ij},\ \underline{q_S} + A_{ij},\ A_{ij} + B_{ij} - min\,(A_i, B_j)\ /\ q_r,\ \underline{q_S})$$

Sachant que $min\,(X, Y) = X + Y - max\,(X, Y)$, la preuve est terminée.

Remarque 3.1 : Les conditions (3.5) et (3.6) ne sont pas des conditions nécessaires pour $M_1 \leq_{st} M_2$**.** La seule raison qui fait que (3.5) n'est pas une condition nécessaire résulte du fait que $X + Z \leq_{st} Y + Z$ n'implique pas que $X \leq_{st} Z$ aussi bien si X, Y et Z sont indépendants.

Proposition 3.1 [32]: Pour deux tâches adjacentes i et j, l'équation (3.5) est satisfaite par chacune des deux conditions suivantes :

i) $P[\,min\,(A_i, B_j) \leq min\,(A_j, B_i)\,] = 1$

ii) $1/E(A_i) - 1/E(B_i) \geq 1/E(A_j) - 1/E(B_j)$, quand les temps d'exécution sont des v.a indépendantes et exponentiellement distribuées.

Preuve : Sous la condition i) on a

$P[-min\,(A_j, B_i) \leq -min\,(A_i, B_j)] = 1.$

Comme : $min\,(X, Y) = X + Y - max\,(X, Y)$, alors on en déduit que $W_1 \leq_{st} W_2$. Du fait que l'ordre stochastique est préservé pour toute fonction croissante en particulier la fonction maximum, alors la condition (3.5) est établie.

La justification de la condition ii) est basée sur la propriété de « sans mémoire » de la loi exponentielle, qui suit la même démarche de la preuve de la proposition 2.2.

- Gourgand et al.[33] traitent de l'évaluation des performances et de l'ordonnancement dans un flow shop stochastique à « m » machines afin de

minimiser l'espérance du makespan. Pour l'évaluation des performances, ils proposent un modèle markovien et un modèle de simulation à événements discrets.

3.2. Flow shop stochastique à 2-machines sans espace de stockage

Aucun espace tampon n'existe entre les deux machines. Quand une tâche a terminé son exécution sur la première machine, elle sera bloquée sur cette dernière si la seconde machine n'a pas terminé son exécution et par conséquent, on ne peut pas exécuter une nouvelle tâche sur la première machine.

● *Pinedo* [27] considère les deux cas de distribution de temps d'exécution, différente et identique sur les deux machines dont l'objectif est de minimiser l'espérance du makespan.

i). Si le problème est considéré sous l'assertion que le temps d'exécution est une v.a indépendante de distribution identique sur les différentes machines alors, la valeur de l'espérance des temps d'exécutions des tâches influe sur la séquence optimale des tâches.

Le théorème suivant minimise l'espérance du makespan quand les temps d'exécution des tâches sont stochastiquement ordonnés.

Théorème 3. 3 [27] **:** Si $X_1 \leq_{st} X_2 \leq_{st} \ldots \leq_{st} X_n$ alors la séquence des tâches 1 3 5 ... n ...6 4 2 et 2 4 6 ... n ....5 3 1 minimise l'espérance du Makespan

Sous la condition de non chevauchement, l'évaluation de $E(C_{max})$ est obtenue.

Théorème 3.4 [27]**:** Si les temps d'exécution sont non chevauchant alors une séquence minimise l'espérance du makespan si et seulement si c'est une séquence SEPT-LEPT et pour toute séquence SEPT-LEPT on a la valeur

$$E(C_{\max}) = \sum_{i=1}^{n} \mathrm{E}(\mathrm{X_i}) + \max_{\mathrm{i}=1,\ldots,\mathrm{n}} (\mathrm{E}(\mathrm{X_i}))$$

Notons par T la formule de $E(C_{max})$, donnée ci-dessus.

Preuve : La longueur d'ordonnancement d'une séquence dans un flow shop sans espace de stockage est plus grande ou égale à celle obtenue avec espace. Le blocage subit sur la première machine augmente la longueur d'ordonnancement. De plus si l'espérance du makespan d'une séquence est égale à T, on sait qu'elle est optimale. La preuve que l'évaluation de E (C_{max}) sous une séquence SEPT-LEPT est égale à T est similaire à celle du théorème 2.2) .On montre la condition nécessaire par l'absurde en effet, on suppose qu'il existe une séquence qui n'est pas SEPT-LEPT est qui est optimale. Dans cette séquence on peut trouver une tâche j qui s'exécute entre deux tâches successives i et k d'espérance plus grande que la tâche j. Sous cette séquence, la tâche j s'exécute après la tâche i, cependant, elle est bloquée sur la première machine le temps que la tâche i termine son exécution sur la seconde machine. Par conséquent l'espérance du temps d'exécution de la tâche k pour commencer son exécution sur la première machine est égal à au moins $\sum_{i=1}^{k-1} \mathrm{E}(\mathrm{X_i})$. D'où l'espérance du makespan est strictement plus grande que T.

ii). Si le problème est considéré sous l'assertion que le temps d'exécution est une variable aléatoire indépendante de distribution différente sur les différentes machines alors le flow shop stochastique peut être formulé comme un problème déterministe du voyageur de commerce.

On définit par *M*, la longueur d'ordonnancement.

Soient X_i^1 et X_i^2 les temps d'exécution de la tâche « *i* » respectivement sur la première machine de distribution F_i^1 et la seconde de distribution F_i^2.

Du fait que la zone tampon n'existe pas entre les deux machines, il est clair qu'à chaque fois qu'une tâche commence son exécution sur la première machine, la tâche précédente l'est aussi sur la seconde machine.

Soit *I*, le temps total durant l'intervalle [0, M] pour lequel une seule machine travaille.

Notons qu'une machine est dite inoccupée (idle) si elle est libre ou lorsqu'une tâche sur la première machine est bloquée par une autre tâche sur la seconde. D'où le résultat suivant : $2M = \sum_{i=1}^{n} X_i^1 + \sum_{i=1}^{n} X_i^2 + I$.

Exemple 3.1: Trois tâches J_1, J_2, J_3 sont à exécuter sur deux machines disposés en flow shop sans espace de stockage. Les temps d'exécution A_i et B_i de la tâche « i » respectivement sur la première et la seconde machine sont donnés dans le tableau suivant:

Tableau 3.1: Durées d'executions des tâches

Tâches	A_i	B_i
J_1	3	2
J_2	4	5
J_3	2	4

Une solution selon la liste L = (J_1, J_2, J_3) de longueur d'ordonnancement C_{max} de 16 unités de temps est représentée par le diagramme ci-dessous.

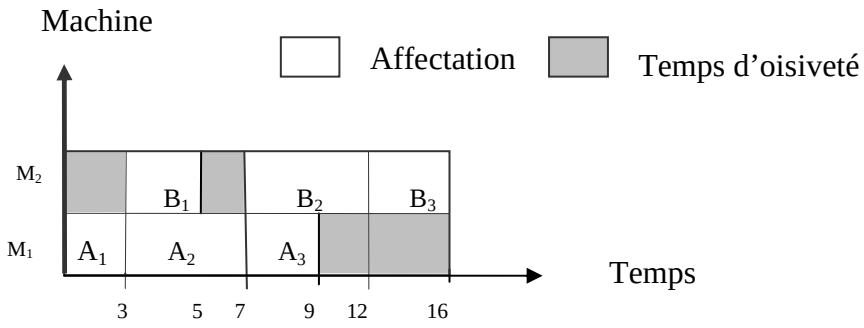

Figure 3.2: Diagramme de Gantt sous la séquence (1, 2, 3)

La tâche 3 est bloquée par la tâche 2.

La longueur du makespan *M* sous la séquence (1, 2, 3) pour le problème ci-dessus est de 16 unités de temps.

Le temps total I durant l'intervalle [0, 16] pour lequel une seule machine travaille est représenté par la somme de tous les temps qui correspondent au cellules coloriée en gris dans le schéma. D'où *I = 3 +2 + 3+ 4 = 12* unités de temps. Et deux fois la longueur du makespan est égale à

$$2M = \sum_{i=1}^{3} X_i^1 + \sum_{i=1}^{3} X_i^2 + I = 9 + 11 + 12 = 32 .$$

Comme $\sum_{i=1}^{n} X_i^1 + \sum_{i=1}^{n} X_i^2$ est constant, alors minimiser *E (M)* revient à minimiser E (I).

Supposons que la tâche *i* précède la tâche *j* dans la séquence.
Durant la période du temps où la tâche *j* occupe la première machine, il peut exister un certain temps où seulement une machine travaille. En effet, deux cas peuvent être considérés.

Dans le cas où $X_j^1 > X_i^2$, La première machine doit continuer l'exécution de la tâche *j* au moment où la tâche *i* se libère de la seconde machine.
La figure 3.3 illustre cette représentation.

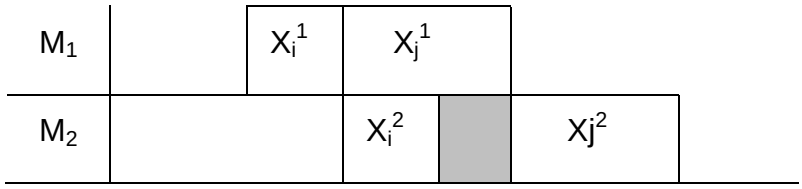

Figure 3. 3. Représentation des tâches i et j lorsque $X_j^1 > X_i^2$

Dans le cas où $X_i^2 > X_j^1$, la seconde machine doit continuer à exécuter la tâche *i* lorsque la tâche *j* s'est libérée de la première machine. La figure 3.4 illustre cette représentation.

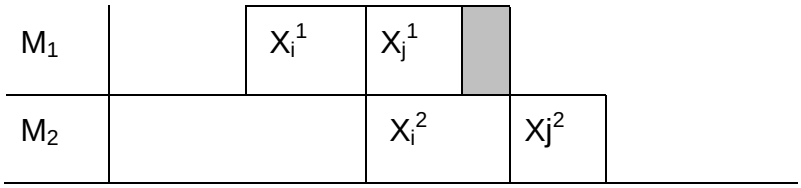

Figure 3. 4. Représentation des tâches i et j lorsque $X_j^1 < X_i^2$

On constate que la valeur d'espérance du temps durant lequel une seule machine est occupée est égale *à* $E(X_j^1) + E(X_i^2) - 2E(\min(X_j^1, X_i^2))$.

Considérant le problème du voyageur de commerce qui commence son trajet par la ville 0 et a pour objet de visiter les *1, 2…, n* villes et revenir à la cité 0.

L'espérance du temps durant lequel une seule machine travaille est définit par d_{ij} ; soit la distance entre la ville i et la ville j, donnée par :

$$\begin{cases} \mathrm{E}(\mathrm{X}_{\mathrm{j}}^{1}) + E(X_i^{2}) - 2E(\min(X_j^{1}, X_i^{2})), i \neq 0, j \neq 0 \\ \mathrm{d}_{0\mathrm{j}} = E(X_j^{1}) \\ d_{i0} = E(X_i^{2}) \end{cases}$$

Déterminer le tour qui minimise la distance totale pour ce voyageur de commerce est équivalent à déterminer une séquence optimale pour minimiser l'espérance du temps lorsqu'une seule machine travaille. Cependant le problème du voyageur de commerce est NP- complet qui implique qu'il n'y a pas de méthode efficace pour déterminer le minimum du makespan.

- *Jia* [34], a considéré le cas du blocage où n tâches ont des temps de service aléatoires. Les ordonnancements optimaux pour minimiser l'espérance de la différence en temps d'attente jusqu'à ce que l'exécution sur la seconde machine ne s'entamera sont établis quand les tâches peuvent être stochastiquement ordonnées.

CHAPITRE 4
CONTRIBUTION AU FLOWSHOP STOCHASTIQUE A 2-MACHINES

4.1. Introduction

Les résultats théoriques du flow shop stochastique arbitraire ne sont pas abondants et sont même rares et la détermination de l'espérance du makespan d'une suite de tâches est difficile. L'utilisation des heuristiques nous fournit une solution approchée.

Quand les *temps d'exécutions* des tâches suivent une loi de distribution *exponentielle* et sont indépendants entre eux, de moyenne égale à l'inverse du paramètre de la loi, le problème est résolu exactement par la règle de Talwar [02]. Cette règle sera exposée au paragraphe deux. Elle stipule qu'une tâche i précède une tâche j si et seulement si

$$1/ E(A_i) - 1/ E(B_i) \geq 1/E(A_j)-1/ E(B_j).$$

L'optimalité de cette règle a été prouvé par Cunningham et Dutta [03].

Pour évaluer l'espérance du makespan E (Cmax), ces auteurs ont proposé une formule récurrente utilisant les chaînes de Markov et les équations de Chapman-Kolmogorov. Ils fournissent, dans le paragraphe trois, un schéma itératif pour le flowshop à deux machines.

Dans le paragraphe suivant, une heuristique appelée « Pseudo Déterministe de Johnson » notée PDJ élaborée par Portougal (2006) [04] est présentée. C'est la version stochastique de l'algorithme de Johnson quand les données sont les espérances des temps d'exécutions. Nous vérifions expérimentalement qu'elle est asymptotiquement convergente en espérance vers la solution optimale avec une

probabilité égale à ″ un ″ dite aussi « presque sûrement ». Nous la présentons ci-dessous.
Les résultats expérimentaux et la performance de cette heuristique sont donnés au paragraphe cinq.

L'étude et la performance d'une heuristique de résolution pour la minimisation de l'espérance mathématique du makespan nous ont incité à déterminer la séquence optimale par la règle de Talwar et évaluer l'espérance du makespan par un schéma itératif détaillé ultérieurement.
Cette heuristique a été mise en œuvre sur des problèmes à 2-machines pour lesquels on dispose d'une solution optimale.

Nous proposons une méthodologie de recherche de la séquence optimale de Talwar.

4.2. Détermination de la séquence optimale par la règle de Talwar

Nous rappelons qu'à chaque tâche « i » est associée deux temps d'exécution A_i et B_i respectivement sur la première et la deuxième machine.
α_i, β_i représentent respectivement les paramètres de la loi exponentielle associés aux temps d'exécution A_i et B_i de la tâche « i » sur la première et la deuxième machine.

Quand les temps d'exécutions des tâches sont des variables aléatoires de distribution quelconque, l'évaluation de l'espérance mathématique *E (Cmax)* du makespan ne peut être faite de manière exacte. Seulement, quand les temps d'exécutions des tâches sont exponentiels des résultats existent.
Dans le cas exponentiel, le problème de minimiser l'espérance du makespan est résolu exactement par la règle de Talwar [02], avec $E(A_i) = a_i$ et $E(B_i) = b_i$. Cette règle stipule qu'une tâche i précède une tâche j si et seulement si

$$1/a_i - 1/b_i \geq 1/a_j - 1/b_j \qquad (4.1)$$

D'une manière équivalente, une tâche i précède une tâche j si et seulement si

$$E(\min(A_i, B_j) = \frac{a_i b_j}{a_i + b_j} \leq \frac{a_j b_i}{a_j + b_i} = E(\min(A_j, B_i) \qquad (4.2)$$

En effet, pour tout $t > 0$ on a :

$$P(A_i > t) = e^{-\frac{1}{a_i}t}$$

$$P(B_j > t) = e^{-\frac{1}{b_j}t}$$

Si A_i et B_j sont deux v.a indépendants de distribution exponentielle alors, la distribution du minimum de ces variables est aussi exponentielle.

En effet,

$$P(\min(A_i, B_j) > t) = P(A_i > t, B_j > t) = P(A_i > t).P(B_j > t)$$

$$= e^{-\frac{1}{a_i}t} . e^{-\frac{1}{b_j}t}$$

$$= e^{-(\frac{1}{a_i} + \frac{1}{b_j})t}$$

On en déduit que $E(\min(A_i, B_j) = \frac{1}{1/a_i + 1/b_j}$

D'une façon analogue $E(\min(A_j, B_i) = \frac{1}{1/a_j + 1/b_i}$

Par conséquent la condition (4.2) est établie

Un ordonnancement optimal S* est déterminée par des transpositions de couples de tâches d'une séquence initiale $S_0 = (t_1, t_2, .., t_i, \ldots, t_j, \ldots, t_n)$, ne remplissant pas la condition de Talwar.

Les temps d'exécutions des tâches respectivement sur la première et la seconde machine sont générés exponentiellement selon la méthode d'inversion. Le principe de cette méthode est présenté dans le cas d'une distribution F(x) continue. En effet, F(.) étant monotone croissante, et par conséquent inversible, on peut générer la suite $\{x_i\}$ à partir de la procédure ci dessous.

Si U est une valeur générée uniformément sur l'intervalle [0,1] par la fonction Random du langage de programmation, $X = F^{-1}(U) = -\frac{1}{\lambda}\log(U)$ est une v.a de distribution exponentielle de paramètre λ.

La moyenne de tous les temps d'exécutions générées représente l'espérance d'un temps d'exécution.

Soit [i] l'indice de la tâche dans la séquence S_0 à la position i. L'organigramme de la figure 4.1 schématise la recherche de la séquence optimale S*.

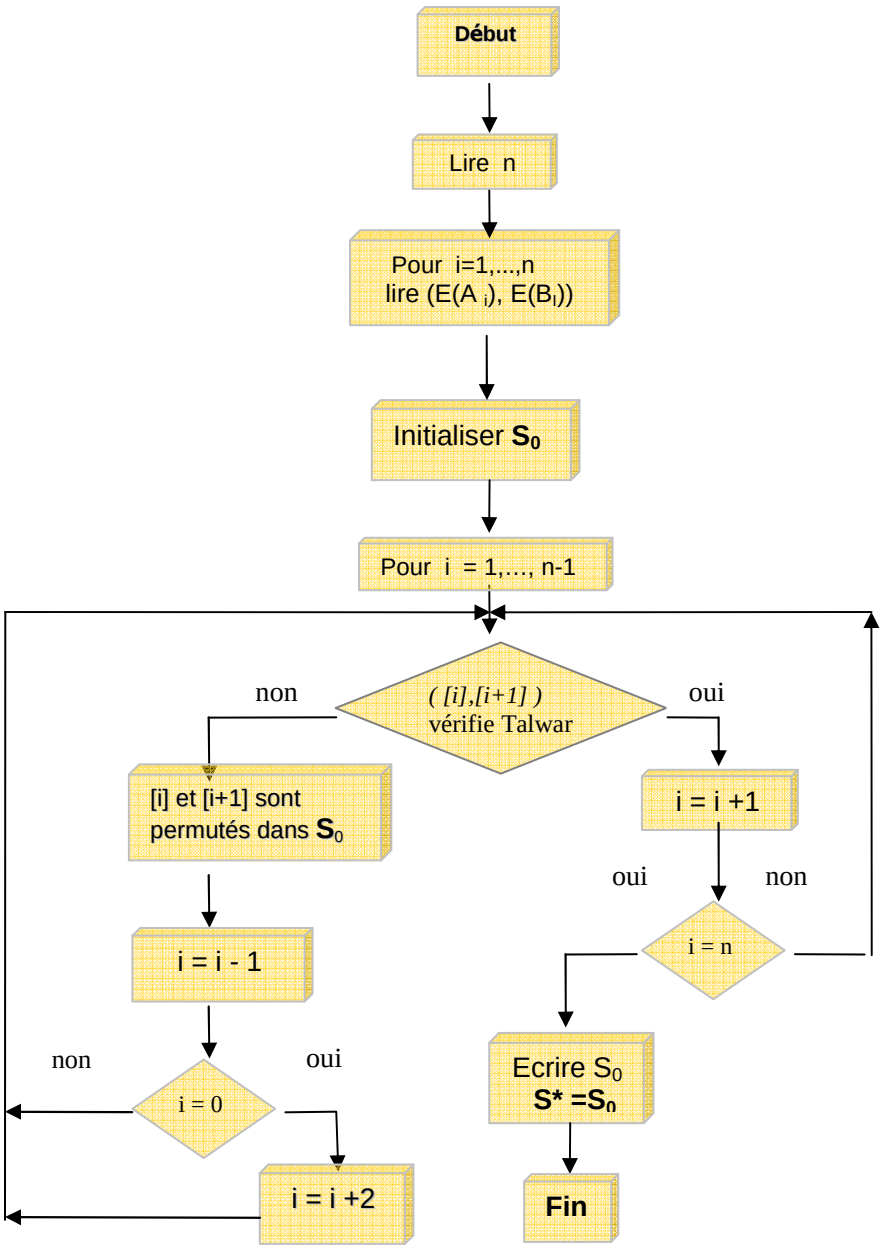

Figure 4.1 : Méthodologie de la séquence optimale de Talwar**.**

Une application de la détermination de la séquence optimale est illustrée dans l'exemple 4.1 ci-dessous. Les valeurs dans ce tableau sont les résultats d'un tirage aléatoire fait par nos soins. Un autre utilisateur, avec un autre tirage, trouvera sûrement des valeurs différentes.

Exemple 4. 1 : Soit cinq tâches à exécuter sur deux machines disposées en séries.
$E(A_i)$, $E(B_i)$ représentent la valeur de A_i respectivement de B_i générée par la loi exponentielle (un seul tirage).

Tableau 4.1. Espérances de temps d'exécutions

Tâches i	Espérances	
	$E(A_i)$	$E(B_i)$
1	0.7357792	10.74001
2	0.0159666	0.1827835
3	13.412781	0.2313543
4	0.3447451	0.3592366
5	7.3848394	0.0771587

Soit S_0 = (1, 2, 3, 4, 5) une séquence initiale de tâches. La condition de Talwar sur la première paire de tâches (1, 2) n'est pas vérifiée. Permutons ces deux tâches. (2, 1, 3, 4, 5) est la séquence obtenue. Parmi ce qui suit dans la séquence, le premier couple ne remplissant pas la condition, est celui de la paire (3,4) permutant les et (2, 1, 4, 3, 5) est la séquence améliorée. La paire précédente (1,4) satisfait la condition, répétons la procédure pour la paire (3,5) suivante, elle vérifie la règle, la séquence reste inchangée. La séquence améliorée correspond à la séquence S* de Talwar.

4.3. Evaluation de $E(C_{max})$

Cunnigham et Dutta ont proposé une formule récurrente utilisant les chaînes de Markov et les équations de Chapman-Kolmogorov[03]. Ils fournissent, dans ce qui suit, un schéma itératif pour cette évaluation.

Considérant « n » tâches qui s'exécutent sur 2-machines sous la séquence $S_0 = (1, 2,\ldots, n)$. L'espérance du makespan est évaluée en utilisant la procédure de la figure 4.2 ci-dessous.

Soient la donnée (q, r), (q, r = 1, 2,…, n+1 ; q ≥ r), l'état du flow shop (la position de la tâche r et q dans le système) et $\alpha_{q,r}$,$\beta_{q,r}$ des constantes.

Déterminons les valeurs initiales suivantes :

$$\alpha_{1,1} = 1/a_1 \quad , \quad \beta_{1,1} = -1/(a_1)^2,$$

Pour tout q, $1 \leq r < q \leq n+1$, calculons les expressions données par les formules :

$$\alpha_{q,r}=(a_{q-1}\,\alpha_{q-1,r}+b_{r-1}\,\alpha_{q,r-1})\,/(a_q+b_r)$$

$$\beta_{q,r} = -(\,\alpha_{q,r} - a_{q-1}\,\beta_{q-1,r} - b_{r-1}\beta_{q,r-1})\,/\,(a_q+b_r)$$

Pour tout q, $2 \leq q \leq n$,

$$\alpha_{q,q}= b_{q-1}\,\alpha_{q,q-1}/\,a_q$$

$$\beta_{q,q}= -(\,\alpha_{q,q} - b_{q-1}\,\beta_{q,q-1}\,)/\,a_q$$

$$\text{où} \quad a_{n+1} = b_0 = 0 \ ,$$

La valeur de l'espérance de la longueur d'ordonnancement est :

$$E(Cmax) = -b_n\beta_{n+1,n}$$

Figure 4.2 : Schéma itératif pour l'évaluation de $E(C_{max})$

Pour les preuves et les justifications, le lecteur est invité à consulter [03].

L'exemple 4.2 illustre une application de l'évaluation de l'espérance du makespan.

Exemple 4.2 : Trois tâches sont à exécuter sur deux machines disposées en séries.

a_i et b_i représentent les paramètres de la loi exponentielle associés aux temps d'exécution A_i et B_i de la tâche « i » sur la première et la seconde machine, tirés aléatoirement.

S_0 = (1, 2,…, n) une séquence initiale de tâches.

Evaluons l'espérance du makespan sous la séquence S_0.

Tâches i	Paramètres	
	a_i	b_i
1	0.1	0.2
2	0.3	0.4
3	0.6	0.8

De notre calcul, les valeurs initiales sont : $\alpha_{1,1} = 10$, $\beta_{1,1} = -100$.

Initialisons r par 1

Pour q = 2 ; $\alpha_{2,1} = 2$, $\beta_{2,1} = -24$, $\alpha_{2,2} = 1.3333$, $\beta_{2,2} = -20.4444$.

En incrémentant r d'une unité r = 2

Pour q = 3 ; $\alpha_{3,1} = 0.75$, $\beta_{3,1} = -9.9375$, $\alpha_{3,2} = 0.55$, $\beta_{3,2} = -8.6708$,

$\alpha_{3,3} = 0.3666$, $\beta_{3,3} = -6.3916$.

En Réincrémentant r d'une unité r = 3

Pour q = 4 ; $\alpha_{4,1} = 2.25$, $\beta_{4,1} = -41.0625$, $\alpha_{4,2} = 1.95$, $\beta_{4,2} = -38.4125$,

$\alpha_{4,3} = 1.25$, $\beta_{4,3} = -25.5625$.

La valeur de l'espérance de la longueur d'ordonnancement est :

$$E(Cmax) = 20.45.$$

L'espérance du temps de fin d'exécution de la dernière tâche sous n'importe qu'elle séquence S peut être calculer directement en utilisant les relations récurrentes ci-dessus évoquée.

4.4. Heuristique Pseudo Déterministe de Johnson [04]

Pour la détermination d'une borne sur $E(C_{max})$, l'heuristique PDJ a été étudiée. Elle consiste en l'application de la règle de Johnson sur les espérances des temps d'exécutions. Nous appliquons cette règle au cas stochastique. La séquence générée est notée par S.

On définit le « Pseudo Déterministe Makespan » par PDM obtenue lorsque les données sont les espérances de temps d'exécution. PDM_S est le PDM obtenu sous la séquence S. Son évaluation est déterminée par les équations de récurrence suivantes :

$$\left.\begin{aligned} C_{i,[1]} &= \sum_{l=1}^{i} E(p_{l,[1]}), \qquad \text{i} = 1,2 \\ C_{1,[k]} &= \sum_{l=1}^{[k]} E(p_{1,[l]}), \qquad \text{k} = 1,..., \text{n} \\ C_{2,[k]} &= Max\,(C_{1,[k]}, C_{2,[k-1]}) + E(p_{2,[k]}) \quad ; \ \text{k} = 2, ..., \text{n}. \end{aligned}\right\} \qquad (4.3)$$

où $C_{2,[n]}$ est le temps de fin d'exécution de la dernière tâche [n] sur la seconde machine qui représente le PDM_S.

$P_{i,[j]}$ est le temps d'exécution de la tâche [j]sur la $i^{iéme}$ machine.

La valeur PDM_S ainsi obtenue est comparée à l'optimale.

De l'exemple 4.1 et en appliquant la règle de Johnson sur les espérances de temps d'exécutions on obtient :

Tâche 2 ordonnancée : 2 — — — —

Tâche 5 ordonnancée : 2 — — — 5

Tâche 4 ordonnancée : 2 4 — — 5

Tâche 1 ordonnancée : 2 4 1 — 5

Tâche 3 ordonnancée : 2 4 1 3 5

La séquence S générée par l'heuristique PDJ sera (2, 4, 1, 3, 5).

Dans la figure 4.3. Le Pseudo-Déterministe Makespan PDM_S sous cette séquence S est égale à 21.971271 unités de temps.

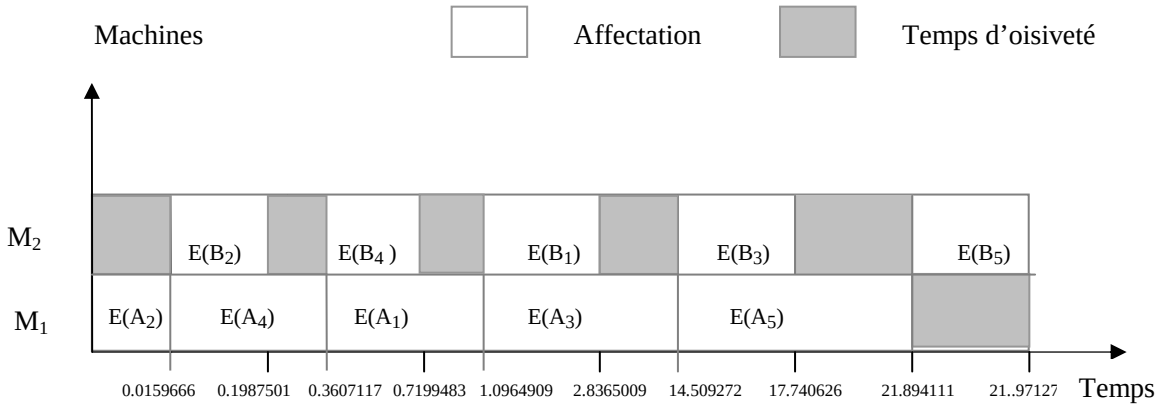

Figure 4.3. Diagramme de Gantt d'une séquence générée par PDJ

Nous proposons la comparaison de cette heuristique PDJ [04] avec la méthode exacte de Talwar. On montre que son évaluation est une borne inférieure pour l'espérance du makespan. Portougal [04] montre la convergence asymptotique vers la solution exacte. Nous la confirmons expérimentalement. L'heuristique de résolution PDJ, la recherche de la séquence de Talwar et l'évaluation de l'espérance du makespan ont été implémenté en utilisant la procédure rand() du logiciel scilab-3.1.1 de calcul numérique, copyright (c) 1989-2005 Consortium Scilab (INRIA, ENPC). Elle est utilisée pour générer des variables aléatoires. Les programmes ont été déroulés sur un processeur Intel Pentium 4 cadencé à 2.1GHz et 128 Mo de RAM sous Windows XP.

4.5. Résultats expérimentaux et performance de l'heuristique

L'heuristique PDJ a été testée sur un nombre de tâches allant de 20 à 1000 tâches. Pour chaque tâche i, les durées d'exécution A_i et B_i suivent la lois exponentielle. Les valeurs obtenues sont la moyenne de 100 expériences effectuées et les résultats sont représentés dans le tableau 4.2.

PDM_S et M(S) représentent respectivement le Pseudo-Déterministe Makespan de la séquence S et la longueur d'ordonnancement sous la même séquence.
Le traitement des tâches se fait sur deux machines alignées en série.

Tableau 4.2. Comparaison entre l'heuristique et la solution optimale

n	20	50	70	100
PDM_S	463.60329	792.50275	1373.0189	1439.2715
Solution optimale de Talwar **(E(M(S*))**	489.41099	809.50875	1402.9457	1488.7006
L'espérance de la longueur **(E(M(S))**	490.27052	811.51024	1405.7393	1491.8395

Une représentation graphique peut s'obtenir en associant l'axe des abscisses au nombre de tâches et l'axe des ordonnées à la valeur du makespan prise en moyenne sur 100 expériences effectuées. L'allure des courbes est affine par morceau.

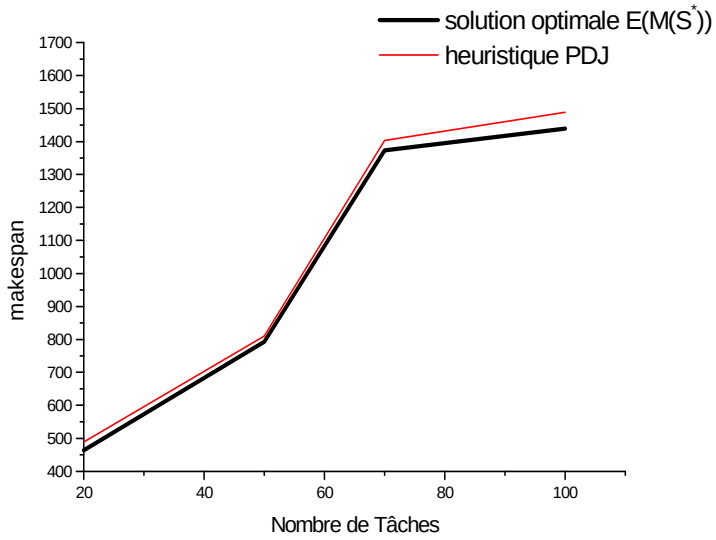

Figure 4.4.Comparaison graphique de l'heuristique et de la solution optimale

Nos résultats expérimentaux montrent que le PDM_S est une borne inferieure pour l'espérance du makespan. Ce résulat confirme celui donné par l'inégalité de « Jensen » $PDM_S \leq E(M(S))$, qui se traduit par « L'espérance de la longueur est supérieure à la longueur des espérances ».

Comme la séquence de Talwar S^* *est optimale* alors l'espérance du makespan sous la séquence S est une borne supérieure pour l'objectif $(E(M(S^*)) \leq (E(M(S))$.

La figure 4.5. représente un histogramme de comparaison entre l'evaluation de l'éspérance du makespan sous la sequence S obtenue de l'heuristique et celle de la solution optimale.

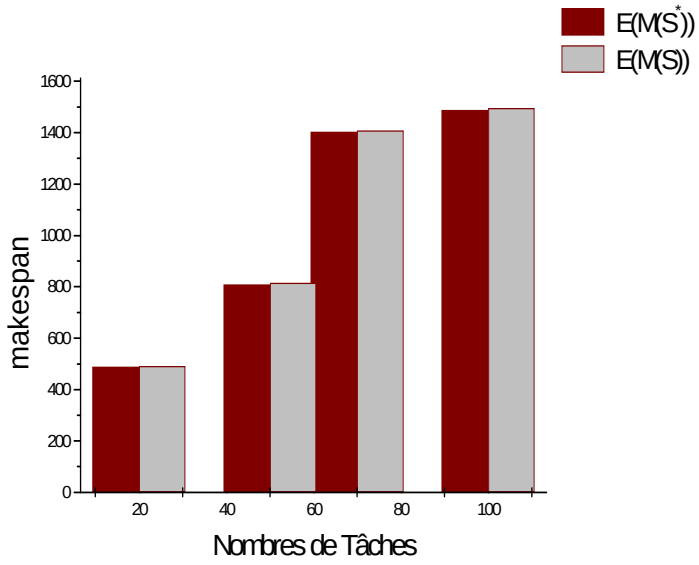

Figure 4.5. Comparaison entre l'évaluation de l'espérance de la séquence optimale et celle de l'heuristique.

Nous avons testé la performance de l'heuristique PDJ en calculant l'erreur définie par l'expression :

$$erreur = \frac{E(M(S)) - E(M(S^*))}{E(M(S^*))} .$$

Elle représente un écart relatif, où $E(M(S))$ et $E(M(S^*))$ définissent respectivement l'espérance du makespan sous la séquence S obtenue par le schéma itératif et la solution optimale de Talwar. La figure 4.6 interprète la performance de l'heuristique pour les problèmes à 100, 300, 500, 1000 tâches.

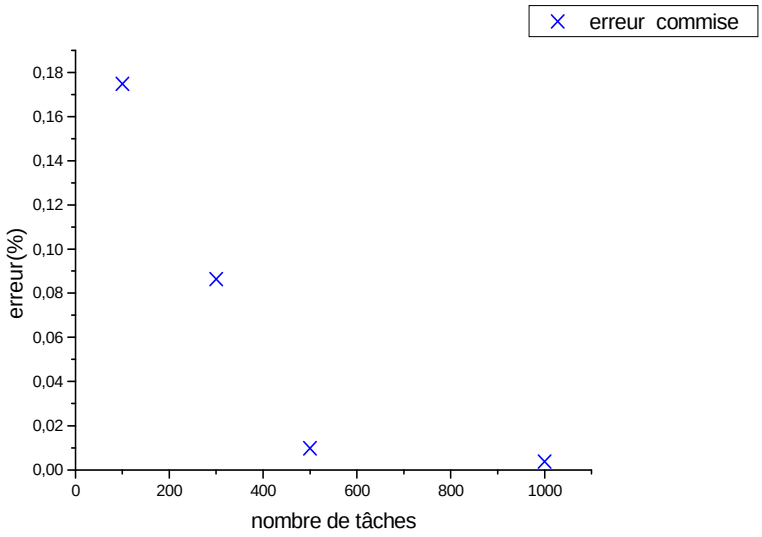

Figure 4.6. Performance de l'heuristique.

A partir de l'évaluation de l'erreur pour les problèmes à plus de 500 tâches, nous constatons que l'erreur devient négligeable et par suite l'évaluation de l'heuristique converge asymptotiquement vers la solution optimale de Talwar.

Après avoir testé cette heuristique sur des problèmes de grandes tailles Nous confirmons expérimentalement qu'elle est asymptotiquement optimale. En effet, cette heuristique est suffisante pour sélectionner une bonne séquence qui minimise l'espérance du makespan. L'étude expérimentale nous fournisse une borne inférieure pour minimiser l'espérance du makespan.

Remarque : Ce chapitre a fait l'objet d'un :

1. résumé accepté comme « poster » au « Séminaire International De Mathématiques Appliquées et Simulations », 22-25 Avril 2007, Oum El Bouaghi, Algérie.
2. article publié à CPI07, 5ième conférence Internationale sur la conception et la production intégrées, Thème 9: Planification de la production et ordonnancement, Rabat, Maroc, 22, 23 & 24 Octobre 2007, pp.1-15.

CHAPITRE 5
RECHERCHE RECENTE : CAS DES TEMPS D'EXECUTIONS DE TACHES DE DISTRIBUTION DE PROBABILITE DE WEIBULL

Nous rappelons que dans le cas où les temps d'exécutions sont déterministes, la règle de Johnson est suffisante et optimale pour la minimisation de l'espérance du makespan. Quand ils sont aléatoires, indépendants et de distribution exponentielle, la règle de Talwar nous fournira un ordonnancement stochastiquement optimal. Si les temps d'exécutions des tâches sont indépendants et de distribution de probabilité de Weibull, Kalczynski et Kamburowski(2006) [05] ont suggéré une nouvelle règle d'ordonnancement où les deux règles de Johnson et Talwar en dérivent. Cette règle se confond avec une heuristique quand les temps d'exécution sont caractérisés par leur moyenne et ayant un même coefficient de variation.

Des résultats de simulation montrent que cette règle donne des résultats « encourageants » pour minimiser l'espérance du makespan.

5.1. Propriété de la loi de Weibull

Une v.a X suit la loi de Weibull de paramètres respectivement d'échelle $\lambda > 0$ et de forme $c > 0$ si elle est de densité :

$$f(x) = \begin{cases} \lambda c e^{-\lambda x^c} x^{c-1} & x \geq 0 \\ 0 & \text{sinon} \end{cases}$$

Sa fonction de répartition est définit par : $F(x) = P(X < x) = 1 - e^{-\lambda x^c}$, pour tout $x > 0$.

La loi de Weibull est souvent utilisée pour caractériser la fiabilité des matériels dans l'industrie. Son espérance mathématique, sa variance et son coefficient de variation, sont respectivement donnés par les expressions suivantes :

$$E(X) = \frac{\Gamma(1+\frac{1}{c})}{\lambda^{1/c}},$$

$$Var(X) = D^2(X) = \frac{\Gamma(1+\frac{2}{c}) - \Gamma^2(1+\frac{1}{c})}{\lambda^{2/c}},$$

$$C_V(X) = \frac{D(X)}{E(X)} = \sqrt{\frac{\Gamma(1+\frac{2}{c})}{\Gamma^2(1+\frac{1}{c})} - 1}$$

où Γ(p) représente la fonction gamma définie par la valeur de l'intégrale $\int_0^{\infty} x^{p-1} e^{-x} dx$.

Soient A_i et B_i les deux v.a associées aux temps d'exécution d'une tâche i sur la première et la seconde machine. Elle sont de distribution de Weibull d'espérance respectivement a_i et b_i et de paramètres α_i, β_i.

Proposition 5.1 [05]. Si A_i et B_j sont indépendantes alors, le minimum de ces variables est une v.a et est aussi de Weibull.

Preuve : Soit l'événement que la v. a A_i est supérieure à une valeur d'instant t, la probabilité de sa réalisation est $P(A_i > t)$. Cette probabilité est donnée par la formule suivante :

$P(A_i > t) = e^{-\alpha_i t^c}$, pour tout $c > 0$, $\lambda > 0$, $t > 0$, où $E(A_i) = a_i = \frac{\Gamma(1+\frac{1}{c})}{\alpha_i^{1/c}}$. (5.1)

et $P(B_j > t) = e^{-\beta_j t^c}$, où $E(B_j) = b_j = \frac{\Gamma(1+\frac{1}{c})}{\beta_j^{1/c}}$. (5.2)

Pour deux tâches i et j, si A_i et B_j représentent deux v.a indépendantes alors la probabilité que leur minimum soit supérieur à t est égale au produit des probabilités que les deux v.a soient supérieur à t. Par conséquent

$$P(\min(A_i, B_j) > t) = P(A_i > t, B_j > t) = P(A_i > t)P(B_j > t) = e^{-\alpha_i t^c} e^{-\beta_j t^c} = e^{-(\alpha_i + \beta_j)t^c} .$$

D'où $E(\min(A_i, B_j)) = \dfrac{\Gamma(1+\frac{1}{c})}{(\alpha_i + \beta_j)^{1/c}}$ (5.3) .

5.2. Règle sur les distributions de temps d'exécution de tâches de Weibull

Kalczynski et Kamburowski [05] ont présenté une règle d'ordonnancement qui stipule qu'une tâche i doit précéder une tâche j si et seulement si

$$E(\min(A_i, B_j) = \frac{a_i b_j}{(a_i{}^c + b_j{}^c)^{1/c}} \le \frac{a_j b_i}{(a_j{}^c + b_i{}^c)^{1/c}} = E(\min(A_j, B_i) .$$

Cette inégalité est équivalente à celle de :

$$1/a_i{}^c - 1/b_i{}^c \ge 1/a_j{}^c - 1/b_j{}^c . \quad (5.4)$$

En effet, en substituant α_i et β_j respectivement de la formule (5.1) et (5.2) de la proposition (5.1), on trouve que :

$$\alpha_i = \frac{\Gamma^c(1+1/c)}{a_i{}^c}, \quad \beta_j = \frac{\Gamma^c(1+1/c)}{b_i{}^c} . \quad (5.5)$$

En remplaçant les formules (5.5) dans celle de (5.3), nous obtenons que :

$$E(\min(A_i, B_j)) = \frac{1}{(\frac{1}{a_i{}^c} + \frac{1}{b_j{}^c})^{1/c}} = \frac{a_i b_j}{(a_i{}^c + b_j{}^c)^{1/c}} .$$

D'une façon analogue, et en permutant les indices des v.a, on en déduit que :

$$E(\min(A_j, B_i)) = \frac{1}{(\frac{1}{a_j{}^c} + \frac{1}{b_i{}^c})^{1/c}} = \frac{a_j b_i}{(a_j{}^c + b_i{}^c)^{1/c}} .$$

Par conséquent l'inégalité $E(\min(A_i, B_j) \le E(\min(A_j, B_i)$ est équivalente à:

$$\frac{1}{(\frac{1}{a_i{}^c} + \frac{1}{b_i{}^c})^{1/c}} \le \frac{1}{(\frac{1}{a_j{}^c} + \frac{1}{b_i{}^c})^{1/c}}$$, qui est aussi équivalente à :

$$\frac{1}{a_i{}^c} - \frac{1}{b_i{}^c} \ge \frac{1}{a_j{}^c} - \frac{1}{b_j{}^c} \quad .$$

Remarques 5.1 : - Soit c = 1 et μ représentant son espérance, une distribution de Weibull se réduit à la loi exponentielle.

- Pour des valeurs de c au voisinage de ∞, elle se réduit au cas déterministe à valeur μ (dans ce cas la densité f(x) se réduit au point 0).

D'où la règle (5.4) contient les deux règles de Johnson et Talwar à la fois, comme cas particuliers.

- Une propriété remarquable des deux règles de Johnson et Talwar est la transitivité entre les tâches.

Si $E(\min(A_i, B_j)) \le E(\min(A_j, B_i))$ et $E(\min(A_j, B_k)) \le E(\min(A_k, B_j))$

alors $E(\min(A_i, B_k)) \le E(\min(A_k, B_i))$

Qui se traduit par : Une tâche i précède une tâche j et une tâche j précède une tâche k implique que la tâche i précède la tâche k.

Ce qui nous conduit à dire que si la condition est optimale, elle est évidemment transitive mais si elle n'est pas transitive, l'optimalité n'est pas assurée.

Même si la condition **«** E(min(A_i , B_j)) ≤ E (min(A_j , B_i)) » assure un ordre transitif entre les tâches pour les temps d'exécution de distribution de Weibull, ceci ne garantie pas l'optimalité de la règle (5.4). Un exemple sera fournie ci dessous. Une façon de mesurer la performance de cette règle approchée consiste à faire une analyse expérimentale. On propose d'appliquer cette règle comme une heuristique pour la minimisation de l'espérance du makespan.

5.3. Test d'optimalité

L'optimalité de la règle (5.4) qui stipule qu'une tâche i doit précéder une tâche j si et seulement si $\frac{1}{a_i^{\,c}} - \frac{1}{b_i^{\,c}} \geq \frac{1}{a_j^{\,c}} - \frac{1}{b_j^{\,c}}$ est testée par simulation. Ceci nous conduit à prouver qu'elle peut nous fournir une séquence de tâches optimale.

Le test a été pris lorsque la condition $E(\min(A_i, B_j)) = E(\min(A_j, B_i))$ pour toute paire de tâches i et j. Si la règle (5.4) minimise notre objectif alors toutes les séquences de tâches conduisent à une même évaluation de cette espérance. L'ordre des tâches n'affecte pas la valeur de l'espérance du makespan.
Nous présentons un exemple de quatre tâches où un test a été effectué pour mesurer cette performance.

Exemple 5.1. On considère le cas de quatre tâches qui s'exécutent sur deux machines disposées en flowshop. A_i et B_i représentent le temps d'exécutions de la tâche « i » respectivement sur la première et la seconde machine, de distribution de Weibull de paramètre de forme c et de paramètres d'échelle respectifs :

$$\alpha_1 = 1 \ , \ \alpha_2 = 4/5, \ \alpha_3 = 3/5, \ \alpha_4 = 2/5$$
$$\beta_1 = 9/10 \ , \ \beta_2 = 7/10 \ , \ \beta_3 = 1/2 \ , \ \beta_4 = 3/10.$$

Pour toute paire de tâches i et j on a :

$$E(\min(A_i, B_j)) = E(\min(A_j, B_i)) = \frac{\Gamma(1+\frac{1}{c})}{(\alpha_i + \beta_j)^{1/c}} = \frac{\Gamma(1+\frac{1}{c})}{(\alpha_j + \beta_i)^{1/c}}.$$

Si la séquence de tâches est déterminée par la règle optimale (5.4), toutes les 4! (24) permutations de tâches doivent fournir la même évaluation, et cela pour tout c > 0.

Nous avons réalisé un générateur de jeux d'essais dont l'intention est d'atteindre l'objectif ci dessus. Les paramètres α_i et β_i sont fixés comme ci-dessus et les paramètres de formes « c » sont donnés aléatoirement. Les temps d'exécutions A_i et B_i associés à une tâche « i » sont simulés pour une distribution de Weibull de « paramètre de forme » « c » et de «paramètre d'échelle » respectifs α_i et β_i selon la procédure suivante basée sur la méthode d'inversion :

Procédure 5.1

1. tirer par la fonction Rand, une valeur U uniformément distribué sur [0, 1].
2. $X = [-\frac{1}{\lambda}\log U]^{1/c}$ suivra une loi de Weibull de « paramètre de forme » « c » et de « paramètre d'échelle » λ.

La moyenne de tous les temps d'exécutions générées représente expérimentalement l'espérance du temps d'exécution X.

Pour un échantillon de 10000 jeux d'essai, on simule les 24 espérances de la longueur d'ordonnancement pour les valeurs aléatoires de paramètres c = 1.5, 2 et 2.5.

Les espérances du temps de fin d'exécution de la dernière tâche sont simulés et sont données par le tableau 5.1 (un seul échantillon).

Tableau 5.1. Espérances simulés du makespan

Séquences de tâches	c = 1,5	c = 2	c = 2,5
(1 2 3 4)	6,9044581	6,0064839	5,8198486
(1 2 4 3)	7,0369835	6,0142045	5,8152691
(1 3 2 4)	6,9044581	6,0064839	5,8198486
(1 3 4 2)	7,0197282	6,0767249	5,7826558
(1 4 2 3)	7,0322536	6,0211448	5,7568952
(1 4 3 2)	7,0322536	6,0288429	5,8169859
(2 1 3 4)	6,9044581	6,0064839	5,7376507
(2 1 4 3)	7,0369835	6,0064839	5,7726598
(2 3 4 1)	6,9637445	6,0485989	5,7568952
(2 3 1 4)	6,9044581	6,0064839	5,7376507
(2 4 1 3)	6,9044581	6,0033195	5,8198595
(2 4 3 1)	7,0262697	6,0064839	5,8152694
(3 1 2 4)	6,9044581	6,0033195	5,7376507
(3 1 4 2)	7,0197282	6,0064839	5,8159824
(3 2 4 1)	6,9637443	6,0485989	5,7376507
(3 2 1 4)	6,9044581	6,0064839	5,8198486
(3 4 1 2)	7,0290144	6,0188429	5,7376507
(3 4 2 1)	7,0290144	6,0188429	5,7726598
(4 1 2 3)	6,9745892	6,0253563	5,7376507
(4 1 3 2)	7,0197282	6,0126538	5,8186957
(4 2 3 1)	6,9745589	6,0256841	5,8186957
(4 2 1 3)	6,9044581	6,0064839	5,7376507
(4 3 1 2)	7,0197282	6,0356241	5,8156895
(4 3 2 1)	6,9637443	6,0213568	5,8156985

Les résultats de l'écart respectivement du minimum et du maximum simulé de l'espérance du makespan ainsi que le pourcentage de l'erreur commise sont donnés dans le tableau suivant :

Tableau 5.2. Ecart et pourcentage de l'erreur entre le minimum et le maximum de la longueur d'ordonnancement.

Paramètre c	Ecart	Pourcentage de l'erreur
1.5	0.09845	1.98 %
2	0.06962	1.68 %
2.5	0.08219	1.75 %

Les cellules, représentées et encadrées dans le tableau 5.1, fournissent les valeurs de l'objectif simulé.

Les résultats expérimentaux montrent que pour les différents paramètres de formes c, toutes les évaluations de l'espérance du makespan simulés des 24 séquences de tâches sont proches les uns des autres. En effet l'écart entre le minimum et le maximum d'espérance du makespan est de moyenne égale à 0.0834246 unité de temps.

Pour c = 2.5, le minimum et le maximum d'espérance du makespan simulé sont de 5.7376507 et 5.8198486 respectivement et le pourcentage de l'erreur calculé est de 1.75 %.

On remarque que pour ce test, dans 40 % des cas on a obtenu la solution optimale, et dans les 60 % restant la solution l'avoisine à moins de 0.07 unité de temps en moyenne.

Les résultats de la simulation sont satisfaisants. Ce qui confirme que la règle (5.4) peut fournir une séquence optimale.

5.4. Distribution de temps d'exécution arbitraire

Pour une distribution de temps d'exécution des tâches arbitraire la formule « $E(\min(A_i , B_j)) \leq E(\min(A_j , B_i))$ » n'assure pas toujours un ordre transitif entre les tâches. L'exemple (5.2) est exhibé dans le cas où les temps d'exécutions des tâches sont de distribution de Gamma pour montrer cet effet. Cette raison, parmi tant d'autres, fait aussi la difficulté de déterminer dans le cas général une règle d'ordonnancement qui minimise l'espérance du makespan.

Si X représente une v.a de distribution Gamma et d'espérance μ, sa densité est définit par l'expression $f(x) = \frac{s^s x^{s-1}}{\mu^s \Gamma(s)} e^{-sx/\mu}$ pour $x \geq 0$, $s > 0$.

La variance et le coefficient de variation de la v.a X sont données par :

$$\text{Var}(X) = \mu^2/s, \quad C_V(X) = 1/\sqrt{s}\,.$$

Nous rappelons que la loi exponentielle correspond à la distribution de la v.a associée au temps séparant l'apparition de deux évènements. La distribution Gamma est la distribution d'une somme de variables aléatoires exponentiellement distribuées. Elle est associée à la distribution du temps écoulé entre la $k^{ième}$ et la $(k + r)^{ième}$ apparition d'un évènement. Elle est d'usage utilisée comme distribution de probabilité de la durée de vie des appareils domestiques. L'usure est un phénomène naturel de détérioration des véhicules automobiles, des machines mécaniques etc.

Si s, le paramètre de la loi est de valeur « n », la distribution Gamma se réduit à une distribution d'Erlang d'ordre n.

Proposition 5.2 [05] **:** L'espérance du minimum de deux v.a X_1et X_2 de distribution d'Erlang de paramètre s et d'espérance μ_1 et μ_2 respectivement est donnée par

$$E(\min(X_1, X_2)) = (\mu_1 + \mu_2)\sum_{r=0}^{s-1} \binom{2r}{r} \times \frac{1}{r+1}\left[\frac{\mu_1\mu_2}{(\mu_1+\mu_2)^2}\right]^{r+1}.$$

Exemple 5.2 : On considère trois tâches de distribution de temps d'exécution d'Erlang. Fixons s = 2, les espérances de temps d'exécution des tâches sur la première et la seconde machine sont données dans le tableau 5.3) suivant :

Tableau 5.3. Les données sur la distribution d'Erlang

Tâches i	Espérances	
	$E(A_i)$	$E(B_i)$
1	4.6832	6.7264
2	2.6651	3.0565
3	1.1623	1.2081

Après calcul, on constate que

$E(\min(A_1, B_2)) = 2.2914 < E(\min(A_2, B_1)) = 2.2968$, et

$E(\min(A_2, B_3)) = 1.0097 < E(\min(A_3, B_2)) = 1.0102$ Mais

$E(\min(A_1, B_3)) = 1.1169 > E(\min(A_3, B_1)) = 1.1155$.

De cet exemple, nous confirmons que la condition « $E(\min(A_i, B_j)) \leq E(\min(A_j, B_i))$ » n'assure pas la transitivité entre les tâches. D'où la difficulté de déterminer une règle optimale de minimisation de l'espérance du makespan. La seule règle exacte dont nous disposons actuellement est quand les temps d'exécution de tâches sont de distribution exponentielle.

CONCLUSION ET PERSPECTIVES

La résolution des problèmes de flowshop stochastiques à 2-machines se fait en général par l'utilisation d'au moins trois classes de méthodes stochastiques : les méthodes de type recuit simulé, les algorithmes génétiques et les processus bandits, des processus à 2-décisions markoviens etc… Dans ce mémoire, nous avons introduit une approche alternative par les ordres stochastiques, un nouvel outil de résolution. Cette approche consiste à comparer des variables aléatoires selon un ordre stochastique et par suite comparer les ordonnancements qui sont formés d'une suite finie de variables aléatoires. Des dominances stochastiques sont définies entre les tâche à qui on leurs associe un temps de traitement aléatoire sur les deux machines, afin de minimiser la longueur d'ordonnancement. De notre recherche bibliographique, une synthèse des résultats retrouvés dans les livres de Barlow [14], Marshall [15] et Shaked et Shantikumar [17] est fournie. Des travaux des chercheurs précurseurs, continuateurs et contemporains tel Talwar[02], Pinedo[27], Ku et Niu [29], Komburowski[32] etc…sont exposés. Pour évaluer l'espérance du makespan, une heuristique a été étudiée, implémentée et testée sur un nombre élevé de données. Nous avons proposé sa comparaison avec la solution optimale. Nous avons vérifié expérimentalement qu'elle est asymptotiquement convergente en espérance vers la solution optimale avec une probabilité égale à « un » dite aussi « presque sûrement ». Son évaluation constituera une borne inférieure pour l'espérance du makespan. Un exemple est donné où un test a été effectué dans le cas d'une distribution de probabilité de Weibull.

La seule règle exacte dont nous disposons actuellement est quand les temps d'exécution de tâches sont de distribution exponentielle. Pour une distribution des temps d'exécution des tâches arbitraire, les résultats théoriques du flow shop stochastique sont rares et la détermination de l'espérance du makespan d'une suite de tâches est difficile. Des problèmes sont encore ouverts dans le cas où la fonction

objectif est différente de l'espérance du makespan, de type flowtime, flowtime moyen, nombre de tâches en retard etc… Une généralisation au cas du flow shop stochastique à « m » machines est à espérer et peut faire l'objet d'une étude ultérieure.

LISTE DES SYMBOLES ET DES ABREVIATIONS

n	Nombre de tâches à exécuter .
A_i	temps d'exécution de la tâche « i » sur la première machine.
B_i	temps d'exécution de la tâche « i » sur la seconde machine.
C_{max}	longueur d'ordonnancement, soit le makespan.
SPT	ordre croissant du temps d'exécution.
LPT	ordre décroissant du temps d'exécution.
P(X)	probabilité de X.
X /Y	événement conditionné.
E	espérance mathématique.
$X \geq_{st} Y$	ordre stochastique.
$X \geq_{as} Y$	ordre presque sûrement.
$X \geq_{hr} Y$	ordre du taux de hasard.
$X \geq_{lr} Y$	ordre du rapport de vraisemblance.
S	séquence de tâches générée par l'heuristique.
S^*	séquence optimale de Talwar.
PDJ	Pseudo Déterministe de Johnson.
PDM	Pseudo Déterministe makespan.
PDM_S	PDM de la séquence S.
M(S)	longueur d'ordonnancement sous la séquence S.

REFERENCES

1. Johnson, S.W., « Optimal two and three-stage production schedules with setup times included », *Naval Research Logistic Quarterly 1*, 1954, pp. 61-68.

2. Talwar, P.P., «A note on sequencing problems with uncertain job times ». *Journal of Operations Research Society of Japan 9*, 1967, pp. 93-97.

3. Cunningham, A.A., Dutta, S.K., «Scheduling jobs with exponentially distributed processing times on two machines of a flowshop », *Naval Research Logistics Quarterly16*, Vol.20, No.1, 1973, pp. 69-81.

4. Portougal, V., D, Trietsch., «Johnson's problem with stochastic processing times and optimal service level ». *European Journal of Operational Research*, (2006), 751-760.

5. Kalczynski, P.J., Kamburowski, J., «A heuristic for minimizing the expected makespan in two-machine flow shops with consistent coefficients of variation ». *European Journal of Operational Research* 169, 2006, pp. 742-750.

6. Mehdi Ouafia et Derbala Ali. «Résolution du flowshop stochastique à 2-machines par les ordres stochastiques ». CPI07, 5ième conférence Internationale sur la conception et la production intégrées, Thème 9: Planification de la production et ordonnancement, Rabat, Maroc, 22, 23 & 24 Octobre 2007, pp.1-15.

7. Mehdi Ouafia et Derbala Ali. « Contribution au flowshop stochastique à deux machines ». Séminaire International De Mathématiques Appliquées et Simulations, Centre Universitaire Larbi Ben M'hidi d'Oum El Bouaghi, 22-25 Avril 2007, accepté la communication pour être présentée en Poster.

8. Mehdi Ouafia et Derbala Ali. « Le flow shop stochastique à 2-machines», Séminaire Hebdomadaire du Département de Mathématiques, faculté des sciences de l'université de Saâd Dahlab de Blida, 04 mars 2007.

9. Baker, K.R., « Introduction to sequencing and scheduling », *Wiley* Publishing*, New York* 1974.

10. Carlier, J., Chrétienne, P., « Problèmes d'ordonnancement. Modélisation, complexité et algorithmes », Paris, Masson, 1988.

11. Gotha., «Les problèmes d'ordonnancements », *Rairo Recherches Opérationnelles,* Vol. 27, No.1, 1993, pp.77-150. [28].

12. Pinedo, M.L., «Scheduling: Theory, Algorithms and systems ». Prentice Hall, Englewood cliffs, New Jersey, 1995.

13. Graham R.L, Lawler E.L. et Lenstra J.K. et Rinnoy Kan A.HG., « Optimisation and approximation in deterministic sequencing and scheduling: asurvey », Discrete Mathematics 5, (1979), 287-326.

14. Barlow, R.E., Proschan, F., «Statistical theory of reliability and life testing », New York, 1975.

15. Marschall et Olkin, I., « Inequalities:Theory of majorization and its applications», *Academic Press*, 1994, Orlando, F.L.

16. Ross, S.M., « Introduction to dynamic programming », *Academic Press, Wiley,* 1983.

17. Shaked, M., Shanthikumar, G., « Stochastic orders and their applications », *Academic Press* 1994, Boston, pp 381-427.

18. Oukid, N., «Comparaisons stochastiques de files d'attente », Thèse de Magister en Mathématiques Appliquées, Département de Mathématiques, Université de Blida, 1995.

19. Conway, R.C., Maxwell, W.L., Miller, L.W., «Theory of scheduling ». *Addison Wesley*, 1967.

20. Banerjee, B.P., «Single facility sequencing with random execution times », *Operations research 13*, 1965, pp. 358-364.

21. Makino, T., «On a scheduling Problem ». *Journal of Operation Research Society of Japan 8*. 1965, pp. 32-44.

22. Bagga, P.C., « n-Jobs, 2-machine sequencing problems with stochastic service times », *Operations Research 7*, 1970, pp.184-197.

23. Garey, M.R.D., Johnson, D.S., et Sethi, R., «The complexity of flowshop and jobshop scheduling », *Mathematics of Operations Research*, 1976. Vol.1, No.2, 117-129.

24. Levner, E., «Optimal planning of parts machining on a number of machines ». *Automatic Remote Control 12*, 1969, pp. 1972-1981.

25. Weber, R.R., «The Interchangeability of tandem ./M/1 queues in series ». *Journal of Applied Probability 16*. 1979, pp. 690-695.

26. Muth, E.J., «The reversibility property of production lines ». *Management Science*, 1979, vol.25, No.2, pp. 152-158.

27. Pinedo, M., «Minimizing the expected makespan in stochastic flow shops ». *Operations Research 30*, 1982, pp. 148-162.

28. Tembe, S.V., Wolff, W, W., «The optimal order of service in tandem queues », *Operations Research* 22, 1974, 824 – 833.

29. Ku, P.S., Niu, S.C., «On Johnson's two-machine flow shop with random processing times ». *Operations Research* 1986, vol. 34, No. 1, pp. 130-136.

30. Ramudhin, A., Bartholdi III, J.J., Calvin, J.M., Vande Vate, J.H et Weiss, G., «A probabilistic analysis for two-machine flowshops ». Operations Research, 1996, Vol. 44, No.6, pp. 889-908.

31. Elmaghraby, S.E., Thoney, K.A., «The two-machine stochastic flowshop problem with arbitrary processing time distributions", *IIE Transactions* 31, 1999, 467-477.

32. Kamburowski, J., «Stochastically minimizing the makespan in two-machine flow shops without blocking", *European Journal of Operational Research* 112, 1999. 304 - 309.

33. Gourgand, M., Grangeon, N., et Norre, S., « Une contribution au problème d'ordonnancement du flow shop stochastique". *Congrès ROADEF*, école des mines de Nantes, France, 2000. pp. 146-147.

34. Jia, C., «Minimizing variation in stochastic flow shop" Operations Research Letter 23, 1998, pp. 109-111.