وزارة التعليم العالي والبحث العلمي

BADJI MOKHTAR- ANNABA UNIVERSITY

UNIVERSITE BADJI MOKHTAR ANNABA

جامعة باجي مختار- عنابة

Année : 2014

Faculté: Sciences de l'ingéniorat

Département: Electronique

Thèse

Présentée en vue de l'obtention du diplôme de : Doctorat

Intitulée

Modélisation et conduite des systèmes hybrides. Application aux systèmes manufacturiers

Spécialité: Automatique

Présentée par : KHALDOUNA Zahia

DIRECTEUR DE Thèse: DJEGHABA Messaoud Pr Université d'ANNABA

DEVANT Le JURY

PRESIDENT : DEBBACHE Naceredine Pr Université d'ANNABA

EXAMINATEURS:

MOUSS Leila Hayet Pr Université de BATNA
MOSTEFAI Mohamed Pr Université de SETIF
MEHANAOUI Med Lamine Mc Université de SKIKDA

Remerciements

Qu'il me soit d'abord permis de remercier et d'exprimer ma gratitude envers le bon Dieu, qui m'a donné la patience et le courage pour que je puisse continuer ce travail.
Je tiens à exprimer toute ma gratitude à Monsieur D*jeghaba Messaoud,* Professeur au sein du département d'électronique à l'université d'Annaba, qui a assuré la direction de ce travail. Je le remercie pour ses conseils pertinents et éclairés. Qu'il trouve ici l'expression de ma profonde reconnaissance.
Mes vifs remerciements vont également aux membres de jury de soutenance composé de: *Mr Debbache Naceredine*, Professeur au sein du département d'électronique à l'université d'Annaba, *Mme Mouss Leila Hayet* Professeur à l'université de Batna, *Mr Mostefai Mohamed* Professeur à l'université de Sétif, *Mr Mehanaoui Mohamed Lamine* Maitre de conférence à l'université de Skikda. Je les remercie chaleureusement pour leur présence et pour avoir accepté d'examiner la présente thèse.
Je remercie aussi tous les enseignants et les responsables du Département d'Electronique de l'Université de Annaba pour leur aide et leur encouragement.
Sans oublier mes collègues, je tiens à les remercier vivement.
Enfin, Je voudrais associer à mes remerciements toutes les personnes qui ont contribué de prés ou de loin à l'aboutissement de ce travail

ملخص

في نظم الإنتاج الحالية عدة الصعوبات متعلقة باحتياجات المستخدم (البحث عن حل شامل ودقيق ...) ، بخصائص المشكلة التي يجري تناولها وبحسابات الوقت المهم. وقد تم حل هذه الصعوبات بالعديد من الدراسات وذلك باستخدام أساليب التحسين المختلفة.
وللتغلب على هذه الصعوبة ، فقد اخترنا أن نركز على تطوير الأساليب العشوائية ودراسة تطبيقاتها على تصميم يواجه مشاكل نظام الإنتاج.
إن فعالية هذه الطرق يعتمد على اختيار و سيطرة المعلمات. هذا التعديل معقد، وخاصة عندما يكون عدد المعلمات كبير و عندما يتم توسيع نطاق الاختلاف عند كل معلمة.
ترتكز هذه الأطروحة على الطرق الرئيسية و الواعدة للطرق العشوائية مثل: الخوارزميات الجينية ، محاكاة الصلب و البحث المحرمات . من خلال هذه الأساليب، يمكن أن توفر الحلول لمشاكل التحسين التقريبية الكلاسيكية لأكبر عدد من التطبيقات التي كان من المستحيل التعامل معها من قبل .
ولحل مشكلة التحسين، أدلى ببعض المساهمات الأصلية، الذي هو تحديد العوامل التي يعتقد انها تؤثر على الأداء والمقارنة من خلال تغيير هذه العوامل، كلها تقدم حلا وسطا جيدا بين زمن الاستجابة وارضاء المعيار المختار، مثل تحقيق أقصى قدر من عدد البنود و تقليل وقت إنجاز (Makespan) .
نحن من ثم قارنا بين هذه المناهج ، للتمييز بين تلك التي تقدم أفضل الحلول بالمعنى المقصود في معايير اختيارها ، لعدد كبير من العينات.
الخوارزمية الجينية هي أفضل من غيرها من النهجين .
اقترحنا أيضا إدماج الطرق الثلاثة في قرار واحد و نظام للتقييم و اختيار في كل لحظة واحدة هته التي توفر أفضل النتائج.
لنفس العدد من العينات ، ووجدنا تحسنا للدخل الشامل ، مع مساهمة كل النهج.

الكلمات الرئيسية : نظام تحسين الإنتاج، ألمراقبة ، الأساليب العشوائية ، الخوارزمية الجينية ، البحث المحرمات ، محاكاة الصلب

Résumé

Les problèmes d'optimisation en système de production présentent plusieurs difficultés liées aux besoins de l'utilisateur (recherche d'une solution globale, précision de la solution …), aux caractéristiques du problème traité et aux temps de calculs importants. La résolution de telles difficultés a fait l'objet de nombreux travaux en utilisant diverses méthodes d'optimisation.
Pour lever cette difficulté, nous avons choisi de nous intéresser au développement des méthodes stochastiques et d'étudier leurs applications aux problèmes de conception rencontrés en système de production.
L'efficacité de ces méthodes dépend du choix de ses paramètres de contrôle. Ce réglage est complexe, surtout quand le nombre de paramètres est élevé et quand la plage de variation de chacun de ces paramètres est étendue.
Cette thèse s'intéresse aux principales et les plus prometteuses des méthodes stochastiques : les algorithmes génétiques, recuit simulé et la recherche tabou. Grâce à ces méthodes, on peut proposer des solutions approchées pour des problèmes d'optimisation classiques de plus grande taille et pour de très nombreuses applications qu'il était impossible de traiter auparavant. Elles comportent souvent plusieurs paramètres contrôlant les différents opérateurs et leur influence sur les processus stochastiques.
Pour résoudre notre problème d'optimisation, quelques contributions originales ont été apportées, qui consistent à identifier les facteurs supposés influencer les performances et à effectuer des comparaisons en faisant varier ces facteurs, toute en offrant un bon compromis entre le temps de réaction et la satisfaction du critère choisi, tels que la maximisation de nombre de pièces produites et la minimisation du temps de réalisation (Makespan).
Nous avons ensuite comparé entre ces approches, pour distinguer celle qui offre la meilleure solution au sens des critères choisis, sur un nombre important d'échantillons.
L'algorithme génétique devance les deux autres approches.
Nous avons proposé aussi d'intégrer les trois méthodes dans un même système de décision et d'évaluer et de choisir à chaque instant celle qui offre le meilleur résultat. Sur le même nombre d'échantillon, nous avons constaté une amélioration du résultat global, avec la contribution de chaque approche.

Mots clés : *Système de production, Optimisation, contrôle, méthodes stochastiques, algorithme génétique, recherche tabou, recuit simulé*

Abstract

Optimization problems in production systems present several difficulties related to the needs of the user (search for a comprehensive solution, accuracy of the solution ...), characteristics of the problem being addressed and important time calculations. The resolution of such difficulties has been the subject of numerous studies using various optimization methods. To overcome this difficulty, we have chosen to focus the development of stochastic methods and study their applications to design problems encountered in the production system. The effectiveness of these methods depends on the choice of its control parameters. This adjustment is complex, especially when the number of parameters is large and when the variation range of each parameter is extended.

This thesis focuses on the main and most promising stochastic methods such as genetic algorithms, simulated annealing and taboo search. Through these methods, we can provide approximate solutions for classical optimization problems larger and for many applications it was impossible to deal with before. They often have several parameters that control the various operators and their influence on stochastic processes.

To solve our optimization problem, some original contributions were made, which is to identify the factors believed to influence performance and comparison by varying these factors, all offering a good compromise between response time and satisfaction the chosen criterion, such as maximizing the number of items and the minimization of completion time (Makespan).

We then compared between these approaches, to distinguish the one that offers the best solution within the meaning of the selected criteria, a large number of samples.

The genetic algorithm is ahead of the other two approaches.

We also proposed to integrate the three methods in one decision system and to evaluate and choose at every moment one that offers the best results. The same number of samples, we found an improvement of comprehensive income, with the contribution of each approach.

Keywords: System of production, Optimization, Control, Stochastic methods, Genetic algorithm, Taboo search, Simulated annealing

LISTE DES ABREVIATIONS

AG	Algorithme génétique
FIFO	First In First Out
FMS	Flexible Manufacturing System
RdP	Réseaux de Pétri
RdPCC	Réseau de Pétri continu à vitesses constantes
RdPCV	Réseau de Pétri continu à vitesses variables
RdPH	Réseau de Pétri Hybride
RdPHT	Réseau de Pétri hybride temporisé
RS	Recuit simulé
RT	Recherche tabou
RTHu	Recherche tabou de Hu
RWS	Roulette Wheel Sélection
SDED	Systèmes dynamiques à évènements discrets
SDH	Systèmes dynamiques hybrides
SED	Systèmes à évènements discrets
SFPM	Système Flexibles de Production Manufacturière

Liste des symboles

x_i , i = 1..n	variables d'optimisation
n	nombre de variables d'optimisation
$x_0,\ldots, x_0^n$	vecteur d'optimisation
t_i	la transition i
$\{t_1, t_2,..., t_m\}$	l'ensemble de transition
P_i	la place i
$\{p_1, p_2,..., p_n\}$	l'ensemble de place
M_i	marquage de la place i
M_0	marquage initiale
Pre	matrice d'incidence avant
Post	matrice d'incidence arrière
W	matrice d'incidence du réseau
$d_j(t)$	la temporisation associée à la transition t_j
$U_j(t)$	la fréquence de franchissement associée à la transition
S	séquence de franchissement
V (t)	vitesse instantanée de franchissement
N	nombre de solutions dans l'espace de recherche
f(x)	fonction coût de la solution x
C_{max}	Makespan à optimiser
N(s)	voisinage de la solution s
P_c	probabilité de croisement
P_m	probabilité de mutation
T_0	température initiale
L	longueur de la liste tabou
s	solution du problème
s^*	solution optimale
$O_{i,j,k}$	la $j^{ème}$ opération de la tâche i sur la machine k
$E_{i,j}$	temps opératoire
P(x,y)	probabilité de Boltzmann
T	paramètre de contrôle (température)

Liste des tableaux

Tableau 3.1 : Lois de décroissance de la température les plus utilisées..... **59**
Tableau 3.2 : Tableau représentant l'ensemble des tâches..... **70**
Tableau 4.1: Table de comparaison du Makespan par application des deux méthodes..... 8**9**
Tableau 4.2: Table de comparaison de la productivité par application des deux méthodes **89**
Tableau 4.3:Résultats obtenu par l'application des deux opérateurs sélection et croisement.. **90**
Tableau 4.4:Résultats obtenu par l'application des deux opérateurs sélection et mutation..... **91**
Tableau 4.5:Résultats obtenu par l'application des trois opérateurs de l'algorithme génétique**91**
Tableau 4.6:Résultats obtenu par l'application de la méthode recuit simulé..... **92**
Tableau 4.7:Résultats obtenu par l'application de la méthode recherche Tabou..... **93**
Tableau 4.8: Nombre de pièces produites par chaque algorithme **94**
Tableau 4.9: Tableau récapitulatif obtenu par utilisation des trois méta-heuristiques..... **95**

LISTE DES FIGURES

Figure 1.1: Schéma d'un système de production **5**
Figure 1.2 : Schéma d'un processus d'optimisation **20**
Figure 1.3 : La simulation **21**
Figure 2.1 : Exemple de réseau de petri **28**
Figure 2.2: Éléments composant un RdP hybride **37**
Figure 2.3: Structure physique du système de transport **39**
Figure 2.4: Le modèle Réseau de Pétri hybride du convoyeur **39**
Figure 2.5: RdP lots modélisant un fonctionnement pour le système de transport **40**
Figure 2.6: Cohérence entre les lots **41**
Figure 2.7: Nouveaux types de nœuds dans un Réseau de Petri lot **41**
Figure 3.1 : Processus de sélection par décimation **51**
Figure 3.2 : Principe du croisement en un point **52**
Figure 3.3 : Principe du croisement en deux points **52**
Figure 3.4: Exemple de croisement uniforme **52**
Figure 3.5 : Exemple sur la mutation binaire **54**
Figure 3.6 : Exemple de codage des chromosomes **73**
Figure 4.1: Schéma du Système à modéliser **80**
Figure 4.2: Représentation d'une ligne de production **80**
Figure 4.3: Présentation de la cellule de production **81**
Figure 4.4: Modèle de la cellule de production **83**
Figure 4.5: Exemple de modèle de fonctionnement des gammes de fabrication **85**
Figure 4.6 : Schéma de comparaison du Makespan des deux méthodes **89**
Figure 4.7 : Schéma de comparaison de la productivité des deux méthodes **89**
Figure 4.8 : Schéma du Makespan obtenu par application des deux opérateurs S + C **91**
Figure 4.9 : Schéma du Makespan obtenu par application des deux opérateurs S + M **91**
Figure 4.10 : Schéma du Makespan obtenu par application des deux opérateurs S+C+M **92**
Figure 4.11 : Schéma du Makespan obtenu par application de la méthode recuit simulé **93**
Figure 4.12 : Schéma du Makespan obtenu par application de la méthode recherche tabou **94**
Figure 4.13 : Comparaison du Makespan fournis par les trois algorithmes (critère 1) **94**
Figure 4.14 : Schéma du Makespan obtenu par utilisation des trois méta-heuristiques **95**
Figure4.15:Taux d'utilisation de chaque méta-heuristique dans le programme d'optimisation **96**

SOMMAIRE

Introduction générale **1**

CHAPITRE I : Les systèmes de production flexible

1- Introduction **3**

2- Les systèmes de production **4**
2.1- Historique **4**
2.2- Les systèmes de production classiques **5**
2.3- Nécessité de la flexibilité **5**
3- Les systèmes de production flexible **7**
3.1- Définition **7**
3.2- Caractéristiques des systèmes de production **7**
4- Problématique de la conduite de ces systèmes **10**
4.1- Environnement incertain **10**
4.2- Ressources à aptitudes multiples **10**
4.3- Autres contraintes **11**
4.4- Nécessité d'un modèle de conduite **11**
4.4.1- Une classification des formalismes **11**
4.4.2- Formalismes pour la modélisation structurelle **12**
4.4.3- Formalismes pour la modélisation dynamique **12**
4.4.4- Formalismes hybrides ou mixtes **13**
5- Optimisation combinatoire **14**
5.1- Choix d'une méthode **15**
5.1.1- Les méthodes déterministes **15**
5.1.2- Les méthodes stochastiques **16**
5.2- Les méta-heuristiques **17**
6- Simulation et analyse des résultats **20**
6.1- Application aux systèmes de production **21**
6.2- Simulation des systèmes dynamiques hybrides **21**
7- Conclusion **22**

CHAPITRE II : Les outils de modélisation

1- Introduction **23**

2- Les outils de modélisation **23**

2.1- Les outils de modélisation utilisés dans les Systèmes discrets **23**

2.2- Les outils de modélisation utilisés dans les Systèmes continus **25**

2.3- Les outils de modélisation utilisés dans les Systèmes Dynamiques Hybrides **26**

3- Réseau de Petri **27**

3.1- Marquage des places **29**

3.2- Comportement du franchissement dans un réseau de Petri **29**

3.3- Réseaux de Petri et robustesse des SED **30**

4- Classes de réseau de petri **30**

4.1- Les temporisations dans les réseaux de pétri **30**

4.1.1- Les réseaux de petri temporisées **31**

4.1.2- Réseaux de Petri stochastiques **32**

4.2- Les réseaux de Petri Colorés **32**

4.3- Les réseaux de Petri continus **33**

4.3.1- Introduction **33**

4.3.2- Réseau de pétri continu autonome **34**

4.3.3- RdP continu à vitesses constantes (RdPCC) **34**

4.3.4- RdP continu à vitesses variables (RdPCV) **34**

5- Les réseaux de petri hybrides **35**

5.1- Réseau de pétri hybride autonome **36**

5.2- Les RdP hybrides temporisés **37**

6- Réseau de pétri lots **38**

7- Validation et vérification des modèles utilisés **42**

8- Conclusion **42**

CHAPITRE III : Les méthodes de conduite

1- Introduction **43**

2- Les méthodes d'optimisation **43**

2.1- Les méthodes exactes et leurs limites **44**
2.1.1- Les méthodes Branch & Bound **44**
2.1.2- Méthodes mathématiques **44**
2.1.3- La méthode des directions conjuguées de Powell **44**
2.2- Les méta-heuristiques et leurs adaptations à un problème d'optimisation **45**
3- Les Méta-heuristiques étudiées **46**
3.1- Les Algorithmes Génétiques **46**
3.1.1- Principes de base **47**
3.1.2- Codage du chromosome **48**
3.1.3- Évaluation des individus **49**
3.1.4- Génération de la population initiale **49**
3.1.5- Opérateur de sélection **49**
3.1.6- Opérateur de croisement **51**
3.1.7- Opérateur de mutation **53**
3.1.8- Remplacement générationnel **55**
3.1.9- Critères d'arrêt **55**
3.2- Le recuit simulé (Simulated Annealing) **55**
3.2.1- Probabilité de Boltzmann **56**
3.2.2- Température **58**
3.2.3- Décroissance de la température **58**
3.2.4- Nombre d'itérations à température constante **59**
3.2.5- Critères d'arrêt **59**

3.3- La recherche tabou 61

3.3.1- Principe de base **61**
3.3.2- Mémoire à court terme **61**
3.3.3- Mémoire à long terme **62**
3.3.4- Critère d'aspiration **62**
3.3.5- Critère d'arrêt **63**
3.3.6- La recherche taboue de Hu **64**
4- Approche proposée **65**
4.1- Problème job shop **65**
4.1.1- Formulation du problème **65**
4.1.2- Allocation de tâches **66**

4.1.3- Application des méta-heuristiques pour l'ordonnancement de la cellule **67**
4.2- Implantation de l'algorithme génétique **68**
4.3- Implantation du Recuit Simulé **75**
4.4- Implantation de la Recherche Tabou **77**
5- Conclusion **78**

CHAPITRE IV : Application

1- Introduction **80**

2- Présentation de la cellule de production pilote **80**
2.1- Le modèle d'un système de production **80**
2.2- Présentation de notre atelier de fabrication **81**
3- Modélisation de la cellule de production pilote **82**
4- Résultats et commentaires **88**
4.1- Influence des paramètres des algorithmes **96**
4.2- Comparaison des méta-heuristiques **97**
5- Conclusion **97**

Conclusion générale **99**

Références bibliographiques **102**

Annexes **107**

INTRODUCTION GÉNÉRALE

INTRODUCTION GÉNÉRALE

Le développement technologique en perpétuelle croissance a complètement modifié nos modes de consommation. En effet la durée de vie commerciale d'un produit a été réduite considérablement, par l'arrivée rapide sur le marché de produits concurrents ou équivalent, de moindre coût et de qualité souvent supérieure. Cette évolution a contribué à créer de plus en plus de nouveaux besoins des consommateurs.
Ainsi, pour pérenniser et développer leurs activités, les entreprises sont dans l'obligation de s'y adapter. Cette adaptation passe par une plus grande flexibilité, tant sur le plan organisationnel, managérial que technologique. Il s'agit alors non seulement d'être en mesure de répondre à une demande immédiate ou urgente, sur un ou une famille de produits et/ou de services donc d'être réactif, mais aussi de pouvoir provoquer ou susciter le besoin, et donc d'être proactif.
Les systèmes de productions flexibles représentent des dispositifs capables de prendre en charge quelques unes de ces questions, notamment celle relative à la flexibilité de la production. Ils se caractérisent par la polyvalence des moyens de production et de transport et sont sujets à différents aléas dus notamment à un environnement changeant.
La conduite de ces systèmes (cellule de production flexible) est confrontée à de multiples contraintes nées de la complexité de l'environnement (évènements aléatoires notamment les pannes, la disponibilité des ressources, des objectifs variables, etc…). Ces difficultés génèrent des conflits par rapport aux ressources communes donc partageables. Il s'agit alors de trouver le meilleur ordonnancement et la meilleure affectation possible des ressources aux différentes tâches à réaliser, dans des temps très courts, par rapport à la dynamique du système. On choisit le plus souvent le temps global de réalisation et/ou le taux de production L'arbitrage influe directement sur les performances de ce système. C'est pourquoi, il y a lieu de bien choisir les règles de décision. L'approche par les méthodes exactes est vite abandonnée pour des problèmes réels au vu de la combinatoire générée. On se rabat alors sur des approches heuristiques ou méta heuristiques, qui proposent des décisions acceptables et quelque fois optimales, en un temps de traitement raisonnable.
L'opération de conduite s'appuie sur un modèle du procédé. Les systèmes de production flexible (Flexible Manufacturing System, FMS), appartiennent généralement à la classe des systèmes dynamiques à évènements discrets (SDED), même si l'on peut y déceler des parties continues. Parmi les outils de modélisation disponibles pour prendre en charge cette famille de systèmes, il y a les Réseaux de Pétri, qui se déclinent en plusieurs variantes. L'aspect continu et discret de ces systèmes pourra être modélisés par le réseau de Pétri hybride, par exemple. Les réseaux de Pétri sont considérés actuellement comme l'outil le mieux adapté pour la modélisation de ces systèmes. Il prend en charge la synchronisation, le parallélisme des opérations et offre à l'exploitant à tout moment un état réel du processus modélisé.

La problématique abordée par cette thèse traite des questions liées à la recherche de la meilleure décision d'affectation et d'ordonnancement, au sens d'un ou de plusieurs critères, d'un ensemble d'opérations à différentes ressources. Le champ d'application est une cellule de production flexible.

Ce problème connu, et étudié depuis longtemps, intéresse toujours une partie de la communauté scientifique. La raison est qu'il n'y a pas de solution exacte dans un contexte industriel, où le temps de prise de décision est un facteur essentiel, au vu de la dynamique des processus. Alors, en fonction des développements de nouvelles techniques ou approches, s'appuyant particulièrement sur l'intelligence artificielle, il est proposé dans la littérature

spécialisée des solutions intéressantes et qui répondent aux soucis des industriels. Certes il s'agit de solutions non optimales mais offrant un bon compromis entre le temps de réaction et la satisfaction du critère choisi. C'est dans ce contexte que nous inscrivons notre contribution.

Ces approches s'appellent les heuristiques. Elles comportent souvent plusieurs paramètres contrôlant les différents opérateurs et leur influence sur les processus stochastiques. L'efficacité d'une heuristique dépend du choix de ses paramètres de contrôle. Ce réglage est complexe, surtout quand le nombre de paramètres est élevé et quand la plage de variation de chacun de ces paramètres est étendue.
Enfin pour un jeu de paramètres de contrôle donnés, l'aspect stochastique fait que les résultats varient d'une exécution à une autre.
Les méta-heuristiques permettent d'aider à la conception de méthodes heuristiques pour un problème d'optimisation. Elles sont adaptables et applicables à une large classe de problèmes. Elles sont représentées essentiellement par les méthodes de voisinage comme le recuit simulé, la recherche tabou, et les algorithmes évolutifs comme les algorithmes génétiques.
Une méta-heuristique est constituée d'un ensemble de concepts fondamentaux (par exemple, la liste taboue et les mécanismes d'intensification et de diversification pour la méta-heuristique tabou). Grâce à ces méta-heuristiques, on peut proposer aujourd'hui des solutions approchées pour des problèmes d'optimisation classiques de plus grande taille et pour de très nombreuses applications qu'il était impossible de traiter auparavant. On constate, depuis ces dernières années, que l'intérêt porté aux méta-heuristiques augmente continuellement en recherche opérationnelle et en intelligence artificielle.

La thèse présente un état de l'art des techniques évolutionnistes et un ensemble d'application sur un système de production. Cette thèse s'intéresse aux principales méta-heuristiques : les méthodes de voisinage, les algorithmes évolutifs. Nous possédons bien des comparaisons entre méta-heuristiques sur différentes classes de problèmes mais nous ne savons généralement pas prévoir l'efficacité d'une méta-heuristique sur une instance donnée. Pour améliorer cette situation, une tendance se dessine qui consiste à favoriser les tests descriptifs : ceux-ci consistent à identifier les facteurs supposés influencer les performances et à effectuer des comparaisons en faisant varier ces facteurs.
Après l'introduction d'un état de l'art sur les problèmes imposés dans les systèmes de production flexible, on s'intéresse au problème d'ordonnancement d'un atelier, dans lequel l'objectif était d'optimiser un critère donné, donc une étude s'appuie sur l'analyse, la modélisation, l'optimisation et la simulation d'ateliers apporte une aide rapide et efficace à la résolution du problème posé (chap 1). Le système de production que nous étudions est composé d'une partie discrète et d'une partie continue. Nous présentant la modélisation de ce système de production à l'aide du modèle basé sur les réseaux de pétri hybrides (RdP continu pour modéliser l'aspect temps employer au système et RdP discret pour modéliser les moyen de production) illustré au (chap 2). Deux grandes classes de méthodes sont présentées : les méthodes déterministes et les méthodes stochastiques. Les caractéristiques principales de chaque classe, et leurs points forts et leurs points faibles sont montrés. Nous détaillons les différentes méta-heuristiques appliquées à notre système dans le but d'optimisation, nous analysons ces méthodes pour dégager quelques principes fondamentaux, certaines améliorations sont proposées pour la résolution de notre problème (chap 3). L'intérêt et l'efficacité des méthodes décrites dans le chapitre précédent serons testées sur le problème étudié, une comparaison des performances de ces méthodes est établie, ainsi une proposition des pistes pour guider le choix d'une méta-heuristique en pratique est ajoutée (chap 4). Pour conclure, nous discutons enfin les limitations des méta-heuristiques et préconisons quelques voies de recherche.

CHAPITRE 1

Les systèmes de production flexible

Chapitre 1 Les systèmes de production flexible

1. Introduction

Pour s'adapter à un environnement industriel, les entreprises doivent fixer des objectifs notamment:

- une grande souplesse de production obtenue par une flexibilité accrue d'équipements qui associés à l'informatique (automates programmables, robots, cellules et ateliers flexibles) sont pris en charge par un système de gestion de production performant. D'où la nécessité d'un modèle destiné à fournir un support pratique d'analyse pour la prise de décision face à des perturbations.
- une diminution des arrêts machine par l'utilisation optimale des moyens de production tout en exigeant une grande disponibilité,
- une bonne qualité du produit intégrant les incertitudes sur les paramètres d'entrée,
- une réduction des coûts de production, en diminuant les coûts de maintenance toute en diminuant les arrêts.

Afin de réaliser ces objectifs, une entreprise doit s'adapter aux changements apportés aux systèmes de production et savoir anticiper et réagir face à une incertitude ou une complexité. Les systèmes de production manufacturière sont constitués de procédés à contraintes de temps. La commande de ces procédés doit garantir, pour chaque opération, les spécifications sur les durées opératoires afin d'assurer la conformité des produit et le taux de production. Il en résulte un besoin de méthodologies et d'outils pouvant aider les décideurs de mieux respecter la notion de performance et à évaluer ce dernier lors de la conception, de l'exploitation, et de modelage d'un système de production.

La flexibilité recherchée pour un atelier de production a comme effet de créer de nombreuses situations de conflits nécessitant un choix pour leur résolution, ce qui permet d'augmenter les possibilités (solutions) du système. Ceci ne peut être envisagé qu'à l'aide d'un outil de simulation qui s'avère souvent indispensable dans ce contexte.

En effet, ainsi que l'écrit Jacquet Lagrèze, «la simulation ne permet pas de trouver une solution optimale, mais seulement une solution jugée satisfaisante pour illustrer la célèbre et importante distinction faite par Simon » [47]. De son côté, Anderson [9] définit la simulation comme une manipulation numérique d'un modèle symbolique conçu pour représenter l'évolution d'un système dans le temps. Ainsi qu'il le souligne par ailleurs, les processus mis en œuvre sont souvent de nature complexe et aléatoire, et les problèmes traités par cette approche sont typiquement peu structurés.

La simulation est un processus qui consiste à distinguer quatre phases distinctes:

- Concevoir un modèle du système (réel) qui permet la représentation du système étudié, c'est une étape essentielle pour la simulation
- la programmation, qui permet de fournir des résultats testé sur se système
- Mener des expérimentations sur ce modèle (et non pas des calculs),
- Interpréter les observations et les résultats fournis par le déroulement du modèle et formuler des décisions relatives au système.

Donc pour étudier le comportement d'un système réel on construit son modèle. Les systèmes réels sont généralement caractérisés par des aspects continus ou discrets. Les systèmes à double composante comportementale sont appelés Systèmes Hybrides. Ce qui fait apparaître deux parties dans la modélisation de tels systèmes: le modèle continu et le modèle discret. Le modèle continu traduit la physique de chacune des opérations. Tandis que le modèle discret décrit les trajectoires de la matière (enchaînement d'opérations) au cours de son passage dans l'unité de production et en fonction du temps, un changement de phases entraîne une discontinuité lors d'un passage d'un état à un autre. Le résultat fournit par ce modèle est une suite de changements d'état exprimée en termes d'activités et d'événements.

Ces événements sont connus soit à l'avance, soit liés à des variables d'état du système, et confèrent aux procédés des caractéristiques discrètes qui dans le cadre de la simulation dynamique, sont couplées à des étapes continues pour former des Systèmes Dynamiques Hybrides.

La simulation hybride peut alors se révéler une aide précieuse dans la mise en place des produits de fabrication. Pour décrire ces produits, plusieurs formalismes peuvent être utilisé: Grafcet, machines à états, réseaux de Pétri… Les plus élaborés sont les réseaux de pétri. Ils permettent de décrire le comportement d'un système de production vu leur capacité de modéliser les systèmes dynamiques hybrides toute en utilisant les RdP hybride.
Des méthodes d'optimisation ou d'apprentissage associé au modèle des systèmes permettent de résoudre des problèmes auxquels les méthodes classiques n'apportent pas de réponses satisfaisantes, et qui permet de fournir une solution assez bonne pour la résolution d'un problème d'optimisation.

Ce travail de doctorat s'inscrit dans ce champ et s'intéresse au problème d'ordonnancement d'un atelier. Il nécessite la prise en compte de plusieurs critères (minimisation du temps moyen de séjour des produits, minimisation des retards des produits sur les ressources de production, minimisation des stockages de produits en durée et nombre, maximisation des taux d'utilisation des équipements, qualité des produits,….). Nombreuses études ont été orientées vers ce type de problème d'ordonnancement dans lequel l'objectif était d'optimiser un critère donné.

Cette démarche qui s'appuie sur l'analyse, la modélisation, l'optimisation et la simulation d'ateliers afin d'apporter une aide rapide et efficace à la résolution du problème posé.
Dans ce travail de recherche, nous nous sommes intéressés plus particulièrement aux problèmes d'optimisation en environnement dynamique qui sont actuellement d'un intérêt particulier. Dans ce premier chapitre, nous précisons le contexte de l'étude. Les Système Flexibles de Production Manufacturière (SFPM) y sont présentés ainsi que leurs caractéristiques principales. Un état de l'art rapide sur les systèmes de production manufacturière, les travaux concernant l'ordonnancement, les problèmes d'optimisation, seront établi dans les lignes qui suivent pour positionner notre contribution.

2. Les systèmes de production

2.1 Historique

L'espace d'un système de production est constitué de produits, de moyens de production et d'opérateurs en interdépendance entre eux. Le contrôle et l'exploitation des liens entre eux sont les activités des algorithmes de pilotage. La tendance actuelle s'oriente vers des systèmes produisant des petites et moyennes séries capables de s'adapter aux changements de production, afin de répondre aux exigences de diversité, de productivité et de qualité, exigées par la concurrence actuelle du marché. Ces systèmes, appelé Systèmes Flexibles de Production Manufacturière (SFPM). Le but des systèmes de production est de fabriquer et de transformer une matière première en produit fini ou semi fini. Le schéma simplifié des systèmes de production est représenté par la figure 1.

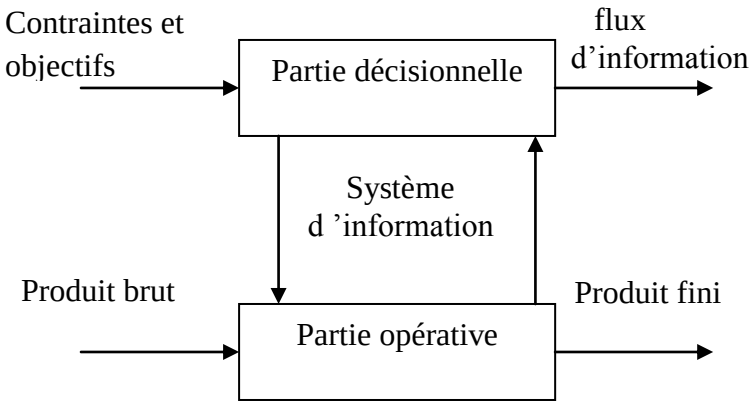


Figure 1.1: Schéma d'un système de production

Deux parties composent les systèmes de production :
- la partie décisionnelle du système: représente le système de décision (la commande), cette partie regroupe l'ensemble des moyens logiciels, matériels, et les informations permettant la gestion du système. Les décisions portent sur les actions à mener par le procédé sur le produit, ces actions sont élaborées à partir des informations (comptes-rendus) réunies sur le procédé ou sur le produit. Le système d'information traite et mémorise les informations, les données et les décisions.
- La partie opérative : représente le procédé de production, ou le système physique regroupe l'ensemble des organes physiques ayant pour but la transformation des matières premières. Selon la nature de flux de produit [69], la partie opérative peut être de trois types:

♦ discrète: le produit représente une matière solide et discrète, et constitue un ensemble de matière localisable individuellement.
♦ continue: le produit est un flux continu de matière.
♦ mixte ou hybride: le processus est à la fois discret et continu.

La transformation passe de produit brut (qui représente la matière première avant transformation) en produit finis (qui représentent les entités à transformer), et les entités transformées doivent répondre aux exigences de qualité, de coût, et de délai, et qui doivent satisfaire les caractéristiques illustré dans le paragraphe suivant.

2.2 Les systèmes de production classiques

Le système de production est décomposé en deux parties complémentaires:
La première partie est la partie opérative, qui désigne le flux matériel, elle est constituée des entités appartenant à trois populations: la population des produits, la population des moyens de production et la population des opérateurs de production [72], elle a pour fonction de fabriquer (qualité, quantité, délais) des produits finis à partir des produits bruts avec les moyens de production et les opérateurs de production.
La deuxième partie est la partie conduite, qui traite le flux informationnel, elle a pour fonction d'élaborer les ordres nécessaires à la partie opérative.

Le rôle principal de la fonction conduite des systèmes de production est de diriger, guider, et piloter, de manière à assurer la pertinence et la cohérence du système dans un environnement donné.

2.3 Nécessité de la flexibilité

Dans un monde qui va vite, l'entreprise est soumise à l'évolution du marché. Aussi et pour y faire face, le système de production est orienté vers des familles de produits et non vers un

seul type de produit. Cette aptitude exige une grande flexibilité tant matérielle qu'organisationnelle. La flexibilité est l'aptitude d'un système de production à fabriquer plusieurs variantes de produit, et se caractérise aussi par sa capacité d'adaptation à la production de nouveaux produits. Pour ce faire une bonne conduite du système est alors exigée afin de rapporter un meilleur coût de production face aux variations des systèmes flexibles. Cela nécessite, un système de conduite élaboré, afin de faire face aux différentes contraintes.

a. Fondements du pilotage industriel

Piloter une partie de l'entreprise ou toute l'entreprise, revient à organiser et entamer des actions pour corriger en permanence l'écart entre l'objectif visé et le résultat atteint. On trouve les fondements du pilotage dans des domaines d'automatique et le contrôle de gestion. Le pilotage partage un certain nombre d'analogies avec la commande en automatique. On parle ainsi :

. d'objectif (consigne) pour caractériser les sorties espérées du système piloté (commandé),

· de variables essentielles (variables d'état) pour caractériser l'état du système,

· de variables d'action (grandeurs de commandes) pour caractériser les entrées sur lesquelles agir pour modifier l'état du système,

· de système de pilotage (système de commande) pour fixer, en fonction de l'écart enregistré, les grandeurs de commande du système piloté (commandé),

· des aléas (perturbations) pour caractériser les modifications de l'état du système hors des actions du système de commande,

· de performance (d'erreur) pour caractériser la correspondance entre les sorties réelle et les sorties espérées.

Le système piloté contient toutes les activités qui assurent la transformation des produits effectué sur un ensemble des moyens et flux qui élaborent les plans d'action en fonction des objectifs et des performances du système piloté. D'une façon générale «un plan d'action décrit les actions à mener et les moyens nécessaires pour atteindre les objectifs quantitatifs et qualitatifs d'une unité de travail» [34], il est aussi défini comme «le groupe d'activités directement responsable de la gestion de la transformation d'ordres de fabrication planifiés en pièces sorties de l'atelier» [54].

Deux types de pilotages peuvent être distingués :

-Le pilotage industriel : oriente les moyens disponibles en vue de l'amélioration de la performance. Il faut alors définir les objectifs des différentes activités, exécuter et enchaîner ces activités afin d'atteindre l'objectif stratégique fixé à l'entreprise.

-le pilotage technique : le système physique concerné peut être les ateliers, les lignes de production, les processus opérationnels... et doit atteindre des objectifs en termes essentiellement de quantité, délai, et qualité.

Donc pour conclure on peux dire que le pilotage consiste en un ensemble d'actions effectuées pour diriger et guider de manière à assurer la pertinence et la cohérence du système en présence de perturbations afin de réaliser un ensemble d'opérations sur un flux de produit. Deux types de perturbations peuvent être distingués: les perturbations externes et les perturbations internes.

Les perturbations externes sont définies, dans le cas général, comme étant les perturbations affectant l'entrée d'un système ou d'un procédé. Il s'agit des changements de gamme, des changements d'objectifs pour une gamme, de la variation des instants d'arrivée des produits.

Les perturbations internes sont celles qui affectent la constitution du procédé ou son fonctionnement: les pannes des machines, les changements des durées opératoires, la suppression de ressources pour maintenance préventive…

L'entité du système de pilotage est identifiée par un système de décision qui permet, de préparer les actions ou exécuter les ordres, qui pilotent localement chaque entité sur la base d'un objectif déduit de l'objectif global.

b. système d'aide à la décision

La partie conduite d'un système de production doit disposer d'un modèle de la partie opérative du système, afin de réaliser son objectif. Les produits sont modélisés par un ensemble d'opérations nécessaires à leur réalisation. L'ajout d'un système de décision est nécessaire, permet l'affectation de chaque opération un équipement physique capable d'accomplir cette opération, et de faire exécuter par le système physique l'ensemble des opérations de fabrication qui lui sont affectées toute en répondant aux objectifs de production fixés doivent être respecté au mieux, en satisfaisant les contraintes temporelles et de coûts.

Ainsi il permet de définir les décisions dans lesquels il précise les opérations à accomplir, les grandeurs sur lesquelles il est possible d'agir, ainsi que les méthodes à suivre.

Afin de gérer le déroulement des opérations sur les moyens de production, la conduite doit respecter les contraintes techniques illustrées ci-dessous :

- une machine exécute au plus une opération à la fois (contrainte d'exclusion mutuelle) ;
- une opération commencée sur une machine doit être terminée sans interruption sur la même machine (contrainte de non-préemption) ;
- la fabrication d'une pièce nécessite un ensemble d'opérations qui doivent être effectuées dans un ordre spécifique (contrainte de précédente).

L'aide à la décision est indispensable pour résoudre des conflits pouvant apparaître, par l'allocation des ressources limitées, par la rechercher des solutions dans un espace globale. Donc l'ordonnancement est la fonction qui décide quoi faire, quand et où, il est dynamique lorsqu'il s'intéresse à un environnement changeant, où il peut y avoir, des retards d'arrivées sur ressource, insertion de nouvelles tâches, apparition des pannes sur machines...

3. Les systèmes de production flexible

3.1 Définition La description du fonctionnement d'un système de production consiste sur la définition de certaines grandeurs caractéristiques et à l'établissement des relations mathématiques qui gouvernent l'évolution de ces grandeurs. Un modèle modélisant le comportement du système doit être établi. Les concepteurs utilisent dans leurs modèles deux types de variables: des variables continues et/ou des variables discrètes et ceci selon le type de variables utilisées dans le modèle. Le système sera alors qualifié de « continu », de « discret », ou de « hybride » si son modèle requiert les deux types de variables simultanément.

3.2 Caractéristiques des systèmes de production

L'entreprise est souvent confrontée à des changements importants et quotidiens mettant les systèmes de production face à des contraintes et investissement qui ont généré une importante évolution de la fonction production, est cela est due:

- à une concurrence augmentée
-aux nouvelles technologies
-à l'amélioration continue de la qualité
-à la recherche de coûts minimums

La situation économique dans lequel les entreprises évoluent nécessite des systèmes de production basés sur des nouvelles caractéristiques, telles que la flexibilité, la réactivité, proactivité et robustesse.

a. Flexibilité

La flexibilité d'un système, est une propriété qui recouvre généralement deux aspects complémentaires mais distincts [31]:
-un aspect interne lié à une capacité de changement, de déformation, et à une variété d'états possibles;
-un aspect externe lié à une capacité d'adaptation à des modifications de l'environnement, et à des perturbations.
Le premier type de flexibilité est lié à la nature même de l'objet alors que le second concerne les interactions avec son environnement. Les auteurs utilisent respectivement les notions de flexibilité statique et flexibilité dynamique pour désigner ces flexibilités interne et externe.

• La flexibilité statique pouvant avoir plusieurs formes [15], nous distinguons:
-la flexibilité temporelle, concernant les dates de début des tâches dans le temps; elle est souvent implicite en ordonnancement. En effet, lorsqu'un ordonnancement est déterminé, tout en satisfaisant les contraintes de ressource, impose le respect des performances temporelles.
-la flexibilité séquentielle qui autorise une modification de l'ordre des tâches sur les ressources; elle impose de manipuler non pas un seul ordonnancement, mais une famille d'ordonnancements.
-la flexibilité sur les affectations qui, dans le cas où une tâche peut être réalisée par une ressource à choisir où être associée à plusieurs modes d'exécution, de laisser libre certains choix d'affectation, par exemple suite à la défection d'une ressource, une tâche est déplacée de la ressource défectueuse vers une autre de façon dynamique. Elle peut aussi être associée à un voisinage de solutions pouvant être examiné lors de l'exécution, ou à une famille d'ordonnancements, sans privilégier un ordonnancement en particulier.

• La flexibilité dynamique d'un ordonnancement réside dans la capacité de l'ordonnancement réactif à adapter l'ordonnancement déterminé, en présence de perturbations, de façon à satisfaire au mieux les exigences propres à ce niveau de décision.

Plusieurs autres types de flexibilité [3] ont été mis en évidence suivant leurs incidences sur l'objectif qui est le produit fini, et sur les moyens de production permettant la réalisation de ce produit.
•flexibilité de produits : offre la possibilité d'une reconfiguration du système pour la prise en compte d'un nouveau produit ou famille de produits permettant ainsi un gain de productivité ;
•flexibilité de mélange: c'est la possibilité de produire simultanément un ensemble de produits ayant des caractéristiques de base communes; cette flexibilité peut être mesurée par le nombre de produits différents qui peuvent être fabriqués simultanément ;
•flexibilité de quantité: il s'agit de la capacité du système à faire face aux fluctuations de la quantité des produits à fabriquer en modifiant les rythmes, ainsi que les temps de passage et d'engagement des outils;
•flexibilité d'ordre des opérations: permet de changer l'ordre des opérations en cours de production (ce qui suppose l'existence d'une gamme principale et des gammes secondaires) ou de choisir la destination suivante après chaque opération ;
•flexibilité d'expansion: autorise une extension et une modification de l'architecture du système;

•flexibilité des ressources: c'est la capacité des ressources à effectuer plusieurs tâches élémentaires et de permettre la reprogrammation.
Le contrôle très élevé des ressources technologiques relevant de ce système à cause de leur surdimensionnement à l'installation entraîne une baisse de productivité. Il faut donc faire un compromis entre flexibilité et productivité.

b. Réactivité

Le plus important pour le client est de recevoir sa livraison dans les délais impartis et ce quel que soit le carnet de commande. Afin de satisfaire une telle exigence, ceci impose que le système de production doit d'être réactif, c'est-à-dire capable de répondre rapidement et économiquement à un changement (fabrication multi-produit, introduction d'une commande urgente, modification d'une norme etc.) ou à une perturbation. Ces perturbations peuvent provenir soit du système de production (défauts d'alimentation, défauts de réalisations d'une tâche, pannes des machines, déchets) soit de son environnement (approvisionnements des matières premières).

Il faut donc adopter et disposer d'une excellente connaissance sur la composition interne du système, son environnement, ses interactions, ses aspects technologiques, humaines, opérationnels, organisationnels, décisionnels, et économiques, selon un horizon temporel d'évolution.

Afin d'assurer la réactivité du système de production, trois fonctions annexes s'avèrent nécessaires :
•une fonction d'observation : collecte les variables nécessaires au suivi afin de connaître l'état courant du système (disponibilité et état des produits, disponibilité et état des moyens de production) ;
•une fonction de surveillance : détecte (suite au résultat d'une observation) et interprète les écarts et les changements entre le plan prévisionnel et le plan courant par anticipation ;
•une fonction de correction : tente à tout instant de corriger les écarts entre ces plans, ce qui implique un ordonnancement dynamique.

c. Proactivité

[41] La complexité croissante des processus de production et l'évolution rapide de l'environnement conduisent à considérer comme nécessaire une adaptation permanente, dans un environnement où les perturbations sont toujours présentes. La réactivité est donc nécessaire mais elle n'est pas suffisante, et les systèmes de production doivent présenter une nouvelle propriété : la proactivité.

La proactivité d'un système de production est définie comme sa capacité à anticiper les changements d'état, à adapter ses règles de fonctionnement, et à se réorganiser.
Notre approche sera de vérifier à priori la capacité du système à absorber ces perturbations.

d. Robustesse

La robustesse d'un système de fabrication est la capacité à préserver certaines propriétés d'intérêt face à des perturbations. L'acquisition de ces propriétés est liée à une réorganisation importante du système de production existant, particulièrement au niveau de la conduite du système par la prise en compte des nouvelles approches. Ce qui nous emmène à la bonne conduite des systèmes de production flexible.

Différentes définitions de la robustesse ont été proposées, chacune des définitions répond à un objectif donné dans un domaine donné. Dans le domaine des statistiques, la robustesse est caractérisée comme étant une insensibilité à toutes les dérives par rapport aux hypothèses [46]. En recherche opérationnelle, dans le contexte d'aide à la décision, B. Roy propose le

concept de robustesse qui vise à élaborer des éléments de réponse à un problème auquel est confronté un décideur en tenant compte des incertitudes sur les paramètres du problème [73]. En ce qui concerne la conception du système, la robustesse est caractérisée par l'ajustement des paramètres du système, afin de garantir un niveau de performance acceptable.

4. Problématique de la conduite de ces systèmes

Un problème de conduite consiste à ordonnancer et affecter des tâches aux ressources. Le problème d'ordonnancement consiste à organiser dans le temps la réalisation de tâches, compte tenu de contraintes temporelles (délais, contraintes d'enchaînement,...) et de contraintes portant sur l'utilisation et la disponibilité de ressources requises. Un problème d'ordonnancement peut être défini de la manière suivante : il s'agit de contrôler l'activité d'un ensemble de ressources disponibles en quantité limitée, en gérant les conflits d'utilisation que pose la réalisation d'un ensemble d'activités sur un horizon de temps généralement imposé.

Donc il faut programmer au mieux (c'est-à-dire employer la définition de la fonction d'optimisation) l'ensemble des tâches à réaliser, en leur allouant les ressources nécessaires et en fixant leur date de début.

La méthode utilisée pour l'élaboration d'un ordonnancement vise souvent à satisfaire un ou plusieurs objectifs (coûts, délais, qualité), exprimés au sein d'un ou plusieurs critères de performance. La programmation de cette méthode se base sur des variables que sont les *tâches,* les *ressources,* les *contraintes* et les *objectifs.*

4.1 Environnement incertain

La fonction de conduite d'un système de production peut avoir plusieurs objectifs à réaliser toutes en veillant au respect des contraintes de temps et de ressource. Il peut s'agir de satisfaire des objectifs économiques, des objectifs temporelles, de gérer au mieux le risque en présence d'incertitudes dans un environnement incertain. Elle doit produire une solution admissible, c'est-à-dire celle qui satisfait l'ensemble des contraintes formulées, ou la solution produite doit non seulement être admissible, mais aussi permet de minimiser ou maximiser la valeur du ou des critères considérés.

Les critères d'évaluation numériques pour des approches d'optimisation peuvent être:
-soit liés au temps, comme dans le cas de la minimisation du temps total d'exécution (C_{max});
-soit liés aux ressources, comme dans le cas de la minimisation de la quantité de ressources nécessaires à la réalisation des tâches ou la charge de chaque ressource;
-soit liés aux coûts (de production, de transport, etc.) qu'un ordonnancement induit.

Dans le cadre de cette thèse on s'intéresse beaucoup plus à la minimisation du temps total d'exécution (C_{max}), et à la minimisation de la charge de chaque ressource toute en réduisant le temps d'accès des taches a la ressource. Un ordonnancement déterministe est construit hors-ligne. Cet ordonnancement est ensuite utilisé en ligne pour guider les décisions. Il est alors probablement adapté en temps réel pour tenir compte des perturbations.

4.2 Ressources à aptitudes multiples

Une ressource k est un moyen humain ou technique, disponible en quantité limitée, destiné à être utilisé pour la réalisation de plusieurs tâches. Une ressource est dite renouvelable si après avoir été utilisée par une ou plusieurs tâches elle est de nouveau disponible, dans le cas contraire elle est dite consommable (Ex. matières premières). La notion d'état de ressource disponible ou non est également utilisée afin de prendre en compte des contraintes technologiques liées à son utilisation. Un changement d'état de la ressource peut être produit

par le début ou la fin de certaines tâches du problème, ou par l'occurrence d'événements externes incontrôlables (panne sur machine, maintenance, prévention...).

4.3 Autres contraintes

La notion de contrainte est relative à un ensemble de variables de décision, elle exprime une limitation sur les domaines de valeur de ces variables. Il y a deux sortes de variables : le premier type de variables liées aux décisions de localisation des tâches dans le temps (variables d'ordonnancement); le deuxième type de variables est liées aux décisions d'affectation des tâches sur les ressources (variables d'affectation). Dans le cadre de cette thèse, nous nous intéressons à ces deux types de variables qui sont caractérisés par les contraintes temporelles et les contraintes de ressources.

-Les contraintes temporelles permettent d'exprimer les interdépendances temporelles entre tâches, qui caractérisent l'organisation d'une production. Ce type de contrainte est aussi appelé contrainte de précédence généralisée qui permet d'exprimer des intervalles temporels minimaux ou maximaux entre les variables.

-Les contraintes de ressources permettent de représenter à la fois les caractéristiques de consommation des tâches sur les ressources et les caractéristiques de disponibilité des ressources.

4.4 Nécessité d'un modèle de conduite

Plusieurs travaux ont été étudiés pour l'intégration du modèle de simulation dans une procédure d'optimisation. L'utilisation de l'outil modélisation est une voie importante de recherche pour la détermination de la robustesse dans la conduite des systèmes de production. On parle alors de deux types de robustesses : la robustesse externe, correspond à la capacité d'un système à faire face aux variations prévues ou imprévues, et la robustesse interne du système, correspond à sa capacité face à des variations internes. Elle consiste à garantir à priori les taux de production, les temps de séjour des produits dans le système. La nécessité d'un modèle du système est bien évidente.

Un modèle est une description de la structure du système physique, il permet une représentation comportementale ou fonctionnelle de chacun de ses composants [58].

Une représentation comportementale est constituée de relations de causes à effets entre diverses variables du système. Une représentation fonctionnelle est plus abstraite puisqu'elle ne s'adresse qu'aux objectifs prévus que le système physique doit remplir. Les niveaux comportemental et fonctionnel comprennent des relations entre des grandeurs physiques, et permettent de mettre en évidence la présence d'un événement anormal ou anomalie. Le niveau structurel s'appuie sur la structure réelle du système physique et décrit les interconnections entre ses différents éléments ou constituants.

4.4.1 Une classification des formalismes

La classification des formalismes a pour but de démontrer que certains aspects ne sont pas pris en compte de la même façon. Elle est dédiée à la spécification de la dynamique des systèmes en fonction de trois critères: la manipulation de variables continues ou discrètes, la représentation de l'espace continu ou discret et la présence du temps continu ou discret.
Afin d'établir cette classification, il est nécessaire de déterminer :

-les propriétés des systèmes:
Discret ou continu: cette propriété est à considérer selon le changement d'états, d'espace, le temps, etc. Dans ce dernier cas, le temps qui est défini comme une variable discrète ou continue n'implique pas les mêmes outils de modélisation.

- Modélisation:
Déterministes ou stochastiques: les transitions d'états sont défini soit de manière déterministe soit de manière stochastique.

4.4.2 Formalismes pour la modélisation structurelle

La modélisation structurelle est la représentation des entités formant le système et leur mise en relation, ou la description des variables caractéristiques des processus du système. Cette définition donne naissance à la modélisation fonctionnelle ou à contraintes et la modélisation déclarative.

La première forme identifie les processus ou les lois gouvernant le système à partir desquels elle identifie les variables caractéristiques. La modélisation comportementale décrit ensuite les relations fonctionnelles entre les variables (Ex. à l'aide d'équations différentielles).

La deuxième forme a pour objectif de décrire les entités et leurs relations.
Nous allons donc nous intéresser à cette deuxième forme en raison de l'approche de modélisation que nous avons choisie.
Des définitions des concepts d'entité et de relation qui sont à la base de la modélisation structurelle sont données ci-après.

a. Une entité

Une entité est un élément d'un système qui possède un état et un comportement. Cette définition est très proche de celle de système, mais la différence réside dans le fait qu'une entité est un système, mais tout système n'est pas une entité. Le système est appréhendé au travers de ces variables caractéristiques et de la dynamique de ces dernières. Pour conclure, nous proposons de définir une entité comme un élément discret d'un système.

b. Une relation

Une relation est un lien sémantique entre plusieurs entités. La notion de lien sémantique couvre principalement les aspects suivant:
– un lien d'interdépendance: deux entités se connaissent, elles pourront alors s'échanger des informations;
– un lien de composition: une entité est formée de l'assemblage de plusieurs entités, une relation hiérarchique est alors établie ;

4.4.3 Formalismes pour la modélisation dynamique

La modélisation dynamique est la description du comportement et les interactions des entités du système ou des contraintes sur les variables caractéristiques du système.
Ce qui forme des nécessités complémentaires entre la description des processus ou la description des lois. Cette partition des approches nécessitent les formalismes associés aux équations différentielles.

Les systèmes d'équations différentielles

Les équations différentielles sont l'outil le plus favorisé vu leur puissance d'expressions mathématiques et les outils formels qui les accompagnent. A partir d'un système d'équations différentielles, on peut déduire des propriétés de convergence ou de stabilité à l'infini sous certaines conditions, et de déduire l'évolution temporelle des variables composant le système, si ce dernier possède de bonnes propriétés.

La critique que nous pouvons énoncer envers les équations différentielles est qu'il n'existe pas toujours le bon théorème ou le bon outil pour le système que l'on construit.

Dans ce cas, il n'est plus possible d'en déduire quoi que ce soit sans passer par des outils d'analyse numérique qui procèdent par simulation numérique, ce qui implique l'utilisation des méthodes par simulation.

En ce qui concerne la représentation d'entités discrètes, il est très difficile, avec des outils mathématiques courants, de représenter des entités discrètes et de représenter les processus les mettant en jeu. Les équations différentielles sont, dans la grande majorité des cas, utilisées pour représenter des lois ou des contraintes entre variables du système. On ne peut pas faire de description directe des processus.

4.4.4 Formalismes hybrides ou mixtes

Les formalismes hybrides ou mixtes sont constitués de formalismes où la séparation entre structure et comportement est difficile. Parmi les modèles qui sont aptes de traiter ce type de formalismes, on mentionne ces deux exemples: les réseaux de Petri hybrides et les automates cellulaires.

a. Les réseaux de Pétri hybrides

Le réseau de pétri hybride est une extension des réseaux de pétri qui offre une représentation graphique associée à des variables continues et à des variables discrètes [41]. Donc les réseaux de pétri hybrides est une coopération de deux modèles : le modèle discret dont les places et les transitions sont discrètes, et le modèle continu dont les places et les transitions sont continues.

Il y a également deux types de transitions. Les transitions discrètes (équivalentes aux transitions ordinaires) qui ne peuvent être liées à des places continues que par des boucles élémentaires (la place est simultanément place d'entrée et place de sortie et avec le même poids sur les arcs). Et des transitions continues, sur lesquels un débit maximum est associé à chaque transition continue qui permet de faire varier de façon continue les marquages des places continues, donc le marquage s'écoule continûment de la place d'entrée vers la place de sortie.

Tant que la place d'entrée continue d'une transition continue est non vide, la transition fonctionne à son débit maximal. Si la place d'entrée est vide mais qu'elle est alimentée en amont par une transition continue, alors le débit est le minimum des débits des deux transitions.

Le pilotage de la partie continue par la partie discrète se fait par l'intermédiaire de boucles élémentaires reliant une place discrète à une transition continue. Les équations différentielles décrivant les flots passant d'une place d'entrée continue à une place de sortie continue ne seront actives que lorsque la place discrète contiendra au moins un jeton. Le changement d'états est guidé par le franchissement des transitions. Ce problème d'évaluation a déjà été étudié à plusieurs reprises en utilisant les réseaux de Petri comme modèle. Ils offrent, ainsi, la manière d'exprimer les changements d'états.

b. Les automates cellulaires

Les automates cellulaires décrits des systèmes où l'espace est une composante importante. La notion d'espace est prise ici au sens large: espace de variations d'une variable, ou espace topologique où des entités sont localisées. Ce formalisme hybride décrit la structure spatiale du système à l'aide d'une grille multidimensionnelle de cellules interconnectées et décrit le comportement local de chaque cellule de l'automate, en fonction de son voisinage.

5. Optimisation combinatoire

L'optimisation combinatoire est une étude qui combine différentes techniques de mathématiques discrètes et de l'informatique afin de résoudre des problèmes d'optimisation. Ce genre de problème apparaît dans des domaines divers, tels que l'industrie, le transport, les télécommunications,...

La résolution d'un problème d'optimisation combinatoire consiste à explorer un espace de recherche afin de maximiser (ou minimiser) une certaine fonction donnée sur un ensemble fini de solutions. Cet ensemble des solutions du problème, permet de représenter diverses structures combinatoires, comme par exemple des chemins, des cycles, des arbres, etc, dans les graphes.

Bien qu'il soit fini, l'ensemble de solutions d'un problème d'optimisation combinatoire peut avoir un nombre très grand de solutions. Les complexités (en taille ou en structure) relatives de l'espace de recherche et de la fonction (appelée fonction de coût, fonction d'adéquation ou fitness) à maximiser ou à minimiser (appelée optimum global) conduisent à utiliser des méthodes de résolution radicalement différentes. Une méthode énumérative consiste à l'énumération des solutions du problème ne peut donc être étudiée même si la puissance des calculateurs augmentait considérablement. Des outils plus performants ont été développés pour aborder ce type de problèmes, comme la programmation linéaire, les méthodes de la recherche opérationnelle,...

L'optimisation combinatoire se trouve ainsi au carrefour de plusieurs études telles que:

- la combinatoire,
- l'algèbre linéaire,
- la programmation linéaire,
- la programmation en nombres entiers,
- la théorie des graphes,
- la complexité des algorithmes,

Pour un certain type de problèmes d'optimisation combinatoire, on ne connaît pas d'algorithmes efficaces de résolution. Il y a donc peu d'espoir de pouvoir trouver une méthode efficace, ils sont généralement de type NP-difficiles. Pour ces problèmes, résoudre un problème d'optimisation consiste à trouver la ou les meilleures solutions vérifiant un ensemble d'objectifs et de contraintes définis par l'utilisateur.

Certains des problèmes d'optimisation peuvent être résolus par des méthodes exactes simples, qui permettent de trouver la (ou les) meilleure(s) solution(s) ou solution(s) optimale(s). Mais la plupart des problèmes étudiés en optimisation appartiennent à la classe des problèmes NP-difficiles [14]. Cette classe rassemble des problèmes pour lesquels on ne connaît pas d'algorithme rapide dont la résolution exacte n'est pas possible en un temps de calcul proportionnel à N^n, où N désigne le nombre de paramètres inconnus du problème, et n est un entier.

Pour résoudre ces problèmes d'optimisation combinatoire NP-difficiles, deux grandes classes de méthodes générales existent : les méthodes arborescentes et les heuristiques.
Les méthodes arborescentes (branch and bound methods), elles peuvent atténuer le manque des algorithmes polynomiaux pour des problèmes de taille moyenne, mais pour des problèmes de grande taille, leur durée d'exécution devient excessive. Dans ce cas, il faut se tourner vers les méthodes heuristiques.

En première approximation, on peut dire qu'une méthode déterministe est adaptée à un espace de recherche petit et complexe, et qu'un espace de recherche grand nécessite plutôt

une méthode de recherche stochastique (recuit simulé, algorithme génétique,...). Par la suite nous étudierons ces méthodes afin de choisir le mécanisme adéquat pour traiter le problème d'optimisation.

5.1 Choix d'une méthode

La nature des variables, des domaines de définition, et des critères à optimiser vont tous influencer sur le choix de la méthode d'optimisation qui va être appliqué dans le problème d'optimisation. Il existe deux grandes familles de méthodes d'optimisation : les méthodes déterministes et les méthodes stochastiques. On va donner dans les paragraphes qui suivent un aperçu général sur ces deux types de méthodes d'optimisation.

5.1.1 Les méthodes déterministes

Définition Les méthodes déterministes se caractérisent par une exploration systématique de l'espace de recherche.

Il existe de nombreuses méthodes d'optimisation déterministes : les méthodes locales qui assurent la convergence vers l'optimum de la fonction le plus proche de la solution courante en explorent son voisinage, et les méthodes globales qui s'attachent à faire converger la solution vers l'optimum globale de la fonction.

● Parmi les méthodes de recherche locale on trouve:
Les méthodes de gradient, la méthode multistart et la méthode du simplexe. D'une manière générale ces méthodes obéissent à l'algorithme suivant :

1. choix d'une première solution courante i admissible
2. génération d'une solution j dans le voisinage de i
3. si f(j) est meilleur que f(i) alors j devient la solution courante puis retour en 2)
4. l'algorithme se termine lorsqu'il n'y a plus d'amélioration de la solution courante.

a. les méthodes du gradient

Historiquement, les méthodes du gradient sont les plus anciennes. Elles permettent de résoudre des problèmes non linéaires [59] et sont basées sur une hypothèse forte: la connaissance de la dérivée de la fonction objectif en chacun des points de l'espace.

b. La méthode multistart

La méthode multistart effectue une recherche locale à partir de plusieurs points répartis dans l'espace de recherche. Avant de lancer le processus de recherche, il faut réaliser un maillage de l'espace de recherche. L'efficacité de cette méthode dépend de la bonne adéquation entre le maillage et la forme de la fonction objectif. Si le maillage est trop grand, la probabilité de trouver l'optimum global sera faible car certaines vallées ne seront pas traitées. Si le maillage est trop petit, la recherche globale sera inefficaces car plusieurs points vont être présents dans la même vallée et convergeront donc vers le même optimum.

c. La méthode du simplexe

L'intérêt principal de la méthode du simplexe par rapport aux deux précédentes est qu'elle ne nécessite pas de calcul de gradient. Elle est uniquement basée sur l'évaluation de la fonction. Cela la rend utilisable pour des fonctions bruitées.

Cette méthode locale effectue une recherche multidirectionnelle dans l'espace d'état [61]. Soit une fonction f à minimiser, on appelle *simplexe* de R^n tout ensemble $(x_0, x_1,.., x_n)$ de points de R tel *que* $f(x_0) \geq f(x_i)$ $\forall$ i $\in$ [1..n]. x_0 est donc le meilleur élément.

● Les méthodes de recherche globale qui sont la méthode de séparation évaluation (branch and bound), et la méthode de tunneling. Les méthodes homotopiques, l'algorithme A*, et la programmation linéaire sont également des méthodes d'optimisation globale.
Nous allons présenter brièvement les deux premières méthodes:

a. L'algorithme de séparation - évaluation : branch and bound

Le principe de cette méthode est de découper (branch) l'espace initial de recherche en domaines de plus en plus restreints afin d'isoler l'optimum global (principe de séparation). L'algorithme de recherche forme ainsi un arbre dont chaque nœud représente une partie de l'espace. Ensuite chaque nœud est évalué de façon à déterminer sa borne (bound) inférieure en fonction d'un critère d'évaluation. Si cette borne n'est pas meilleure que la solution courante alors la recherche sur ce nœud est stoppée, sinon la séparation continue.

b. La méthode du "Tunneling"

Cette méthode recherche l'optimum global d'une fonction en effectuant des recherches successives d'optima locaux telles que la valeur de la fonction s'améliore. L'algorithme se décompose en deux phases : une phase de recherche d'un optimum local et une phase dite de tunneling.

Si on se place dans le cas d'une fonction f à minimiser, la première phase peut utiliser une méthode de gradient pour trouver le minimum local $f^* = f(x^*)$.
La deuxième phase consiste à trouver un point x dans une autre vallée à l'aide d'une fonction T de tunneling du type : $T(x) = f(x) - f^* \leq 0$

5.1.2 Les méthodes stochastiques

Définition Les méthodes stochastiques sont caractérisées par :
- Un processus de création aléatoire ou pseudo-aléatoire des points dans l'espace d'état,
- Une heuristique permet de guider la convergence de l'algorithme.

Ces méthodes sont utilisées dans des problèmes où on ne connaît pas d'algorithme de résolution en temps polynomial et pour lesquels on espère trouver une solution approchée de l'optimum global.
Ces atouts principaux sont les suivants :
- Facilité d'implantation,
- Flexibilité par rapport aux contraintes des problèmes,
- Qualité élevée des solutions.

Pour un problème d'optimisation combinatoire, une méthode approchée ou heuristique est un algorithme qui a pour but de trouver une solution réalisable, mais sans garantir l'optimalité, tout en tenant compte des critères à optimiser et des contraintes du problème. Il existe un très grand nombre d'heuristiques selon les problèmes à traiter. Ces méthodes sont classées en trois classes [39].

a) Méthodes constructives : construisant une seule solution par une suite de choix partiels et définitifs (c'est-à-dire sans retours en arrière), on les appelle méthodes gloutonnes (greedy), elles cherchent à chaque itération à faire le choix le plus avantageux.

b) Recherches locales (local search) : on part d'une solution initiale et par transformations successives, on construit une suite de solutions de coûts décroissants. Le processus s'arrête quand on ne peut plus améliorer la solution courante.

c) Métaheuristique : c'est une méthode de recherche locale qui peut être piégée dans un minimum local. Elle peut s'échapper de ces minima locaux en construisant une suite de solutions, dans laquelle la fonction objectif peut temporairement augmenter. Ces méthodes, comprennent notamment la méthode du recuit simulé, les algorithmes génétiques, la recherche tabou, les colonies de fourmis, etc,… . Ces méthodes partagent ensemble être inspirées par le fait des analogies avec la physique (Ex. recuit simulé), avec la biologie (Ex. les algorithmes génétiques), ou avec l'éthologie (Ex. les colonies de fourmis). Elles ont au moins une partie stochastique qui permet d'offrir des possibilités face à l'explosion combinatoire, et capables de guider, dans une tâche particulière, une autre méthode de recherche spécialisée (Ex. une autre heuristique).

Ces méthodes ont aussi des inconvénients parmi lesquels on mentionne, les difficultés de réglage des paramètres de la méthode, et le temps de calcul élevé.

Dans le cadre de notre travail de thèse, nous avons étudié dans le chapitre III trois méthodes méta-heuristiques avec beaucoup plus de détails, afin de nous permettre de choisir la meilleure méthode qui offre de meilleurs résultats pour résoudre notre problème d'optimisation.

5.2 Les méta-heuristiques

Les méta-heuristiques sont des techniques d'amélioration progressive d'une première solution, elles permettent de traiter des problèmes de grande taille, tout en obtenant des solutions excellentes, souvent optimales [51].

Ces méthodes peuvent être classées en deux classes: les méta-heuristiques de voisinage, qui améliorent une seule solution en recherchant dans leur voisins (recherche tabou, recuit simulé,…), et les méta-heuristiques distribuées, qui traitent simultanément toute une population de solutions (algorithme génétique,…) [74].

En effet, il se traite à travers ces approches trois problématiques liées à :

- La résolution des conflits sur ressource de production entre différentes gammes de production par le respect des contraintes liées au système de production.
- Le partage de ressource, considéré comme un espace de recherche, en termes de charge de travail liée au nombre de produits en conflits, tout en minimisant le temps de réalisation.
- L'affectation des ressources entre différents produits, en minimisant les augmentations des temps d'attentes et la charge de travail des ressources.

Ci-après on va donner une brève description des méthodes méta-heuristiques qu'on va exploiter dans le chapitre III.

a. Le recuit simulé

La méthode du recuit simulé est basée sur une analogie avec le processus physique de recuit des matériaux cristallins. Ce dernier consiste à amener un solide à basse température après l'avoir élevé à forte température. Lorsque le solide est à une forte température, chaque particule possède une très grande énergie et peut effectuer de grands déplacements aléatoires dans la matière. Au fur à mesure que la température est abaissée, chaque particule perd de l'énergie et sa capacité de déplacement se réduit. Les différents états transitoires de refroidissement permettent d'obtenir des matériaux très homogènes et de bonne qualité.

Appliquée aux problèmes d'optimisation, on considère une solution initiale à partir de laquelle on recherche dans son voisinage une autre solution de façon aléatoire. L'originalité de cette méthode est qu'il est possible de se diriger vers une solution voisine de moins bonne

qualité avec une probabilité non nulle, ce qui permet à cette technique de sortir des optima locaux [29].
Les déplacements aléatoires de chacun des points vont être liés à une probabilité dépendante d'une variable *T* représentant la température du matériau. L'algorithme est appliqué en suivant les étapes suivantes :

Algorithme

1) A partir d'un point initial $x_{0,}$ on effectue un déplacement aléatoire (changement d'état).
2) Si le déplacement mène à x_1 tel que $f(x_1) < f(x_0)$ alors x_1 est accepté.
Sinon, il est accepté avec une probabilité $p = exp(-|\Delta f(x)| / kT)$.
3) effectuer un nouveau déplacement aléatoire vers un autre voisin
4) s'il n'y a plus de déplacement à effectuer, diminuer la température T selon la loi de décroissance et retourner à 2)

Avec $\Delta f(x)$ représente la distance de déplacement : x_1-x_0.
T est assimilé à une température décroissante au cours du temps.
k est une constante.

Au début de la simulation, les solutions qui représentent l'espace de recherche ont une grande capacité d'exploration de l'espace d'état car l'algorithme accepte des déplacements très importants. De nombreuses solutions n'améliorant pas leur valeur dans *f* sont acceptées car la probabilité *p* est grande. Au fur et à mesure que *T* diminue *(p* augmente), la capacité de déplacement d'une solution diminue et les solutions améliorant leur valeur sont de plus en plus nombreuse. Quand $T \rightarrow 0,$ seules les solutions améliorant leur valeur sont acceptées.

b. Les Algorithmes génétiques

Les algorithmes génétiques (AG) sont inspirés du phénomène naturel. Basée sur la loi de l'évolution énoncée par Darwin, seuls les individus les plus forts d'une population survivent par suite de leur meilleure adaptation à leur milieu naturel. Le principe est de reproduire l'évolution naturelle d'individus [29], génération après génération, en respectant les lois de l'hérédité et le concept de sélection naturelle, autrement dit « la survie de l'individu le plus fort ou le mieux adapté à l'environnement ».

En vue d'imiter les processus d'évolution observés dans la nature, la première adaptation des algorithmes génétiques aux problèmes d'optimisation combinatoire a été réalisée par Holland dans les années 70 [43]**.** Une analogie entre un problème d'optimisation et le phénomène naturel nous conduit de faire un échange structuré d'informations entre les individus pour reproduire de nouveaux éléments, qui sont meilleurs au sens du critère choisi, c'est-à-dire au sens de la fonction objectif à optimiser [38].
Pour un problème d'optimisation donné, une solution est un individu qui représente un point de l'espace de recherche (appelé aussi espace d'état), à chaque individu on associe la valeur du critère à optimiser (son adaptation), un ensemble de solutions correspond à une population d'individus. On génère ensuite de façon itérative des populations d'individus sur lesquelles on applique des processus de sélection, de croisement, et de mutation. La sélection a pour but de favoriser les meilleurs éléments de la population, le croisement et la mutation assurent une exploitation et une exploration efficace de l'espace de recherche. L'exploration désigne les processus visant à récolter de l'information sur le problème optimisé. L'exploitation (ou l'intensification) vise à utiliser l'information déjà récoltée pour définir et parcourir les zones intéressantes de l'espace de recherche. L'algorithme appliqué est illustré à partir des étapes suivantes:

Algorithme

1) Genèse: représente la première phase de l'algorithme dans laquelle la population initiale est construite d'une manière aléatoire, ou à travers des résultats issus d'autres techniques d'optimisation.

2) Evaluation: consiste à calculer la valeur de la fonction de coût pour chaque individu.

3) Sélection: choix des éléments les plus adaptés pour la formation de la nouvelle génération.

4) Recombinaison et mutation: phase de reproduction dans laquelle une nouvelle population est construite à partir des individus sélectionnés via des opérateurs de croisement et de mutation.

5) Arrêt: il s'agit d'un test de l'efficacité de l'algorithme à travers une valeur de la fonction objectif à atteindre, le nombre d'itérations, ou le temps d'exécution. La solution courante est prise quand ce test est vérifié, sinon l'algorithme passe à l'itération suivante qui commence à partir de l'étape 2).

c. La méthode Tabou

La recherche Tabou [37] est une méthode de recherche de voisinage. L'élément fondamental de la méthode tabou est l'utilisation d'une mémoire dynamique dite liste tabou (liste interdite) d'où le nom de la méthode. Elle permet d'enregistrer les informations adéquates des étapes de recherche précédentes. Cette liste empêche le blocage de la recherche sur un optimum local en interdisant de revenir sur des solutions précédemment visitées de l'espace d'état, ces solutions dites taboues c'est-à-dire interdite.

En pratique, la méthode enregistre dans une liste taboue T les propriétés des dernières solutions visitées. Dans l'itération qui suit la nouvelle solution représenté par la meilleure des voisines enlève la solution la plus ancienne de la liste (Ex. la pile). Dans d'autres cas, la méthode mémorise les mouvements réalisés plutôt que les solutions, ensuite on interdit les mouvements inverses.

Algorithme

1) générer une liste taboue T

2) A partir d'un point initial x_0, on effectue un déplacement aléatoire (changement d'état).

3) Si le déplacement mène à x_1 tel que $f(x_1)$ est la meilleure valeur de tous les voisins de x_0, alors ajouter x_1 à la liste tabou, x_0 reçoit la valeur de x_1

4) mise à jour de la liste tabou

5) si le critère d'arrêt n'est pas satisfait (nombre d'itérations à effectuer) alors retourner à 2)

La méthode Tabou est souvent utilisée avec des techniques dites d'intensification et de diversification. L'intensification est fondée sur l'enregistrement des propriétés des situations a priori de bonne qualité afin de les exploiter par la suite. La diversification cherche à diriger l'algorithme vers des régions de l'espace de recherche non encore explorées.

Remarque Dans le cas d'utilisation d'une méthode d'optimisation, le chercheur doit choisir entre maintien de la diversité et convergence de la recherche dans l'espace des solutions de façon à ne laisser aucune partie de l'espace des états non visitée, alors que le décideur doit choisir entre une méthode qui fournira un résultat optimal ou proche de l'optimum, et de faire le choix d'une méthode qui est capable de donner un résultat satisfaisant en un temps de calcul relativement court.

6. Simulation et analyse des résultats

Un système de production est constitué : d'un système opérant (partie physique), d'un système de conduite (partie commande), et d'un système d'informations reliant ces deux derniers, traversé par un flux d'informations (présence d'une pièce, état d'une machine) et un flux physique (matière première, pièces).

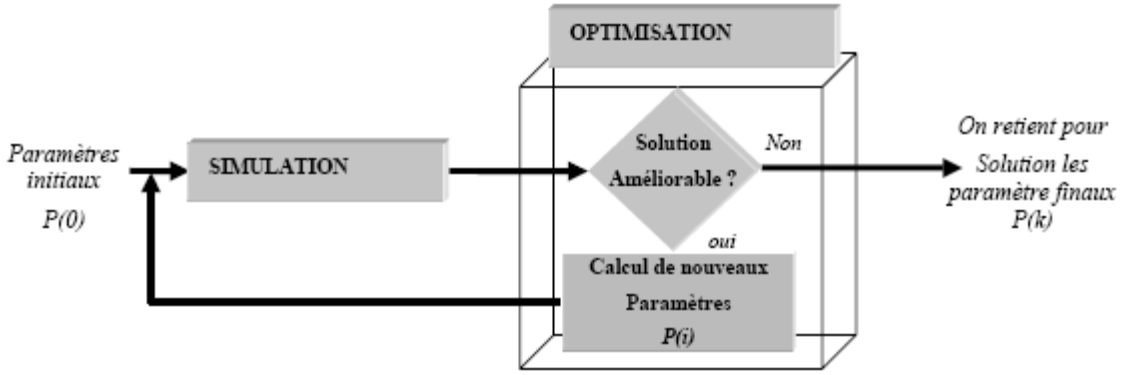


Figure 1.2 : Schéma d'un processus d'optimisation [65]

La représentation de la circulation des flux de produits est définie comme suit :

- Le flux est ralenti par des activités que mobilisent les ressources (après avoir attendu leur disponibilité) pendant un certain temps (durées opératoires, temps de transfert, ...),
- Le flux est contraint par des règles opératoires (gammes, contraintes technologiques),
- Le flux est dirigé par les règles de conduite (système de contrôle).

La circulation de produits exploite ces cordonnées pour calculer le makespan (intervalle de temps entre le début et la fin de la production des pièces) afin d'évaluer la performance qui se base souvent sur le temps de cycle (nombre moyen de pièces par unité de temps).

La modélisation et la simulation de la partie opérative du système permet de fournir les informations nécessaires au système de conduite.

Un modèle permet de décrire le fonctionnement du système (sa structure et son comportement dynamique) avec des détails nécessaires à la résolution du problème posé. [35] Il est utilisé quand on veut comprendre un système réel que l'on ne peut pas observer ou expérimenter directement, parce que le système n'existe pas encore ou parce que cela est trop difficile à manipuler.

L'état du système à un instant donné est caractérisé par l'ensemble des valeurs des variables et des attributs de tous les objets. Dans le cas d'une évolution continue de l'état, le modèle est continu, la description se fait via des équations différentielles. Dans le cas où les changements d'état s'effectuent instantanément à des instants discrets dans le temps, le modèle est à événements discrets, la description se fait à l'aide d'un réseau de Petri. Certains problèmes nécessitent des modèles dits hybrides, où apparaissent conjointement des comportements continus et des comportements dus à des événements discrets. Le modèle reproduit l'évolution au cours du temps de l'état du système sous l'effet des activités qui y sont réalisées.

La simulation est l'un des outils les plus puissants d'analyse des systèmes complexes, elle est conçue pour représenter l'évolution d'un système dans le temps. Elle est définie comme une manipulation numérique d'un modèle, dont les simulations faites sur ce modèle à pour but de comprendre le comportement du système et/ou d'évaluer différentes stratégies pour différentes opérations du système, et de mesurer l'impact relatif de chacune de ces composantes sur la performance globale du système de production.

6.1 Application aux systèmes de production

A partir de circulation des flux de produits définie précédemment, on obtient des valeurs des indicateurs de performances, qui seront par la suite joints pour des prises de décisions relatives pour l'aide à la conception, et à la conduite. Ces indicateurs de performances sont liés au volume de production (nombre et type de pièces produites), à l'engagement des ressources (le taux de leur utilisation) et probablement au plaisir du client.

Afin d'améliorer le système, il faut agir soit sur la capacité des ressources, soit sur la manière dont on utilise ces ressources (règles de gestion).

Ont peut agir sur les données à utiliser suivantes:

- Produits (type, gamme),
- Moyens de production (capacité des machines, coût, taux de production, disponibilité),
- Stocks (nature, capacité, gestion),
- Opérateurs (nombre, temps),
- Stratégie(s) de gestion de production,

En production, la simulation permet d'évaluer les effets suivants:

- suppression/adjonction de machines/de main d'œuvre;
- modification du processus de fabrication (gammes, temps de fabrication);
- présence d'aléas de fabrication (pannes machines, rupture des stocks, commandes urgentes);
- capacité des stocks;
- ordonnancement (politiques de lancement, règles de gestion des files d'attente, affectation des ressources)
- insertion de nouveau produits/suppression de produits existants.

6.2 Simulation des systèmes dynamiques hybrides

La simulation dynamique hybride est composée de plusieurs étapes: la simulation de la partie discrète, la simulation de la partie continue et la mise en place du modèle global.

- La simulation de la partie discrète consiste à associer un modèle discret, ou les marques se déplacent sur le graphe en franchissant les transitions, les marques sur se modèle représente l'état du système.
- La simulation de la partie continue consiste à résoudre un système d'équations composé d'équations différentielles ordinaires et partielles et d'équations différentielles algébriques.

La mise en place du modèle global est obtenue par enchaînement des sous-modèles actifs du système. La structure qui le caractérise est par conséquent souvent variable puisque le nombre d'équations et le nombre de variables sont menés à changer au cours de la simulation.

La modélisation implique de prendre en compte la question de validation dans les modèles proposés. La validation de tout modèle pose un problème réel du fait de la complexité des différents processus représentés, et consiste à confronter les données obtenues avec le modèle de la réalité [30]:

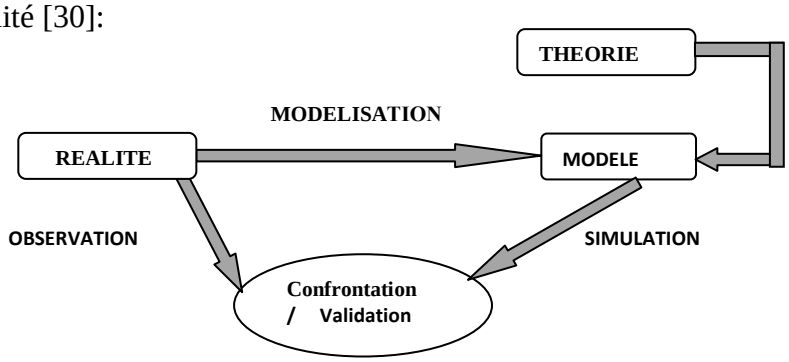


Figure 1.3 : la simulation [44]

La validation est utilisée pour la vérification de la comptabilité du modèle au système réel, et d'être vue comme une valeur de jugement fondée à partir des expériences faites avec un modèle. La validation opérationnelle peut être s'envisager par :
- une comparaison du modèle avec des modèles déjà validés,
- par une analyse de flexibilité du modèle c'est à dire comment le modèle évolue face aux modifications,
- une validation interne du modèle en utilisant un ensemble de données tests pour s'assurer si les sorties sont consistantes à chaque pas de la simulation.

7. Conclusion

Dans cette section, nous avons donné un état de l'art sur les systèmes de production. Nous avons exposé les différents types de problèmes d'ordonnancement ainsi que les principales méthodes de résolution existantes dans la littérature. De fait, les calculs d'ordonnancement reposent sur des ressources chacune caractérisée par des besoins et des contraintes spécifiques. Les systèmes d'ordonnancement se préoccupent du cadre d'application dans lequel le problème se situe dont la fonction est de produire des solutions efficaces pour l'allocation de tâches sur des ressources tout en respectant certains objectifs et contraintes.

Pour trouver une solution réalisable, un système d'ordonnancement procède à une analyse des données sur le problème posé. Cette analyse dépend du point de vue du système sur les ressources à gérer (modélisation du problème), et de la base théorique retenue pour générer ces solutions (méthodes exactes, heuristiques, etc.).

Au final ce chapitre a proposé un aperçu général sur les systèmes dynamiques hybrides. Il a dans un premier temps, positionné ce domaine d'étude par rapport aux systèmes à dynamique continue et discrète. Il a ensuite introduit les aspects « modélisation » et a proposé, dans ce contexte, un ensemble de formalismes permettant de les représenter. Les formalismes proposés pour représenter les systèmes dynamiques hybrides sont nombreux. Ils se différencient par la technique de modélisation adoptée qui repose soit sur une approche visant à étendre un modèle discret ou un modèle continu, soit sur une approche visant à utiliser un modèle mixte.

En revanche, la technique de modélisation reposant sur une approche mixte est plus générale et offre un potentiel de modélisation élevé dans chaque domaine (continu et discret). Cet état de l'art sur les SDH, proposé dans ce chapitre, nous permet ainsi d'aborder plus aisément la problématique de cette étude, qui est d'optimiser les systèmes dynamiques hybrides.

L'outil de simulation peut donc s'envisager selon différentes phases: conception du modèle, conception de la solution opérationnelle, vérification par l'expert informatique, et validation par l'expert du domaine.

CHAPITRE 2

Les outils de modélisation

1. Introduction

Les systèmes de productions industriels ont des comportements de nature très différente selon leurs domaines d'application dans l'industrie. Ils ont des comportements discrets et /ou continus. Les ateliers de production flexibles se caractérisent par un fonctionnement où se côtoient plusieurs situations relatives au parallélisme des opérations, au synchronisme et au changement d'état, suite à des événements, par exemple. Afin de prendre en charge au mieux toutes ces situations, l'utilisation d'un outil de modélisation performant est nécessaire.

Il doit être à la fois un moyen de description et un moyen d'analyse du fonctionnement qui se traduit par un changement d'état suite à l'occurrence d'un événement. Ce dernier peut être de deux types: l'un est uniquement déterminé par rapport à la variable temps, tandis que l'autre dépend des conditions d'évolution du système et s'obtient par des règles.

Parmi les outils de modélisation disponibles, l'on s'accorde à dire que parmi les plus performants, il y a le Réseau de Pétri (RdP). Il est reconnu comme étant un outil puissant de modélisation des systèmes de production par rapport à sa capacité de représenter les parallélismes, les conflits et le partages de ressources. Il se caractérise aussi par ses divers extensions, adaptées aux divers environnements (stochastiques, déterministe, discret, continu, notamment).

Dans un RdP discret, l'état et les grandeurs d'un système sont discrets et habituellement représentés par des variables logiques ou des nombres entiers naturels (les systèmes manufacturiers, les automates ou les calculateurs numériques). Le marquage d'une place peut correspondre soit à l'état booléen d'un dispositif (par exemple une ressource est disponible ou pas), soit à un nombre entier (nombre de pièces dans un stock). Par contre, dans une représentation continue, l'état du système appartient à un domaine continu et les grandeurs caractéristiques évoluent de façon continue à l'intérieur de ce domaine (les processus chimique et les asservissements analogiques, par exemple). Dans un RdP continu, les marquages de places sont des nombres réels et le franchissement de transitions est un processus continu.

Mais il peut arriver que pour un même système, les aspects continus et discrets coexistent et sont complémentaires. Le système de production que nous étudions, fait partie de cette catégorie. Il est composé d'une partie discrète et d'une partie continue. On l'appelle, système hybride. L'outil de modélisation d'un tel système hybride, doit pouvoir prendre en compte les deux aspects caractérisant ce système, c'est-à-dire les comportements discret et continu. Cet outil existe, c'est le réseau de Pétri hybride et c'est une des extensions des réseaux de Pétri.

Ce chapitre est consacré à une brève présentation des divers outils de modélisation utilisés pour représenter les systèmes à évènements discrets, ensuite sont mis en relief, les extensions des réseaux de Pétri et particulièrement l'outil choisi dans le cadre de cette thèse, qu'est le Réseau de Pétri Hybride.

2. Les outils de modélisation

2.1 Les outils de modélisation utilisés dans les Systèmes discrets

Il s'agit ici des systèmes, dont le comportement peut être représenté sous forme d'occurrences asynchrones d'événements discrets. Un événement discret est défini comme étant un changement qualitatif d'une situation. Les systèmes à évènements discrets (SED) peuvent être vus comme un système dynamique dont l'espace continu a été discrétisé en un espace d'états discrets. Un système à événements discrets est un système, pour lequel le temps aussi bien que les composantes du vecteur d'état sont des variables discrètes.

Le comportement dynamique d'un SED s'attache à l'ordre d'occurrence des événements et ne tient pas compte précisément du temps, ces systèmes sont des SED non temporisés.

Dans d'autres applications, la prise en compte du temps est primordiale et il est intégré aux modèles de façon à désigner les trajectoires d'événements et leurs temps, ces systèmes sont des SED qualifiés de temporisés.

Un système à évènement discret peut être rencontrés dans plusieurs domaines: la production manufacturière, le transport, le domaine de l'informatique, la communication… Les modèles classiques de représentation de ces systèmes sont les Automates Finis, les StateCharts, le Grafcet et les réseaux de pétri.

a. Les automates à états finis et à événements finis

Les automates à états finis constituent l'outil le plus simple mis à la disposition du modélisateur. Les automates permettent de décrire la dynamique à l'aide des notions d'état et de transition. La structure d'un automate est proche de la structure d'un système dynamique définie: $A = \langle X, S, \Delta \rangle$

Où X et S sont respectivement les entrées du modèle et les états du système. La fonction Δ décrit la fonction de transition en indiquant l'état suivant en fonction de l'état courant et de l'entrée du système. Les entrées sont des symboles, mais il est facile de passer de la notion de symbole à celle d'événement et ainsi de disposer d'un outil de spécification à événements discrets. Nous pouvons représenter graphiquement un automate à états finis à l'aide d'un graphe où les sommets représentent les états et les arcs les transitions.

Les automates à événements finis constituent un point de vue complémentaire où les sommets représentent les événements et les arcs la succession possible des événements.
Les arcs portent, en général, une durée. Cette durée représente le temps qui s'est écoulé entre les deux événements.

b. les state charts: Les diagrammes dynamiques UML

La modélisation par objets et UML propose une extension objets des automates à états finis: les state charts. Comme pour la majorité des diagrammes d'UML, il est possible d'utiliser le formalisme selon différents niveaux de détails. Les state charts peuvent être utilisés comme un automate à états finis. Dans sa version complète, le formalisme s'intègre dans une démarche de modélisation objets. Le formalisme des state charts est une combinaison des formalismes à base d'états et de transitions. Nous retrouvons:
– des transitions déclenchées sur des événements: si le modèle reçoit un événement, un message dans le vocabulaire objets, provenant d'un autre state chart alors la transition est franchie ;
– des transitions temporisées : la transition est franchie au bout d'un certain temps ;
– des transitions conditionnelles : si la condition exprimée en fonction du vecteur d'états est vraie alors la transition est franchie.

c. Le Grafcet

Le grafcet est un outil normalisé sur le plan international depuis 1988 [19]. Il est représenté par un graphe cyclique, composé alternativement d'étapes (états du système) et de transitions (changements d'état), ces éléments étant reliés entre eux par des arcs. Son avantage majeur est qu'il exprime efficacement les phénomènes de synchronisation et de parallélisme. Il constitue également un outil très bien adapté aux systèmes automatisés.

Au final, le nombre d'étapes et de transition croît linéairement avec l'ajout de machine parallèle et non pas exponentiellement.
Toutefois, le nombre d'états reste identique à celui de l'automate à états finis.

d. Les Réseaux de Petri

Les Réseaux de Petri (RdP) sont introduits par Carl Adam Petri en 1962, ils constituent un outil de représentation formel, graphique et mathématique très bien adapté pour la modélisation, l'analyse et le contrôle des SED. Ils permettent la description efficace des contraintes de synchronisation, de parallélisme et de séquencement, et ils disposent de nombreuses techniques de vérification automatique des propriétés (borné, vivant,..).

Un réseau de Petri est un graphe orienté composé de deux types de nœuds: les places (représentés par des cercles) et les transitions (représentés par des traits). Des arcs orientés, relient les places aux transitions, auxquels un poids (nombre entier strictement positif) est éventuellement associé à chaque arc. Lorsque le poids n'est pas signalé, il est par défaut égal à un. Ce graphe constitue la partie statique du réseau. La partie dynamique est introduite par les jetons qui se déplacent sur le réseau. Leur position sur les places à un instant donné est représentée par un vecteur qui définit le marquage du réseau.

2.2 Les outils de modélisation utilisés dans les Systèmes continus

Les systèmes continus sont perçus au travers de la représentation des variables d'état continues et une variable temporelle, continue ou discrète.

[10] L'expression «procédés continus» à un sens mathématique par rapport au temps. Ces procédés assurent une production permanente

.

De manière générale, un système dynamique continu peut être vu comme un système multivariables caractérisé par un vecteur d'état réel de dimension finie. Il suppose un certain niveau de continuité (dérivabilité), l'évolution des variables sont continues dans le temps (Ex: la température). Parmi les modèles modélisant un système dynamique continu on site:

a. Les Modèles à Equations différentielles et Algébriques

Un système continu peut être décrit par un ensemble d'équations différentielles ordinaires et/ou algébriques et finalement s'écrire sous la forme implicite:

$$F\left(x^n, x, p', u, t\right) = 0$$

$$X(t_0)=x_0$$

$$X^n(t_0)= x_0^n$$

Avec F : équations (différentielles et/ou algébriques)

x : vecteur des inconnues du système

x^n: vecteur des dérivées à l'ordre n par rapport à la variable indépendante (t)

p': ensemble de paramètres opératoires du système

u : vecteur des paramètres de contrôle

t : variable indépendante (généralement le temps)

Cette représentation générale permet de décrire des phénomènes non linéaires, très présents en génie des procédés. La connaissance au point initial t_0, du vecteur X et de ses dérivées successives (respectivement $x_0,\ldots, x_0^n$), des paramètres opératoires et de contrôle, permet de déterminer l'état ultérieur du système par intégration.

b. Les Bond Graphs

Les bonds graphs constituent un outil graphique permettant de décrire les échanges d'énergie dans le système étudié. Issu des domaines de la mécanique, l'hydraulique et l'électrique, cet outil de modélisation dynamique est applicable à de nombreux domaines de la physique, facilitant ainsi les échanges d'énergie de part sa nature graphique.

Avec les bonds graphs, les phénomènes physiques sont décrits sous forme de graphes plutôt que par la constitution de bibliothèques de modèles basés sur des équations différentielles non linéaires. A l'aide d'un langage universel, les bonds graphs permettent d'afficher la nature des échanges de puissance dans le système tels que les phénomènes de stockage, de transformation et de dissipation de l'énergie, tout en mettant en évidence la nature physique et la localisation des variables d'état.

c. Extension des modèles continus

Dans le cadre de l'extension des modèles continus vers les systèmes dynamiques hybrides (SDH), une première extension proposée, repose sur l'utilisation de variables mixtes. Cela consiste à introduire des variables booléennes ou entières au sein du modèle continu (Ex: l'état «ouverture» ou «fermeture» d'une vanne peut être représenté par une variable booléenne égale respectivement à un ou à zéro).

Une autre extension consiste à introduire des éléments de commutation dans le modèle continu, ce qui donne naissance à des graphes d'états représentant les modes de fonctionnement du système.

Ils sont associés d'une part, à des systèmes d'équations différentielles modélisant le comportement de la partie continue, et d'autre part, à des fonctions de saut traduisant les discontinuités lors des changements de modes.

2.3 Les outils de modélisation utilisés dans les Systèmes Dynamiques Hybrides

La plupart des procédés industriels réels évoluent selon des entrées dépendant des sous processus continus qui sont lancés, arrêtés par des commandes à états discrets. La différence avec les systèmes continus, réside dans le fait que ces procédés comportent des séquences de transfert et de conditionnement, relevant de systèmes à événements discrets, et d'opérations continues pendant un certain temps.

Les procédés industriels ont rarement un comportement purement discret ou purement continus mais plutôt un mixte des deux. Ces systèmes dynamiques à composante mixte (dynamique continue et événementielle) sont nommés: systèmes dynamiques hybrides (SDH). Comme le précise Zaytoon [79], la représentation continue ou discrète des modèles est directement liée à la nature des variables d'état et temporelle, qui peuvent être: soit continues, soit discrètes, soit symboliques. C'est à partir de cette classification sur les variables d'état que découlent les modèles des systèmes à dynamique continue ou à événements discrets.

Dans les systèmes hybrides on se concentre sur les problèmes à dynamique continue puis on essaye de prendre en compte les aspects discrets qui font parties des aspects événementiels pour y inclure les éléments continus. Le comportement à dynamique continue est régulièrement représenté par un système d'équations différentielles et algébriques tandis que le comportement à dynamique discrète est plutôt traduit par un ensemble d'états et de transitions.

L'aspect continu du traitement apparaît dans les activités de transfert, de régulation et de transformation. Dans le cas des systèmes manufacturiers la durée d'une activité est fixé et connu à l'avance, Mais, dans le cas des systèmes batch, la durée d'une activité n'est pas fixée et connue à l'avance, elle dépend généralement de l'état du système au début de celle-ci. (Ex. le temps de remplissage d'une cuve va dépendre du niveau de la cuve située en amont, c'est à dire pression de sortie du fluide plus ou moins forte, en début de phase de remplissage).

L'aspect discret vient du fait que ces activités ont une date de début et une date de fin, la fin d'une ou plusieurs activités peut déclencher une nouvelle configuration du système afin de réaliser l'activité suivante.

Les méthodes permettant la description des Systèmes Dynamiques Hybrides (SDH) sont capables de prendre en compte tous les types de grandeurs ainsi que leurs interactions, qui peuvent être représenté par un modèle basé sur les RdPH.

Par la suite on va donner un bref aperçu sur les réseaux de petri, ensuite on va entamer les réseaux de petri hybrides qui constitue l'axe principal de notre modélisation.

3. Réseau de Petri

Pour la modélisation de la dynamique des systèmes on a besoin d'un modèle qui a un formalisme visuel, un support mathématique, une spécification hiérarchique et une description facile des systèmes. Le réseau de petri est l'outil de modélisation le plus adapté pour répondre à ces exigences grâce à ces avantages. De plus, leur représentation graphique permet de visualiser le comportement dynamique du système pendant sa simulation.

Le réseau de Petri (RdP) est l'un des outils les plus utilisés pour la modélisation, il a connu depuis sa création un grand succès à cause de sa simplicité mathématique [64]**,** ainsi ses avantages pour la représentation graphique [60].

Le réseau de Petri est un formalisme qui joue un rôle important pour la description des systèmes à événements discrets, car il est capable de modéliser des propriétés telles que : le synchronisme, le parallélisme, le conflit, et le partage de ressources.

Il s'appuie sur des relations et règles simples basée sous un outil théorique et un formalisme mathématique (Ex. les équations différentielles). Il permet de représenter des comportements très complexes on s'appuyant sur des graphes particulièrement adaptés à la modélisation et à l'analyse des propriétés des systèmes.

La modélisation est donc obtenue par un modèle qui représente l'expression graphique de la structure du système, elle permet :

- une analyse des propriétés dynamiques : la bornitude, l'absence de blocage, les invariants,…
- de représenter un comportement donné d'un processus tout en modélisant le fait que les ressources sont partagées, que des priorités interviennent, qu'il peut y avoir présence d'une gamme de fabrication,…
- de faire apparaitre l'objectif de production qui devra être optimisé, pour ce faire il faut minimiser l'utilisation des ressources, maximiser le taux de production, minimiser le temps au plus tard de production pour limiter les stocks de pièces,… .

Pour aboutir à de meilleurs résultats, il faut que le produit respecte les contraintes définies dans le modèle. Et il faut étudier l'ordonnancement des opérations tel qu'il est spécifié dans le modèle.

Un réseau de Petri (RdP) est un graphe constitué de deux sortes de nœuds: Les places (représentées par des ronds ◯) et les transitions (représentées par des barres ▬). Le graphe est orienté par des arcs qui relient un nœud à l'autre (jamais de places à places, ou de transitions à transitions directement), voir la figure 1.

A chaque place est associée un marquage qui est un nombre entier correspondant au nombre de jetons dans la place. Un jeton est un petit disque noir (représenté par ●) qui représente généralement une ressource disponible dans la place ou il se trouve. Le marquage correspond à l'ordre croissant des indices, c'est-à-dire à ($m_1 \ldots m_i$). La transition T_i est sensibilisée s'il y a au moins un jeton dans chaque place d'entrées de cette transition. Le franchissement consiste à retirer un jeton de chaque place d'entrée et à rajouter un jeton à chaque place de sortie de la transition franchie.

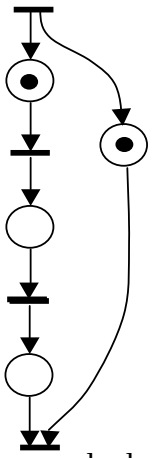


Figure 2.1 : exemple de réseau de petri

Définition 1 [41] Un réseau de Petri (RdP) est un outil graphique et mathématique décrivant les relations entre des conditions et des événements.
Un réseau de Petri peut être assimilé à un système composé de deux parties distinctes:
· Une partie statique (structurelle).
· Une partie dynamique (comportementale).

La partie statique (structurelle) se constitue d'un graphe orienté biparti valué < P, T, Pre, Post > avec:
· P, un ensemble fini de places, $\{p_1, p_2,..., p_n\}$, *représentées par des cercles.*
· T, un ensemble fini de transitions, $\{t_1, t_2,..., t_m\}$, *représentées par des barres.*
Ces deux ensembles formant les sommets du réseau, avec $P \cap T = \emptyset$, *les ensembles de P et T sont disjoint.*
· Pre : P x T $\rightarrow \aleph$ est l'application d'incidence avant, correspondant aux arcs directs reliant les places aux transitions, *c'est l'application des places précédentes.*
· Post : P x T $\rightarrow \aleph$ est l'application d'incidence arrière, correspondant aux arcs directs reliant les transitions aux places, *c'est l'application des places suivantes.*
Lorsque Pre et Post prennent leurs valeurs dans l'ensemble {0, 1}, le réseau est alors qualifié de réseau ordinaire.

En utilisant une notation matricielle, les matrices d'incidence avant et arrière peuvent être respectivement définies comme suit:
· Pre = $[\delta_{ij}]$ $(1 \le i \le n, 1 \le j \le m)$ avec $[\delta_{ij}] = Pre(p_i, t_j)$,
· Post = $[\varepsilon_{ij}]$ $(1 \le i \le n, 1 \le j \le m)$ avec $[\varepsilon_{ij}] = Post(p_i, t_j)$.
Ainsi, la matrice d'incidence du réseau $W = [w_{ij}]$ $(1 \le i \le n, 1 \le j \le m)$,
avec $w_{ij} = Post(p_i, t_j) - Pre(p_i, t_j)$
Elle permet de donner une représentation de la structure de réseau. Cette matrice est de dimension n×m. Chaque colonne indique la modification du marquage lors du franchissement de la transition correspondante.

Nous adopterons la notation suivante:
· °t (resp. t°) représentera l'ensemble $\{p \in P / Pre(p, t) \neq 0\}$ (resp. $\{p \in P / Post(p, t) \neq 0\}$) correspondant à l'ensemble des places d'entrée (resp. sortie) de la transition t.
· °p (resp. p°) représentera l'ensemble $\{t \in T / Post(p, t) \neq 0\}$ (resp. $\{t \in T / Pre(p, t) \neq 0\}$) correspondant à l'ensemble des transitions d'entrée (resp. sortie) de la place p.
La composante statique est complétée par un marquage initial:
· M_0: P $\rightarrow \aleph$, $M_0(p)$ est le nombre de marques contenues initialement dans la place p.

La partie dynamique (comportementale) du réseau consistera alors à faire évoluer ce marquage. On peut aussi dire que :

· les propriétés de structure elles sont indépendantes du marquage initial,
· les propriétés de comportement elles sont dépendantes du marquage initial.

Soit S une séquence de franchissement de transitions. Le vecteur caractéristique de la séquence S, noté s, est le vecteur de dimension m dont la composante j correspond au nombre de franchissements de la transition T_j dans la séquence. Si la séquence S est réalisable à partir d'un marquage M_i, alors on obtient le nouveau marquage M_{i+1} tel que : $M_{i+1}= M_i+ W.S$.

Les RdP permettent également la représentation des informations quantitatives, car les marques peuvent être accumulées dans les places.

3.1 Marquage des places

Le marquage d'un réseau de Petri est une opération qui consiste à assigner des jetons dans les places, donc le marquage donne la position des jetons dans les places. Un marquage M (p) est le nombre de marques (jetons) contenus dans la place p, et le marquage initial donne la position initiale des jetons.

L'état du système modélisé par le RdP est représenté par le marquage définissant le nombre de jetons contenus dans chaque place. Ces jetons vont circuler dans les places selon certaines règles. Cette circulation des jetons symbolise l'évolution dynamique du système, d'où l'évolution de l'état (représentant la dynamique du système) correspond à l'évolution du marquage (produit par le franchissement de transitions).

3.2 Comportement du franchissement dans un réseau de Petri

Dans un réseau de Petri, l'état du système modélisé à un instant donné est représenté par marquage des places. Ce marquage peut être modifié au fur et à mesure de l'occurrence de franchissements de transitions. Pour pouvoir être franchie, une transition doit être franchissable.

Le franchissement (tir) d'une transition T_j ne peut s'effectuer que si le marquage de chacune des places P_i directement en amont de cette transition vérifie la règle suivante:
On dit qu'une transition t est franchissable (sensibilisée) si et seulement si
$\forall p \in P \quad M(p) \geq Pre(p, t)$

Le franchissement (tir) de T_j consiste à retirer Pré (P_i, T_j) jetons dans chacune des places directement en amont de T_j et à ajouter Post (P_k, T_j) jetons dans chacune des places P_k directement en aval de T_j.
[41] Si t est franchissable pour le marquage M, le franchissement (tir, firing) de t donne le nouveau marquage M' tel que :
$\forall p \in P \quad M'(p) = M(p) - Pre(p, t) + Post(p, t)$

Des conflits apparaissent lorsqu'une place est en amont de plusieurs transitions, et que le marquage de la place commune ne permet pas de franchir toutes les transitions qui la partagent. Il existe deux types de conflit, le conflit structurel et le conflit effectif, qui sont définis par la suite;

Définition 2 [20] Un conflit structurel correspond à un ensemble de transitions qui ont au moins une place d'entrée en commun.

Définition 3 [20] Un conflit effectif correspond à l'existence d'un conflit structurel avec un nombre de marques dans la place du conflit est inférieur au nombre de marque nécessaire au

franchissement de toutes les transitions de sortie de cette place qui sont validées par le marquage M.

Ce type de conflit correspond au problème rencontré dans le modèle de notre cellule de production étudiée. Lorsque plusieurs transitions demandent le partagent d'une même ressource, un outil de prise de décisions est associé au modèle de RdP, qui permet de résoudre le conflit en choisissant la transition qui doit être franchie.

3.3 Réseaux de Petri et robustesse des SED

La robustesse d'un système est définie comme étant sa capacité à préserver certaines propriétés d'intérêt face à des perturbations. C'est la robustesse d'un système de production à contraintes qualitatives.

La synthèse de la commande des systèmes de production consiste à :

· L'établissement d'un modèle de spécification des contraintes du procédé à commander,
· La résolution des choix d'affectation des ressources et leur séquencement.
· La détermination des dates de lancement des opérations à effectuer correspondent aux instants de tir des transitions (pour les commandes temporelles).

À chaque étape de la synthèse de la commande, la robustesse doit prendre en compte:

· Les imprécisions sur les durées opératoires.
· Le choix d'un séquencement des ressources, qui doit aboutir à un séquencement admissible.
· Le contrôle des instants de début des opérations qui doit tenir compte des perturbations agissant sur le procédé.

Pour ce faire plusieurs classes de réseaux de Pétri prenant en charge la notion de robustesse sont apparues.

4. Classes de réseau de petri

Les systèmes de production ont connu de structures physiques plus compliquées, ainsi que leurs commandes. De nombreux travaux ont également conduit à l'apparition d'autres extensions des réseaux de Petri pour augmenter la puissance d'abstraction et répondre à la modélisation de problèmes spécifiques. Chacune de ces extensions apporte un élément de réponse à tel ou tel autre aspect de la complexité des systèmes étudiés. Le choix d'un modèle de représentation dépendra du système de production à étudier et des propriétés à analyser.

Plusieurs classes de réseaux de Petri ont été développées, chacune d'elles cherchant à décrire une vue des systèmes flexibles de production, de leur conception, et de leur conduite.
La complexité des structures physiques et de leurs commandes donne lieu aux études sur les réseaux de Petri colorés [5], les RdP dépendants du temps [70], etc... Les systèmes présentant des aspects continus trouvent un début de réponse avec les RdP continus [23]. On se qui concerne les systèmes hybrides, ils trouvent leur réponses dans les RdPH.

4.1 Les temporisations dans les réseaux de pétri

Afin de représenter le comportement dynamique des systèmes, il est indispensable de modéliser le temps. Ainsi, l'outil devient plus performant pour la spécification et la validation des systèmes.

Le modèle réseau de Petri intégrant l'aspect temps permet de représenter les mécanismes temporels associés à un processus. Cette approche aide à traiter les questions liées à l'analyse et à l'évaluation des performances aux travers des méthodes analytiques.

Dans un RdP, on associe un état qui est représenté par une place a une certaine durée (durée de l'opération), et on associe un changement d'état à une transition. On peut aussi associer la

durée d'une opération à une place et on associe le temps d'attente d'arrivé d'un événement à la transition qui est franchie quand cet événement se produit.

Dans ce chapitre nous allons présenter le principal type de modèle réseau de Petri qui utilise le temps explicitement: le RdP temporisé. Cet outil de modélisations permet de spécifier pour chaque opération soit une durée minimale associée aux transitions (modèle RdP T-temporisé) [70] ou aux places (modèle RdP P-temporisé) [75].

4.1.1 Les réseaux de petri temporisées

Ramchandani [70] fut le premier à introduire les réseaux de Petri temporisés. Plus particulièrement, il a associé une durée à chaque franchissement de transition conduisant au modèle T-temporisé. Ces derniers ont surtout été utilisés dans un cadre d'évaluation de performances.

Ainsi pour toute transition t à laquelle est associée une temporisation d_t le franchissement de cette dernière entraînera les opérations suivantes:
En supposant que la décision de franchir t intervient à l'instant τ.
1. Prélever Pre (p, t) jetons de toute place p élément de °t, à l'instant τ.
2. Ajouter Post (p, t) jetons à toute place p élément de t° à l'instant $\tau + d_t$.
Sifakis **[75]** quant à lui a décidé d'associer aux différentes places du réseau une durée traduisant le temps de séjour des jetons contenus dans ces dernières.

Plus récemment Julia [48] a proposé un modèle associant les temporisations aussi bien aux places qu'aux transitions du réseau : le modèle P-T-temporisé. Son cadre d'application est la conception et le pilotage de cellules flexibles à fonctionnement répétitif.

Les temporisations associées aux transitions représentent les durées associées aux opérations à effectuer, ces durées sont des constantes propres au problème considéré. Les temporisations associées aux différentes places du RdP symbolisent quant à elles d'éventuelles attentes avant le franchissement des transitions et représenteront alors des variables à définir.

Définition 4 [20] Un réseau de Petri temporisé avec n places et p transitions est un doublet RT = <R, D> avec:
R est un réseau de Petri < P, T, Pre, Post> avec un marquage initial M_0
D est la fonction durée minimale de séjour d'une marque dans une place donnée: D: $P \rightarrow Q^+$ qui à chaque place fait correspondre un nombre rationnel positif décrivant la durée d'indisponibilité des jetons.

La sémantique est que les marques doivent rester dans la place p_i au moins le temps d_i associé à cette place. Pendant d_i la marque est indisponible, elle ne participe pas à la validation des transitions, d_i représente donc :
• la durée d'indisponibilité de la marque pour la validation des transitions
• le temps minimum de séjour d'une marque dans une place.

a. Règles de fonctionnement

Un état est un doublet < M, I> où
- M est une application de marquage assignant à chaque place du réseau un certain nombre de marques $\forall p \in P, M(p) \geq 0$,
- I est une application de temps d'indisponibilité, assignant à chaque marque k dans la place p_i un temps θ_i^k où θ_i^k est la durée qui reste à la marque k pour terminer son temps de séjour minimal dans la place p_i,

Les temporisations associées aux places permettent de prendre en compte les durées opératoires minimales. Par conséquent une transition validée au sens des RdP autonome peut ne pas être obligatoirement franchie. Une transition est franchissable si elle est validée au sens des RdP autonomes et si les marques qui la valident sont disponibles.

b. Interprétations

Soit t le temps de séjour du jeton dans la place p et d (p) la durée associée à la place p. Il existe donc deux états possibles pour un jeton :

○ non disponible si $t < d(p)$

● disponible si $t \geq d(p)$

Le temps où un jeton n'est pas disponible permet de spécifier la durée nécessaire pour effectuer une opération sur un produit donné. L'état de disponibilité d'un jeton permet de modéliser l'état d'un produit qui est disponible pour la prochaine tâche.

La littérature définit les réseaux de Petri stochastiques (RdPS) qui sont obtenus à partir des réseaux de Petri classique [76] en associant des durées de franchissement aléatoires aux transitions pour l'évolution du marquage [80]. Ce type de RdPS est bien adapté pour la modélisation des phénomènes aléatoires où le temps entre deux événements n'est pas fixe.

4.1.2 Réseaux de Petri stochastiques

Dans le cas du réseau de petri temporisé définie précédemment les durées sont associées aux transitions ou aux places du réseau. Elles représentent une partie de l'ensemble des contraintes à respecter. Ces durées correspondre à des valeurs fixes dans le cas d'un RdP temporisé déterministe. Ces modèles sont bien adaptés lorsque les aspects quantitatifs sont bien définis. Mais pour certains phénomènes imprévisibles (tels que l'apparition de pannes sur des machines ou la connaissance imprécise de certains paramètres), ils ne peuvent plus représenter correctement les processus. Dans ce contexte, les réseaux de Petri stochastiques ont été introduits par Florin et Natkin [32] afin de modéliser efficacement les phénomènes aléatoires. Le RdPS permet de prendre en compte l'occurrence des défaillances et leur influence sur le comportement du système.

Dans un RdP stochastique, les durées de sensibilisation associées aux transitions résultent d'un tirage aléatoire. L'hypothèse la plus couramment employée est que cette durée est distribuée selon des lois exponentielles (Ex. valeurs moyennes). Par conséquent, même si le processus n'est pas déterministe, il faut trouver un résultat qui garantit le bon fonctionnement sur une plage. Une application utilisant ce formalisme pour modéliser et analyser les performances d'un atelier de maintenance est traitée dans [2].

Remarque Dans un RdP temporisé le temps représentant une durée opératoire est associé à une place ou à une transition. Le franchissement des transitions dépend du temps minimum de séjour d'une marque dans la place qui est spécifié (durée minimale d'une opération). Le non respect de ces contraintes entraîne une dégradation de la qualité du produit et par la suite son évacuation au niveau du contrôle. Ainsi les RdP temporisés permettent de spécifier les contraintes du temps minimum.

4.2 Les réseaux de Petri Colorés

Les réseaux de Petri Colorés, [10] dérivés des réseaux de Petri Ordinaires [17] sont des graphes bipartis. Les nœuds sont appelés places (représentées par des cercles) ou transitions (représentées par des rectangles) et sont connectés par des arcs quantifiés. La quantification

des arcs en entrée permet d'indiquer les conditions d'activation d'une transition alors que celle des arcs en sortie détermine l'effet de l'action représentée par la transition.

Définition 5 [10] Un réseau coloré est un 5-uplet R = <P, T, C, Pré, Post> où
- P est un ensemble fini dont les éléments sont appelés places,
- T est un ensemble fini disjoint de P dont les éléments sont appelés transitions,
- C est une famille indexée par P ∪ T d'ensembles C(x); C(x) est appelé domaine de couleurs associé à la place ou à la transition x,
- Pré et Post sont deux familles indexées par PxT d'applications

Pré (p, t) et Post (p, t) telles que pour tout (p, t) de P x T, Pré (p, t) et Post (p, t) aient pour domaine C(t) et codomaine Bag(C(p)) { Bag (A): un multi-ensemble a sur un ensemble non vide A est une fonction a $\in[A\rightarrow \aleph]$. Intuitivement, un multi-ensemble est un ensemble qui peut contenir plusieurs occurrences du même élément. Bag (A) désigne l'ensemble des multi-ensembles finis sur A}. Pré et Post sont appelées respectivement fonction d'incidence avant et fonction d'incidence arrière.
Chaque place p contient un ensemble de marques. L'ensemble des marques contenues dans l'ensemble des places du modèle définit le marquage du modèle.

Définition 6 [10] Règles de franchissement d'un réseau de Petri coloré
Soit R un réseau de Petri, M un marquage de R, on dit qu'une transition t de R est franchissable à partir de M pour la couleur x de C(t) si et seulement si:
- $\forall p\in$ P, M (p) ≥ Pré (p, t) (x)

On note M [t (x) > la franchissabilité de t à partir de M pour la couleur x.
Si t est franchissable à partir de M pour la couleur x, le marquage M' obtenu par le franchissement de t est défini par :
- $\forall$ p ∈ P, M' (p) = M (p) - Pré (p, t) (x) + Post (p, t) (x)

On note M [t (x) > M' le franchissement de t à partir de M pour la couleur x.
Si à partir d'un marquage M, plusieurs transitions sont candidates, n'importe laquelle d'entre elles peut être activée.

4.3 Les réseaux de Petri continus [24]

4.3.1 Introduction

Un RdP continu est caractérisé par un vecteur de marquage, dont les composantes correspond à des nombres réelles positives ou nulles, et non plus des entiers. [24] montre que le comportement d'un RdP continu est un cas limite du RdP discret, obtenu en coupant chacune des marques du réseau en k jetons avec k tend vers l'infini, les places et transitions continus sont représentées par une double ligne.
une transition est représenté par: ▭ une place est représenté par: 

Le franchissement d'une transition t_j consiste alors à enlever une quantité X_j des places en amont de t_j et à ajouter cette quantité X_j dans les places en aval de t_j, d'où le franchissement d'une transition s'effectue donc comme un flux continu.

Une transition t_j est validée si nous pouvons franchir une quantité X_j de t_j avec $0 \le X_j \le \min (M(p_a);...; M(p_b))$ ($\{p_a,...,p_b\}$ l'ensemble des places d'entrée de t_j).
X_j est appelée quantité de franchissement.
Soit $d_j(t)$ la temporisation associée à la transition t_j à l'instant t d'un réseau continu temporisé.
Nous définissons la fréquence de franchissement $U_j(t)$ associée à cette transition tel que:

$$U_j(t) = 1/ d_j(t)$$

Cette fréquence est homogène à une vitesse: $V_j = k / d_j(t)$

4.3.2 Réseau de pétri continu autonome

Définition 7 La définition d'un réseau de pétri continu autonome est similaire à celle d'un réseau de pétri discret sauf que les fonctions Pré, Post, et le marquage initial prennent leurs valeurs dans $\Re^+$ au lieu de $\aleph$.

Remarque a) Dans un RdP continu, une séquence de franchissement S à partir d'un marquage M implique une trajectoire correspondant à une suite de marquages successifs.
Le vecteur caractéristique s d'une trajectoire est un vecteur dans lequel chaque composante est un nombre réel correspondant à une quantité de franchissement de la transition correspondante.
Le marquage M', atteint par le franchissement d'une séquence S à partir de M, obtenu par la relation fondamentale : $M' = M + W \cdot S$

La relation fondamentale pour un RdP continu est identique à celle d'un RdP discret. Ainsi, toutes propriétés des RdP discrets tirées de cette relation peuvent être transposées aux RdP continus.
b) les notions de RdP vivant, borné, sans blocage, sont similaires pour les réseaux de pétri continus et discrets.

Selon l'interprétation des contraintes temporelles pour la validation des transitions qui dépend des marquages des places en amont de la transition considérée, nous définissons deux réseaux de Pétri continu, l'un à vitesse constante, noté RdPCC, l'autre, à vitesse variable, noté RdPCV.

4.3.3 RdP continu à vitesses constantes (RdPCC) [24]

Le marquage d'une place dans un réseau de pétri continu à vitesse constantes (RdPCC), est une fonction continu du temps. Pour déterminer le comportement dynamique d'un RdP continu à vitesses constantes, on construit un graphe d'évolution dans lequel un nœud est associé à un vecteur de vitesse. Ce vecteur de vitesses est constant pendant une certaine période, puisque les évolutions des marquages sont des fonctions linéaires du temps. Cette caractéristique est la propriété la plus importante de ce modèle.

Définition 8 [41] Un RdPCC est caractérisé par des intervalles de temps avec les vitesses de franchissement (ou quantité de franchissement par unité du temps) constantes.
Un RdPCC est un sextuplet $R=<P, T, V, Pré, Post, M_0>$
où: P, T, Pré, Post ont les mêmes définitions que celle des RdP discrets, excepté que Pré et Post peuvent correspondre à des réels.
M_0 est le marquage initial des RdP continu.
V est une application de l'ensemble T des transitions de R dans l'ensemble des nombres $\Re^+ \cup \{\infty\}$.

4.3.4 RdP continu à vitesses variables (RdPCV) [24]

Définition 9 [41] Dans le RdPCC, la vitesse maximale de franchissement d'une transition T_j est constante. Alors on ne tient pas compte de la valeur des marquages des places en amont de cette transition. Par contre dans un RdP continu à vitesses variables, les vitesses instantanées de franchissement dépendent à chaque instant du nombre de marques continues dans les places en amont des transitions considérées.
L'idée est la suivante. Supposons qu'une transition T_j, dont le taux de franchissement est U_j, possède une seule place d'entrée P_i. S'il y avait toujours un jeton dans cette place (discrète), le taux de franchissement serait $U_j = 1/d_j$. Le RdPCV donne une vitesse instantanée $v_j = Uj \cdot m_i$,

ce qui correspond exactement au taux moyen de franchissement du RdP discret dont on veut approximer le fonctionnement.

La relation qui exprime la variation du marquage en fonction du temps: $\frac{dm}{dt} = W.v(t)$ est vraie pour n'importe quelle expression de v (t), ce qui conduit à une expression qui s'applique au marquage m du réseau; et qui peut être toujours s'appliquer si l'on dispose de v(t).

La vitesse instantanée de chaque transition est donnée par les relations suivantes:

- pour le mode implicite (limitations des transitions à un seul franchissement à la fois)
  $v_j(t) = U_j \min_i(1, m_i(t))$ avec i tel que $P_i \in {}^\circ T_j$
- pour le mode explicite (pas de limitations de franchissement)
  $v_j(t) = U_j \min_i(m_i(t))$ avec i tel que $P_i \in {}^\circ T_j$

L'intérêt du RdPCV vient du fait que les valeurs atteintes en régime stationnaire (marquages constants) correspondent aux valeurs moyennes du RdP discret correspondant. Lorsqu'il s'agit de modéliser un système discret, l'approximation fournie par RdPCV est donc meilleure que celle donnée par le RdPCC.
Dans un RdP continu, pour une vitesse $V_j(t) > 0$ une transition t_j est franchie continûment.
A un instant (t + dt) le marquage d'une place p_i est :

$$m_i(t + dt) = m_i(t) + \sum_{k=1}^{m} [Post(P_i, T_k) - Pré(P_i, T_k)]V_k(t)dt \qquad 1 \leq i \leq n$$

Soit la relation fondamentale: $dM / dt = W . V(t)$
où: W : matrice d'incidence du RdPC
V (t) : vitesse instantanée de franchissement. $V(t) = (V_1(t), \ldots, V_m(t))$

$M(t) = (m_1(t), \ldots, m_n(t))$ $\quad dm_i / dt = \dot{m}_i$

Notre cas d'étude faisant partie des Systèmes Dynamiques Hybrides, nous nous intéresserons à la représentation de cette catégorie de système.

La modélisation d'un SDH, nous permettra de visualiser les parties statiques et dynamiques du système, donc le fait de disposer d'un modèle rend possible la simulation du comportement d'un système pour le prévoir, l'influencer et chercher à l'optimiser. Il permet aussi, sous certaines conditions, de l'analyser pour détecter a priori d'éventuels dysfonctionnements et/ou d'en faire la commande pour obtenir les performances souhaitées.

Un RdP hybride est l'outil capable de modéliser un système hybride SDH, sous un seul formalisme qui comporte le formalisme discret et le formalisme continu en même temps, afin de simuler et analyser les systèmes complexes. Il permet de réunir les propriétés des réseaux de Petri continus et discrets. La partie continue est adaptée pour modéliser un flux continu ou un flux discret comportant un nombre important de marques (Ex. nombre de pièces dans un stock par un nombre évoluant continûment), et la partie discrète modélise ce qui ne peut l'être en continu (Ex. l'état opérationnel d'une ressource, l'apparition d'une panne, par une grandeur booléenne possédant que deux valeurs 0 ou 1).

5. Les réseaux de petri hybrides

Définition 10 [66] Un système est dit hybride s'il possède deux types de variables d'état: des variables d'état continues et des variables d'état discrètes [77].
De manière générale, les systèmes dynamiques faisant intervenir explicitement et simultanément, des phénomènes ou des modèles de type dynamiques continus et

événementiels, sont appelés systèmes dynamiques hybrides. Ces systèmes sont classiquement constitués de processus continus interagissant avec, des processus discrets.

Un système dynamique hybride est l'interaction entre des algorithmes discrets de planification et des algorithmes continus de commande. Dans sa description, il utilise de fonctions du temps continues par morceaux (pour la partie continu) et de fonctions à valeurs discrètes (pour la partie discrète).

Il est nécessaire, dans certains cas, de représenter sur un même modèle des phénomènes à la fois continus (comme le traitement d'un flux de matière continu) et discrets (comme l'occupation d'une ressource). Une telle représentation ne peut être effectué uniquement, avec le formalisme RdP discret, ou seulement avec le formalisme RdP continu. Cela a ainsi conduit à la naissance de la définition d'une nouvelle extension, les réseaux de Petri hybrides [52] dans lesquels coexistent des places et des transitions continues et discrètes, et permet de représenter les deux aspects continu et discret en même temps.

Dans un RdP continu, les marquages de places sont des nombres réels (Ex. nombre de pièces dans un stock par un nombre réel), et le franchissement de transitions est un processus continu. Un RdP continu peut être autonome (le temps n'est pas pris en compte), ou avec le temps est pris en compte dans ce cas des vitesses de franchissement associées aux transitions. Plusieurs modèles de RdP continus temporisés ont été définis: ils diffèrent de la façon de définir et de calculer les vitesses instantanées de franchissement des transitions. Ils fournissent de bonnes approximations pour l'évaluation de performances du système quand un RdP contient un grand nombre de jetons. Le franchissement d'une transition consiste à retirer une quantité q de marques de chaque place d'entrée et à ajouter la même quantité q à chaque place de sortie. D'autres auteurs ont ajouté des temporisations aux places dans les réseaux de pétri continus [18].

Dans un RdP discret, l'état d'une machine opérationnelle ou en panne, ou plus généralement une condition logique, ne peut pas être modélisée par un nombre réel. La modélisation de ces systèmes hybrides conduit naturellement aux RdP hybrides contenant une partie discrète et une partie continue [53]**.**

Dans la littérature [24] nous trouvons deux types de réseaux de Petri hybrides: les RdP hybrides autonomes et les RdP hybrides temporisés.

5.1 Réseau de pétri hybride autonome

Définition 11 Un réseau de pétri hybride autonome est un sextuplet Q = {P, T, Pre, Post, M_0, h} tel que:

P= {P_1, P_2,…, P_n} est un ensemble fini et non vide de places ;

T= {T_1, T_2,…,T_m} est un ensemble fini et non vide de transitions ;

$P \cap T = \phi$

Pré : $P \times T \rightarrow \Re^+$ où $\aleph$, est l'application d'incidence avant ;

Post : $P \times T \rightarrow \Re^+$ où $\aleph$, est l'application d'incidence arrière ;

M_0: $P \rightarrow \Re^+$ où $\aleph$, est le marquage initial.

h : $P \cup T \rightarrow$ {D, C}, appelée « fonction hybride », indique pour chaque nœud s'il est discret ou continu

Le graphisme adopté, pour la représentation des places et des transitions en continu et en discret, est reproduit à la figure suivante:

Figure 2.2: Éléments composant un RdP hybride [66]

Dans les définitions de Pré, post, et M_0: $\aleph$ correspond au cas ou $P_i \in D$, et $\Re^+$ correspond au cas ou $P_i \in C$.
Les fonctions Pré et Post doivent satisfaire le critère suivant: si P_i et T_i sont une place et une transition telles que $P_i \in D$ et $T_i \in C$, alors l'égalité Pre (P_i, T_i)= Post (P_i, T_i) doit être vérifiée.

Définition 12 [41] Une D-transition est validée si chacune des places P_i de ${}^{o}t_j$ vérifie :
$M(P_i) \geq$ Pré(p_i, t_j).

Définition 13 [41] Une C-transition est validée si chacune des places p_i de ${}^{o}t_j$ vérifie :
si P_i est une D-place, $M(p_i) \geq$ Pré(p_i, t_j);
si P_i est une C-place $M(p_i) > 0$.

Dans un réseau de pétri hybride, le vecteur caractéristique s d'une séquence S est un vecteur dans lequel chaque composante est soit un entier correspondant au nombre de franchissements d'une transition discrète, soit un nombre réel correspondant à une quantité de franchissement d'une transition continu. Un marquage M obtenu par le franchissement d'une séquence S à partir d'un marquage M_0, peut être obtenu par la relation fondamentale:
$M = M_0 + W \cdot s$.
Cette relation fondamentale est identique à celle d'un RdP discret. Donc nous pouvons transposer toutes les propriétés des RdP discrets aux RdP hybrides.

Par ailleurs, il est possible d'avoir des arcs inhibiteurs dans un RdP hybride [6]. On définit ainsi l'extension d'un réseau de Petri Hybride.

Extension du réseau de petri hybride [71]
Dans l'extension du réseau de petri discret, l'arc inhibiteur dont son poids r de la place P_i vers la transition T_j, permet le franchissement de T_j si et seulement si le marquage de P_i est inferieur à r. La même chose s'applique lors de l'utilisation d'un RdP hybride. (Pour r = 1, il correspond au test zéro, donc il faut que le marquage de la place P_i soit : $M_i < 1$, si M_i est un entier alors il faut que $M_i = 0$, si M_i est un réel, le test de zéro est donné par 0^+ qui représente un poids infiniment petit mais non nul).

5.2 Les RdP hybrides temporisés

Lors de l'analyse et l'évaluation des performances des systèmes dynamiques hybrides où nous intégrons l'aspect temps, on adopte les RdP hybrides temporisés qui sont des outils largement utilisés dans ce domaine de modélisation.

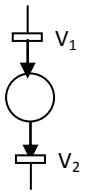
Un réseau de petri hybride temporisé RdPHT avec n places et p transitions est défini comme un RdPH pour lequel on associe des durées minimales de franchissement aux transitions discrètes, et des fréquences maximales de franchissement aux transitions continues.

Les temporisations sont associées soit aux places soit aux transitions (les deux cas sont équivalents [24]).

Dans de nombreux cas, les réseaux de petri hybrides permettent une représentation homogène des aspects continus et discrets au sein de même formalisme. Ces propriétés leur permettent de représenter efficacement les systèmes comportant des transferts de produits ou de pièces.

Définition 14 [41] Donc un RdP temporisé est un couple <H, Tempo> où H est un RdP hybride et Tempo est l'application qui associe à chaque transition (place) un nombre rationnel positif :
Tempo(t) = d_j, avec d_j: temps associé à la transition t_j.
♦ La temporisation associée à une D-transition t_j est d_j
♦ La vitesse maximale de franchissement associée à une C-transition t_j est: $V_j = 1/ d_j$

Un système continu pur est modélisé par le réseau de petri continu, ensuite la partie discrète sera additionnée, et le système complet sera modélisé par un réseau de petri hybride
$M(t+dt) = M(t) + (V_1 – V_2)\, dt$
à l'état initial: $M(t) = M(0) + (V_1 – V_2)\, t$
V : vitesse maximale, v : vitesse instantanée



♦ Le marquage M(t) à un instant donné se déduit à partir du marquage initial M_0, par la relation fondamentale:

$$M(t) = M(0) + W \cdot [\sigma(t) + \int_{u=0}^{t} V(u)\, du]$$

avec: σ (t): nombre de franchissement de chaque D-transition entre l'instant initial et l'instant t. Les composantes associées aux C-transitions sont nulles.
V(t): vecteur vitesse de franchissement instantanée des C-transitions à l'instant t.

Dans le cas général, $w = \begin{bmatrix} wD & 0 \\ wCD & wC \end{bmatrix}$ avec :
w^D correspond au arcs des nœuds discrets, w^C correspond au arcs des nœuds continus, et w^{CD} correspond aux arcs entre les places continues et les transitions discrètes; les arcs entre les places discrètes et des transitions continus correspond au sous-matrice 0, quand w^{CD} =0 le RdP hybride est élémentaire.

Pour conclure on peut dire que, les systèmes industriels complexes trouvent leurs réponses de modélisation par l'utilisation des modèles basées sur les RdP hybrides qui constituent les outils les plus aptes et qui donnent de meilleures performances. Ils permettent la prise en compte les aspects continus d'un système de fabrication dont ces variables continues prennent leurs valeurs dans l'ensemble des réels $\Re$, et son aspect discret dont ces variables discrètes prennent leurs valeurs dans l'ensemble des entiers $\aleph$.

6. Réseau de pétri lots [4]

Les matières premières sont continues, mais elles sont traitées par lot, c'est à dire de façon discrète. Le modèle décrivant ce type de système doit prendre en compte à la fois les aspects discrets et les aspects continus. C'est à dire qu'il doit manipuler des variables discrètes et continues et être capable de décrire leurs évolutions.
Pour transporter un lot de produit vers un stock intermédiaire ou vers une ressource, il doit être disponible sur le convoyeur de l'amont vers l'aval de ce convoyeur.

Le convoyeur joue un rôle important du fait de transporter les profilés entre machines et zones intermédiaire de stockage, le flux physique des produits sont conditionnés par des phénomènes d'accumulation dynamique et par la variation de vitesse d'entrainements des éléments de transfert (chariot filo-guidés, convoyeur, etc).

La vitesse de déplacement de ceux – ci peut donc soit être égale à la vitesse d'entraînement du convoyeur (état non accumulé ou régime libre), soit inférieure (état accumulé). Dans ce dernier cas, on parle de vitesse apparente des pièces [27]. Pour un système mono produit sans accumulation, souvent, entre deux ou plusieurs stations de travail d'une ligne de fabrication, les produits circulent à travers un système de convoyage qui assure le transport des pièces ou des sous–ensembles de pièces parmi des zones de stockage intermédiaires (Fig.), [25].

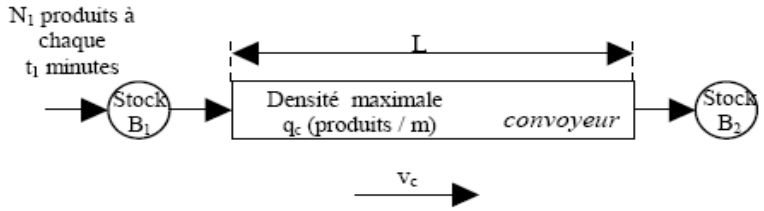


Figure 2.3: Structure physique du système de transport

Soit le convoyeur dont les caractéristiques suivantes : sa longueur est L, sa vitesse d'entraînement v_c et la capacité maximale de transport Q_c produits (avec une densité q_c pièces/m).

Pour ce système, il est possible de concevoir un modèle hybride dans lequel le marquage des places continues P_2 et P_5 représente le nombre de produits existant dans les deux zones de stockage. L'entrée, respectivement la sortie du système de convoyage, sont représentées par les transitions continues T_3 et T_4, pour lesquels les vitesses de franchissement maximal ont la valeur :

$V_3 = V_4$(pièces/min) = vitesse du convoyeur × densité maximale = v_c (m/min) × q_c (pièces/m).

On associé à la transition discrète t_2 un délai qui représente le temps instantané sur le convoyeur, et il a comme valeur :

délai (min) = longueur du convoyeur (m) / vitesse du convoyeur (m / min)

la capacité maximale des places disponible sur le convoyeur Q_c à comme valeur :

Q_c (pièces) = longueur du convoyeur (m) × densité maximale (pièces/m)

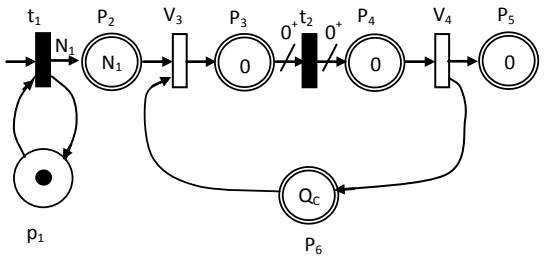


Figure 2.4: Le modèle Réseau de Pétri hybride du convoyeur

La place P_4 modélise la capacité maximale de transport et la durée t_2 associé à la transition discrète T_2 a comme correspondance l'intervalle de temps nécessaire pour un produit à parcourir l'entière longueur du convoyeur ($t_2 = L / v_c$).

Un arc $P_3 – T_2$ et $T_2 – P_4$, avec un poids représenté par « 0^+ », pour une très faible quantité d'un produit transporté qui arrive sur le convoyeur, la transition T_2 est validée et son franchissement, à la fin de temporisation t_2 – transfère le marquage entre les places P_3 et P_4.

Cette manière de représentation permet - par la structure P_3–T_2–P_4 - la modélisation effective du transport des produits.

Par ailleurs, une partie du lot peut être accumulée à la sortie, si la totalité du flux sortant du convoyeur ne peut être absorbée par le convoyeur ou par la machine située en aval (phénomène d'accumulation). En revanche, les réseaux de Petri lots étendent les RdP hybrides, permet de modéliser les systèmes de production dans lesquels existent les phénomènes d'accumulation et de retard pur, définissent ainsi un nouveau type de place (place lot) et de transition (transition lot). Des vitesses de franchissement sont associées aux transitions. Un exemple d'application utilisant les RdP lots pour l'optimisation du fonctionnement des lignes de production à haute cadence est traité dans [11].

[49] Les processus lots discrets jouent un rôle important dans les processus industriels. Ils caractérisent une large classe de systèmes à événements discrets (SED) tels que les systèmes de production, les systèmes de stockage. Un lot est une quantité de produits, caractérisé par une taille (nombre fini). La taille d'un lot correspond au nombre d'entités (produits, commandes, tâches…) qui le constituent. Les différents flux caractérisant ces systèmes tels que le flux matériel et le flux de la matière première évoluent en quantités (lots) finies de tailles généralement variables.

Les réseaux de Pétri lots introduisent dans les Pétri hybrides un nouveau type de place : la place lot, et un nouveau type de transition : la transition lot (Fig.) [8], [27]. On peut représenter par un modèle Réseau de Pétri lots (Fig.), [79].

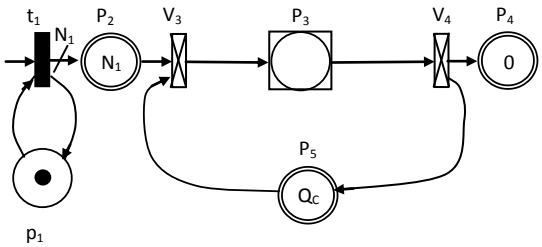


Figure 2.5: Réseau de Pétri lots modélisant un fonctionnement pour le système de transport

Le comportement d'une transition-lots représenté par une vitesse de franchissement lorsque cette transition est validée, la vitesse maximale associée à ces transitions est:
V = vitesse du convoyeur × densité maximale (exprimée ici en pièces/mn)

Le marquage d'une place-lots est défini par un ensemble de lots, chaque lot étant défini par trois paramètres fonctions du temps (soit quatre paramètres avec le temps).

Ils permettent (par leur formalisme et leur dynamique hybride à événements discrets et à temps continus) de connaître à tout instant la position des fronts d'accumulation et l'état global du système.

Le flux de produits est conditionné par la vitesse d'entraînement et les débits d'entrée et de sortie. Un lot cohérent interne, est défini comme un ensemble de pièces ayant les mêmes caractéristiques de répartition sur le convoyeur pendant un intervalle de temps, [27], [79].

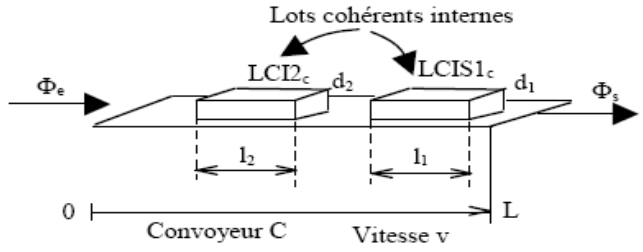


Figure 2.6: cohérence entre les lots

L'état d'un lot se caractérise par une longueur l_n, une densité d_n, une coordonnée x_n et la date t à laquelle il se trouve à cette coordonnée (lot cohérent interne $LCIn_c$ (t) = [l_n (t), d_n (t), x_n (t)]$_c$). Le convoyeur, à son tour, est caractérisé par sa longueur, sa capacité maximale, sa vitesse d'entraînement et par son flux d'entrée et son flux de sortie (C: L, Q_c, V, Φ_e, Φ_s).
L'aspect hybride est défini sur les lots par un avancement temporel de type événementiel et par des équations continues dévolutions.

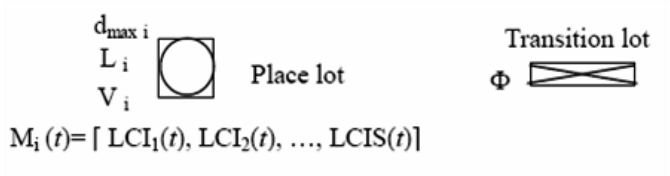


Figure 2.7: Nouveaux types de nœuds dans un Réseau de Petri lot

On peut définir un réseau lot temporisé [27] par un ensemble RPL = < RP, f, c, Tempo, M_0> tels que:
a) RP = <P, T, Pre, Post> est un réseau de Pétri où P est un ensemble de places et T un ensemble de transitions, avec P∩ T = Ø et P ∪ T≠ Ø, Pre (P_i, T_j) est la fonction d'incidence avant et Post (P_i, T_j) la fonction d'incidence arrière,
b) f : P ∪ T → {D, C, L}, appelée fonction lot, indique pour chaque nœud s'il est discret, continu ou lot,
c) c: P $\rightarrow \Re^+.\Re^+.\Re^+$ avec f(P) = L; $P_i \rightarrow \{V_i, d_{max\ i}, L_i\}$, la fonction caractéristique, qui associe à chaque place lot trois valeurs continues et constantes: une vitesse d'entraînement, une densité maximale et une longueur,
d) Tempo est l'application qui associe à chaque transition une valeur temporelle: si $f(T_j)$ = D, alors Tempo (T_j) = d_j, durée associée à la transition discrète T_j, exprimée en unités de temps, si $f(T_j)$ = C ou L, alors tempo (T_j) = Φ (T_j) = Φ_j, flux maximum associé à la transition T_j, exprimée en unité de marques (pièces) / unités de temps,
e) M_0 est le marquage initial.

M(t) est défini, à un instant donné t, par un nombre entier pour une place discrète, un réel positif pour une place continue et à travers un ensemble ordonné de lots cohérents internes pour une place lot.
Une condition nécessaire de validation d'une transition lot est que chacune de ses places lots amont doit comporter un lot de sortie. Pour le franchissement des transitions discrètes et continues, on enlève et on ajoute des marques (réels ou entières) à leurs places discrètes ou continues amont et aval, alors que pour le franchissement d'une transition lot, des lots sont crées ou supprimés à l'intérieur des places lots.

Un lot est défini à un instant t par: la longueur (c'est-à-dire longueur du convoyeur qu'il occupe), la densité (si la transition-lots d'entrée sur le convoyeur est faiblement validée, les

pièces constituant le lot ne sont pas déposées sur le convoyeur à une vitesse suffisante pour présenter la densité maximale), et la position de la tête de lot sur convoyeur:
lot i = (longueur$_i$ (t), densité(t), x_i(t), t).
Si le convoyeur est vide, m le nombre de pièces est défini par:
m = longueur du convoyeur × densité maximale (exprime le nombre de pièces)

Après l'établissement du modèle de système, une étape de validation est doit être disposé, afin de vérifier la compatibilité et le bon fonctionnement du modèle.

7. Validation et vérification des modèles utilisés

La validation est le rapport d'affinité d'un modèle par rapport au cahier des charges qu'il est supposé de représenter. Cette activité aide à l'amélioration de la sûreté de fonctionnement des systèmes modélisés puisqu'elle permet de détecter au plus tôt des erreurs de conception, et donc de renforcer la confiance accordée, par les utilisateurs et les concepteurs, aux modèles établis pour ces systèmes.
La simulation est utilisée pour la validation des modèles, puisque elle permet d'observer leur comportement, il reste seulement le problème du choix adéquat des grandeurs d'entrées à faire évoluer de façon à reproduire fidèlement le comportement du système modélisé et son environnement.

8. Conclusion

Dans ce chapitre nous nous somme intéressés à la modélisation des systèmes de production manufacturiers par les réseaux de pétri, qui sont connus pour être l'outil puissant de modélisation et d'analyse. Les réseaux de pétri hybrides, extensions du modèle de base, offrent la possibilité de modéliser la partie discrète et continue simultanément, ainsi que les systèmes dynamiques hybrides sur les mêmes bases conceptuelles. Le partage des ressources génère souvent des conflits. Ils sont résolus par des ordonnancements sur la ressource en question. Parmi les approches utilisées pour la prise en charge de ces questions, on exploite l'association de méthodes d'aide à la décision, et notamment les méta_heuristiques.
Elles doivent répondre à un maximum d'exigences, dont l'évaluation de la performance, la modélisation des contraintes temporelles, la synthèse d'une commande temporelle (c'est-à-dire le réglage des instants de tirs des transitions), la vérification du respect des contraintes etc.
Les méthodes d'aide à la décision permettant de traiter cette problématique feront l'objet du chapitre suivant.

CHAPITRE 3

Les Méthodes de conduite

1. Introduction

L'environnement actuel des entreprises, les oblige à être de plus en plus compétitives. Pour cela, elles doivent optimiser leur organisation, leur management, et leurs systèmes de production pour faire face à des problèmes de plus en plus complexes. Nous nous intéressons dans ce travail, à la partie production de l'entreprise. Les données sont variables et quelque fois même aléatoires. Optimiser la production veut dire, favoriser l'exécution des opérations simultanément lorsque cela est possible, exploiter au mieux les ressources communes ou moyens de production, en quantité limitée forcément, etc.. En pratique, il s'agit en fait de garantir un ordonnancement de tâches et une exploitation des moyens de façon optimale, sous le respect des quantités et des délais imposés.
Ce problème classique et connu est pris en charge par deux approches, l'une classique s'appuyant sur les méthodes exactes et l'autre exploite les heuristiques et méta-heuristiques.

- les méthodes exactes (complètes) garantissent la complétude de la résolution. Elles ont permis de trouver des solutions optimales pour des problèmes de taille raisonnable, mais elles rencontrent généralement des difficultés face aux applications de taille importante. Le temps de calcul nécessaire pour trouver une solution peut augmenter exponentiellement avec la taille du problème. C'est une des contraintes qui limite son utilisation et favorise la recherche d'autres voies.
- les méthodes approchées (incomplètes) qui perdent la complétude pour gagner en efficacité. Elles constituent une alternative très intéressante pour traiter les problèmes d'optimisation de grande taille. Il est souvent nécessaire de trouver des modes de résolution qui fournissent une solution de bonne qualité dans un laps de temps raisonnable. C'est ce que font les heuristiques. Les heuristiques les plus efficaces sont appelées les méta-heuristiques et s'adaptent particulièrement au type de problème posé.

Parmi les méta-heuristiques apparaissent deux approches ayant une grande utilité dans la résolution de nombreux problèmes d'optimisation. La première de ces approches est la Recherche Locale (ou méthodes de voisinage). Elle comprend la technique du Recuit Simulé [50] et la Recherche Taboue [36]. La deuxième approche est la Méthode Évolutive. Elle est plus récente que les précédentes, et est reconnue plus par le nom d'algorithmes génétiques dont les origines remontent aux travaux de Holland [42].
Nous avons choisi d'exploiter et d'adapter ces trois méthodes dans le cadre de ce travail.
Leur amélioration est également une de nos motivations. Leur description ainsi que les adaptations proposées sera décrites et détaillées par la suite dans ce chapitre.

2. Les méthodes d'optimisation

La fonction d'optimisation spécifie le choix optimal. Elle est définie par toutes les méthodes qui servent à déterminer l'optimum d'une fonction avec (ou sans) contraintes. Elle consiste à trouver le meilleur choix entre un nombre fini de choix exprimé par une fonction (dite fonction coût) à minimiser (ou à maximiser) avec (ou sans) contraintes, sur un ensemble de définitions finies. Parmi les méthodes d'optimisation, on trouve celles qui sont basées sur les méthodes exactes. Un des avantages de ces méthodes réside précisément dans la possibilité de contrôler le temps de calcul ou la qualité de la solution trouvée qui tend à s'améliorer progressivement au cours du temps, mais présentent l'inconvénient de ne pas résoudre les problèmes qui prennent un temps de calculs trop important. Il existe des méthodes qui évitent de se trouver dans cette situation, comme nous le verrons plus loin, ce sont les méthodes basées sur les méta-heuristiques. Elles contiennent souvent une technique ou procédure permettant d'éviter de se retrouver piégé dans des minima locaux, tout en explorant davantage

l'ensemble de l'espace des solutions de façon à augmenter la probabilité de rencontrer le minimum optimal, c'est-à-dire le minimum global.

2.1 Les méthodes exactes et leurs limites

Les méthodes exactes s'appuient sur une direction de recherche qui peut être fournie par les dérivées de la fonction objectif. Elles peuvent être efficaces lorsque la solution initiale est proche de l'optimum recherché. Parmi les méthodes exactes on trouve :

2.1.1 Les méthodes Branch & Bound [62]

Elles s'appuient sur l'algèbre de Moore ou algèbre des intervalles qui permet d'estimer l'intervalle de variation d'une fonction à partir de son expression analytique et des intervalles de variation des paramètres [55]. Elles sont déterministes d'ordre zéro et global mais ne s'appliquent que pour des modèles analytiques. Les plus évoluées sont capables de localiser tous les optima de la fonction avec une certaine précision liée à la taille des subdivisions de l'espace et aux incertitudes d'estimation des intervalles de variation de la fonction objectif.

Elles sont généralement coûteuses en temps de calcul et ne peuvent s'appliquer que pour un nombre réduit de paramètres.

2.1.2 Méthodes mathématiques [62]

Pour déterminer un optimum, les méthodes mathématiques se basent sur la connaissance d'une direction de recherche donnée souvent par le gradient de la fonction objectif par rapport aux paramètres. Elles génèrent une suite de points (X_k, $k \in \aleph$) qui convergent vers un minimum local X^* de la fonction f vérifiant l'équation suivante ou φ (X^*, r) = 0 suivant le traitement des contraintes utilisées.

L'inconvénient principal des méthodes à base de gradient est que la dérivée de la fonction f n'est pas toujours connue, dans ce cas, il faut l'estimer par les différences finies.

$$\frac{\Delta f}{\Delta x_i} = \frac{f(x_1,...,x_i+\lambda,...,x_n) - f(x_1,...,x_i,...,x_n)}{\lambda}$$

Dans ces conditions le choix du pas du gradient λ est très important, il conditionne la bonne détermination de la direction de recherche.

Parmi ces méthodes, la méthode du gradient conjugué, la méthode quasi-Newton, la méthode SQP et la méthode de Powell sont présentées brièvement.

2.1.3 La méthode des directions conjuguées de Powell [62]

Lorsqu'il n'est pas possible de calculer le gradient, la méthode de direction conjuguée propose de trouver l'optimum uniquement par des recherches linéaires [67]. Elle effectue n recherches linéaires successives suivant des directions conjuguées qui sont modifiées à chaque itération pour accélérer la convergence.

Ces méthodes convergent rapidement et précisément vers l'optimum si celui-ci est proche du point initial. Si ces méthodes sont intéressantes en raison de leur grande rapidité de convergence, elles ont plusieurs inconvénients :

1. les valeurs de la fonction objectif et éventuellement de ses dérivées doivent être accessibles.
2. lorsque le gradient de la fonction n'est pas calculable directement, sa détermination par la méthode des différences finies est toujours délicate à cause de problèmes liés au choix du pas de variation pouvant conduire à des problèmes de convergence.
3. ces méthodes nécessitent la résolution de systèmes matriciels pouvant être mal conditionnés.

4. la convergence est exclusivement locale. L'optimum trouvé dépend du point initial. La sensibilité par rapport aux conditions initiales est importante.

Il existe de nombreux problèmes d'optimisation combinatoire pour lesquels la détermination d'une solution exacte risquerait de prendre un temps de calculs trop important. Soit lorsque la fonction objectif possède plusieurs optimums, elles peuvent converger vers un optimum local. Soit quand le nombre de combinaisons possibles devient exponentiel par rapport à la taille du problème, dans ce cas le temps de calcul devient rapidement critique. On dit qu'ils ne peuvent pas être résolus en un temps polynomial par les algorithmes exacts. Ils font ainsi partie de la classe des problèmes dits NP-Difficiles. On veut alors disposer d'une solution de bonne qualité (c'est-à-dire assez proche de l'optimale) en utilisant une quantité de temps de calculs et/ou de mémoire limitée. L'optimalité de la solution ne sera pas garantie.

Cependant le temps nécessaire pour obtenir cette solution sera beaucoup plus faible. Il s'agit des méthodes méta-heuristiques. Elles sont particulièrement utiles pour les problèmes qui nécessitent une solution pour résoudre des problèmes difficiles sur des instances numériques de grande taille.

2.2 Les méta-heuristiques et leurs adaptations à un problème d'optimisation

Les méta-heuristiques d'optimisation sont des algorithmes applicables à une grande variété de problèmes. Elles sont apparues à partir des années 1980 dont le but est de résoudre au mieux des problèmes d'optimisation difficile. Elles disposent d'une solution de bonne qualité (assez proche de l'optimale) en utilisant une quantité de temps de calculs et/ou de mémoire limitée. Mais il est le plus souvent impossible de prévoir avec certitude l'efficacité d'une méthode donnée, c'est-à-dire que l'optimalité de la solution ne sera pas garantie quand elle est appliquée à un problème donné. Elles ne nécessitent pas de point de départ, elles s'appuient sur des mécanismes probabilistes et aléatoires qui explorent efficacement l'espace de recherche, et convergent vers l'optimum global.

Ces méthodes ont en commun les caractéristiques suivantes:
- une partie stochastique qui permet de faire face à l'explosion combinatoire des possibilités;
- inspirées par des analogies avec la physique pour le recuit simulé, avec la biologie pour les algorithmes génétiques, ou avec l'éthologie pour les colonies de fourmis, etc..
- difficultés de réglage des paramètres mêmes de la méthode,
- généralité et application possible à une large classe de problèmes,
- efficacité pour de nombreux problèmes,
- possibilité de compromis entre qualité des solutions et le temps de calcul,
- nécessité d'adaptation de la méthode au problème traité,
- difficulté de prévoir la performance.
- optimum non garanti,

L'adaptation d'une méthode méta-heuristique à un problème d'optimisation particulier nécessite de modéliser adéquatement certains points:
- L'ensemble S des solutions : il définit le domaine dans lequel la recherche d'un optimum va s'effectuer.
- La fonction objectif f à minimiser : son rôle principal est de mesurer la valeur des solutions admissibles dans un espace de recherche composé de mauvaises solutions et de solutions qui se trouvent dans des régions autour d'un minimum local.
- le voisinage d'une solution : il indique les déplacements possibles dans S. L'exploration de S consiste à parcourir un chemin qui se dirige à chaque pas d'une solution vers une solution voisine.

- L'opérateur de combinaison : il permet de générer de nouvelles solutions à partir des solutions présentes dans la population courante.

Afin de ne peut être piégé par un minimum local, une idée qui est à la base de toutes les méta-heuristiques dites de voisinage (Ex. recuit simulé, recherche tabou), autorise de temps en temps, des mouvements de remontée, ou d'une autre manière d'accepter une dégradation temporaire de la situation lors du changement de la configuration courante. À chaque méta-heuristique on associe un mécanisme de contrôle des dégradations, qui permet dès que possible de s'extraire du piège que représente un minimum local, pour partir explorer une autre "vallée" plus prometteuse (Ex. critère d'aspiration pour la recherche tabou).

Un autre type de méta-heuristiques dites distribuées (Ex. les algorithmes génétiques) ont elles aussi des mécanismes permettant d'éviter d'être figé dans un minimum local de la fonction objectif. Ces mécanismes (Ex. la mutation dans les algorithmes génétiques) que représente le contrôle en parallèle de toute une "population" de solutions, permettent de lutter contre les minima locaux.

Nous allons illustrer et détailler ces trois méthodes dans la suite dans ce chapitre.

3. Les Méta-heuristiques étudiées

3.1 Les Algorithmes Génétiques

Les principes fondamentaux de ces algorithmes ont été exposés par Holland [42] dans le cadre de l'optimisation mathématique. L'édition en 1989 de l'ouvrage de référence écrit par D.E. Goldberg (1989), décrivant l'utilisation de ces algorithmes pour la résolution de problèmes concrets. Ces algorithmes s'inspirent du fonctionnement de l'évolution naturelle, notamment la sélection de Darwin, et la procréation selon les règles de Mendel. Les algorithmes génétiques sont des algorithmes d'optimisation stochastiques, appartenant à la famille des algorithmes évolutionnaires, fondés sur les mécanismes de sélection naturelle et de la génétique. Son fonctionnement se base sur l'exploration d'une population de solutions (chromosomes) initiales, qu'on évalue leur performance relative (fonction fitness) et ainsi quantifier leur qualité. Sur la base de ces résultats on crée une nouvelle population de solutions potentielles, obtenue en appliquant les opérateurs évolutionnaires: la sélection, le croisement et la mutation. Ce cycle sera répété jusqu'à l'obtention d'une solution satisfaisante. A chaque itération apparaissent dans la population les individus les plus adaptés à leur environnement, c'est-à-dire les optimaux pour la fonction de performance. Par analogie avec l'optimisation, les algorithmes génétiques, considèrent l'environnement comme la fonction de performance à optimiser, et les solutions potentielles du problème comme des chromosomes. La qualité de la solution correspondant à chaque chromosome est quantifiés via sa propre fitness, et seuls les plus aptes survivent à la sélection naturelle et peuvent se reproduire. Les algorithmes génétiques peuvent ainsi traiter une grande variété de problèmes numériques.

En passant à la pratique, l'application de l'algorithme génétique, doit établir les composantes suivantes:

- représenter sous forme de chromosome, les solutions.
- initialiser la population de solutions candidates.
- évaluer la qualité de chaque individu, en termes de fitness, par une fonction objectif.
- appliquer l'opérateur de sélection des parents.
- appliquer des opérateurs de recombinaison qui produisent l'ensemble des nouveaux individus.
- établir un opérateur de remplacement.
- associer un critère d'arrêt à l'algorithme génétique.

Parmi ces composantes, certaines dépendent entièrement de l'application (évaluation, représentation, initialisation, croisement et mutation), alors que les autres sont définis indépendamment du domaine d'application (sélection, remplacement).

3.1.1 Principes de base

La génétique affecte à chaque individu un ensemble de données qui constitue son code génétique. L'ensemble des gènes caractérise parfaitement l'individu, ou le chromosome. Chaque individu se voit ainsi doté de capacités lui permettant d'évoluer dans son environnement naturel.

Si l'individu est bien adapté, il a une plus grande chance de procréer dans la génération future.

Une des caractéristiques principales des algorithmes génétiques est la robustesse [33]: ils permettent de fournir une ou plusieurs solutions de «bonne» qualité (pas nécessairement optimale, mais suffisante en pratique) à des problèmes très variés.

La simulation du processus d'évolution part d'une population de N solutions du problème représentées par des individus. Cette population est appelée population parent. Le degré d'adaptation d'un individu à l'environnement est exprimé par la valeur de la fonction coût f(x), où x est la solution que l'individu représente. Le choix de la fonction d'évaluation ou fonction coût est guidé pour le désir d'avoir un modèle qui relie les données et possède une bonne performance. Plus le coût de la solution que représente l'individu est faible plus il est mieux adapté à son environnement. Au cours de cette population, intervient alors la sélection de parents, qui favorise la survie des individus les mieux adaptés au détriment des plus faibles, en utilisant une procédure de sélection qui tient compte de l'adaptation respective de chaque individu. De nouvelles solutions produites à travers les opérateurs génétiques, tels que le croisement et la mutation des individus selon une probabilité donnée. Ces nouvelles solutions constituent la nouvelle population qui est obtenue par le choix de N individus parmi les populations parent et enfant, est appelée aussi génération suivante. Ce processus est répété pendant un certain nombre de générations, en espérant que les optima de F (fitness) apparaissent dans la population. En itérant ce processus, on produit donc, une population plus riche en individus mieux adaptés. Le principe de base d'un algorithme génétique est décrit par le programme suivant:

```
POP, POP' et Fitness: tableaux de taille N
        POP:= initialisation de la population
        f(x) := min [f(x_i)]
                1≤ i ≤N

        f_min := f(x)
        x_min := x
        Répéter
           Fitness (évaluer les individus de la population POP)
                Répéter (phase de reproduction génétique)
                Sélection
                Opérateurs de reproduction (croisement, mutation)
                Jusqu'à ce que POP' remplie
           POP:= sélectionner nouvelle population à partir de (POP, POP')
           f(x) := min[f(x_i )]
                   1 ≤i≤ N

           Si   f(x) < f_min
```

$f_{min} := f(x)$
$x_{min} := x$
Fin de Si
Jusqu'à conditions d'arrêt satisfaites

Pseudo code d'un algorithme génétique de base

Le principe d'une méta-heuristique est de trouver une configuration adéquate au problème. Avec les algorithmes génétiques, cette configuration inclue la taille de la population, le type de sélection, le type et le taux de croisement P_c, le type et le taux de mutation P_m (habituellement le taux de mutation entre 0.01% et 0.1%, mais l'utilisateur peut adapter ces variables selon ses besoins, en exécutant plusieurs tests de la méta-heuristique). Le remplacement est la phase dans laquelle aucun des individus de la population ne se retrouve dans la population n + 1.

La mise en œuvre d'un algorithme génétique nécessite de fixer convenablement les paramètres gérant l'évolution des populations au cours des générations: taille de la population, nombre de générations, taux de croisement et taux de mutation.
L'algorithme génétique se base principalement sur la définition de plusieurs paramètres que nous allons détailler par la suite:

3.1.2 Codage du chromosome

Le choix du codage des données dépend de la spécificité du problème traité. C'est la représentation manipulable par l'algorithme génétique en vue de son évolution, et conditionne fortement l'efficacité de l'algorithme.

Ce dernier opère sur un ensemble de chromosomes numériques, chargés de trouver la séquence optimale, l'utilisateur interprète ensuite le chromosome et lui restitue une valeur de la force d'adaptation.

Plusieurs codes d'informations pour les variables numériques sont disponibles: codage binaire naturel (représentation sous forme de chaîne binaire) qui est le plus fréquemment utilisé [22]**,** codage réel (représentation directes des valeurs réels de la variable), et le codage de Gray.

● La représentation binaire

Le codage binaire est le cadre général des AG traditionnels. Chaque individu est représenté par un vecteur binaire (ou chaîne de bits), ou chaque élément prend la valeur 0, 1:
$I = (a_1, a_2, \ldots, a_l) \in \{0,1\}$, où l est la taille du vecteur (nombre de bits).
Cette représentation s'adapte bien à des problèmes où les solutions potentielles ont une représentation binaire canonique, comme les problèmes booléens.

Néanmoins, ce codage présente trois inconvénients majeurs:

· Les performances de l'algorithme sont diminuées lorsque la longueur de la chaîne augmente.

· Deux nombres décimaux voisins peuvent être très éloignés dans un codage binaire naturel (1000 et 0111): falaise de Hamming. Ce problème peut être réglé en remplaçant le code binaire naturel par le code binaire de Gray.

● Codage de Gray

Avec le codage binaire, deux configurations proches dans l'espace des paramètres peuvent avoir deux chromosomes très distincts, (Ex. les chaînes 1000 et 0111) même si elles

correspondent à deux configurations réelles voisines. Cette caractéristique peut s'avérer pénalisante pour la recherche locale par croisement.
L'utilisation du code de Gray a été recommandée pour répondre à ce problème [12]. En effet, avec ce code, les entiers adjacents ne diffèrent que d'un seul bit. Le passage entre deux configurations réelles voisines ne nécessite que de modifier un seul bit dans le chromosome.

Le passage du code binaire au code de Gray est effectué de la manière suivante:

$$b_j^{gray} = \begin{cases} b_j & si\ j = l(x_i) \\ b_j \oplus b_{j-1} & si\ j \le l(x_i) \end{cases}$$

où $\oplus$ représente l'addition modulo 2.
$l(x_i)$ est le nombre de bits du gène numéro i correspondant au paramètre x_i.
La transformation inverse s'obtient avec l'équation suivante:

$$b_j = \bigoplus_{k=j}^{l(x_i)} b_k^{gray}$$

Si on considère que le chromosome est représenté en code de Gray, on effectuera d'abord la transformation de l'équation avant un décodage binaire standard.

● **Codage réel**

Ce codage présente des avantages majeurs. Il est plus précis que le codage binaire, et l'espace de recherche est le même que l'espace du problème. Il a le mérite d'être simple à utiliser, et l'évaluation de la fonction coût est plus rapide.

3.1.3 Évaluation des individus

La performance d'un individu est mesurée par une fonction de performance ou d'adaptation (fitness) qui se déduit de la fonction objectif. La construction d'une fonction d'adaptation appropriée à partir de la fonction objectif est très importante, pour garantir un bon fonctionnement des AG, et elle reflète la capacité de l'individu à survivre dans son environnement. La phase d'évaluation consiste donc, à calculer la «force» d'adaptation de chaque individu de la population. On calcule la qualité de chaque individu de la population pour déterminer sa probabilité de sélection. Plus la qualité est élevée mieux l'individu est adapté. On obtient cette qualité à partir de la fonction de coût f par une transformation telle que : $\text{Qualité}(x) = f_{max} - f(x)$
Ou bien, la force de l'individu à partir de la fonction de coût C, par une transformation telle que:

$$F(x) = C_{max} - C(x)$$

3.1.4 Génération de la population initiale

La population initiale constitue le point de départ de l'algorithme génétique. En effet, il est intuitivement préférable d'avoir une génération initiale d'individus qui ne viole pas les contraintes du problème et qui constitue l'ensemble des solutions potentielles du problème.

Des connaissances à priori sur le problème traité permettent ainsi de générer les individus dans un domaine particulier afin d'accélérer la convergence de l'algorithme génétique.
Plusieurs méthodes sont employées pour la génération de la population initiale, parmi lesquelles on trouve: le tirage aléatoire, l'utilisation d'heuristiques, ou directement introduite par l'utilisateur.

3.1.5 Opérateur de sélection

La sélection est un opérateur principal utilisé dans l'AG dont le premier rôle est de maintenir les meilleurs individus dans la population. Elle consiste à choisir les individus de la population courante qui vont survivre et se reproduire. L'opérateur de sélection joue ainsi un

rôle primordial dans la détermination de la performance des nouvelles générations et donc dans l'amélioration de la qualité des solutions et de progresser vers la bonne direction. Elle est réalisée en fonction de la valeur de la fonction de coût qui évalue chaque solution.

Plusieurs techniques de sélection sont utilisées par les chercheurs. Elles peuvent être déterministes ou stochastiques [16]:

● **La sélection déterministe**

Elle consiste à garder les meilleurs individus au sens de leurs coûts et à rejeter le reste, ce qui implique leur classement ou « ranking ». C'est la méthode de sélection la plus simple à mettre en œuvre, puisqu'elle consiste à choisir les n meilleurs individus parmi la population, avec n, un paramètre fixé (Ex. égal à la moitié de la population).

● **Une sélection stochastique**

Elle peut être réalisée par la technique de la roulette pondérée [38],

● **Sélection par tournois**

L'idée de cette méthode est de sélectionner aléatoirement un nombre *k* d'individus dans la population, et de choisir parmi eux celui qui a la meilleure performance. On organise autant de tournois que d'individus à sélectionner. On peut décider de faire des tirages avec, ou sans remise. Le tirage sans remise permet d'éviter de favoriser excessivement un individu. Dans la mesure où seul l'ordre des éléments importe dans la sélection par tournois, il est parfois possible de simplifier l'évaluation de la fonction objectif, et donc de gagner en temps de calcul.

● **Roulette (sélection proportionnelle)**

La première sélection mise en place pour les AG est le tirage à la roulette, ou Roulette Wheel Sélection (RWS) introduite par Goldberg. C'est une méthode stochastique qui exploite la métaphore d'une roulette de casino, qui compterait autant de case que d'individus dans la population. La largeur de la case d'un individu $\overrightarrow{x_i}$ est proportionnelle à sa performance f($\overrightarrow{x_i}$): $\frac{f(\overrightarrow{x_i})}{\sum_j^n f(\overrightarrow{x_j})}$. La roue étant lancée, l'individu sélectionné est désigné par l'arrêt de la roue sur sa case.

L'espérance n_i de nombre de copies d'un élément $\overrightarrow{x_i}$ de la population courante est donné par l'expression : $n_i = \frac{N}{\sum_j^N f(\overrightarrow{x_j})} f(\overrightarrow{x_i})$

où N est le nombre d'individus. L'espérance maximale Max (n_i) dans l'ensemble des éléments de la population est appelée la pression sélective.

Cette méthode favorise les meilleurs individus. Mais tous les individus ont toujours des chances d'être sélectionnés. Cependant, la méthode peut causer une perte de la diversité de la population si la pression sélective (ou n_i du meilleur) est élevée.

● **Sélection par rang**

Elle consiste à attribuer à chaque individu son classement par ordre croissant des valeurs de la fonction objectif. On prélève ensuite une nouvelle population à partir de cet ensemble d'individus ordonnés, en utilisant des probabilités indexées sur les rangs des individus:

$$probabilité\ de\ sélection\ (parent\ i) = \frac{rang\ (parent\ \ i)}{\sum_{j \epsilon population} rang\ (parent\ j)}$$

● **Sélection aléatoire**

La sélection se fait aléatoirement, uniformément et sans intervention de la valeur d'adaptation. Chaque individu a donc une probabilité uniforme (1 / N) d'être sélectionné. En général, la convergence de l'algorithme génétique est lente en utilisant cette méthode.

● **Décimation**

Seuls les p meilleurs individus de la population sont sélectionnés et autorisés à se reproduire par croisement et mutation jusqu'à obtenir une population de même taille N que la population initiale [68].

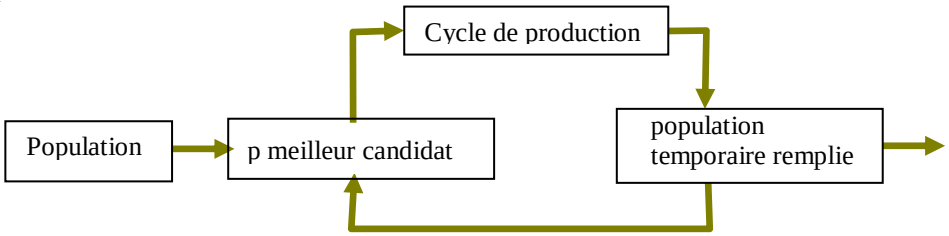


Figure 3.1 : Processus de sélection par décimation

L'inconvénient de la sélection est qu'elle fait sélectionner les meilleurs individus de la population, ce qui conduit à la perte de la diversité des solutions. L'algorithme risque ainsi de converger brusquement. Pour avoir une bonne exploration de l'espace de recherche, des opérateurs de croisement et de mutation, sont appliqués aux individus sélectionnés, pour en créer d'autres. C'est le rôle des opérateurs de croisement et de mutation.

Les individus sélectionnés seront utilisés pendant la phase de reproduction, et la structure de leurs chromosomes sera modifiée pour construire les nouveaux individus de la génération suivante. La phase de reproduction est composée de deux principaux opérateurs: le croisement et la mutation. Le croisement réalise une opération qui nécessite deux parents, tandis que la mutation est une opération unaire, utilisée pour introduire une faible variation dans la solution afin de changer la direction de recherche.
Les opérateurs génétiques se distinguent selon le type de codage, binaire ou réel.

3.1.6 Opérateur de croisement

Le croisement est une étape essentielle pour l'algorithme génétique, car il permet l'exploration de l'espace de recherche. Une fois que la population intermédiaire (choisie par l'opérateur de sélection) est déterminée, les individus seront répartis en couples. Les chromosomes sont alors copiés et recombinés de façon à former, en général, deux descendants possédant des caractéristiques issues des deux parents. On forme ainsi la génération suivante.

L'opérateur de croisement s'opère avec une probabilité p_c, fixée selon le problème concerné. Plus ce taux est élevé, plus il y a de nouvelles structures qui apparaissent dans la population. Mais, s'il est trop élevé, les bonnes solutions risquent d'être modifiées trop vite par rapport à l'amélioration que peut apporter la sélection. D'autre part, si le taux de croisement est très faible, la recherche risque de stagner, à cause du faible taux d'exploration.
Parmi les méthodes de croisement les plus utilisées on peut souligner :

● **Croisement binaire**

Plutôt que de travailler sur l'espace de recherche lui-même, les algorithmes génétiques peuvent s'appliquer sur des chaînes de bits. Dans ce cas, le croisement consiste à échanger une partie du matériel génétique des deux parents pour former deux nouveaux individus

(enfants). En pratique, l'échange n'est effectué que si un nombre aléatoirement, tiré entre 0 et 1, est inférieur à p_c. Plusieurs types de croisement binaire sont possibles.

Croisement en un point

Pour chaque couple, un point de croisement est choisit au hasard. Les composantes situées à gauche de ce point sont conservées et celles à droite sont échangées entre les deux individus. Les enfants ainsi constitués sont placés dans la population suivante P (g+1).

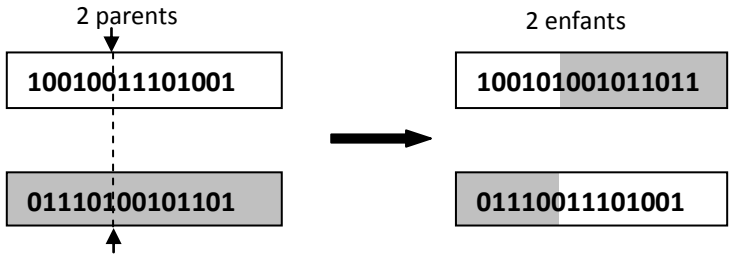


Figure 3.2 : Principe du croisement en un point

Croisement en deux points

Les deux points de croisements sont également choisis au hasard puis, les séquences des chromosomes situées entre les deux points sont échangées. Elle est généralement considérée comme plus efficace que le précédent.

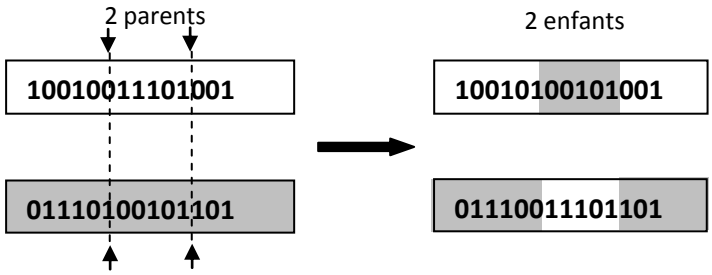


Figure 3.3 : Principe du croisement en deux points

Croisement multiple

Le croisement précédent peut être généralisé en croisement en k-points de coupe, [26] générant k+1 sous chaînes pour chaque chromosome. Les deux chromosomes fils sont obtenus par concaténation des sous chaînes en alternant les parties venant de chaque parent.

Croisement uniforme

Il opère à l'aide d'un masque qui représente les tirages aléatoires, pour décider la transmission de la valeur de l'allèle à l'un ou l'autre des descendants. Si, à la même position que l'allèle, la valeur du masque est égale à 1, l'allèle du parent 1 passe à celui de l'enfant 1 et l'allèle du parent 2 passe à l'enfant 2. Sinon, c'est l'inverse qui se produit.

Parent 1	1	1	0	0	1	1	1	1	1	0
Parent 2	1	1	1	1	0	0	0	0	0	1
Masque	1	0	0	1	1	1	0	0	1	1
Enfant 1	1	1	1	0	1	1	0	0	1	0
Enfant 2	1	1	0	1	0	0	1	1	0	1

Figure 3.4: Exemple de croisement uniforme.

Le croisement adaptatif

Pour créer un croisement adaptatif, Spears propose un mécanisme qui met en valeur le croisement uniforme et le croisement en deux points.

Un bit de décision choisi aléatoirement est ajouté en dernière position du chromosome des individus de la population. Le croisement à appliquer est déterminé en fonction de la valeur du bit de décision des deux parents reproducteurs:

- Si les bits de décision des deux parents valent 0, ceux-ci sont recombinés à l'aide d'un croisement uniforme.
- Si les bits de décision des deux parents valent 1, ceux-ci sont recombinés à l'aide d'un croisement à deux points.
- Si les bits de décision des deux parents diffèrent, le type de croisement à appliquer est fixé aléatoirement de façon équiprobable.

● Croisement réel

La différence entre le croisement réel et le croisement binaire réside dans la nature des éléments qu'il modifie. Il ne s'agit plus des bits qui sont échangés à droite du point de croisement, mais de variables réelles.

Des opérateurs simples du type croisement réel en un point, en deux points, multiple ou uniforme peuvent être implantés de la même manière que dans le cas d'un codage binaire. La seule différence réside dans le point de coupe qui doit être choisi entre deux variables du vecteur, ce qui revient à permuter des variables entre 2 chaînes. L'opérateur de croisement en un point sera présenté.

$x = \langle x_1, x_2, \ldots, x_k, x_{k+1}, \ldots, x_n \rangle$ $\quad x' = \langle x_1, x_2, \ldots, x_k, y_{k+1}, \ldots, y_n \rangle$

$y = \langle y_1, y_2, \ldots, y_k, y_{k+1}, \ldots, y_n \rangle$ $\quad y' = \langle y_1, y_2, \ldots, y_k, x_{k+1}, \ldots, x_n \rangle$

Principe de croisement réel en un point

Ces techniques ont un taux d'exploration relativement limité car elles ne font apparaître aucune nouvelle valeur pour les paramètres. Pour palier à cet inconvénient majeur, deux techniques sont applicables :

***Croisement arithmétique**

Le croisement arithmétique est propre à la représentation réelle. Il s'applique à une paire de chromosomes et se caractérise par une pondération aléatoire des chromosomes des deux parents:

Soient x et y deux parents et p un poids appartenant à l'intervalle [0,1] qui garantit que les descendants auront bien leurs gènes x_i dans $[x_{im}, x_{iM}]$. Alors les enfants sont:

$x' = p \cdot x + (1 - p) \cdot y$

$y' = (1 - p) \cdot x + p \cdot y$

x' est constitué de p% du parent x et de (100 - p)% du parent y, et réciproquement pour y'.

***Croisement uniforme**

Elle échange chaque composante entre les deux individus avec une probabilité égale à 0.5.

3.1.7 Opérateur de mutation

Les deux opérateurs précédents, sélection et croisement, permettent à la phase dite d'exploitation de concentrer les éléments de la population courante dans les zones de forte performance. Mais, rien n'assure que ces lieux de concentrations correspondent à la zone dans

laquelle se trouve l'optimum de la fonction considérée. En d'autres termes, elles peuvent se faire sur des optima locaux. Il est donc essentiel, à chaque étape de l'algorithme (à chaque génération) d'exploiter des zones jusqu'alors négligées pour s'assurer que la convergence ne se fasse pas sur un optimum local. Ainsi pour le maintien de la diversité, l'utilité d'une bonne exploration du domaine de recherche, on à recours à l'opérateur de mutation, que l'on applique sur certains éléments. Il consiste à faire perturber l'élément sur lequel il est appliqué.

Il permet d'introduire une certaine information dans la population, qui a pu être perdue lors de l'opération de croisement. L'opérateur de mutation s'applique avec une certaine probabilité, appelée taux de mutation P_m, typiquement compris entre 0.01 et 0.10. Ce faible taux de mutation permet de dire que la mutation est considérée comme un mécanisme d'adaptation secondaire pour les algorithmes génétiques.

● Mutation binaire

La mutation binaire est définie par la modification aléatoire de la valeur d'un allèle dans un chromosome. Elle joue le rôle de bruit, empêche l'évolution de se figer, évite une convergence prématurée vers les optimums locaux et garantit que l'optimum global peut être atteint. Il est appliqué avec une probabilité fixée, p_m. Le taux de mutation rend la recherche trop aléatoire s'il est trop élevé. Par ailleurs, s'il est trop faible, la recherche risque de stagner.

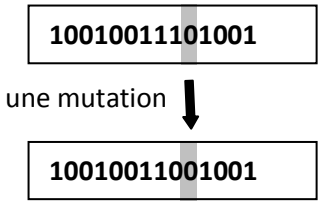


Figure 3.5 : Exemple sur la mutation binaire.

● Mutation réelle

La différence ente la mutation réelle et la mutation binaire réside sur la nature de l'élément qu'elle modifie. Ce n'est plus un bit qui est complémenté, mais une variable réelle qui est de nouveau tirée au hasard dans son intervalle de définition.
On distingue deux types d'opérateur de mutation, uniforme et non uniforme.

Mutation uniforme

Elle sélectionne au hasard un gène k dans la chaîne $x =< x_1, x_2, ..., x_k, ... x_n >$ et génère la nouvelle chaîne $x' = < x_1, x_2, ..., x'_k, ..., x_n >$ où x'_k est une valeur aléatoire prise dans l'intervalle $[x_{km}, x_{kM}]$.

Mutation non uniforme

La mutation non uniforme possède la particularité de modifier les éléments qu'elle altère dans un intervalle de définition variable et de plus en plus petit. Plus les générations avancent, moins la mutation écarte les éléments de la zone de convergence. Cette mutation adaptative offre un bon équilibre entre l'exploration du domaine de recherche et l'affinement des individus. Le coefficient d'atténuation de l'intervalle est un paramètre de cet opérateur.
Il génère une nouvelle valeur x'_k pour le gène k sélectionné au hasard, donné par la formule:

$$x'_k = \begin{cases} x_k + \Delta(g, x_{kM} - x_k) \\ ou \\ x_k - \Delta(g, x_k - x_{km}) \end{cases}$$

où g représente la génération courante et Δ(g, y) une fonction qui retourne un nombre aléatoire dans [0, y] et dont la probabilité de renvoyer un nombre proche de 0 augmente lorsque g augmente. Cette propriété de Δ permet à l'opérateur de chercher de façon uniforme dans l'espace des solutions dans les premières générations puis de façon plus locale en fin d'exécution. Michalewicz [57] a proposé pour Δ la fonction:

$$\Delta(g, y) = y.(1 - rand^{(1-\frac{g}{G})^b})$$

où G est le nombre maximum de génération, rand un nombre aléatoire dans [0,1] et b un paramètre de l'opérateur qui détermine le degré de dépendance au nombre d'itérations.

3.1.8 Remplacement générationnel

Dans ces études sur la performance de cette technique, De Jong a proposé de définir un paramètre G (Generation Gap), spécifiant le taux de remplacement à chaque génération, afin d'augmenter la durée de vie des meilleurs individus. G=1 correspond au remplacement générationnel total. Avec cette approche, la population des enfants remplace entièrement la population parente. La durée de vie d'un individu est alors d'une seule génération.

3.1.9 Critères d'arrêt

Le critère d'arrêt est une caractéristique essentielle des AG. Un critère peu performant peut en effet conduire à de nombreuses évaluations inutiles de la fonction d'adaptation. Les critères proposés dans la littérature sont:

1. nombre maximal de générations (itération). Ce nombre doit être suffisant pour permettre une exploration correcte de l'espace de recherche, mais pas trop grand pour ne pas devenir pénalisant sur le plan du temps de calcul.
2. temps écoulé
3. le meilleur individu de la dernière génération atteint un seuil critique (meilleur).

3.2 Le recuit simulé (Simulated Annealing)

Ces origines remontent aux expériences réalisées par Metropolis et al dans les années 50 pour simuler l'évolution d'un processus de recuit physique [56]. Elle est considérée comme la plus ancienne des méta-heuristiques. Cette méthode a été proposée en 1983 par Kirkpatrick [50] elle cherche un état d'énergie minimale qui correspond à une structure stable du solide, pour la résolution des problèmes d'optimisation combinatoire. Elle met en œuvre une stratégie d'évitement des minima locaux.

La méthode du recuit simulé est utilisée en métallurgie pour améliorer la qualité d'un solide. A partir de l'état liquide du solide, mis sous haute température, il est refroidi progressivement jusqu'à à retrouver sa forme solide par une diminution progressive de la température. A chaque progression, la température est maintenue jusqu'à ce que la matière trouve un équilibre thermodynamique.

Les thermodynamiciens ont remarqué qu'à partir du même liquide, on peut obtenir différents solides selon la vitesse de décroissance de la température. En effet,

- si la baisse de la température est progressive, on peut atteindre le minimum global d'énergie.
- si la baisse de la température est brutale, le matériau est amorphe et ses atomes sont figés dans un état désordonné, et il produira un minimum local d'énergie.

En le réchauffant (recuit), le métal peut être retravaillé de nouveau pour lui donner la forme désirée. Il faut baisser lentement la température en marquant des paliers suffisamment longs pour que le corps atteigne l'équilibre thermodynamique à chaque palier de la température.

Cela permet d'obtenir à la fin du processus un matériau dans un état cristallin bien ordonné correspondant à un état d'énergie minimum.

Le comportement des atomes a été caractérisé par une loi statistique de distribution proposée par Boltzman: pour une température donnée T, la probabilité pour qu'un système d'atomes soit dans un état d'énergie E est proportionnelle à $\exp(-\frac{E}{T})$. Ainsi quand la température décroît et devient proche de zéro, seuls les états d'énergie minimum ont une probabilité non nulle d'apparaître.

Le processus du recuit simulé repose sur une procédure itérative qui cherche des configurations de coût faible, mais n'empêche pas aussi d'accepter de manière contrôlée des configurations qui dégradent la fonction de coût. A chaque nouvelle itération, on génère un voisin *s'* de la configuration courante *s* tel que *s'∈N(s)*, de manière aléatoire, et de performance *f(s')*. Si ce voisin est de performance meilleure que celle de la configuration courante, f (s') ≤ f(s), il est retenu. Sinon, s', est accepté avec une probabilité p (Δf, T) qui dépend de deux facteurs:

- d'une part l'importance de la dégradation Δf = f(s') - f(s*)* (les dégradations plus faibles sont plus facilement acceptées),
- d'autre part un paramètre de contrôle T, la température (une température élevée correspond à une probabilité plus grande d'accepter des dégradations).

La température est contrôlée par une fonction décroissante qui définit un schéma de refroidissement. Un critère d'arrêt est associé à l'algorithme, qui s'arrête et retourne la meilleure configuration trouvée lorsque aucune configuration voisine n'a été acceptée pendant un certain nombre d'itérations à une température ou lorsque la température atteint la valeur zéro.

Les paragraphes suivants présentent le recuit simulé dans sa forme originale pour l'optimisation combinatoire, et ainsi quelques améliorations proposées.

3.2.1 Probabilité de Boltzmann

La probabilité de Boltzmann [1], notée P_T mesure la probabilité de trouver un système dans une configuration i avec une énergie E_i, à une température T donnée, dans l'espace des configurations S. Elle est définie par:

$$P_T(X = l) = \frac{1}{Z(t)} \exp(\frac{-E_i}{KT})$$

Où X est une variable stochastique qui désigne l'état actuel du solide, K est appelé la constante de Boltzmann, et Z(T) est une fonction appelé fonction de répartition définie par:

$$Z(T) = \sum_{j=1}^{t} \exp(-\frac{E_j}{KT})$$

Où t représente tous les états énergétiques possibles.

Dans cette expression, le facteur KT montre que lorsque la température est très élevée, tous les états sont à peu près équiprobables, c'est-à-dire qu'un grand nombre de configurations sont accessibles. Au contraire quand la température est basse, les états à haute énergie deviennent peu probables par rapport à ceux de faible énergie.

Le recuit simulé exploite généralement le critère défini par l'algorithme de Metropolis [56] pour l'acceptation d'une solution obtenue par perturbation de la solution courante. L'énergie est remplacée par la fonction objectif. Et l'obtention d'un solide à énergie minimum est équivalente à la recherche de l'optimum global de la fonction objectif.

Après chaque passage d'une configuration X à une configuration Y, la variation de la fonction objectif est $\Delta f = f(Y) - f(X)$. La transformation est acceptée selon la probabilité p(X, Y) telle que: $p(X,Y) = e^{\frac{-\Delta f}{T}}$

Lorsque la variation Δf est négative ou nulle, l'exponentielle est supérieure ou égale à 1 et la nouvelle configuration est acceptée.

Si $\Delta f > 0$, p(X, Y) est comparé à un nombre aléatoire rand $\in [0,1]$:
Si rand < p(X, Y) la configuration Y est acceptée;
Sinon elle est rejetée et une autre configuration est essayée.

Remarque : Le cas $\Delta f > 0$ permet à l'algorithme de ne pas rester bloqué dans un minimum local. Il permet en effet de garder des solutions qui paraissent mauvaises mais qui peuvent se révéler utiles par la suite.

L'algorithme est décrit par les étapes suivantes:

```
Si  f(y) ≤  f(x)
    f(x) := f(y)
    x := y
Sinon
    p = exp (-Δf/T)
    r := solution aléatoire entre [0, 1]
    Si   r ≤ p
        f(x) := f(y)
        x := y
    Fin de Si
```

Pseudo code de la règle de Metropolis

Le paramètre de contrôle T (température) influe sur la probabilité d'accepter une solution plus mauvaise.

À T élevée, la probabilité d'acceptation d'un mouvement quelconque tend vers 1, c'est à dire tous les mouvements seront acceptés (recherche aléatoire dans l'espace des configurations).

La température est diminuée lentement au fur à mesure du déroulement de l'algorithme (processus de refroidissement). La diminution sera suffisamment lente pour que l'équilibre thermodynamique soit maintenu. Nous présentons ci-dessous l'algorithme du recuit simulé.

```
x : solution choisie aléatoirement
f(x) : valeur de la fonction de performance
f_min := f(x)
x_min := x
T:= initialiser la température (assez élevée)
Répéter
   Répéter
      générer un voisin x ∈ voisinage S(x)
      appliquer la règle de Metropolis
      Si f(x) < f_min
         f_min := f(x)
         x_min := x
      Fin de Si
   Jusqu'à équilibre thermodynamique atteint
   T:= décroître température
```

Jusqu'à conditions d'arrêt satisfaites

Pseudo code de l'algorithme du recuit simulé

Un ensemble de paramètres conduit à la convergence de l'algorithme. Ces paramètres sont:
• Valeur initiale du paramètre de contrôle T_0 (température initiale)
• Facteur de réduction de la température
• Nombre d'itérations à température constante
• Taille de voisinage N_s
• Critère d'arrêt

Nous détaillons ci-dessous les paramètres de base de la méthode du recuit simulé.

3.2.2 Température

La température représente un paramètre de contrôle important. Pour appliquer l'algorithme du recuit simulé, il faut en premier lieu initialiser la température. Cette dernière doit avoir une valeur assez haute dans le but d'éviter les minimums locaux, et doit être suffisamment basse pour réduire le temps d'optimisation. La diminution lente de la température, permet au système de rechercher les "bassins d'attraction" dont la préférence est toujours donnée à celui dont le coût est minimal, mais entraîne une convergence très lente de l'algorithme. Une diminution brutale de la température risque de piéger l'algorithme dans un minimum local.

La loi de décroissance de la température est importante pour l'efficacité de l'algorithme, elle laisse le temps au système de tester au maximum les configurations, pour aboutir au minimum global.

La température initiale est notée par T_0. Ce paramètre de contrôle doit être suffisamment élevé pour permettre à toutes les configurations d'être acceptées, c'est à dire pour permettre la localisation de la région où se trouve le minimum global. Elle est déterminée, sinon fixée arbitrairement. D'après Kirkpatrick [50], T_0 doit être choisie de sorte que la probabilité d'acceptation de la plus mauvaise solution, noté P_r, soit environ de 80%. Ensuite, dans l'expression de p(X, Y), l'utilisateur doit fixer l'accroissement maximal acceptable de la fonction objectif, noté Δf^+, pour en déduire la valeur de T_0. L'expression suivante est obtenue:

$$T_0 = \frac{\Delta f^+}{\ln(p_r)}$$

Cette relation peut être déduite de la probabilité de Boltzmann. Heck [40] propose de générer aléatoirement un certain nombre de configurations initiales, ensuite, la température initiale est calculée de façon à pouvoir accepter les configurations dont la fonction objectif est comprise dans l'intervalle $[-3\cdot\sigma \; +3\cdot\sigma]$ autour de la valeur moyenne, avec σ l'écart type des valeurs initiales de la fonction objectif. La probabilité d'acceptation P_r est réglée à 0.01 pour avoir une température initiale suffisamment élevée. Ainsi, la température initiale vaut:

$$T_0 = \frac{-3\sigma}{\ln(p_r)}$$

3.2.3 Décroissance de la température

La principale difficulté rencontrée dans la résolution d'un problème d'optimisation par la méthode du recuit simulé pour obtenir une meilleure performance est liée à la détermination du schéma de refroidissement. De nombreux schémas théoriques et pratiques ont été proposés, parmi lesquels on trouve:

La réduction par paliers, ou chaque température est maintenue égale pendant un certain nombre d'itérations, et décroît ainsi par paliers.

Le changement de température de T_k vers T_{k+1} est effectué au moment où l'équilibre thermique est détecté. La variation de température se fait par paliers comme son nom l'indique, suivant la fonction de décroissance utilisée. Les fonctions les plus couramment rencontrées dans la littérature sont les fonctions linéaires, discrètes ou exponentielles: voir tableau

Type	Fonctions	Paramètres
Linéaire	$T_{k+1} = a \, . \, T_k$	a<1 (type 0.8 à 0.99)
Discret	$T_{k+1} = T_k - \Delta T$	$\Delta T > 0$
Exponentielle	$T_{k+1} = T_k . \exp(\frac{-\lambda . T_k}{\sigma_k})$ Avec $0 \le \frac{T_{k+1}}{T_k} \le 1$	σ_k est l'écart type des fonctions objectifs des configurations acceptées à la température T_k, λ est un paramètre de réglage fixé par l'utilisateur, positif

Tableau 3.1 : Lois de décroissance de la température, les plus utilisées.

La plus utilisée est la loi linéaire qui permet d'avoir une décroissance rapide au début de l'optimisation et lente proche de la convergence. *T* peut décroître linéairement, à chaque itération, mais on peut aussi envisager une décroissance par paliers, c'est-à-dire en gardant *T* constant, tant qu'un certain nombre de conditions n'ont pas été remplies, de façon à atteindre l'équilibre thermodynamique. Une fois cet équilibre atteint, on abaisse légèrement la température et on recommence une nouvelle chaîne de calculs à ce nouveau palier.

Pour la fonction discrète, la décroissance est indépendante de la valeur de l'état précédent du système et dépend uniquement de ΔT.

La décroissance exponentielle permet de tenir compte de l'état précédent, par l'utilisation de l'écart type, des valeurs de la fonction objectif obtenue au palier de température précédent. Au début de la recherche, presque toutes les configurations sont acceptées. Comme ces configurations peuvent être très dispersées dans l'espace de recherche, l'écart type peut être alors relativement grand et donc la température décroît plus lentement selon la valeur donnée au paramètre λ.

3.2.4 Nombre d'itérations à température constante

Cette valeur doit être assez élevée pour atteindre l'équilibre thermodynamique et suffisamment faible pour réduire le temps d'optimisation.

3.2.5 Critères d'arrêt

Plusieurs critères d'arrêt existent. Le processus d'optimisation est arrêté quand aucune amélioration sensible n'est apportée sur la simulation, où bien quand le paramètre de contrôle T est inférieur à une certaine valeur choisie.

Dans la pratique, le processus est stoppé lorsque la température atteint la valeur nulle ou bien lorsque plus aucun mouvement accroissant l'énergie n'a été accepté au cours du palier. Vanderbilt et al [78] ont suggéré comme critère d'arrêt l'expression suivante:

$$\frac{\overline{f} - f_{opt}}{\overline{f}} \le \varepsilon$$

Où f_{opt} est la valeur de la fonction objectif à l'optimum courant, $\overline{f}$ est la moyenne des fonctions objectifs des configurations générées depuis le début de l'optimisation et ε est la précision fixée par l'utilisateur.

Painton, en 1994 [63], propose d'arrêter l'algorithme lorsque après dix décroissances successives de la température, il n'y a pas d'amélioration de la configuration optimale.
Dolan, en 1990 [28], propose d'arrêter l'algorithme du recuit simulé lorsque la température de recuit est inférieure à une valeur choisie.

• Procédure d'implémentation d'un recuit simulé:

Choisir arbitrairement une solution initiale *s*
Poser $T \leftarrow T_0$ initialisation de la température initiale
Répéter
Répéter
Choisir aléatoirement $s' \in N(s)$
Générer un nombre réel aléatoire *r* dans [0,1]
Les cas suivants se présentent:

· Si $f(s')<f(s)$ alors $e^{\frac{f(s)-f(s')}{T}}$ >1, donc r est toujours inférieur à cette valeur, et on accepte la solution *s'* (une meilleure solution est donc toujours acceptée).

· Si $f(s')>f(s)$ et *T* est très grand, alors $e^{\frac{f(s)-f(s')}{T}} \cong 1$, il y a de fortes chances d'accepter *s'*, Alors on teste Si $r < e^{\frac{f(s)-f(s')}{T}}$ alors Poser $s \leftarrow s'$

· Si $f(s')>f(s)$ et *T* est très petit, alors $e^{\frac{f(s)-f(s')}{T}} \cong 0$, et on va donc probablement refuser *s'*, Alors on teste Si $r < e^{\frac{f(s)-f(s')}{T}}$ alors Poser $s \leftarrow s'$

Jusqu'à ce que l'équilibre thermodynamique soit atteint
Mettre à jour *T* (décroitre le paramètre de contrôle selon la loi de décroissance choisie)
Jusqu'à ce que le critère de terminaison soit satisfait
Fin

• Et voici l'interprétation de son fonctionnement:
Dans un premier temps, on initialise la température *T*, généralement choisie de valeur très grande. Dans ce cas beaucoup de solutions, même celles dégradant la valeur de *f* sont acceptées, et permettent à l'algorithme une visite aléatoire de l'espace des configurations. Mais à mesure que la température baisse selon la loi de décroissance appliquée à l'algorithme, la plupart des mouvements qui augmentent l'énergie seront refusés, et l'algorithme se ramène à une amélioration itérative classique. Lorsque la température intermédiaire est atteinte, l'algorithme autorise de temps à autre des transformations qui dégradent la fonction objectif. Cela permet de laisser une chance au système de s'extraire d'un minimum local.
On constate qu'une diminution de température suffisamment lente, permet de visiter un grand nombre d'itérations, ce qui conduit à l'amélioration de la solution.

Avantages et inconvénients

Le recuit simulé est une méthode simple à programmer et à paramétrer. Elle a l'avantage d'offrir des solutions de bonne qualité. Elle peut trouver la meilleure solution si on la laisse chercher indéfiniment. Elle offre aussi une souplesse d'emploi, qui lui permet d'inclure facilement des contraintes dans le corps du programme.

L'inconvénient de cette méthode est qu'une fois l'algorithme piégé à basse température dans un minimum local, il lui est impossible de s'en sortir tout seul. Pour résoudre ce

problème, on permet de brusque remontée de la température, de temps en temps, afin de relancer la recherche sur d'autres régions plus éloignées, où il est également possible d'empêcher la température de descendre trop bas: on lui donne une valeur minimale au delà de laquelle on ne change plus de palier de température.

3.3 La recherche tabou

La méthode Tabou est une technique d'optimisation combinatoire, dont les principes ont été proposés par Fred Glover dans les années 80. Cette méthode de recherche locale, fait appel à un ensemble de règles et de mécanismes généraux pour guider la recherche de manière intelligente au travers de l'espace des solutions, et ceci par le recours à l'historique des solutions visitées, qui rend la recherche un peu moins aveugle. Il devient donc possible de s'extraire d'un minimum local, mais, pour éviter d'y retomber périodiquement, certaines solutions sont bannies, elles sont rendues « taboues ».

A l'inverse du recuit simulé qui génère de manière aléatoire une seule solution voisine $s' \in N(s)$ à chaque itération, et s'arrête dès qu'il n'y a plus de voisin s' permettant d'améliorer la valeur de la fonction objectif f, Tabou examine un ensemble de solutions de $N(s)$ et retient la meilleure s' même si s'est plus mauvaise que s. Ce critère autorisant les dégradations de la fonction objectif évite à l'algorithme d'être piégé dans un minimum local. Comme la recherche taboue ne s'arrête pas au premier optimum trouvé, et sa stratégie est de continuer la recherche, elle peut alors entraîner des cycles. Le risque serait alors de revenir à s immédiatement, puisque s est meilleure que s', à l'itération suivante. Pour éviter de tourner ainsi en rond, et de revenir aux solutions déjà visitées, l'approche consiste alors à créer une liste T qui mémorise et conserve pendant un moment la trace des dernières meilleures solutions déjà visitées et qui interdit tout déplacement vers une solution de cette liste. Ces solutions sont déclarées tabou, d'où le nom de la méthode. Et cette mémoire est appelée liste Tabou, représente l'une des composantes essentielles de cette méthode. La valeur L de la taille de la liste taboue dépend du problème à résoudre et peut éventuellement évoluer au cours de la recherche, dans laquelle on mémorise les L dernières configurations visitées et on interdit tout mouvement qui conduit à une de ces configurations.

3.3.1 Principe de base

La méthode de recherche Tabou est considérée comme une des méthodes d'amélioration locales. Elle part d'une solution quelconque s appartenant à l'ensemble de solutions S, et se déplace vers une solution s' située dans le voisinage N(s) de s.

Afin de choisir le meilleur voisin s' dans N(s), l'algorithme évalue la fonction objectif f en chaque point s' de N(s), et retient le voisin qui améliore la valeur de la fonction objectif f, ou au mieux celui qui la dégrade le moins.

Cette solution sera stockée dans une liste de longueur L donnée, appelée liste Tabou. Une nouvelle recherche est lancée de nouveau, la nouvelle solution n'est acceptée que si elle n'appartient pas à cette liste Tabou. On évite ainsi de reboucler l'algorithme durant l'exploration d'un nombre de solutions au moins égal à la longueur de la liste tabou, ce qui lui permet de diriger l'exploration de la méthode vers des régions du domaine de solutions non encore visitées. La liste taboue est gérée comme une liste circulaire qui élimine à chaque itération la solution taboue la plus ancienne, en la remplaçant par la nouvelle solution retenue.

3.3.2 Mémoire à court terme

La mémoire de la liste tabou est utilisée pour stocker et mémoriser les configurations telles quelles, c'est à dire la valeur de la fonction objectif prise pour chacune des configurations. Les solutions ne demeurent donc dans T, que pour un nombre limité d'itérations, égal à la taille de la liste taboue. Pour réaliser cette mémorisation d'une manière simple, on construit

un tableau, dans lequel on stocke une à une les valeurs des solutions que l'on a visitées, et qui demeurent les meilleures solutions, dans une pile FIFO (First In First Out). Cette liste est gérée de façon circulaire, comme expliqué plus haut.

Cette solution de stockage en mémoire permet alors de définir les tabous comme les "transformations" qui ont permis le passage d'une solution à une autre. Pour autoriser de nouveau une solution déjà sélectionnée, on procède comme suit :

Soit T un tableau de taille maximale égale à L. Si $f(s)$ est la valeur de la fonction objectif de la solution *s*, on place cette valeur dans T [$f(s)$]. Il faudra alors attendre L itérations, pour que la solution de valeur $f(s)$ soit autorisée de nouveau.

3.3.3 Mémoire à long terme

Dans la liste Tabou T utilisée, si le nombre L de mouvements interdits est trop court, il y aura un risque de boucle. Mais s'il est trop grand, l'algorithme risque d'éviter d'exploiter en profondeur une vallée durant sa recherche.

Une mémoire statistique, basée sur la fréquence d'apparition de certains mouvements, permet de conserver un nombre de mouvements interdits raisonnables, tout en prévenant la formation de cycle de grande largeur. Plutôt que d'interdire un mouvement trop souvent utilisé, on va le pénaliser. En plus de mémoriser la fréquence d'apparition, on peut également invoquer la récence d'une solution (plus un mouvement a été récemment utilisé, plus il est pénalisé), ou bien mémoriser la qualité des mouvements : plus un mouvement a débouché sur des solutions améliorant significativement la valeur de la fonction objectif, plus il est favorisé.

Cette dernière technique est connue sous le nom de mémoire adaptative, et a été développée par Rochat et Taillard.

Par ailleurs, en s'interdisant ainsi certaines solutions, on risquerait de s'interdire des chemins qui permettraient de s'échapper vers une autre vallée plus profonde et plus prometteuse, par exemple un mouvement qui aboutirait à éventuellement écarter une solution meilleure que toutes celles déjà visitées. Il serait donc tout à fait absurde de s'interdire définitivement une valeur. Pour éviter cela, un critère d'aspiration est inséré dans l'algorithme. Il consiste à tester si la solution engendrée dans le statut tabou présente un coût inférieur à celui de la meilleure solution trouvée jusqu'à présent. Si cette situation se produit, le statut tabou de la solution est levé.

3.3.4 Critère d'aspiration

Ce mécanisme particulier qui est appelé critère d'aspiration, est mis en place pour lever le statut tabou d'une solution, sans introduire un risque de cycles dans le processus de recherche.

La fonction d'aspiration consiste à enlever le statut tabou d'un mouvement si ce dernier permet d'atteindre une solution de qualité meilleure que celle trouvée jusqu'alors. Ce critère est évidemment très sévère, il ne devrait pas être vérifié très souvent. L'inconvénient de recourir trop souvent à l'aspiration, est qu'il apporte peu de changements à la méthode, et peut détruire, dans une certaine mesure, la protection offerte par la liste taboue vis-à-vis du risque de cycle.

Il existe d'autres techniques intéressantes pour améliorer la puissance de la méthode taboue. Il s'agit en particulier, de l'intensification et la diversification. Elles jouent toutes les deux un rôle complémentaire. Elles se basent sur l'utilisation d'une mémoire à long terme et se différencient selon la façon d'exploiter les informations de cette mémoire. L'application de l'intensification consiste à mémoriser une liste de solutions de bonne qualité et à retourner vers une des ces solutions. Tandis que l'application de la diversification cherche à diriger la recherche vers des zones inexplorées. Sa mise en œuvre consiste souvent à modifier

temporairement la fonction de coût pour favoriser des mouvements n'ayant pas été effectués ou à pénaliser les mouvements ayant été souvent répétés.

• L'algorithme est décrit par les étapes suivantes:

s := solution aléatoire
f_{min} := f(s)
s_{min} := s
Tabou := tableau de liste des solutions, de longueur L
Tabou := ∅ (Tableau vide)
Répéter
  Générer un ensemble de solution N(s) ∈ voisinage s
  {s, N(s)} ∉ TABOU
  f(s) = min[f(N(s))]
  ajouter ({s, f(s)}, Tabou)
SI f(s) < f_{min}
  f_{min} := f(s)
  s_{min} := s
Fin de Si
Jusqu'à conditions d'arrêt satisfaites

Pseudo code de la méthode de recherche tabou

L'algorithme général de la méthode de recherche tabou peut être décrit par le pseudo-code précédemment défini. Le domaine de définition du voisinage est l'ensemble des solutions qu'il est possible d'avoir sur une machine, tout en respectant les contraintes d'ordonnancement.

Si on nomme N_T (s) toutes les solutions qui ne sont pas taboues ainsi que celles qui le sont mais dont le statut tabou est levé en raison des critères d'aspiration, N_T (s) est représenté par:
$N_T(s) = \{s' \in N(s)$ tel que $s' \notin T$ ou $f(s') < f(s^*)\}$

• Procédure d'association du critère d'aspiration à la méthode Tabou

s solution initiale, L taille de la liste tabou
Poser T ← ∅ et *s** ← *s*,
Répéter
  Choisir *s'* qui minimise *f(s')* dans $N_T(s)$, s' ∉ L
  Si *f(s')* < *f(s*)* alors poser *s** ← *s'*, Modifier la solution courante *i* avec la meilleure solution trouvée (meilleure solution voisine)
  Poser *s* ← *s'* et mettre à jour T, Ajouter cette solution dans la liste Tabou
  (L← L∪s', elle sera taboue pour les L prochaines itérations)
Jusqu'à ce que le critère de terminaison soit satisfait
Fin

3.3.5 Critère d'arrêt

Comme critère d'arrêt on peut décider par exemple de fixer un nombre maximum d'itérations sans amélioration de *s**, ou bien fixer un temps limite au-delà duquel la recherche doit s'arrêter. Le critère d'arrêt de l'algorithme peut donc être un nombre d'itérations maximal, ou bien une durée à ne pas dépasser.

3.3.6 La recherche taboue de Hu

Dans la littérature, on trouve une structure fondamentale des algorithmes de recherche taboue. Elle est implémentée ici pour l'optimisation avec des variables continues. Cette méthode est la recherche taboue de Hu (RTHu).

En suivant les idées de Glover, Hu [45] a proposé un algorithme de recherche taboue pour l'optimisation de variables continues. Pour générer un mouvement aléatoire en utilisant $y_i = x_i + rand_i \cdot p_i$ avec $-1 \leq rand_i \leq 1$, et définir un voisinage, Hu a proposé un ensemble de pas p_i, $P = [p_1, p_2,..., p_{Na}]$ en utilisant $p_{ij} = (x_{iM} - x_{im}) / c^j$ avec $i = 1 \ldots n$, $j = 1 \ldots L_m$

où c, est le coefficient de réduction supérieur à 1, éventuellement dépendant de l'espace de recherche, et L_m, le nombre d'itérations du vecteur pas à température constante.

Mais le codage d'une liste taboue est encombrant car il faudrait garder en mémoire tous les éléments qui définissent une solution. Pour pallier à cet inconvénient, la liste taboue des solutions interdites est remplacée par la liste des derniers pas p_i correspondant aux solutions ayant amélioré la fonction objectif. Cette recherche taboue ne permet pas de dégrader la fonction objectif, même temporairement.

Les paramètres de RTHu sont la taille de voisinage N_n, le nombre de subdivisions du vecteur pas N_a. Ce dernier nombre est également la taille de la liste taboue T.

L'Algorithme de RTHu.

L'algorithme est comme suit:

1. construire la matrice des pas en utilisant la relation $p_{ij} = (x_{iM} - x_{im}) / c^j$ pour chaque direction.
2. initialisation: soit un point de départ généré aléatoirement X_0, la liste taboue est vide et le meilleur point est le point initial $X^* = X_0$, $f^* = f(X_0)$.
3. générer un mouvement aléatoire faisable dans les voisinages $V(X^*, p_k)$, $k = 1,....., N_a$, de la meilleure solution courante en utilisant les pas non tabous. Soit X_j le meilleur point dans tous les voisinages, $f(X_j)$ est sa fonction objectif et p_j le pas utilisé pour générer X_j.
4. si $f(X_j) \leq f(X^*)$ alors $X^* = X_j$, $f^* = f(X_j)$, ajouter le pas p_j à la liste taboue T.
5. mettre à jour les pas p_i en utilisant $p_{ij} = (x_{iM} - x_{im}) / c^j$ et retourner à l'étape 3.
6. si tous les pas sont dans la liste taboue et le critère d'arrêt n'est pas satisfaisant, alors remise à zéro de la liste taboue T et retour à l'étape 3.

Avec le critère d'arrêt

L'algorithme s'arrête si:

1. la fonction objectif ne change plus de manière significative,

ou si,

2. un grand nombre de points est généré sans aucune amélioration de la fonction objectif.

Avantages et inconvénients

La structure de l'algorithme de base de la méthode recherche taboue est finalement assez proche de celle du recuit simulé. Elles ont en commun la recherche de la solution optimale aux voisinages de la solution courante. Seulement la recherche taboue a l'avantage sur le recuit simulé, d'avoir un paramétrage simplifié qui consiste à trouver une valeur d'itérations pendant lesquelles les mouvements sont interdits. Elle a un fonctionnement simple à comprendre. Par contre, elle exige une gestion de la mémoire, de plus en plus lourde. Les résultats intéressants obtenus par la méthode Tabou pour les problèmes d'optimisation combinatoire, et en particulier, pour des problèmes d'affectation sous contraintes, font d'elle une des meilleures méta- heuristiques pour la résolution de ces problèmes.

Remarques :

En résumé, pour adapter une méta-heuristique à un problème d'optimisation particulier, il est important de bien choisir les principaux constituants de ces techniques qui sont l'espace de solution S, la fonction objectif f, et le voisinage N.

La fonction objectif donne la forme de l'espace dans lequel la recherche d'un optimum est effectuée, alors que le voisinage ou l'opérateur de combinaison permettent de générer de nouvelles solutions à partir de la solution courante ou de la population de solutions courante. Il faut s'assurer que les solutions générées par le voisinage ou l'opérateur de combinaison ne doivent pas trop se différencier de la solution courante ou des solutions parents quant à la valeur mesurée par f, ce qui offre une sorte de continuité dans la fonction objectif.

Nous avons également vu qu'il n'est pas nécessaire d'imposer la réalisabilité des solutions dans S. La fonction objectif doit alors pénaliser les solutions non réalisables.

4. Approche proposée

4.1 Problème job shop

Un problème d'ordonnancement consiste à organiser dans le temps, la réalisation d'une tâche, compte tenu des contraintes temporelles (les délais), des contraintes d'acheminement (l'exécution des opérations respecte un ordre bien définie), et des contraintes portant sur la disponibilité des ressources requises. Il fournit une solution au problème d'ordonnancement par l'allocation des ressources, en satisfaisant un objectif principal qui consiste à minimiser la durée totale de la production de toutes les tâches: minimisation du Makespan (C'est la différence entre la date de fin de la dernière opération et la date de début de la première opération de l'ensemble de tâches).

4.1.1 Formulation du problème

La problématique posée concerne en fait deux aspects liés. Il s'agit du :

- problème d'affectation;
- problème d'ordonnancement;

Les données et hypothèses de travail sont les suivantes :

- un ensemble composé de M machines
- un ensemble de I tâches, représenté par une gamme opératoire composée de $O_{i,j,k}$ opérations.
- une opération $O_{i,j,k}$ représentant la $j^{ème}$ opération de la tâche i sur la machine k, nécessite un temps opératoire $E_{i,j}$.
- l'opération $O_{i,j,k}$ nécessite pour être réalisée une machine k appartenant à {1, 2,…, M}l'ensemble des machines.

Soumises aux contraintes suivantes:

- les machines sont indépendantes les unes des autres;
- une machine ne peut exécuter qu'une seule opération à un instant donné;
- une opération en cours d'exécution ne peut pas être interrompue;
- les tâches sont indépendantes les unes des autres.

Notre travail est de chercher à obtenir une allocation et un ordonnancement, les meilleurs au sens d'un critère, sur chaque ressource partagée.

Soit C_{max} la durée totale d'exécution de toutes les opérations qui composent les I tâche. L'objectif recherché est de minimiser C_{max}.

Un mécanisme d'ordonnancement a deux fonctions principales:

- la gestion temporelle,
- l'allocation des tâches à la machine cible.

Un algorithme générateur d'ordonnancement sera le suivant :
- M: nombre de machine;
- t: tableau de dimension M
- Initialiser la date de début de chaque machine à 0: t(k) = 0 $\forall$ machine k $\in$ [1, M]
- k = 1
- tant que (k $\leq$ M) faire
- J: ensemble des opérations sur la machine k
    - tant que (J > 1) faire
    - chercher l'ensemble des opérations qui peuvent être ordonnancées à l'instant t(k) sur la machine k, lorsqu'elle est de nouveau disponible
    - choisir l'opération O de l'ensemble des opérations pour laquelle le temps total d'exécution sera optimal (par association d'une méta-heuristique au modèle RdP utilisé)
    - Placer (affecter) l'opération O sur la machine k
    - enlever O de l'ensemble J opérations
    - ajouter le temps de traitement de O aux opérations J
    - mise à jour de l'ensemble J
    - fin
- k = k + 1
- fin

Algorithme d'un Générateur d'ordonnancement.

4.1.2 Allocation de tâches

Il s'agit d'allouer un ensemble de tâches à un ensemble de ressources, en tenant compte d'un critère donné, ainsi que des contraintes qui doivent être strictement respectés (contraintes temporelles, localité physiques des ressources,...). L'état de ressource est mesuré par le facteur de charge. Il est défini par le nombre de tâches en attente sur une ressource.

Divers algorithmes sont proposés pour résoudre ce type de problème. Ils représentent les différentes techniques utilisées, appliquées en fonction du modèle des tâches choisi. Ce modèle tient compte de l'ensemble des contraintes, de précédence, temporelles, et de localité. La formulation du problème d'optimisation permet à travers une fonction coût d'exprimer les contraintes de temps. La résolution du problème se fait par la suite en appliquant des algorithmes basés sur les méta-heuristiques modifiés. En fonction de la situation et de l'instant, le choix portera sur l'une ou l'autre des techniques proposées, en recherchant celle qui est la plus efficace et qui permet d'aboutir à la solution optimum rapidement dans l'espace de solutions.
Le mécanisme d'ordonnancement local (ordonnanceur) est basé donc sur l'application des algorithmes des méta-heuristiques (RT, RS, AG) modifiés, destinés à trouver un ordre d'exécution entre les tâches lorsqu'un conflit sur une ressource se présente. La prise en compte des contraintes citées, est primordiale afin de garantir le bon ordonnancement.

Des critères sont donc pris en compte afin de les comparer et d'affiner le résultat souhaité, nous citons la réduction des coûts de production ou la minimisation du temps de réalisation.
Chaque tâche T_i est décrite par:
T_i : < $R_{i,j}$, $E_{i,j}$ > où $R_{i,j}$ est la date à partir de laquelle T_i peut commencer son exécution, $E_{i,j}$ le temps d'exécution pour calculer $C_{i,j}$ (la date à laquelle une tâche i termine la fin d'exécution de l'opération j). Pour exprimer une précédence entre les tâches d'une gamme de production, on le fera à travers les dates $R_{i,j}$ auxquelles l'opération j de la tâche i est prête, ces paramètres

$R_{i,j}$ auront des valeurs qui reflèteront l'ordre d'exécution des opérations selon leur précédence, l'opération j+1ne doit pas être exécuté qu'après l'exécution de l'opération j, elle doit vérifie la condition $R_{i,j+1} > R_{i,j}$.

L'algorithme fonctionne selon les étapes suivantes:
- enregistrement de la tâche demandant la ressource, dans une liste d'attente. A chaque ressource est associée une liste. Lorsqu'une ressource se libère, le mécanisme d'ordonnancement prend la main, et les tâches en attente sont extraites et prisent en compte par l'algorithme d'ordonnancement.
Une méta-heuristique est appliquée à cet espace de recherche afin de connaître $S_{i,j}$, la date à laquelle l'opération peut accéder à la ressource. Le choix de refuser ou d'accepter une tâche est assez délicat car il rentre dans l'approche d'ordonnancement qui doit être respectée.
- Une fois que la tâche est acceptée pour accéder à la ressource par son affectation depuis le processus d'ordonnancement, son ensemble de paramètres nécessaires à l'ordonnancement ($S_{i,j}$, $S_{i,j} + E_{i,j}$) est inséré au temps d'exécution. Quand la date de début d'exécution de la tâche est atteinte, le mécanisme la retire de la liste d'attente et lance son exécution. La liste des tâches refusées sera mise de nouveau dans la liste d'attente en insérant s'il y a lieu d'autres tâches demandant cette ressource, et pour lesquelles l'algorithme d'allocation sera déroulé de nouveau pour les réordonnancer par l'application d'une méta-heuristique d'allocation, qui se déroule comme suit:

Début
. Calcul de la charge sur la ressource partagé
. Retirer une tâche de la liste à ordonnancer
. Calcul de sa fonction d'évaluation
. Parcourir la liste d'ordonnancement à la recherche de la tâche suivante
. Appliquer la méta-heuristique appropriée (RT, RS, AG)
. Si la tâche est acceptée, allez vers son allocation
. Si la tâche est refusée, ajouter le temps de traitement de l'opération à la tâche en question, allez vers la liste de tâches refusées
. Mise à jour de la liste d'ordonnancement (tâches refusées actuellement + autres tâches demandant la même ressource partagée).
. L'exécution de la tâche acceptée, fait associée une durée d'attente (une durée qui est égale au temps d'exécution de l'opération acceptée) au temps de la liste des tâches refusées.
La fonction d'évaluation choisie modélise les écarts des temps d'exécution.

4.1.3 Application des méta-heuristiques pour l'ordonnancement de la cellule

Étant donné que le facteur de charge est mesuré par le nombre de tâches en attente de la ressource et non selon les valeurs du taux de son d'utilisation. Le module de décision implémente trois algorithmes : la recherche tabou, le recuit simulé et les algorithmes génétiques. Il choisit entre ces trois algorithmes le plus approprié en fonction de la valeur du facteur de charge mesuré sur chaque ressource afin de résoudre le problème d'optimisation.

Les résultats montrent que l'insertion des temps morts (temps d'attente de la disponibilité des ressources) peut considérablement améliorer la performance de l'ordonnancement en présence des incertitudes sans trop dégrader la qualité de l'ordonnancement réellement exécuté.
Lorsque plusieurs décisions concurrentes sont possibles à un instant donné, il est nécessaire de faire un choix. Ce dernier conditionne la performance finale de l'ordonnancement. Afin d'aboutir au meilleur choix, on a généralement recours à un système d'aide à la décision. Il a

l'avantage d'aider à la prise rapide de la décision, selon la fonction objectif considérée (minimisation du makespan, minimisation du retard d'attente des opérations sur machine).

Afin d'améliorer la performance du système de décision, on intègre des méthodes d'ordonnancement par règles. Elles consistent en la sélection d'une règle déduite d'un modèle de connaissances obtenu après plusieurs tests de simulation sur le système de production:
- l'idée consiste, au moment d'une prise de décision, à simuler l'application d'un ensemble de méthodes basées sur les méta-heuristiques, en se projetant éventuellement sur le futur proche, puis de sélectionner celle fournissant la meilleure performance.
-Le modèle de connaissances est construit en mode hors ligne lors d'une phase d'apprentissage.

On prend comme espace de recherche, le partage de ressource, considéré en terme de charge de travail liée au nombre de produits en conflit. L'affectation de ces ressources entre différents produits peut conduire à ce que certaines tâches tardent à y avoir accès et voient leurs contraintes temporelles violées. Notre objectif est de minimiser les retards d'accès des tâches aux ressources, ce qui conduit à la minimisation du temps d'utilisation de ces ressources (par minimisation des temps d'attente de la disponibilité de la ressource), ainsi la minimisation du temps de réalisation, et donc la minimisation du coût de production. Le calcul s'appuie donc sur la charge affectée sur chaque ressource de l'atelier de production.

Trois états de charge apparaissent sur l'espace des ressources partagées. Ces états ont été choisies afin d'exprimer les temps d'accès des tâches sur ces ressources:
- Faiblement chargé : les tâches sont peu nombreuses. La méta-heuristique choisie lors de l'étape d'ordonnancement, devrait trouver facilement la tâche à affecter à la ressource.
- Moyennement chargé : l'algorithme développé, à recours à l'ordonnancement des tâches à l'aide de celle, parmi les méta-heuristiques, qui fournit le meilleur résultat.
- Hautement chargé : dans ce dernier cas, l'algorithme a recours à l'ordonnancement des tâches à l'aide d'une combinaison de méta-heuristique, afin d'obtenir la meilleure solution,

Si le nombre de tâches ordonnancées est faible, le temps d'exécution de l'algorithme sera donc réduit.
Le problème d'optimisation se base sur l'association de trois méta-heuristiques qui sont: RS, AG, RT. Grâce à leurs différents avantages liés à la résolution des problèmes d'optimisation, les méta-heuristiques associées ont connu un réel développement dans le domaine de l'ordonnancement en général et dans l'ordonnancement des systèmes de production en particulier.
En effet, ces méta-heuristiques se traitent à travers trois problématiques liées à:

- La résolution des conflits entre les différentes gammes sur les ressources de production.
- Le partage de ressource, est considéré comme un espace de recherche en termes de charge de travail liée au nombre de produits en conflit sur la même ressource, en minimisant le temps de réalisation.
- L'affectation des ressources entre les différents produits, tout en minimisant, les augmentations des temps d'attentes sur la ressource, la charge de travail, ainsi que le temps de réalisation.

4.2 Implantation de l'algorithme génétique.
Notre algorithme est initialisé par une population de N individus. Pour chacun d'eux, nous calculons la valeur de sa fonction objectif (fonction d'évaluation). Ensuite, nous sélectionnons les individus les mieux adaptés selon leurs fonction coûts, en se basant sur le principe de la

sélection déterministe (égal à la moitié de la population, c'est-à-dire, V = N/2 individus si N est un nombre paire, sinon V = N / 2 + 1 individus si N est impaire). Les individus sélectionnés seront traités par l'opérateur de croisement. Ils sont choisis avec une probabilité de croisement $P_c = 1$, et sur lesquels, on applique l'opérateur de croisement arithmétique (la multiplication). Les résultats trouvés peuvent être mutés par l'opérateur de mutation réelle avec une probabilité de mutation $P_m = 0.1$. Les individus issus de ces opérateurs génétiques seront insérés dans la nouvelle population (remplacement générationnel) et nous évaluons les valeurs de leur fonction objectif. Après, un test d'arrêt sera effectué. Il consiste à se limiter à la première génération enfants afin d'éviter que les meilleurs chromosomes ne soient perdus après les opérateurs de croisement et de mutation. Si ce test est vérifié, l'algorithme s'arrête en proposant une solution optimale. Elle est obtenue en employant la méthode de triage des nouveaux individus de la nouvelle génération et qui permet de sélectionner le meilleur individu (ou solution optimale), sinon on recommence le processus.

a. Codage d'une solution de JSP

Il y a plusieurs types de codage qui permettent de représenter les solutions d'un problème d'ordonnancement de type job shop, et de connaitre la localisation temporelle des opérations sur les machines. Nous proposons un codage réel qui permet d'affecter les temps d'exécution $t_{ex,i} = C_{i,j}$ (représente le temps d'exécution obtenu jusqu'à présent pour réaliser les opérations de la tâche i) aux tâches qui sont en conflit sur une ressource partagée.

Pour coder une solution, il suffit d'utiliser une matrice réelle strictement positive de I ligne et M colonne (I × M éléments), leur entrées égale à la durée des opérations des tâches I demandant la ressource M. Une matrice réelle positive ne constitue pas une solution réalisable car elle doit vérifier à la fois des contraintes de précédences liées aux gammes, et qui serons liées entre elles par la succession des opérations sur la même machine.

Ce codage utilisé en ordonnancement, est donné à chaque machine représentée par les opérations qui lui sont affectées. Il permet aussi de représenter les solutions au problème Job Shop, mais il rencontre une difficulté liée à la réalisabilité ou non, de la représentation. Face cela, il y a besoin de concevoir des codages qui correspondent toujours à des solutions réalisables pour la sélection, ainsi qu'aux opérateurs de croisement et de mutation associés.

On parle alors de codage réel direct, puisqu'il y a une correspondance entre l'ensemble des chromosomes codés et l'ensemble de solutions.

Considérons la problématique de l'affectation des ressources. Une solution non complète serait un ensemble d'affectations de la ressource sans définir un ordre particulier. Dans la figure ci-dessous, nous avons représenté un chromosome possible, décrivant les variables d'allocation $O_{i,j,k}$ aux ressource M_k, avec $O_{i,j,k}$ nul si la ressource M_k n'est pas disponible et égal à 1 sinon.

États des tâches en conflit sur les mêmes ressources.

$O_{1,j,1}$	$O_{2,j,1}$	$O_{3,j,1}$	$O_{4,j,1}$	$O_{5,j,1}$	$O_{6,j,1}$	$O_{7,j,1}$
$O_{1,j,2}$	$O_{2,j,2}$	$O_{3,j,2}$	$O_{4,j,2}$	$O_{5,j,2}$	$O_{6,j,2}$	$O_{7,j,2}$
$O_{1,j,3}$	$O_{2,j,3}$	$O_{3,j,3}$	$O_{4,j,3}$	$O_{5,j,3}$	$O_{6,j,3}$	$O_{7,j,3}$

États des ressources partagées

M_1	M_2	M_3	Cv_1	Cv_2	Cv_3

Exemple de codage des chromosomes.

Pour chaque ressource partagée, on effectue un chromosome dont la taille (c'est-à-dire, le nombre de ses gènes) est égale au nombre de gammes demandant la même ressource M_k.

b. Représentation

Considérons l'ensemble de Job Shop composés de trois machines et de 7 tâches:

Tâche i	$O_{i,j,k} \Rightarrow$ Machine (durée opératoire)		
tâche 1	$O_{1,1,1} \Rightarrow 1$ (3)		
tâche 2	$O_{2,1,2} \Rightarrow 2$ (2)		
tâche 3	$O_{3,1,3} \Rightarrow 3$ (6)		
tâche 4	$O_{4,1,1} \Rightarrow 1$ (4)	$O_{4,2,2} \Rightarrow 2$ (2)	
tâche 5	$O_{5,1,1} \Rightarrow 1$ (5)	$O_{5,2,3} \Rightarrow 3$ (7)	
tâche 6	$O_{6,1,2} \Rightarrow 2$ (2)	$O_{6,2,3} \Rightarrow 3$ (5)	
tâche 7	$O_{7,1,1} \Rightarrow 1$ (4)	$O_{7,2,2} \Rightarrow 2$ (2)	$O_{7,3,3} \Rightarrow 3$ (3)

Tableau 3.2 : Tableau représentant l'ensemble des tâches.

La matrice précédente sera comme suit:

$O_{1,j,1}$	0	0	$O_{4,j,1}$	$O_{5,j,1}$	0	$O_{7,j,1}$
0	$O_{2,j,2}$	0	$O_{4,j,2}$	0	$O_{6,j,2}$	$O_{7,j,2}$
0	0	$O_{3,j,3}$	0	$O_{5,j,3}$	$O_{6,j,3}$	$O_{7,j,3}$

Afin de respecter l'ordre des opérations dans leurs gammes opératoires, on tente d'utiliser un vecteur dont la taille est égale au nombre total des opérations de toutes les gammes opératoires, en prenant en compte leurs ordres spécifique.

Tout d'abord, le nombre total d'opérations sur les machines dans cet exemple, est égal à12. Nous accordons à chaque opération un numéro entre 1 et 12 en respectant l'ordre des opérations de chaque tâche. Un tableau est codé de 1 à 12. Sa taille est égale au nombre d'opérations des tâches et qui respecte l'ordre entre les opérations de la même tâche. Ainsi l'opération de la tâche 1 aura comme code 1. Pour la tâche 4, les opérations sont numérotées 4 et 5. Ainsi 1 correspond à l'opération $O_{1,1,1}$, 7 correspond à $O_{5,2,3}$.
On dit qu'un ordonnancement est faisable, s'il respecte la gamme opératoire de la tâche.

c. Génération de la population initiale

La population initiale conditionne fortement la rapidité de l'algorithme. Notre espace de recherche est représenté par l'ensemble des opérations en conflit sur la même ressource (demandant la même ressource partagée).

Un chromosome est construit en utilisant le générateur d'ordonnancement à partir des méta-heuristiques, appliquées sur l'ensemble de solutions, définies par leurs fonctions temps d'exécution t_{ex} (i), inséré dans la population initiale. Un chromosome est constitué des gènes composant les opérations définies par la matrice I × M.

La procédure qui génère le codage d'un chromosome est explicitée ci-dessous:
Entier j, k
Entier I : nombre de tâches
Entier J_i: nombre d'opérations dans la tâche i
Tableau d'entiers Chrom: de taille égale à $\sum_{i=1}^{I} J_i$

```
k = 0
j = 1
pour i = 1 à I  faire
   k = k + Ji
fin pour
pour j = 1 à k
   chrom [j] = j
fin pour
```

Algorithme de génération d'un chromosome

d. Calcul de la fonction d'évaluation

Le Makespan est la durée opératoire totale entre la date de fin de la dernière opération de l'ordonnancement et la date de début de sa première opération.

Pour le calculer il faut connaitre la date de début de chaque opération sur sa machine de production.

Nous définissons respectivement $E_{i,j}$, $S_{i,j}$ et $C_{i,j}$, la durée opératoire, la date début et la date de fin de chaque opération $O_{i,j,k}$.

$E_{i,j}$ est une donnée du problème; $C_{i,j}$ est donné par la formule suivante: $C_{i,j} = S_{i,j} + E_{i,j}$; $S_{i,j}$ est défini par la formule suivante: $S_{i,j} = \text{Max}\ (C_{i,j-1}, C_{h,l})$

Avec $O_{h,l,k}$ est l'opération précédente de $O_{i,j,k}$ sur la machine M_k et $C_{i,j-1}$ est la date de fin de l'opération $O_{i,j-1,k}$, qui précède $O_{i,j,k}$, dans la tâche i.

Les cas suivant se présentent:

- si $O_{i,j,k}$ est la première opération de la tâche i alors $C_{i,j-1} = 0$.
- si $O_{i,j,k}$ est la première opération à exécuter sur la machine M_k alors $C_{h,l} = 0$.

Le makespan est le maximum des dates de fin des dernières opérations pour toutes les tâches:

$C_i = \sum_j C_{i,j}$, c'est le temps d'exécution minimale pour effectuer toutes les opérations de la tâche i.

C_{max}= Max $\{C_i\}$, c'est le temps d'exécution maximale pour effectuer les toutes les tâches.

Comme on définit notre espace de recherche comme l'ensemble des solutions en conflit sur la même ressource, alors soit $[S_{i,j}]$ l'ensemble des tâches à allouer, et $[E_{i,j}]$ leur dates d'exécutions respectives. L'algorithme génère la configuration $S_{i,j,}$ une allocation possible qui représente le point optimal de l'espace des tâches en conflit sur la même ressource.

La fonction d'évaluation appliquée pour évaluer une allocation donnée est la suivante:

$F_i = \alpha_1 F_{nt} + \alpha_2 F_{att} + \alpha_3 F_{ex} + \alpha_4 F_{réel}$

- F_{nt} représente le nombre de tâches placées en attente sur la même ressource, c'est-à-dire, le facteur de charge affecté à cette ressource,
- F_{att} représente le temps d'attente d'une ressource pour qu'elle soit disponible
- F_{ex} représente la durée d'exécution de la future opération de la gamme en conflit
- $F_{réel}$ représente le temps de réalisation de la gamme i en question

$F_i = \alpha_1\, nt + \alpha_2\, d_{i,j}^k + \alpha_3\, E_{i,j} + \alpha_4\, C_{i,j-1}$

Avec:

- *nt* est le nombre de tâches en conflit sur la même ressource k (la charge de la ressource k)
- $d_{i,j}^m$ est la durée d'attente pour effectuer l'opération j de la gamme i sur la machine k
- $E_{i,j}$ est la durée d'exécution de l'opération j de la gamme i sur la machine M_k
- $C_{i,j-1}$ est le temps de réalisation de la gamme i, jusqu'à la réalisation de l'opération j.

Les poids α_1, α_2, α_3, α_4 pondèrent certains paramètres par rapport à d'autres. Si les valeurs de ces poids sont proches, les composants de la fonction coût seront équilibrés. Dans notre stratégie, nous privilégions les paramètres, F_{nt} et F_{att}, c'est pourquoi leurs poids α_1 et α_2 seront plus élevés par rapport aux autres.
La notion de poids dans la formule de calcul de F_i, participe à la prise en compte des préférences du décideur vis à vis d'une stratégie de résolution.
Il est ainsi possible de favoriser un critère par rapport aux autres en jouant sur l'importance relative de sa pondération (α_1, α_2, α_3, ou α_4).

e. Sélection

Notre choix s'est porté sur la sélection déterministe. Elle consiste à associer à chaque chromosome i de la population, une fonction d'évaluation F_i, correspondant à l'individu i (proportionnelle à sa performance d'adaptation au système de production). Ainsi, les individus seront triés par ordre croissant selon leur fonction d'évaluation. On sélectionne la moitié supérieure parmi les individus, qui correspond à ceux qui sont les mieux adaptés. Les individus dont la valeur de la fonction d'évaluation est élevée, seront éliminés.

Le pseudo code de la sélection déterministe est donné par:

```
N: est la taille de la population (nombre d'individus)
w, u, v: sont des constantes
p: tableau de taille N
pop: tableau de taille w
Pour  i = 1…N-1
  Pour j = i+1 …N
     Si  (p[Fi ] > p [Fj ]) alors
        v =p[ Fi]
        P[Fi ] =p[ Fj]
        P[Fj ] = v
     Fin de si
     Incrémente j
  Fin pour
  Incrémente i
Fin pour
Si (N modulo 2 ==0)
    w = N / 2
sinon
    w = N / 2 + 1
fin de si
pour  i = 1…w
    pop[i] = p[i]
fin pour
```

f. Opérateur de croisement

Un autre croisement a été ajouté, appelé croisement heuristique. Cette opération est particulière pour les trois raisons suivantes:
- elle utilise les valeurs de la fonction objectif (valeur de la fonction d'évaluation F_i),
- elle produit un seul enfant,
- elle peut ne pas produire d'enfant.

Nous avons effectué un nouveau type de croisement. Il est semblable au mélange de couleur. On mélange deux couleurs (parent) pour fournir une autre couleur (enfant) obtenu sur la base de ces deux couleurs.

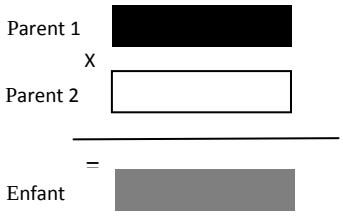


Figure 3.6 : croisement des couleurs.

L'idée ou la règle appliquée à ce croisement, vient de la capacité de l'opération arithmétique de multiplication, qui ne peut générer q'un seul enfant z à partir du produit de ces deux parents *x* et *y*. Cet enfant est plus puissant que ces deux parents, et il porte les caractéristiques de ces deux parents.
Ainsi, le produit entre deux données x et y, produit une troisième donnée z, plus riche en information.
Cet opérateur génère donc, un enfant $O_{i,j+1,k}$ à partir des parents $O_{i,j,k}$ et M_k en appliquant la règle suivante: $chr[O_{i,j+1,k}] = chr[O_{i,j,k}] \times M_k$, ce qui permet de calculer le temps d'exécution futur, de la gamme i après l'affectation de la ressource partagée k: $C_i = \sum_j C_{i,j}$, ainsi que la prise en compte de sa future fonction d'évaluation F_i :
$F_i = \alpha_1 nt + \alpha_2 d^k_{i,j+1} + \alpha_3 E_{i,j+1} + \alpha_4 C_{i,j}$ de la gamme i.

g. Opérateur de mutation
Afin d'éviter d'établir des populations uniformes, incapables d'évoluer, on applique dans l'algorithme, l'opérateur de mutation qui modifie les valeurs des gènes du chromosome d'un seul parent.
Il s'agit d'une mutation réelle uniforme où on modifie le chromosome, représenté par une variable réelle, par une autre variable réelle choisie par la méthode de triage des valeurs des chromosomes représentant les gammes en conflit.
Si un chromosome est choisi pour la mutation avec une probabilité de mutation $P_m = 0.01$, on tire alors un nombre aléatoire $r \in [0,1]$. La procédure suivie est la suivante : si $r < 0.5$, on cherche parmi les tâches en conflit sur la ressource, celle dont la valeur de la fonction d'évaluation est la plus petite, F_{imin}= Min $\{F_i\}$. On échange ensuite les emplacements des opérations de la tâche retenue et ceux de la tâche qui était affecté à la ressource, en respectant les gammes opératoires de chaque tâche. Si (si $r > 0.5$), la même opération est exécutée, sauf que la tâche élue est celle dont la fonction d'évaluation est la plus grande, F_{imax}= Max $\{F_i\}$. De la même manière, on échange les emplacements des opérations entre ces deux tâches en respectant les gammes opératoires de chacune d'elles. Cet opérateur de mutation est explicité dans l'algorithme suivant:

Entier j, c_1, c_2, c_3, k_1, k_2, k_3, k;
k: représente la taille du chromosome; Réel r;
chrom: tableau des chromosome de taille k;
op_1: recherche de l'opération qui opère sur la ressource partagée dont la valeur est F_{imax};
op_2: recherche de l'opération qui opère sur la ressource partagée dont la valeur est F_{imin};
op_3: recherche de l'opération qui opère sur la ressource partagée dont la valeur est F_i;

$k_1 = op_1$
$k_2 = op_2$
$k_3 = op_3$
pour j de 1 à k faire
si (k_1 == chrom [j]) alors
$c_1 \leftarrow j$
fin si
si (k_2 == chrom [j]) alors
$c_2 \leftarrow j$
fin si
si (k_3 == chrom [j]) alors
$c_3 \leftarrow j$
fin si
fin pour
r = rand [0, 1],
si r > 0.5 alors
$j \leftarrow$ chrom [c_1]
chrom [c_1] $\leftarrow$ chrom[c_3]
chrom [c_3] $\leftarrow j$
sinon
$j \leftarrow$ chrom [c_2]
chrom [c_2] $\leftarrow$ chrom[c_3]
chrom [c_3] $\leftarrow j$

Pseudo code de la mutation

En effet, cet opérateur permute l'emplacement des opérations de deux tâches différentes.

h. Méthode d'insertion (remplacement générationnel)

Dans cette approche, le remplacement générationnel est total. À cet effet, la population parents doit être remplacée entièrement par la population enfants. La durée de vie des individus de la population parents se limite alors à une seule génération, ce qui permet d'augmenter la durée de vie des meilleurs individus.

Pour une ressource partagée nous avons utilisé la méthode d'insertion par triage effectuée sur tous les individus de la population enfants et qui fait ressortir le meilleur individu (la meilleure solution ou bien solution optimale) de cette population. Pour cette dernière qui correspond à une opération a réalisée, on lui affecte l'allocation de la ressource partagée. Dès que la ressource est libérée, les opérations restant en attente seront de nouveau les individus de la population courante en lui ajoutant s'il y a lieu les nouveaux individus en conflit sur la même ressource. La procédure de réordonnancement sera de nouveau appliquée.

i. Critère d'arrêt

Le critère d'arrêt, est défini par le nombre maximal de générations (itérations), G = 1 (représente le taux de remplacement de génération), puisque il peut arriver que de nombreuses simulations continuent à être exécutées, ainsi que la recherche de l'optimum alors que ce dernier a déjà été trouvé.

4.3 Implantation du Recuit Simulé

a. Fonction objectif

Notre objectif principal est la minimisation du coût de production. Ce dernier est réalisé par la minimisation du temps de production représenté par la fonction d'évaluation F_i. Si les valeurs de la fonction objectif F_i sont proche les unes des autres et de petite valeur alors il est souvent nécessaire de transformer la fonction objectif F_i en une fonction d'adaptation F_{ia} qui est mieux exploitée, et qui permet de diversifier les valeurs assez proches.

Dans ce cas là, nous utilisons la formule suivante pour la transformation de la fonction d'adaptation [68]: $F_{ia} = F_{Max} - F_i$, avec $F_{Max} = Max \{F_i\}$, avec $i = 1 \ldots N_c$, N_c: représente la taille de la population (le nombre de configurations dans l'espace de recherche) composée des individus en conflit sur la même ressource partagée M_k.

Au début de l'exécution, les valeurs d'adaptation F_{ia} sont proches les unes des autres, même les meilleures solutions engendrent une convergence prématurée lors de l'application. Donc, un opérateur de réajustement (mise à l'échelle linéaire) de la fonction d'adaptation est appliqué afin d'accroître l'écart de performance d'individus [38] :

$F_{ia}' = \alpha \, . \, F_{ia} + \beta$ avec $\alpha = \frac{c.F_{aMax} - \overline{F_a}}{F_{aMax} - \overline{F_a}}$, $\beta = (1 - \alpha).\overline{F_a}$

$c \in [1.2, 2]$: est le nombre de copies souhaitées pour le meilleur individu de la population. Cette valeur de c a été utilisée avec succès [38].

Améliorations apportées

Dans notre cas nous utilisons le meilleur individu avec une seule et unique copie c = 1, comme tous les autres individus de la population. Ceci ce nous conduit à apporter quelques modifications sur l'opérateur de réajustement :

$F_{ia}' = \alpha \, . \, F_{ia} + \beta$ avec $\alpha = \frac{F_{aMax} - \overline{F_a}}{\overline{F_a} - F_{aMin}}$, $\beta = (1 - \alpha).\overline{F_a}$ Avec $F_{aMax} = Max \{F_{ia}\}$, $F_{aMin} = Min \{F_{ia}\}$, et $\overline{F_a} = \frac{1}{N_c}\sum_{i=1}^{N} F_{ia}$: la fonction d'adaptation moyenne de la population.

Il est préférable d'implanter de petits mouvements simples qui permettent de modifier légèrement la solution actuelle, en proposant des zones de recherche. Donc pour orienter la recherche dans une direction efficace qui permet d'exploiter tout l'espace de recherche, on introduit des modifications qui consiste à modifier la variable courante en effectuant un mouvement qui doit être déterminé en utilisant une stratégie efficace.

b. Température initiale T_0

La température initiale T_0 est fixée. Elle représente la configuration où la valeur de la fonction objectif est très élevée, pour permettre à toutes les configurations de l'espace de recherche d'être acceptées au début. Dans notre application nous avons choisi T_0, de valeur : $T_0 = Max \{F_{ia}'\}$,

c. Décroissance de la température

La loi de décroissance de la température utilisée est linéaire, $T_{k+1} = 0{,}9.T_k$. Elle permet d'avoir une décroissance rapide au début de l'optimisation, et lente à l'approche de la convergence. La procédure est initialisée avec la valeur de k = 0, on incrémente ensuite, jusqu'à ce que $T_{k+1} \leq T_{inf}$, et T_{inf} est donnée par l'expression suivante: $T_{inf} = Min \{F_{ia}'\}$.

d. Nombre d'itérations à température constante

Le nombre d'itérations à température constante sera égale au nombre de configurations disponibles dans l'espace de recherche à une température constante N'_c, avec la condition que N'_c soit inférieur ou égale à N_c.

e. Critère d'arrêt

Dolan, en 1990 [28], a proposé d'arrêter l'algorithme du recuit simulé lorsque la température de recuit est inférieure à une valeur choisie. C'est notre choix aussi. Dans l'algorithme proposé, l'arrêt intervient donc, lorsque le paramètre de contrôle T est inférieur à une certaine valeur de température T_{inf} (soit $T \leq T_{inf}$), T_{inf} étant la plus basse température.

La modification que nous avons apportée à la méthode du recuit simulé s'inspire de la méthode de dichotomie (« couper en deux » en Grec). Cette dernière est, en algorithmique, un processus itératif de recherche où, à chaque étape, on coupe en deux parties (pas forcément égales) un espace de recherche que l'on restreint à l'une des deux parties. Donc, au lieu de continuer la recherche dans tout l'espace de solutions, on essaye d'accéder seulement à une partie de l'espace de recherche où il est, le plus probable de trouver la solution optimale.
On suppose bien sûr qu'il existe un test relativement simple permettant à chaque étape de déterminer l'une des deux parties, dans laquelle se trouve une solution. Pour optimiser le nombre d'itérations nécessaires, on s'arrangera pour choisir à chaque étape deux parties sensiblement de même « taille »
L'algorithme s'applique typiquement à la recherche d'un élément dans un ensemble fini. Donc à chaque étape, on coupera l'espace de recherche en deux parties de même taille (à un élément près) de part et d'autre de l'élément médian.
On a représenté l'espace de recherche (l'ensemble de solutions) par un ensemble de températures enregistrées dans un tableau t de taille N_c (nombre de configurations ou de solutions) $t[N_c]$. Les valeurs sont comprises entre une valeur maximale et une valeur minimale, qui sont respectivement T_0, et T_{inf}, avec $T_0 = \text{Max}\{F_i\}$ et $T_{inf} = \text{Min}\{F_i\}$.

Le medium de l'espace de recherche sera donné par la relation suivante :

$$p = T_{inf} + \frac{T_O - T_{inf}}{2} = \frac{T_O + T_{inf}}{2}$$

On débute la recherche par la sélection d'une solution initiale $x^* = x_0 = \text{rand}[T_{inf}, T_0]$ choisie aléatoirement dans l'espace de recherche. Pour aller vers le voisin de la solution courante, on passe par les étapes suivantes. On associe d'abord, un autre tableau contenant la distance séparant chaque solution i du point medium calculé, $d[y_i] = \text{dist}(F_i - p)$. Le minimum des distances $Y = \min | d[y_i] |$ sera ensuite additionné au point medium, pour former le voisin recherché. Cette orientation de recherche a une probabilité élevée de se rapprocher de l'optimum.

f. Algorithme de RS

Finalement, l'algorithme proposé du recuit simulé se déroule comme suit:
1. initialisation: générer un tableau de solutions $t[N_c]$, N_c est le nombre des configurations dans l'espace de recherche, $i = 0 \ldots N_c$, $k = 0$.
Soit x_0 le point initial. Il représente une configuration prise aléatoirement de l'espace de recherche, c'est à dire le point de départ, donné par la relation, $x_0 = \text{rand}[T_{inf}, T_0]$. Le point initial est l'optimum courant $x^* = x_0$, $F^* = F_0$.
2. calcul de : $T_0 = \text{Max}\{F_i\}$, et $T_{inf} = \text{Min}\{F_i\}$.

3. détermination de la valeur du point medium de l'espace de recherche à partir de la relation: $p = \frac{T_O + T_{inf}}{2}$.
4. calcul des distances entre chaque solution et le point medium, $d[y_i] = dist(F_i - p)$. Le minimum de ces distances (le plus proche du point medium) est noté par $Y = \min | d[y_i]$ Récupérer la valeur $d[y_i]$ pour laquelle Y est minimale.
5. générer le nouveau point à partir de la formule suivante: $x_1 = p + d[y_i]$,
6. Si la valeur $\Delta x = x_1 - x_0 < 0$, alors le nouveau point devient l'optimum courant $x^* = x_1$, $T_0 = x_0$; Sinon, calculer la probabilité de Boltzmann $p(x_0, x_1) = e^{\frac{-\Delta x}{T_k}}$ à la température courante.
Si le critère de Metropolis, $p(x_0, x_1) > q$ est vérifié, alors le nouveau point devient le point de départ $x^* = x_1$, $T_0 = x_0$, avec $q \in [0, 1]$. Sinon il est rejeté $x^* = x_0$, $T_0 = x_1$.
7. retourner à 3, N_c fois.
8. réduire la température courante selon la loi de décroissance de température $T_{k+1} = 0{,}9 .T_k$.
9. Si le critère d'arrêt ($T_{k+1} \leq T_{inf}$) n'est pas satisfait retourner à 2. Sinon l'optimum est obtenu.

4.4 Implantation de la Recherche Tabou.

a. Espace de recherche

Dans la recherche Taboue, le voisinage d'un point donné est un mouvement aléatoire dans le voisinage de ce point. Soit $x_i \in S$ (espace de recherche), x_j est le nouveau point obtenu à partir de x_i selon la relation suivante: $F_j = F_i + rand. p$, avec : $-1 \leq rand \leq 1$, et $F_j \geq 0$.
p est la valeur du pas de déplacement dans l'espace de recherche utilisé pour générer un mouvement aléatoire et définir un voisinage. F_j est un mouvement aléatoire faisable. S'il répond aux contraintes, il est noté V_{Fj}.
Pour une configuration F_i et un pas donné p, le voisinage $V(F_i, p)$ est défini de la façon suivante: $V_{Fj} = V(F_i, p) = \{F_j : | F_i - F_j | \leq p\}$, où la configuration F_j est aléatoirement générée dans le voisinage $V(F_i, p)$ de F_i.
Dans notre cas, on va prendre la valeur du pas selon l'expression suivante :

$$p = \frac{\mathbf{Max}\{F_i\} - \mathbf{Min}\{F_i\}}{N_c}$$

La performance de la recherche Taboue dépend de la taille du voisinage et la longueur de la liste Taboue. La taille du voisinage est le nombre de configurations F_j générées dans le voisinage V_{Fi} de F_i avec un pas donné p. La longueur de la liste Taboue est choisie de tel sorte qu'elle ne soit pas trop longue, pour ne pas empêcher l'accès à des zones de recherche plus prometteuse, et pas trop courte pour qu'elle ne tombe pas dans le piège de cycle.
Pour ne pas garder tous les déplacements en mémoire (trop coûteux), on empêche l'accès à certaines solutions pendant un certain nombre d'itérations, et la longueur de la liste Taboue est donc prise à $t = 2/3. N_c$. Cette valeur est choisie après avoir plusieurs essais et simulations.

Dans la recherche Taboue universelle, la meilleure solution dans le voisinage de l'état courant est utilisée comme nouveau point de départ même si la fonction objectif n'est pas meilleure. On autorise ainsi une dégradation de la fonction objectif.

b. Algorithme de RT

L'algorithme se déroule comme suit:
Définissons le voisinage comme l'ensemble des solutions possibles, sur une ressource partagée, sous le respect des contraintes d'ordonnancement. Notons N_c, le nombre des configurations dans l'espace de recherche, et N nombre d'itérations.

1. construire le pas de déplacement en utilisant

$$p = \frac{\mathbf{Max}\ \{\mathbf{F_i}\} - \mathbf{Min}\ \{\mathbf{F_i}\}}{N_c}$$

2. initialisation: la liste Taboue est vide, L[t] = 0, avec t = 2/3. N_c,
3. soit un point de départ généré aléatoirement x_0. Le meilleur point est le point initial $x^* = x_0$, et $F^* = F(x_0) = F_0$.
4. générer un mouvement aléatoire faisable (qui respecte les contrainte de précédence) dans le voisinage V (F_0, p) de la meilleure solution courante, en utilisant le pas p, $F_1 = F_0 + p.\ rand$, avec rand $\in$[-1, 1].
5. Soit x_1 le meilleur point dans tout le voisinage, F_1 est sa fonction objectif.
5. Si $F_1 \leq F^*$ et que la solution x_1 n'est pas Taboue, on retient la solution, en remplaçant la solution courante par la meilleure solution trouvée (meilleure solution voisine), et on pose $x^* = x_1$, $F^* = F_1$.
6. mettre à jour la liste Taboue (ajouter cette solution dans la liste taboue. Elle sera taboue pendant les t prochaines itérations), et retourner à l'étape 4.
7. Si le critère d'arrêt (tant qu'il ne dépasse pas N itérations consécutives) n'est pas atteint, alors retour à l'étape 3.

5. CONCLUSION

L'objectif essentiel de ce chapitre était de proposer une méthodologie de résolution du problème dans le domaine d'optimisation. Le problème a été transformé en un problème d'optimisation avec contraintes en se basant sur un formalisme mathématique.
La difficulté du problème réside dans son aspect combinatoire, puisqu'il comprend un nombre de solutions admissibles important parmi lesquelles on cherche les meilleures par rapport à un critère donné. Le champ de recherche de ces problèmes en environnement dynamique étudie les mécanismes de recherche de l'optimum. De nombreux domaines scientifiques tentent d'appliquer ces méthodes à leurs problèmes spécifiques.

Les méthodes méta-heuristiques sont actuellement en plein développement, et offrent beaucoup de perspectives dans la résolution des problèmes d'optimisation. Il nous a paru pertinent de les intégrer et de les exploiter comme approche ou technique de résolution des problèmes combinatoires et des problèmes d'affectations sous contraintes, comme le problème qui nous préoccupe et que nous avons posé dans ce travail.
Puisque, nous ne pouvons pas prévoir l'efficacité de l'une ou l'autre parmi les méta-heuristiques sur une instance donnée, et il n'existe pas de méthode particulière qui garantisse ce choix, alors, les trois méta-heuristiques sont sollicitées en même temps, et la plus appropriée parmi elles, c'est-à-dire celle qui donne la meilleure solution est choisie.

Pour cela, il y a lieu de favoriser des tests descriptifs, c'est à dire identifier les facteurs pertinents, supposés influencer les performances, et effectuer des comparaisons en faisant varier ces facteurs [13], et décider en fin de compte.

En matière de perspectives de recherche, Il faut d'une part poursuivre des études expérimentales sur les propriétés des méta-heuristiques, et d'autre part renforcer et élargir les recherches dans l'espace de recherche. Ces points ont fait l'objet d'études récentes, et méritent sûrement davantage d'attention.
Les méthodes les plus prometteuses que nous avons utilisé, sont les algorithmes génétiques, le recuit simulé et la recherche taboue.

Nous avons étudié les aspects fondamentaux des algorithmes génétiques standards, et avons proposé des améliorations pour les rendre plus performants. Un nouveau critère a été proposé. Il offre l'avantage de trouver l'optimum d'une fonction objectif avec la localisation de l'optimum.
La méthode du recuit simulé a été détaillée, ainsi que les améliorations apportées. Une méthode de prédétermination du pas de déplacement a rendu l'algorithme plus stable. Un nouveau critère d'arrêt pour la phase à température constante a été proposé. Il est basé sur l'observation de l'évolution de la longueur du vecteur pas.
Enfin, la méthode de recherche taboue fondamentale, RTHu a été étudiée et choisie. Elle fait usage d'une mémoire flexible empêchant temporairement des mouvements qui pourraient faire retourner à une solution récemment visitée.

Cette approche, intégrant les trois méthodes dans le module de décision, sera testée sur une unité pilote et les résultats discutés, dans le chapitre suivant.

CHAPITRE 4

Application

1. Introduction

Le système de production requiert des méthodes de représentation et des techniques d'analyse, permettant de tenir compte de manière efficace des différentes fonctionnalités associées au système, ainsi que de ses caractéristiques temporelles qui constituent le cadre de notre étude. Il y a donc nécessité de pouvoir disposer de méthodes permettant de vérifier un certain nombre de propriétés du système modélisé.

Des opérations de traitement sur machines seront effectuées par le système de production, la validité des résultats à obtenir par association d'un modèle basé sur les réseaux de Pétri ne dépend pas uniquement de l'ordre dans lequel ces opérations seront obtenues, mais aussi du temps qui sera pris pour les obtenir. Les temps de retards constatés sur certaines opérations en demandant une ressource partagée ne se contentent pas seulement d'affecter les performances du procédé modélisé, mais elles peuvent directement porter atteinte à la qualité du produit fini.

En pratique, le lancement de plusieurs produits simultanément nécessite un ordonnancement de plusieurs tâches en parallèle. La résolution de ce problème nécessite une méthode d'optimisation combinatoire qui tient compte d'un certain nombre de paramètres imposés, notamment la quantité de produit à fabriquer, et les types de gammes de fabrication. Les limites de ces approches ont été présentées. C'est pourquoi nous avons, dans notre travail, utilisé de différentes méthodes méta heuristiques, notamment: la recherche tabou, le recuit simulé, et les algorithmes génétiques. Ces différentes techniques, seront comparées par rapport à leurs performances sur le problème de référence, et nous proposons des voies pour guider le choix d'une méta-heuristique par rapport à l'autre en pratique.

2. Présentation de la cellule de production pilote

2.1 Le modèle d'un système de production.

L'unité de production choisie, appartient à l'entreprise de fabrication de profilés Prometech et est implantée au niveau de la zone industrielle d'Annaba.

La fabrication des profilés utilise des supports mécaniques et nécessite plusieurs étapes. Sa description est la suivante :

- stockage de la matière première (profilés bruts),
- traitement sur machine,
- stockage des profilés finis.

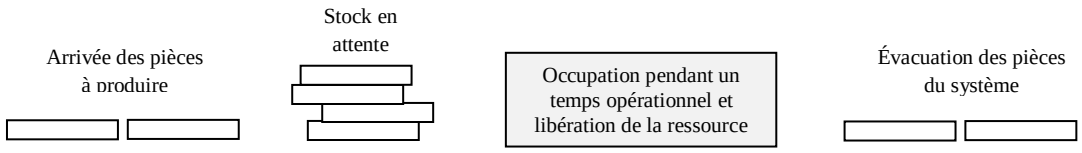


Figure 4.1: schéma du système à modéliser

Pour des installations de type ligne de fabrication, les systèmes de convoyage jouent un rôle important qui peut influer sur les performances globales. La modélisation de ces systèmes est donc nécessaire afin d'étudier leur incidence sur la productivité.

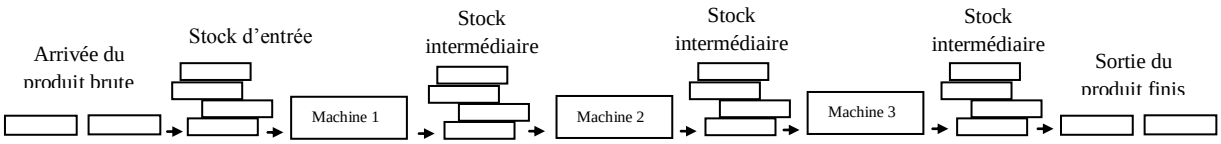


Figure 4.2: représentation d'une ligne de production

Les méthodes d'analyse des performances pour de tels systèmes, déterminent de façon analytique les caractéristiques des lignes de fabrication. Les transferts de produits entre machines et les stations de travail sont composés essentiellement de convoyeurs en série.

Un modèle basé sur les réseaux de Pétri est utilisé pour maintenir une image interne de l'état courant du système. Cet état évolue en fonction des occurrences d'événements. Le réseau de Pétri indique généralement à tout instant l'ensemble des décisions possibles, tout en respectant bien sûr les contraintes de disponibilité des ressources, de précédence entre tâches sur les gammes de production, ainsi que les contraintes portant sur les capacités de stockage, l'utilisation ou la réservation de moyens de transport, etc.

Après une mise en situation, le système de production traite des événements discrets et continus. Il est ainsi qualifié d'hybride. Sa modélisation globale, s'appuie donc sur les réseaux de pétri hybrides, et représente les différentes phases, du traitement d'acier, depuis le stock d'entrée jusqu'au stock de sortie.

2.2 Présentation de l'atelier de fabrication

Un stock d'entrée d'alimentation permet la livraison du profilé (matières premières). Il est supposé de capacité infinie. La chaîne est composée de trois machines M_1, M_2 et M_3. Elles représentent respectivement une machine de sciage à froid, une perceuse horizontale et une perceuse verticale. Des stocks intermédiaires de capacité infinie sont placés entre les machines, et enfin un stock de sortie. Ce dernier permet la livraison de profilés (produits finis) selon la demande des clients.

Les pièces seront évacuées à l'aide un système de transport, assuré par des convoyeurs.

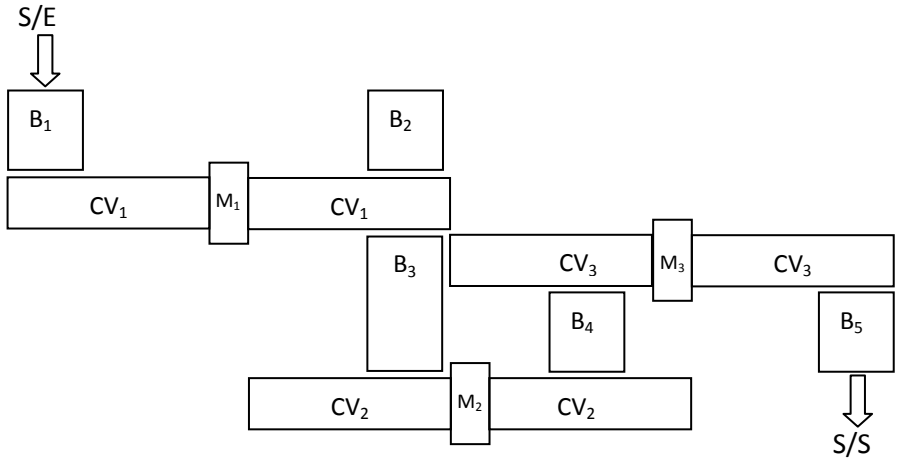


Figure 4.3: schéma synoptique de la cellule de production

Le matériel consiste en: une scierie à froid à course vertical, une perceuse horizontale à broches multiples et une perceuse verticale à broches multiples ainsi que la totalité des rouleaux mécaniques et des stocks intermédiaire destinés à enchaîner le travail entre les unités.

Le stock consiste en :
- les glisseurs nécessaires,
- les dispositifs pousseurs mécaniques,
- le transporteur mécanique à rouleaux.

Ce stock servira, soit à livrer les profilés au transporteur, soit à les transférer au deuxième stock intermédiaire.
Les stocks sont réalisés de façon à ce qu'ils puissent supporter des charges élevées, puisque les profilés peuvent être très lourds, de diverses qualités et épaisseurs.
Une partie transporteuse (convoyeur) constituée d'un prolongement, est destinée à transporter les profilés disposés des deux cotés et de bout à bout de ce prolongement.

Identification des paramètres de qualité des profilés
Pour les profilés, les paramètres fondamentaux et caractéristiques sont :
· Le poids en kilogramme
· Longueur en m
· Diamètre de perçage en mm
Ces paramètres de qualité dépendant du système de transformation, et de la matière première.

3. Modélisation de la cellule de production pilote

Toute modélisation d'un système est généralement dépendante de l'application dans laquelle elle sera utilisée. Nous cherchons donc une commande qui peut assurer un compromis entre les trois critères de performance de tout système de production: quantité, qualité et disponibilité par une approche s'appuyant sur un modèle basé sur le réseau de Pétri hybride.

La partie discrète du modèle décrit les opérations sur machines.
La partie continue du réseau représente le flux du profilé.

Dans ce qui suit nous allons présenter en premier lieu le modèle de la cellule de fabrication décrite précédemment avec les explications et les commentaires nécessaires à la compréhension de son fonctionnement. Ensuite, nous exposerons notre problématique pour construire un modèle qui permet d'intégrer les contraintes du produit fabriqué.

La construction du modèle global sera faite à partir de trois modèles de gammes de fabrication afin de pouvoir bien exprimer les situations de conflits qui apparaîtront sur la demande des ressources partagées (les machines de production et le convoyage).

Grâce au modèle élaboré, on peut analyser le comportement du système et déterminer son évolution. Dans ce modèle, les délais (durées) de traitement, ou vitesses de franchissements sont associés aux transitions. Elles modélisent les opérations ou les états correspondants, qui représentent l'évolution des opérations de transport et d'usinage. Le changement d'état n'a pas de durées, aussi le temps d'attente d'un événement est associé à une place. La transition en aval de la place sera franchie dès que l'événement apparaît.

La figure suivante présente un exemple d'illustration d'un produit fabriqué par trois types de machines M_1, M_2, et M_3 successivement. Leur temps de fabrication est respectivement d_1, d_2, et d_3. L'état du système est comme suit : la présence d'un profilé dans p_1 représente l'attente de la disponibilité de la ressource M_1, dès que la machine est disponible, la transition T_1 est franchie avec une vitesse de franchissement égale à $V_1 = 1 / d_1$, associée T_1. La présence d'un profilé dans p_2 représente l'attente de la disponibilité de la ressource M_2, dès

que la machine est disponible, la transition T_2 est franchie avec une vitesse de franchissement égale à $V_2 = 1 / d_2$, associée à T_2. La présence d'un profilé dans p_3 représente l'attente de la disponibilité de la ressource M_3, des que la machine est disponible, la transition T_3 est franchie avec une vitesse de franchissement égale à $V_3 = 1 / d_3$ associée à T_3. Des stocks intermédiaires sont utilisés pour stocker les produits au cours de l'exécution. On considère les stocks intermédiaires B_1, B_2, et B_3 de capacités de stockages C_1, C_2, et C_3 respectivement. La disponibilité globale du système est déterminée en tenant compte de l'influence des stocks intermédiaires.

Pour déterminer la disponibilité des ressources, on introduit la notion de taux de panne (λ_i) et de taux de réparation (μ_i). Ils sont calculés sur la base des données fournies par l'entreprise. Les places p_5, p_6, et p_7 sont des places indiquant la réparation des machines M_1, M_2, et M_3 respectivement.

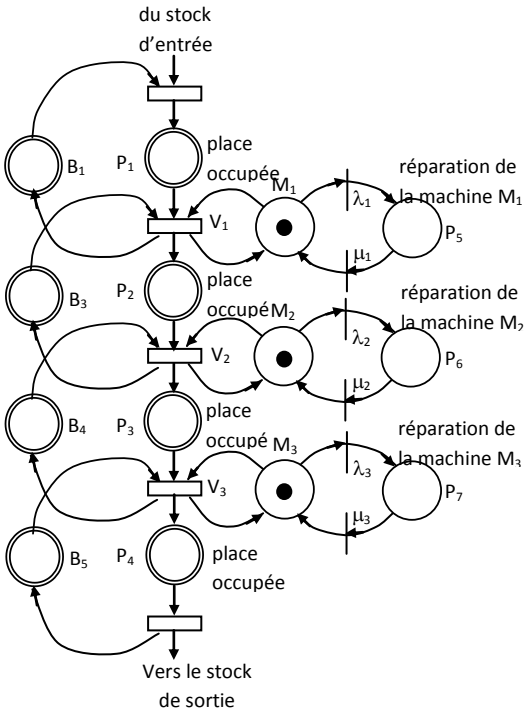


Figure 4.4: modèle de la cellule de production

Pour faire apparaître le problème de partage de ressource (conflit sur les ressources M_1, M_2, et M_3), on s'appuie sur la représentation du modèle composé de trois types de produits, c'est-à-dire, pour les trois différentes gammes de fabrication. Un outil de prise de décision complète le modèle basé sur les réseaux de pétri. Il résout le conflit en précisant quelle transition sera franchie parmi un ensemble de transitions en attente de la ressource partagée. La décision sera prise par l'exécution de l'une des méthodes méta-heuristique choisie, avec l'objectif de minimiser le temps de réalisation.

Un modèle fonctionnel de gamme de fabrication est un outil de simulation permettant le suivi de l'évolution de la production et de ses paramètres. Il offre des informations utiles pour évaluer les performances du système. A partir de ce modèle nous pouvons aussi dégager des scénarii d'optimisation de la production dans le contexte général.

Les deux grandeurs importantes dans la problématique posée, sont:
*Grandeurs d'entrée: nombres de profilés bruts avant traitement.
*Grandeurs de sortie: nombre de profilés fabriqués prêt à être évacués.

Présentation d'une gamme de fabrication

Le produit est représenté par une séquence d'opérations qu'on appelle gamme de fabrication, chaque opération est effectuée sur un type de machine d'usinages M_1, M_2, ou M_3, tout en faisant appel à des convoyeur de transport. L'arrivée des lots de profilés étant aléatoires, nous avons besoin d'un modèle d'ordonnancement dynamique capable de prendre en compte toutes les contraintes temporelles, de précédences, et des perturbations (Ex. panne sur machines, rupture des stocks,…). On introduit ci-après un exemple de trois gammes de fabrication:

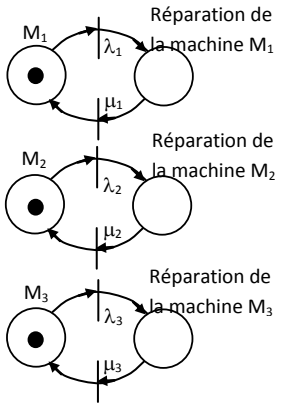
Gamme 1: $IN \xrightarrow{} B_1 \xrightarrow{Cv_1} M_1 \xrightarrow{Cv_1} B_3 \xrightarrow{Cv_2} M_2 \xrightarrow{Cv_2} B_4 \xrightarrow{} OUT$

Gamme 2: $IN \xrightarrow{} B_1 \xrightarrow{Cv_1} M_1 \xrightarrow{Cv_1} B_3 \xrightarrow{Cv_3} M_3 \xrightarrow{Cv_3} B_5 \xrightarrow{} OUT$

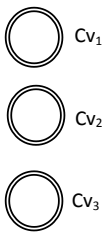
Gamme 3: $IN \xrightarrow{} B_1 \xrightarrow{Cv_1} M_1 \xrightarrow{Cv_1} B_3 \xrightarrow{Cv_2} M_2 \xrightarrow{Cv_2} B_4 \xrightarrow{Cv_3} M_3 \xrightarrow{Cv_3} B_5 \xrightarrow{} OUT$

Une modélisation compacte a conduit au modèle global construit sur la base d'un R.d.P hybride.

Nous avons comme ressources partagées:
- Les machines opérationnelles M_1, M_2, et M_3.



- Les convoyeurs de transport Cv_1, Cv_2, et Cv_3.



- Les stocks intermédiaires B_1, B_2, B_3, B_4 et B_5 de la cellule de fabrication sont de capacités infinies. Elles ne présentent donc pas, un état de conflit sur les ressources partagées.

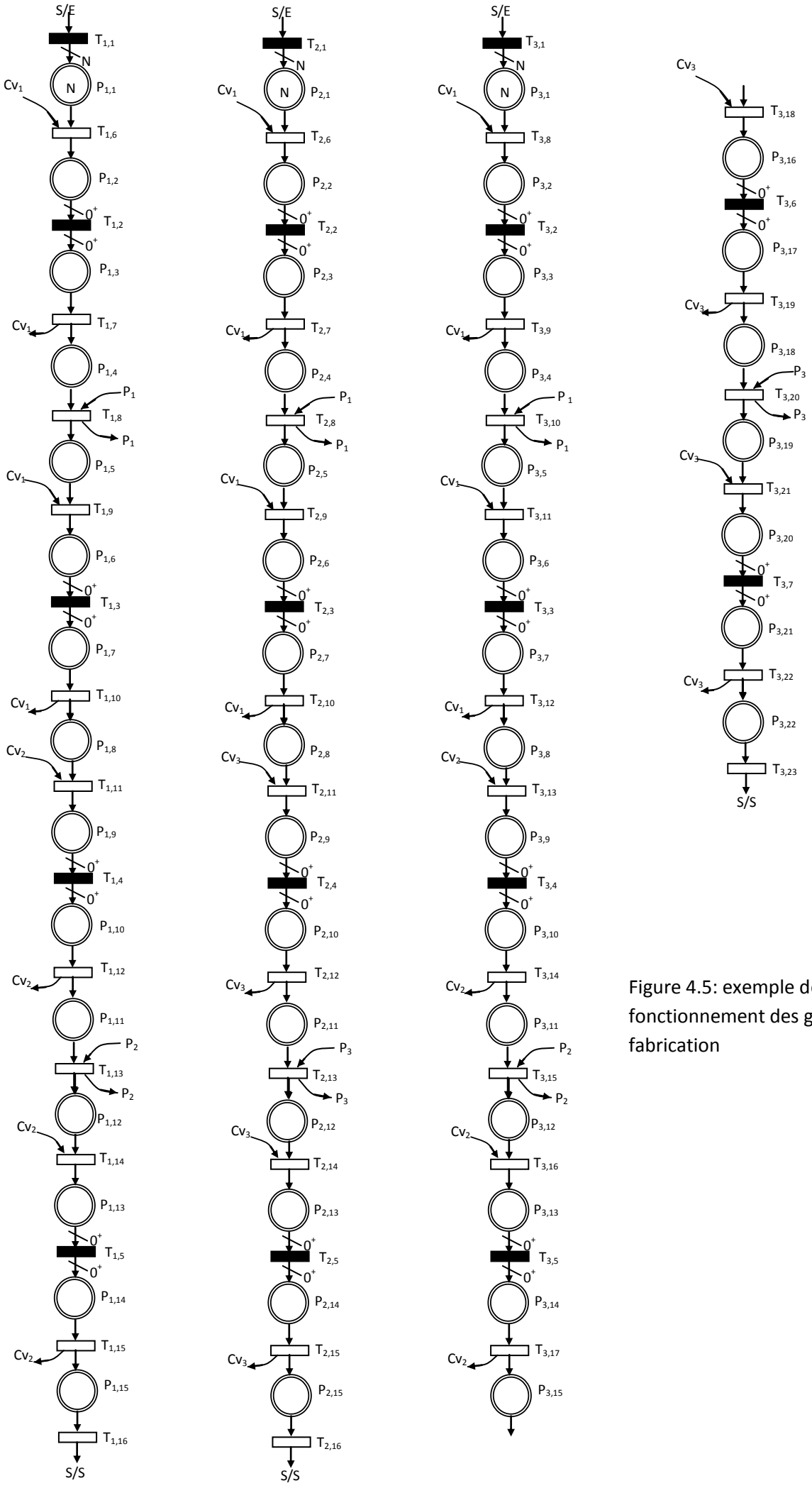


Figure 4.5: exemple de modèle de fonctionnement des gammes de fabrication

Désignations

Places
P_1: traitement sur machine M_1
P_2: traitement sur machine M_2
P_3: traitement sur machine M_3
B_1, B_2, B_3, B_4, B_5 : des stocks intermédiaires de capacités infinies
$P_{1,1}$: représente le nombre de pièces existant dans le stock intermédiaire B_1, disponible au début de course du convoyeur Cv_1
$P_{1,2}$, $P_{1,6,}$ $P_{1,9}$, $P_{1,13}$: quantité de pièces déposée sur le convoyeur
$P_{1,3}$, $P_{1,7}$, $P_{1,10}$, $P_{1,14}$: capacité maximale de transport
$P_{1,4}$: représente le nombre de pièces existant dans la zone de stockage intermédiaire B_1, disponible à la fin de course du convoyeur Cv_1
$P_{1,5}$: représente le nombre de pièces existant dans la zone de stockage intermédiaire B_1, disponible au début de course du convoyeur Cv_1
$P_{1,8}$: représente le nombre de pièces existant dans le stock intermédiaire B_3, disponible à la fin de course du convoyeur Cv_1 et au début de course du convoyeur Cv_2
$P_{1,11}$: représente le nombre de pièces existant dans la zone de stockage intermédiaire B_1, disponible à la fin de course du convoyeur Cv_2
$P_{1,12}$: représente le nombre de pièces existant dans la zone de stockage intermédiaire B_1, disponible au début de course du convoyeur Cv_2
$P_{1,15}$: représente le nombre de pièces existant dans le stock intermédiaire B_4, disponible à la fin de course du convoyeur Cv_2

Transitions
$T_{1,1}$: présence d'une pièce à l'entrée
$T_{1,2}$: temps nécessaire à un produit pour parcourir la longueur totale du convoyeur
$T_{1,3}$: $T_{1,2}$
$T_{1,4}$: $T_{1,2}$
$T_{1,5}$: $T_{1,2}$
$T_{1,6}$: vitesse de franchissement maximale du convoyeur (pièces / min)
$T_{1,7}$: vitesse de franchissement maximale du convoyeur (pièces / min)
$T_{1,8}$: vitesse de la machine M_1
$T_{1,9}$: $T_{1,6}$
$T_{1,10}$: $T_{1,7}$
$T_{1,11}$: $T_{1,6}$
$T_{1,12}$: $T_{1,7}$
$T_{1,13}$: vitesse de la machine M_2
$T_{1,14}$: $T_{1,6}$
$T_{1,15}$: $T_{1,7}$
$T_{1,16}$: présence d'une pièce à la sortie

Notre travail est de chercher à obtenir une allocation et un ordonnancement réalisable sur chaque ressource partagée de façon optimale sur un ensemble de tâches à réaliser. On essaie de trouver le meilleur ordre parmi les tirs de transitions, soumises à un ensemble de contraintes temporelles et correspondant aux tâches à exécuter sur un ensemble de ressources partagées. Le problème se ramène donc à celui de l'ordonnancement des dates de franchissement de ces transitions.

Le modèle réseau de Pétri permet de prendre en compte les contraintes temporelles (durées des tâches sous la forme de durées précises) qui seront exprimées à travers une fonction coût,

ainsi que les contraintes de précédence entre les événements qui doivent être vérifiées par tout comportement correct.

Le but est de maintenir une bonne performance de l'ordonnancement en présence d'incertitudes sur un ensemble de travaux qui est partitionné en sept sous-ensembles en fonction des gammes opératoires:
- le sous-ensemble $\{G_1\}$ composé des travaux n'utilisant que M1;
- le sous-ensemble $\{G_2\}$ composé des travaux n'utilisant que M2;
- le sous-ensemble $\{G_3\}$ composé des travaux n'utilisant que M3;
- le sous-ensemble $\{G_4\}$ composé des travaux passant sur M1, puis sur M2;
- le sous-ensemble $\{G_5\}$ composé des travaux passant sur M1, puis sur M3;
- le sous-ensemble $\{G_6\}$ composé des travaux passant sur M2, puis sur M3;
- le sous-ensemble $\{G_7\}$ composé des travaux passant sur M1, puis sur M2; puis sur M3;

En ordonnancement, une tâche de production est classiquement représentée par le numéro de la tâche i, et le numéro de l'opération j: (i, j). La position de la tâche dans le plan de charge de la machine indique que ce calcul est local à une machine. Cela permet de séparer les aspects locaux (une machine) des aspects globaux (la cellule) pour la mise en œuvre de la résolution.

Le problème consiste à ordonnancer un ensemble T_i de travaux sur un ensemble M_k machines avec k=1..m, $M = \{M_1, \ldots, M_m\}$. Chaque travail est constitué d'un ensemble de tâches devant s'exécuter chacune sur un ensemble de machines M_k, dans un ordre défini par la gamme de production.
L'objectif est de trouver un ordonnancement minimisant le makespan C_{max}, avec C_{max}= Max $\{C_i\}$, C_i : temps total d'exécution de la gamme i (la tâche i).
La stratégie de résolution, est fortement contrainte par le respect absolu des paramètres temps. La stratégie sélectionnée est alors mise en œuvre puis évaluée au vu de ses résultats.

$t_{i,j+1} - t_{i,j} \geq E_{i,j}$ Contrainte d'antériorité
· Numéro de tâche : i
· Numéro de l'opération : j
· Date d'accès à la ressource :$S_{i,j}$
· Durée de l'opération : $E_{i,j}$
· Nombre d'opérations dans la tâche : n_j

On affecte les temps d'exécution à l'évolution du marquage de chaque gamme au cours de la réalisation. Au début, les temps d'exécution des gammes de fabrication sont initialisés à zéro, et à chaque évolution du marquage, le temps d'exécution évolue selon la fonction : $t_{ex,i} = C_{i,j} = S_{i,j} + E_{i,j}$. Si un conflit apparaît sur une ressource partagée, on affecte à chaque gamme en conflit, une fonction d'évaluation $F_i = t_{ex,i}$ *(le temps d'arrivée de l'opération j de la tâche i à la machine considérée).* A l'arrivée d'une nouvelle tâche, le mécanisme d'ordonnancement recherche un nouvel ordre. L'ensemble de solutions obtenu, associé à l'ensemble des fonctions d'évaluations, est traité afin d'être optimisé par les méthodes méta-heuristique choisies. Ces fonctions d'évaluation sont définies par les dates $C_{i,j}$ des tâches. En fait, le principe est un peu plus complexe, puisque il faut également tenir compte des contraintes temporelles sur ces tâches. Le calcul ne sera pas effectué sur toutes les tâches du plan de charge, mais seulement sur celles qui sont en conflit sur la même machine de production. Il s'agit alors de déterminer à partir de l'espace de solutions, la tâche que la méta-heuristique appliquée aura désignée.

La résolution s'appuiera sur les algorithmes d'optimisation, appliqués aux méta-heuristiques modifiées que nous avons proposées.

4. Résultats et commentaires

Notre travail est basé sur une approche englobant l'allocation et l'ordonnancement des tâches.

A. Nous proposons l'utilisation d'un algorithme génétique, pour générer des solutions et choisir, parmi elles, celle qui convient le mieux à notre problème.

Pour l'algorithme génétique, un codage est appliqué sur l'ensemble des solutions trouvées tout en respectant leur nature et l'objectif souhaité. Il introduit un vecteur de nombre réel pour chaque ressource partagée. Chacun des vecteurs représente les tâches allouées à cette ressource sous le respect des relations de précédence. Les individus se reproduisent entre eux grâce à des opérateurs génétiques et donnent ainsi naissance aux autres individus. Le codage des individus et la population initiale générée sont un facteur déterminant pour la rapidité de la convergence vers les bonnes solutions.

Un nouvel algorithme génétique (GA), utilisant de nouvelles stratégies, a été développé dans F.Pezzella [65]. Il nous a donc paru utile de le confronter à l'approche que nous proposons, en traitant les mêmes données, et ainsi les situer l'une par rapport à l'autre à travers les résultats obtenus.

Il s'agit d'affecter les opérations des tâches aux machines, et le séquencement des opérations sur chaque machine, de la façon la plus proche de l'optimal tout en respectant les contraintes structurelles (contraintes de précédence), dans le but de minimiser le temps de réalisation global (Makespan).

Le cas d'étude proposé par F.Pezzella, représente une cellule de production composée de machines, et de produit à réaliser. Les données sont organisées dans une table, où les lignes correspondent aux opérations, les colonnes aux machines, et les entrées aux temps de traitement.

La population initiale est générée par l'approche par localisation, qui consiste à trouver l'affectation initiale par réordonnancement des tâches et des machines, et par la recherche du minimum global de la table d'instance. Elle prend en compte le temps de traitement, et la charge de travail sur la machine. La somme du temps de traitement des opérations est affectée à chaque machine.

Pour ordonner les opérations générées par la population initiale, deux règles sont proposées :

Régle1 : chercher le minimum global dans la table de traitement

Régle2 : permutation aléatoire des opérations et des machines dans la table.

Une fois les deux règles appliquées, il restera à déterminer le séquencement d'opérations sur la machine**.** Trois règles sont adoptées par F.Pezzella,

Règle 1: "The most work remaining (MWR)",

Règle 2: "The most number of operations remaining (MOR)"

Régle 3: "Randomly select job (Random)".

Dans la table suivante, la 1ère colonne représente le nombre de cycles à atteindre NC, la deuxième colonne ($M_{akespan}$AGF) représente la Makespan obtenu par la mise en œuvre de l'algorithme de F. Pezzella. On le compare avec la troisième colonne ($M_{akespan}$AGM) qui représente la Makespan obtenus par notre approche. La quatrième colonne représente l'écart

obtenu entre les deux méthodes ($E_{cart}M\%$) défini par:
$E_{cart} = (M_{akespan}AGM - M_{akespan}AGF) / M_{akespan}AGF *100\%$

NC	$M_{akespan}AGF$	$M_{akespan}AGM$	$E_{cart}M\%$
10	276	266	-3.62
20	570	526	-7.71
30	813	791	-2.70

Tableau 4.1 : table de comparaison du Makespan par application des deux méthodes

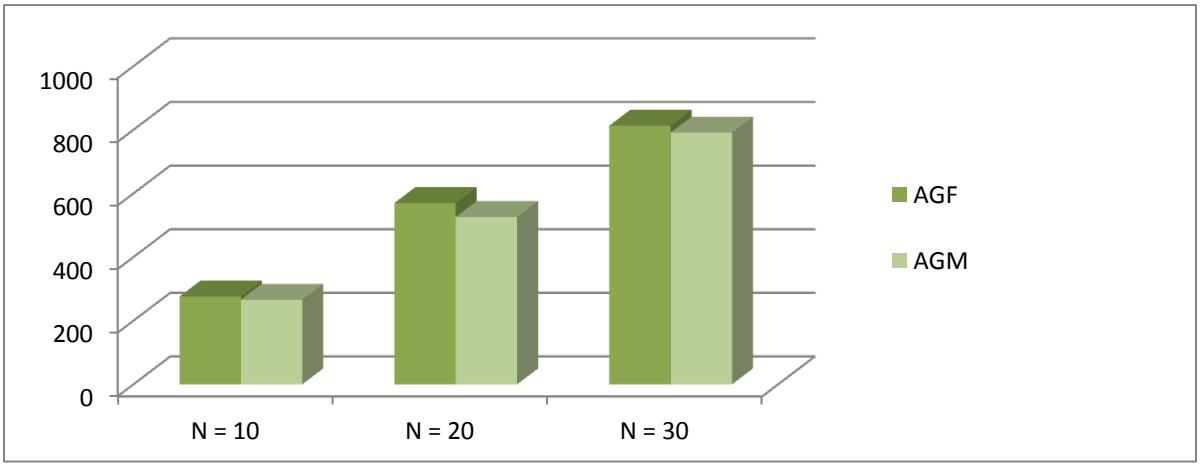


Figure 4.6 : schéma de comparaison du Makespan des deux méthodes

Dans la table suivante la deuxième colonne ($P_{rod}AGF$) et la troisième colonne ($P_{rod}AGM$) représentent respectivement la productivité obtenue par l'algorithme F. Pezzella, et par notre algorithme. La quatrième colonne indique la comparaison des deux méthodes, et définit l'écart de productivité ($E_{cart}P\%$), exprimée par :
$E_{cart}P = (P_{rod}AGM - P_{rod}AGF) / P_{rod}AGF *100\%$

NC	$P_{rod}AGF$	$P_{rod}AGM$	$E_{cart}P$ %
10	78.6	79.6	+1.27
20	80.1	82.7	+3.24
30	82.7	82.8	+0.12

Tableau 4.2 : table de comparaison de la productivité par application des deux méthodes

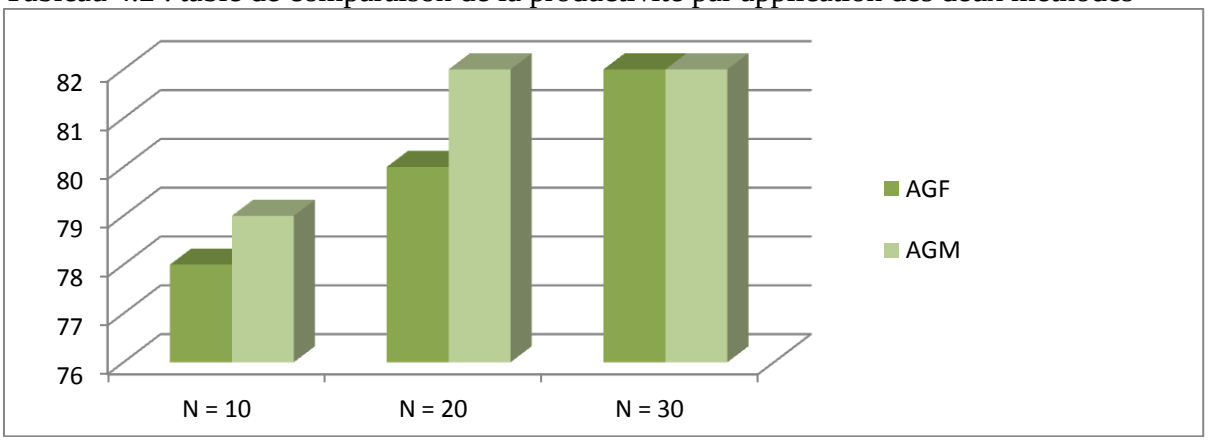


Figure 4.7 : schéma de comparaison de la productivité des deux méthodes

Il s'ensuit des résultats présentés dans les deux tableaux, que notre approche apporte une sensible amélioration, sur le Makespan et sur la productivité.

Il est clair que les techniques s'appuyant sur les algorithmes génétiques peuvent être un outil très intéressant dans la prise en charge des problématiques liées aux cellules de production flexible.

Ce résultat pourrait être encore amélioré, par l'exploration d'autres voies sur le choix des fonctions d'évaluation des individus par exemple, ou par l'utilisation d'une combinaison de différents critères de sélection pour choisir les meilleurs individus d'une génération.

Deux critères sont appliqués sur le choix de la fonction d'évaluation des individus. Ils sont défini par:

$F_{1i} = \alpha_1 nt + \alpha_2 d_{i,j}^{k} + \alpha_3 E_{i,j} + \alpha_4 C_{i,j-1}$

$F_{2i} = \alpha_1 E_{i,j} + \alpha_2 C_{i,j-1}$

Trois tables seront données par la suite, et chacune d'elles permet de faire ressortir l'avantage des opérateurs de sélection, de croisement et de mutation de l'algorithme génétique.
Dans chaque table, la 1ère ligne représente le nombre de cycles à atteindre, la deuxième ligne représente le Makespan obtenu par la mise en œuvre des deux critères de la fonction d'évaluation. La troisième ligne représente la productivité obtenue, et la quatrième ligne représente le taux de performance.

Les facteurs de productivité, fréquence de l'algorithme (vitesse de fabrication d'une pièce) et le taux de performance sont données par :

$$Productivité = \frac{nombre\ de\ pièces\ produites\ dans\ le\ temps\ total}{nombre\ théorie\ de\ pièces\ produites\ dans\ le\ temps\ total}$$

$$fréquence\ de\ l'algorithme = \frac{temps\ de\ fonctionnement\ total\ (Makespan)}{nombre\ de\ pièces\ produites\ dans\ le\ temps\ total}$$

$$taux\ de\ performance = \frac{temps\ net\ de\ fonctionnement}{temps\ brut\ de\ fonctionnement}$$

	Sélection + croisement					
	Critère 1			Critère 2		
Nombre de cycle	10	20	30	10	20	30
Makespan	318	656	983	357	681	1009
Productivité	85.7	82.7	82.5	75.9	79.4	80.3
Fréquence de l'algorithme	5.30	5.46	5.46	5.95	5.67	5.60
Taux de performance	84.9	82.3	82.4	75.6	79.2	80.2

Tableau 4.3 : résultats obtenu par l'application des deux opérateurs sélection et croisement

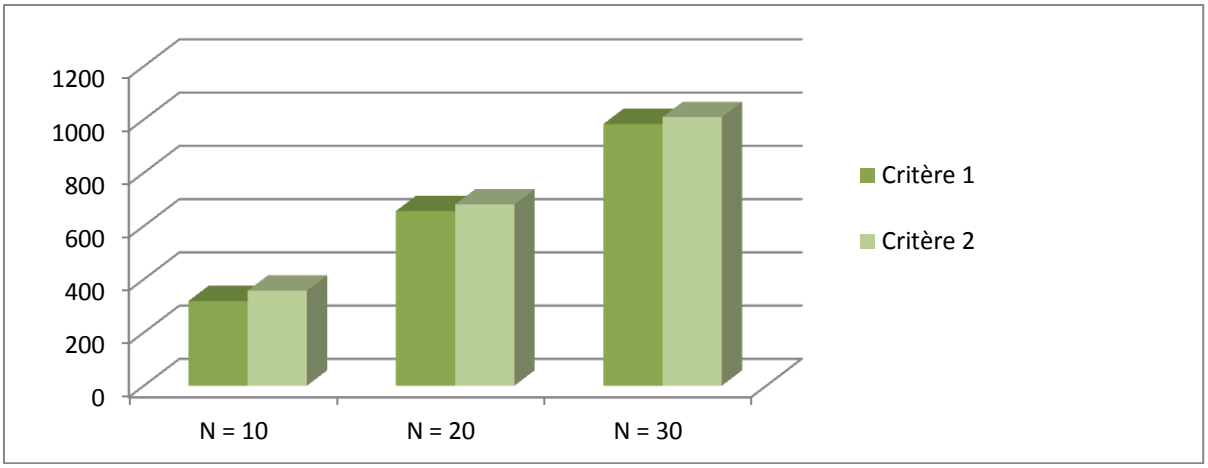


Figure 4.8 : schéma du Makespan obtenu par application des deux opérateurs S + C

	Sélection + mutation					
	Critère 1			Critère 2		
Nombre de cycle	10	20	30	10	20	30
Makespan	297	611	959	346	675	978
Productivité	90.9	88.8	84.5	78.9	80.0	82.9
Fréquence de l'algorithme	4.95	5.09	5.32	5.76	5.62	5.43
Taux de performance	90.9	88.3	84.4	78.0	80.0	82.8

Tableau 4.4 : résultats obtenu par l'application des deux opérateurs sélection et mutation

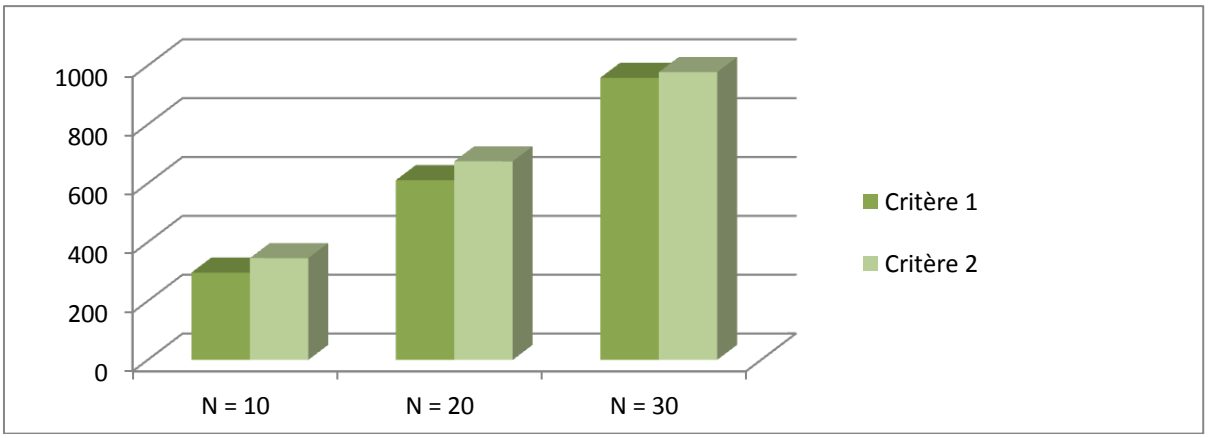


Figure 4.9 : schéma du Makespan obtenu par application des deux opérateurs S + M

	Sélection + croisement + mutation					
	Critère 1			Critère 2		
Nombre de cycle	10	20	30	10	20	30
Makespan	289	559	829	310	635	960
Productivité	93.7	96.7	97.8	88.2	85.1	84.5
Fréquence de l'algorithme	4.81	4.65	4.60	5.16	5.29	5.33
Taux de performance	93.4	96.6	97.7	87.0	85.0	84.3

Tableau 4.5 : résultats obtenu par l'application des trois opérateurs de l'algorithme génétique

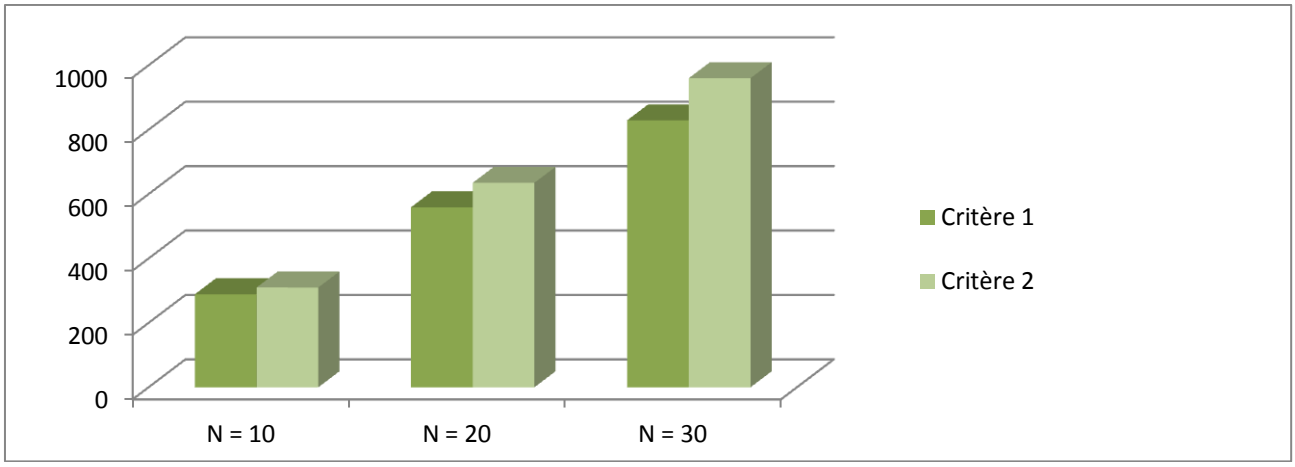


Figure 4.10 : schéma du Makespan obtenu par application des opérateurs S + C +M

Remarque
- d'après les trois tables on peut remarquer que le premier critère donne toujours de meilleurs résultats que le second. Ceci est dû au fait qu'il comporte plusieurs informations sur l'état de la ressource partagée ainsi que les tâches qui sont en conflit. Le choix de la fonction d'évaluation est important puisqu'il influe sur les résultats obtenus.
- l'emploi des trois opérateurs de l'algorithme génétique aide à trouver la meilleure optimisation dans un problème d'ordonnancement.

B. La seconde méthode d'optimisation, qu'on se propose d'utiliser est le recuit simulé. Parmi les solutions générées, le choix portera sur la meilleure, au sens du critère retenu.
Dans le recuit simulé, le facteur de température est remplacé par le temps de production qui est représenté par la fonction d'évaluation F_{1i} (puis F_{2i} respectivement) appliquée sur l'ensemble de solutions c'est à dire l'ensemble des opérations en attente d'une ressource partagée. Un critère de décroissance linéaire de température est utilisé pour permettre la visite de tout l'espace de recherche et d'aller ainsi vers la bonne solution. Un critère d'arrêt est associé à cette recherche et permet de réduire sa durée après que la solution optimale soit trouvée.

Le tableau 4.6 présente les résultats obtenus par application des deux critères F_{1i} et F_{2i} dans la méthode de recuit simulé. Chacun d'eux a été testé dix fois dans la méthode d'optimisation. Leurs moyennes ont été enregistrées dans le tableau qui suit.

	Recuit simulé					
	Critère 1			Critère 2		
Nombre de cycle	10	20	30	10	20	30
Makespan	304	594	877	297	604	907
Productivité	89.5	90.9	92.7	90.9	89.5	89.5
Fréquence de l'algorithme	5.06	4.95	4.87	4.95	5.03	5.03
Taux de performance	85.5	87.5	88.9	87.5	86.0	85.9

Tableau 4.6 : résultats obtenu par l'application de la méthode recuit simulé

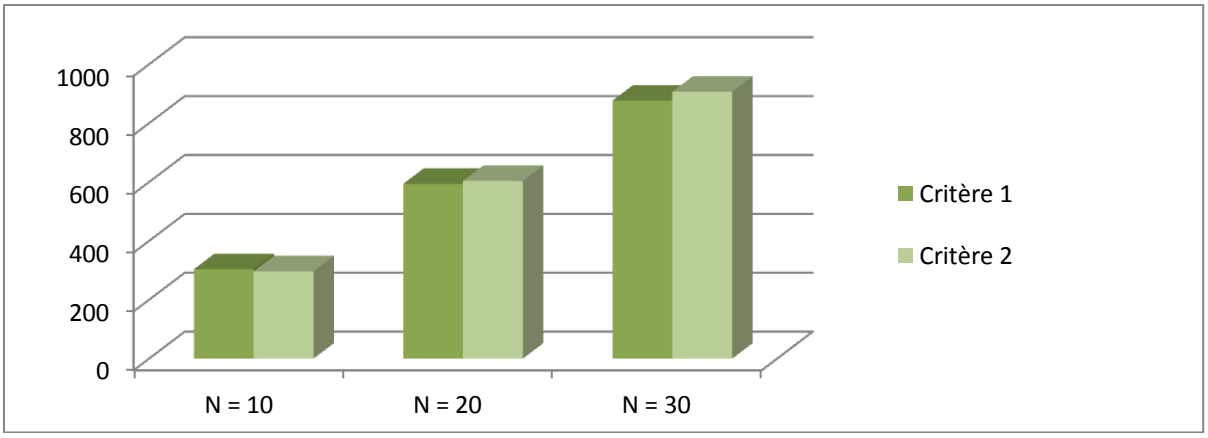


Figure 4.11 : schéma du Makespan obtenu par application de la méthode recuit simulé

Remarque
On remarque que la méthode de recuit simulé donne de meilleur résultat en employant le premier critère d'optimisation. D'où, l'insertion des temps morts (temps d'attente de disponibilité des ressources) dans la fonction d'évaluation peut considérablement améliorer la performance de l'ordonnancement.

C. La troisième méthode d'optimisation est la méthode de recherche Tabou. Il s'agit d'associer une table de mémorisation des solutions déjà visitées, à l'espace de recherche, défini par les fonctions d'évaluation des tâches en attente d'une ressource. L'interdiction d'une solution déjà visitée, nous permet d'explorer tout l'espace de recherche. Une mise à jour efficace est nécessaire pour cette recherche.

Une comparaison des deux critères est inclue dans cette étude. Cette dernière est liée à la taille de la liste Tabou, qui représente un facteur primordial dans la méthode de recherche Tabou. Dans le premier critère, la liste de tabou utilisée est égale au deux tiers de l'espace de recherche, tandis que dans le deuxième critère, la liste Tabou est égale à la moitié de l'espace de recherche. Les résultats obtenus ont été enregistrés dans le tableau 4.7.

	Recherche Tabou					
	Critère 1			Critère 2		
Nombre de cycle	10	20	30	10	20	30
Makespan	297	594	877	300	606	908
Productivité	90.9	90.9	92.7	90.9	89.5	89.5
Fréquence de l'algorithme	4.95	4.95	4.87	5.00	5.05	5.04
Taux de performance	87.5	87.5	88.9	86.6	85.8	85.9

Tableau 4.7 : résultats obtenu par l'application de la méthode recherche Tabou

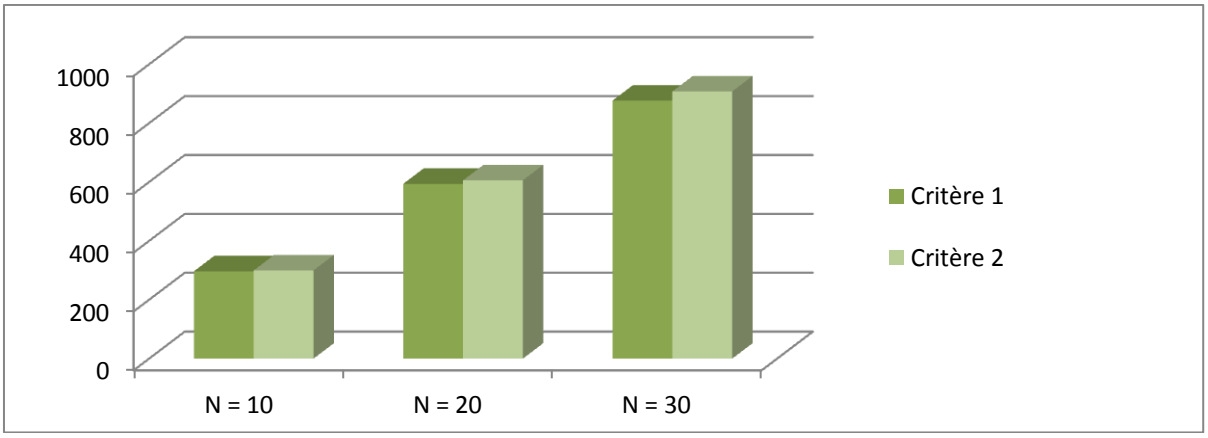


Figure 4.12 : schéma du Makespan obtenu par application de la méthode recherche tabou

Remarque
On remarque que dans la méthode recherche Tabou, la taille de la liste Tabou influe considérablement sur les résultats. Mieux sera choisi la taille de la liste Tabou, mieux serons les résultats obtenus.

D. Un tableau récapitulatif du Makespan des trois méthodes d'optimisation sera inséré par la suite qui donnera un point de vu sur les meilleurs résultats fournis par ces méthodes.

	Makespan					
	Critère 1			Critère 2		
Nombre de cycle	10	20	30	10	20	30
Recherche tabou	297	594	877	300	606	908
Recuit simulé	304	594	877	297	604	907
Algorithme génétique	289	559	829	310	635	960

Tableau 4.8: Nombre de pièces produites par chaque algorithme

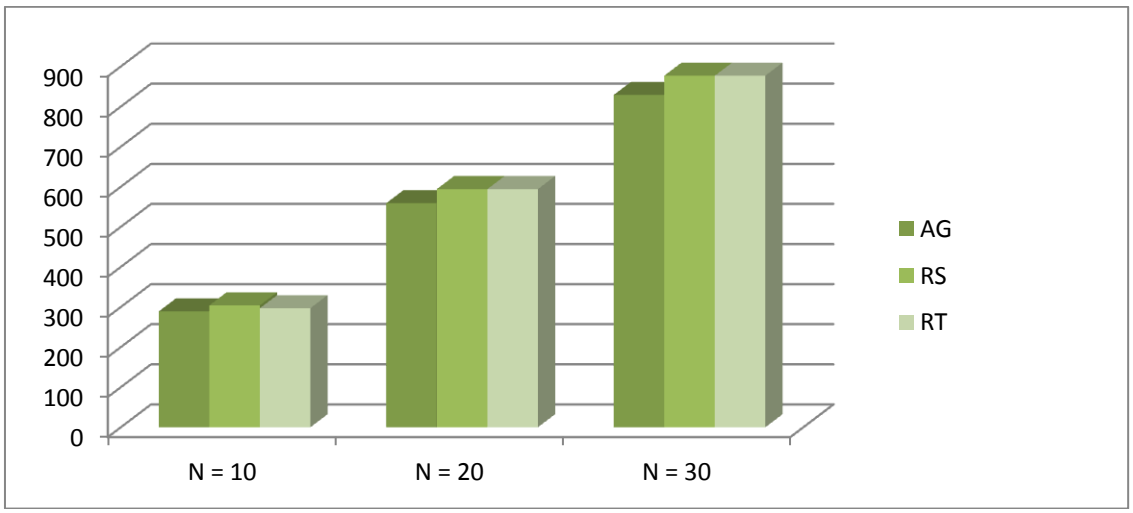


Figure 4.13 : comparaison du Makespan fournis par les trois algorithmes (critère 1)

Les histogrammes précédents montrent que les chances de trouver une solution optimale dans un ensemble de solutions sont améliorées en utilisant des critères définis selon le choix

de l'utilisateur, par application des paramètres de contrôle des opérateurs des méta-heuristiques.
D'après la figure 4.10, nous remarquons que l'amélioration des paramètres de contrôle des opérateurs génétiques accélère la convergence de notre algorithme vers l'optimum, et améliore aussi le temps de réalisation (Makespan).

E. Afin d'illustrer l'efficacité de l'association des trois algorithmes comme un moteur de décision dans un problème d'optimisation, nous allons les faire intégrer dans le même programme. La meilleure décision sera celle qui sera associée à la solution optimale résolvant le conflit.
Les résultats suivants font apparaître l'influence de chaque méta-heuristique sur l'ensemble de solutions, en prenant comme critère d'optimisation le Makespan, la fréquence d'utilisation de l'algorithme, et enfin le taux de performance. Dans le tableau suivant, nous avons effectué plusieurs tests d'optimisation en changeant le critère à optimiser, nous obtenons les résultats suivants:

	Les trois méta-heuristiques (AG+RS+RT)					
	Critère 1			Critère 2		
Nombre de cycle	10	20	30	10	20	30
Makespan	264	524	784	280	560	840
Fréquence de l'algorithme	4.40	4.36	4.35	4.66	4.66	4.66
Taux de performance	98.4	99.2	99.4	92.8	92.8	92.8
Utilisation d' AG (%)	46.66	44.44	44.14	39.13	38.20	38.34
Utilisation de RS (%)	36.66	38.88	40.25	23.91	23.59	23.30
Utilisation de RT (%)	16.66	16.66	15.58	36.95	38.20	38.34

Tableau 4.9: Tableau récapitulatif obtenu par utilisation des trois méta-heuristiques

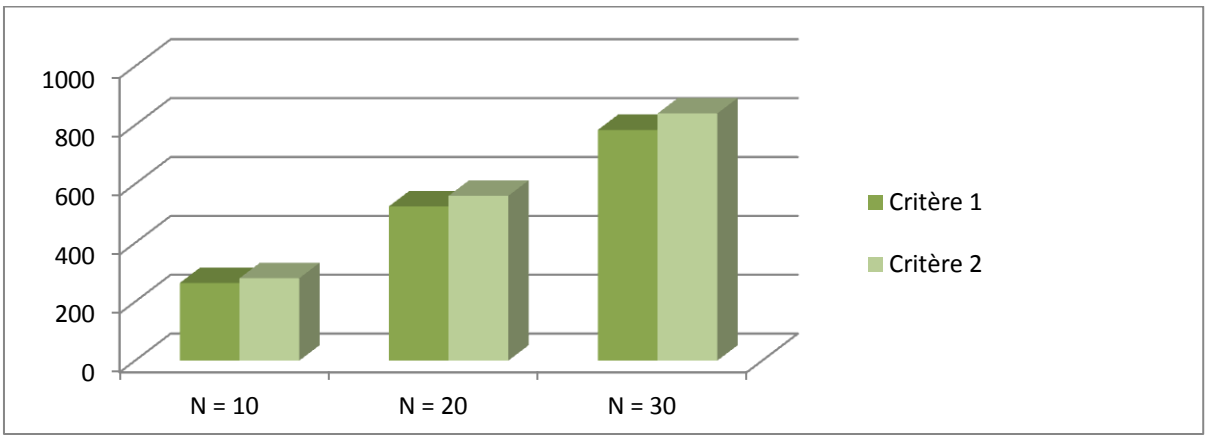


Figure 4.14 : schéma du Makespan obtenu par utilisation des trois méta-heuristiques

Les résultats obtenus du Makespan, de la fréquence de l'algorithme, ainsi du taux de production sont meilleurs en utilisant le premier critère. Par rapport au second, il inclut le temps d'attente dans la fonction d'évaluation. Ainsi les résultats sont beaucoup plus intéressants que ceux trouvés dans les tableaux précédents qui utilisent seulement un seul critère d'optimisation. On peut conclure que l'association et l'exploitation des trois méta-heuristiques dans un même programme d'optimisation, améliore de manière significative les

résultats. A chaque situation, le choix portera sur celle, parmi les trois, qui donne le meilleur résultat.

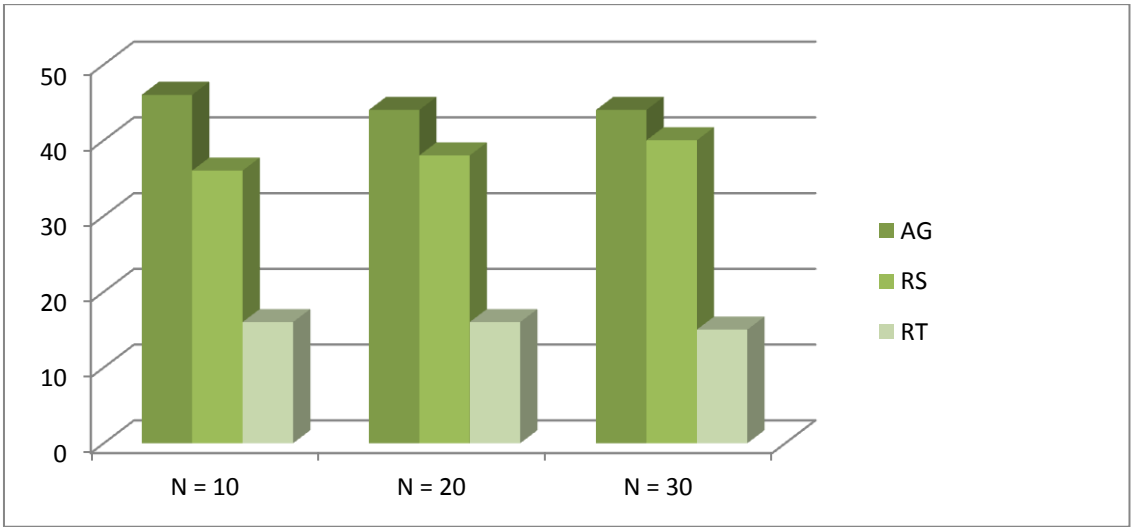


Figure 4.15: Taux d'utilisation de chaque méta-heuristique dans le programme d'optimisation

On constate dans ce graphe, que les algorithmes génétiques sont les plus sollicités pour la résolution du problème d'optimisation.

4.1 Influence des paramètres des algorithmes

- Dans la méthode de l'algorithme génétique, l'application de l'opérateur de croisement est permanente. Par contre, l'opérateur de mutation est affecté d'une probabilité de 0,1, et il est sollicité uniquement dans le cas où les valeurs de la fonction d'évaluation des tâches en conflits sur la même ressource, sont proches.

- Dans la méthode du recuit simulé, les températures supérieures et inférieures du traitement du recuit sont liées aux valeurs maximales et minimales de l'espace de recherche. Il a été appliqué avec une température initiale de $T_0 = F_{imax}$, une température maximale $T_{max} = T_0 = F_{imax}$, et une température minimale $T_{inf} = F_{imin}$, afin d'inclure touts l'espace de recherche de solutions. Le critère de décroissance de température basée sur la loi linéaire $T_{k+1} = T_k \cdot 0.9$ influe sur la rapidité de recherche dans l'espace de solutions.

- La méthode de recherche Tabou dépend essentiellement de la manière de gérer la liste Tabou. Cette dernière ne doit pas être de taille trop longue pour ne pas empêcher l'accès à des zones de recherche plus prometteuse, ni trop courte pour ne pas tomber dans le piège du cycle. Plusieurs test sont effectués sur la taille de la liste Tabou afin d'arriver à celle qui donne le meilleur résultat. La recherche Tabou a été appliquée avec une liste égale au deux tiers du nombre de configurations. Enfin, un critère d'arrêt nous permet d'arrêter la recherche après avoir obtenu la meilleure solution.

Remarque Les résultats montrent que l'insertion des temps morts (temps d'attente de disponibilité des ressources) peut considérablement améliorer la performance de l'ordonnancement en présence des incertitudes, sans trop dégrader la qualité de l'ordonnancement réellement exécuté.

4.2 Comparaison des méta-heuristiques

L'application de chacun des algorithmes RT, RS ou AG au problème d'allocation de tâches soumises à des contraintes de temps et de précédence permet de qualifier une solution de bonne ou de moins bonne, mais ne fournit pas de mécanisme pour déterminer la bonne solution rapidement.

L'algorithme que nous proposons permet de trouver les allocations sur chaque ressource partagée, et de déterminer parmi elles, celle qui réduit le temps de réalisation et le coût de production.

Une comparaison entre les différentes méthodes méta-heuristiques développées a été réalisée sur le problème posé.

La qualité des solutions trouvées par les méta-heuristiques dépend de leur paramétrage. Le choix judicieux des paramètres contribue à la convergence des algorithmes vers un optimum global, ainsi qu'à l'équilibre de la recherche dans l'espace solutions (diversification), et à une exploration locale approfondie (intensification).

Dans notre implémentation, l'ensemble de solutions dans le recuit simulé et la recherche tabou sont exprimées de la même manière que dans l'algorithme génétique, à savoir que les solutions données sont décrites par une matrice où chaque ligne indique l'ordre d'exécution des tâches sur la ressource partagée. La génération de la solution voisine s', est faite par la sélection d'une autre solution (la meilleure trouvée au sens de l'algorithme utilisé) sur la même ressource partagée.

L'algorithme génétique se comporte mieux que le recuit simulé et la recherche tabou, ceci est dû aux opérateurs que nous avons proposés car ils permettent facilement d'identifier la tâche optimale, ainsi la nature de l'algorithme génétique qui prévoie le futur de la tâche, son état et sa future opération à réaliser. L'inconvénient du recuit simulé est qu'il se base beaucoup plus sur l'aspect aléatoire pour la sélection des nouvelles solutions de l'espace de recherche, ainsi que sa température initiale influe sur le temps nécessaire pour trouver une première solution correcte, une température élevée implique un temps d'exécution plus important pour permettre l'algorithme de trouver parmi les allocations celle qui minimise la valeur de la fonction coût. D'autres mesures sont en cours afin d'étudier le comportement de l'algorithme du recuit simulé en fonction de la vitesse de décroissance de température. L'inconvénient de la recherche tabou est qu'il nécessite la sélection de la meilleure solution à chaque étape de recherche, ce qui pénalise la sélection d'autre solutions qui peuvent être efficace pendant la recherche, ainsi la liste tabou constitue le paramètre primordiale pour cette méthode, elle influe également sur la recherche de la solution optimale, ainsi que sur le temps d'exécution

5. Conclusions

Comme pour la plupart des méthodes heuristiques, il n'existe pas de résultats théoriques garantissant la convergence d'une procédure vers un optimum global. Cet état tient à la nature même des méthodes heuristiques. Il y a nécessité d'un effort particulier d'adaptation de la méthode, en intégrant par exemple des connaissances spécifiques au problème posé.

Dans ce chapitre, nous avons adoptés trois méta-heuristiques à l'étape d'ordonnancement. La première est la méthode des algorithmes génétiques qui ont l'avantage d'être appliqués à de nombreux modèles de tâches avec des contraintes diverses. Le codage et les opérateurs de recombinaison étant spécifiques au problème à résoudre, leur élaboration est un facteur déterminant pour une bonne vitesse de convergence. Nous avons présenté une nouvelle variante de l'algorithme génétique. Nous avons proposé un nouveau codage des individus,

spécifique au problème et de nouveaux opérateurs. Ces derniers permettent de manipuler directement l'ordonnancement obtenu sur chaque ressource et réalisent à la fois l'allocation et l'ordonnancement des tâches. Nos expérimentations numériques ont montré l'efficacité de cet algorithme, qui est caractérisé par:

- la façon de coder le chromosome (la représentation d'un individu) qui participe efficacement à la diminution du temps d'exécution d'une part et au respect de l'ordonnancement des opérations de chaque tâche d'autre part,
- le choix de l'opérateur de croisement a permis de générer de meilleurs enfants par la génération d'un individu plus puissant que ces deux parents,
- le choix de l'opérateur de mutation a permis de sortir des minimums locaux et d'obtenir de meilleurs résultats;
- le choix des valeurs de p_m et p_c (avec $p_m = 0.1$ et $p_c = 1$) a fait l'équilibre entre la mutation et le croisement;

La deuxième méthode est celle du recuit simulé. Les résultats montrent que dès les premières itérations, l'algorithme réduit le nombre de tâches non acceptées sur chaque ressource, ce qui permet de trouver des allocations acceptables rapidement. En effet, à la fin de l'exécution de l'algorithme, on obtient, sur les ressources partagées un ensemble de tâches éligibles à l'ordonnancement. La mise en œuvre de l'algorithme a permis d'observer, une convergence rapide qui passe par une réduction du temps global de production.

Enfin, la méthode de recherche Tabou, qui élabore une liste de tâches éligibles, en mémorisant à chaque fois les chemins parcourus pour éviter de les revisiter. Plusieurs tests ont été réalisés avant d'arriver au bon choix de la taille de la liste Tabou utilisée et qui permet une bonne convergence vers la solution optimale. Nous avons aussi proposé un nouveau critère d'arrêt, spécifique au problème qui permet d'arrêter la recherche et de fournir à la fois, l'allocation et l'ordonnancement des tâches.

CONCLUSIONS GÉNÉRALES

CONCLUSIONS GÉNÉRALES

Nous avons présenté dans cette thèse les travaux de recherche sur les méthodes d'optimisation de problèmes. Le but n'est pas de trouver le plus rapidement ou le plus précisément possible un optimum mais d'être capable de suivre l'optimum lors de changement de l'environnement. La plupart des problèmes rencontrés proviennent du fait que les applications des systèmes manufacturiers nécessitent un respect absolu des contraintes de temps et des ressources qui les caractérisent.

Dans cette thèse, nous avons conçu et mis en œuvre, un mécanisme d'ordonnancement de tâches. Souvent les paramètres décrivant les tâches sont connus à l'avance, ce qui implique un certain déterminisme dans leur exécution. Le mécanisme d'ordonnancement que nous avons présenté traduit ce déterminisme par des données appropriées au modèle, qui sont capables de résoudre les problèmes d'ordonnancement aussi bien quand le modèle de tâches est statique ou dynamique.
Les formalismes proposés pour représenter les systèmes dynamiques hybrides sont nombreux. Ils se différencient par la technique de modélisation adoptée qui repose soit sur une approche visant à étendre un modèle discret ou un modèle continu, soit sur une approche visant à utiliser un modèle mixte. Les réseaux de Pétri sont connus pour leur capacité à la représentation des caractéristiques et des interactions entre les différents composants d'un atelier de production manufacturier. Ils permettent de décrire les contraintes de précédence, les choix, les mécanismes de synchronisation et de parallélisme. En termes de modélisation, les outils réseaux de Pétri sont utilisés pour décrire et résoudre plusieurs problèmes réputés « difficiles », en particulier les procédés à contraintes de temps où on doit spécifier le temps d'une opération. La qualité, et même la conformité du produit dépendent directement du respect de ces contraintes.
L'allocation des tâches aux ressources doit être prise en compte dans la phase d'allocation. En effet, une allocation correcte est définie par l'obtention d'ensembles de tâches ordonnançables sur l'ensemble de ressources. Divers algorithmes d'ordonnancement prometteurs, ont été proposés et mis en œuvre en-ligne pour ordonnancer les tâches, un module de décision applique une heuristique pour l'allocation des tâches. Des résultats préliminaires sont présentés.

En bref, l'objectif de ces travaux était de proposer une méthodologie de résolution des problèmes d'optimisation dans les ateliers de production flexible. La transformation du cahier des charges en problème d'optimisation est une tâche critique du concepteur notamment en ce qui concerne le choix de la fonction objectif et des contraintes. Les développements se sont concentrés sur la mise en point de stratégies d'optimisation basées sur l'utilisation des méthodes stochastiques. Ces algorithmes utilisent une connaissance de l'état d'information sur la ressource partagée et ses tâches que doivent être réalisées. L'algorithme fournit des temps satisfaisants même dans les cas de présence d'un nombre important de tâches à ordonnancer. La ressource dans ce cas, aura davantage de temps libre pour l'acceptation et l'exécution des tâches.

Des méthodes différentes, d'optimisation ont été présentées. Les trois méthodes stochastiques les plus prometteuses : algorithmes génétiques, recuit simulé et la recherche taboue ont été implantées et testées. Elles se fondent sur l'aléatoire. Ceci leur permet, d'une part d'explorer le domaine d'optimisation de façon plus robuste que les méthodes déterministes mais, d'autre part, le temps de calcul est élevé.

Pour résoudre ce problème quelques contributions originales ont été apportées. Un nouveau critère de reproduction pour les algorithmes génétiques, a été proposé. L'avantage, est qu'il trouve son chemin vers le bon accouplement des parents. Il produit ainsi les meilleurs enfants à partir d'une population de solution construite sur une approximation de la fonction objectif ayant des propriétés particulières, et il permet de localiser plus rapidement l'optimum.
Pour le recuit simulé, deux améliorations ont été proposées. La première consiste en la détermination de l'espace de recherche, qui rend l'algorithme moins sensible pour les zones moins prometteuse pour le calcul de la solution optimale. La deuxième amélioration se base sur la recherche de la solution voisine. Elle s'appuie sur la notion de calcul de la distance par rapport à la solution courante de l'espace de recherche.

Une étude approfondie sur la méthode de recherche taboue a été réalisée. L'analyse de la méthode de Hu nous a permis d'intégrer des améliorations qui rendent la méthode appliquée plus efficace et prometteuse. Elle fournit une solution avec la précision souhaitée tout en économisant du temps de calcul. Elle utilise efficacement l'historique de la recherche et évite définitivement de revenir à des solutions déjà visitées. En effet, autour de chaque point précédemment visité, une zone taboue est définie en utilisant les deux améliorations apportées. La première concerne l'étude de la taille de la liste taboue, qui sera flexible selon la taille de l'espace de recherche. La deuxième est liée au pas de recherche dans l'espace de solutions, qui rend notre recherche plus efficace et permet de visiter l'espace de recherche et leurs zones les plus prometteuses.
Une comparaison entre l'algorithme génétique proposé et l'algorithme de F.Pezzella a été appliquée, afin de montrer l'efficacité des paramètres introduits. Les résultats obtenus sont meilleurs et nécessitent moins de temps.

Une application de chacune des méthodes stochastiques est réalisée sur notre problème d'optimisation. Une comparaison entre ces différentes méthodes développées, a été réalisée sur les deux formules de la fonction objectif (F_{1i}, et F_{2i}). Les résultats obtenus ont montré que la méthode de l'algorithme génétique est très performante et donne de meilleures solutions pour les deux formules de la fonction objectif par rapport à ceux trouvés par le recuit simulé et la recherche taboue. Les résultats obtenus pour la première formule qui contient des informations sur le temps mort, sont plus performants que ceux trouvés pour la deuxième formule. Ces résultats sont meilleurs si on se réfère au nombre d'évaluations pour atteindre l'optimum tout en respectant les contraintes du problème (lorsque le nombre d'évaluations est grand est que l'optimum est déjà atteint en risque de faire dérouler le programme et de perdre le temps sans apporter des améliorations au résultat trouvé) où lorsque la solution initiale est proche de l'optimum global (l'optimum sera vite trouvé).

Une recherche parallèle est employée comme une nouvelle voie très prometteuse et sur laquelle nous avons effectué plusieurs travaux. Le module de décision contient les trois méta-heuristiques qui seront lancées en parallèle dans la recherche de la solution optimale. L'obtention d'un ensemble de solutions optimales au lieu d'une solution unique permet au concepteur d'affiner ces critères afin d'opérer un choix. Une fois le résultat obtenu pour chaque méthode, ces derniers seront comparés et la meilleure solution sera prise en considération pour l'allocation de la ressource partagée.

Ces travaux nous ont donc, permis de mettre en évidence, d'une part, que les méthodes méta-heuristiques ne sont pas concurrentes mais complémentaires, d'autre part, qu'il y a une nécessaire adéquation entre les propriétés d'un modèle et celle de la méthode d'optimisation employée pour sa résolution.

Au terme de ces travaux, très concrètement, de nombreuses perspectives peuvent être envisagées à la suite des travaux présentés dans ce document, et qui peuvent être misent en œuvre. Il faut poursuivre des études expérimentales sur les propriétés des méta-heuristiques. Une exploitation des complémentarités entre méthodes stochastiques et méthodes déterministes devrait conduire à des implémentations efficaces. Des améliorations apportées aux algorithmes génétiques et au recuit simulé ainsi que la méthode taboue devrait facilement être transposables et bénéfiques au domaine de l'ingénierie. Bien que ces travaux se soient intéressés à l'application de trois méthodes méta-heuristiques à l'optimisation d'un atelier de production flexible, des développements ultérieurs pourraient s'intéresser à l'application d'autres méthodes méta-heuristiques en émergence.

RÉFÉRENCE BIBLIOGRAPHIQUES

RÉFÉRENCE BIBLIOGRAPHIQUE

[1] E. Aarts, J. Korst « Similates Annealing and Boltzmann Machines » John Wiley & Sons, New York, 1990

[2] Abbou R., Simeu-Abazi Z., Di Mascolo M. (2003). Les réseaux de Petri pour la modélisation et l'analyse des performances d'un atelier de maintenance, Modélisation et simulation (MOSIM 2003), 23-25 avril, Toulouse (France).

[3] Adamou M. - *Contribution à la modélisation en vue de la conduite des systèmes flexibles d'assemblage à l'aide des réseaux de Petri orientés objet*, Thèse de doctorat, Université de Franche-Comté, 1997.

[4] Aimed MOKHTARI, Diagnostic méthode associant des systèmes la détection hybrides: par développement classification d'une et la simulation dynamique, THÈSE de Doctorat Préparée au Laboratoire d'Analyse et d'Architecture des Systèmes du CNRS, Soutenue le 23 octobre 2007

[5] H. Alla, P. Ladet, J. Martinez, M. Silva, « Modeling and validation of complex systems by coloured Petri nets : Application to flexible Manufacturing Systems », Advances in Petri nets. Ed Spring Verlag, 1984.

[6] H. Alla, J.M. Flaux, « Modélisation d'une unité de stockage de gaz par réseaux de Petri hybrides ». ADPM'98 , Reims, mars 1998.

[7] Alla, H. and David, R. 1998. Continuous and Hybrid Petri Nets. *Journal of Circuits, Systems & Computers* 8(1): 159–188.

[8] Alla, H., Cavaille, J.-B., Le Bail, J., Bel, G., Les systèmes de production par lot : une approche discret –continu utilisant les réseaux de Petri hybrides, Symposium ADPM'92, Paris, Janvier, 1992.

[9] Anderson J.R. (1989) "Simulation : méthodology and application" *Agricultural economics,*
pp.3-54.

[10] Anton Mitsiouk, Contribution à l'Optimisation des Systèmes Dynamiques : Application au Génie des Procédés. Thèse de doctorat de l'institut national polytechnique de Toulouse, Soutenue le 12 Juillet 2007

[11] Audry N., Ghabri M.-K., Dernongodin L, Prunet F., Ladet P. (1994). *Modelling and control of high throughput production lines*, IFAC Workshop on New Trends in Design of Control Systems,p.392-397, 7-10 Septembre, Smolenice (Slovakia)

[12] Th, Back « Optimal mutation rates in genetic search » S. Forrst (Ed), Proceeding of the 5th International Conference on genetic Algorithms and their Applications, Morgan Kauffman Publishers, pp.2-9, 1993

[13] R.S. BARR, R.L. GOLDEN, J.P. KELLY, M.G.C. RESENDE, W.R. STEWART,
Jr., Designing and reporting on computational experiments with heuristic methods.
Journal of Heuristics 1(1) : 9-32, 1995.

[14] M. Basseur, conception d'algorithmes coopératifs pour l'optimisation Multiobjectif : application aux problèmes d'ordonnancement de type flow-shop. Memoire de thèse. USTL 21/06/2005

[15] J- C. Billaut, A. Mokrim, & E. Sanlaville. Introduction à la flexibilité et à la robustesse en ordonnancement. Dans Flexibilité et Robustesse en Ordonnancement sous la direction de J-C. Billaut, A. Mokrim, et E. Sanlaville, pages 13-32. Hérmes 2005.

[16] C. K. Bounsaythip, « Alorithmes heuristiques et évolutionnaires : application à la résolution du problème de placement de formes irrégulières », Thèse de doctorat en Productique, université des sciences et technologies de Lille, 9 octobre 1998

[17] G.W. BRAMS, *"Réseaux de Petri : Théorie et Pratique" Tomes I et II*, MASSON, Paris, 1982.

[18] BRINKMAN P. L., BLAAUBOER W. A., «Timed cContinuous Petri Nets: A Tool pour Analysis and Simulation of Discrete Event Systems », European Simulation Symposium, Gand. Belgique, Novembre 1990.
[19] CEI-IEC, Preparation of function charts for control systems, Norme Internationale CEI 848,1988.
[20] Collart-Dutilleul Simon, Les Réseaux de Petri P-temporels : Modélisation et validation d'exigences temporelles. N° H642 Université des Sciences et Technologies de Lille. Mémoire d'habilitation, Soutenu le : 28/11/08
[21] A. Corana « Minimising multimodal Functions of Continuous Variables with the Simulated Annealing Algorithm » ACM Trans. Math. Soft., Vol.13, N°.3, pp.262- 280, Sept. 1987
[22] Dagli C. H., Sittisathanchai S. « Genetic neuro-scheduler : A new approach for jobshop scheduling" 1995, International Journal of Production Economics
[23] David, H. Alla, « Continous Petri nets ». Proceeding of the european workshop on application and theory of Petri nets. Universidad de Zaragona, p 275_294, june 1987
[24] René David, Hassane Alla, « Du grafcet au réseaux de Petri ». Hermès, 1992 (2ème édition revue et augmentée).
[25] David, R., Alla, H., *On Hybrid Petri nets,* Discrete Event Dynamic Systems: Theory and Applications, No.11, p.9 – 40, 2001.
[26] K. A. De Jong, W. M. Spears « A formel analysis of the role of multi-point crossover in genetic algorithms » Annuals of Mathematics and Artificial Intelligence Journal, Vol. 5, N°.1, pp.1-26, 1992
[27] Demongodin, I., Les Réseaux de Petri lots : Modélisation des systèmes de production à haute cadence en régime transitoire, Thèse, Université de Montpellier II / LIRM, janvier, 1994.
[28] W. B. Dolan, P. T. Cummings, M. D. Le Van « Algorithmic efficiency of simulated annealing for heat exchanger network design » Comp. Chem. Engng. Vol.14, N°.20, pp.1039-1050, 1990
[29] J. Dréo, A. Petrowski, P. Siarry, E. Taillard. Métaheuristique pour l'optimisation difficile. Edition Eyrolles pages 69-112, 2003.
[30] Drogoul. A. *De la simulation multi agents à la résolution collective de problèmes,* thèse de l'université paris6, 1993
[31] J. Erschler & G. De Terssac. Flexibilité et rôle de l'opérateur humain dans l'automatisation intégrée de production. Rapport laas n°88137, Laboratoire d'Analyse et d'Architecture des Systèmes, Toulouse, France, 1988.
[32] Florin G., Natkin S. (1985). Les Réseaux de Petri Stochastiques, *Technique et Science Informatiques,*VolA, N°l, p143-160.
[33] Fonseca C., Fleming P., Genetic algorithms for multiobjective optimization: formulation, discussion and generalisation. Proc. Of the 5th Int. Conf. on genetics algorithms, 416-423.
[34] Grand dictionnaire terminologique. Office québécois de la langue française, disponible : http://granddictionnaire. Com/btml/fra/r_motclef/index1024_1.asp. accédé le 17/01/04
[35] George DRAGHICI, Nicolae BRINZEI, Ioana FILIPAS, La modélisation et la simulation en vue de la conduite des systèmes de production, UNIVERSITATEA POLITEHNICA DIN TIMISOARA, Les Cahiers des Enseignements Francophones en Roumanie, 1998
[36] Glover, F. *Future paths for integer programming and links to artificial intelligence*. Computers and Operations Research, tome 13, pages 533-549, 1986.
[37] F. Glover and M. Laguna, Tabu Search, Kluwer Academic Publishers, 1997.

[38] D. E. Goldberg « Genetic Algorithms in search, Optimization and Machine Learning » Addison-Wesley, 1989
[39] J.-k. Hao, P. Galinier, M. Habib. Métaheuristiques pour l'optimisation combinatoire et l'affectation sous contraintes. Revue d'intelligence artificielle Vol : N° 1999 http://www.info.univ-angers.fr/pub/hao/papers/RIA.pdf
[40] R. Heckman, Th. Lengauer « A Simulated Annealing Approach to the Nesting Problem in the Textile Manufacturing Industry » In R. E. Burkad, P. L. Hammer, T. Ibakari and M. Queyranne, editors, Annals of Operations Research, Vol.57, pp.103-133, J. C. Baltzar AG science Publishers, Amesterdam, 1995
[41] Hedi Dhouibi, Utilisation des réseaux de pétri a intervalles pour la régulation d'une qualité : application a une manufacture de TABAC, THESE de Doctorat, L.A.G.I.S, UMR 8146- Ecole Centrale de Lille, soutenue le 16 décembre 2005
[42] J. H. Holland « Adaptation in naturel and artificial systems » University of Michigan Press, Ann Arbor, 1975 (ou MIT Press, Cambridge,Mass., 1975.)
[43] J. Holland, "adaptation in natural and artificial systems", Second Edition, MIT Press 1992.
[44] Houda Bleibel, Modélisation Multi-Agents Du Réseau, Universités Francophones AUPELF-UREF, diplôme d'Etudes Approfondies Réseaux de télécommunications, soutenance le 22 décembre 2000
[45] N. Hu « Tabu search Method with random moves for glabally optimal » International Journal for Numerical Methods in Engineering, Vol.35, N°. 5, pp.1055-1070, 1992
[46] Huber P. J., « Robust Statistics », Jhon Wiley et Sons 1981.
[47] Jacquet Lagrèze E. (1989) *Techniques quantitatives de gestion et informatisation.* AFCET/Interfaces, Vol. 84, pp 3-13.
[48] S. Julia. "Conception et Pilotage de Cellules Flexibles à Fonctionnement Répétitif Modélisées par Réseaux de Petri". Thèse de Doctorat, Université Paul Sabatier, Toulouse, juillet 1997.
[49] Karim LABADI, Haoxun CHEN, Lionel AMODEO, Nouvelles Propriétés Comportementales pour les Réseaux de Petri Lots Déterministes et Stochastiques Université de Technologie de Troyes Institut des Sciences et Technologies de l'Information de Troyes (ISTIT) Equipe Optimisation des Systèmes Industriels 12 rue Marie Curie, BP 2060, 10010 Cedex Troyes France
[50] S. Kirkptrick, C. D. Gelatt, M. P. Veccbi « Optimisation by Simulated Annealing » Science, Vol.220, N°.4598, pp.671-680, 1983
[51] P. Lancome, méthodes exactes et approchées pour l'optimisation des systèmes à moyens de transport. Mémoire pour l'obtention de l'Habilitation à Diriger des Recherche de l'Université Blaise Pascal (Clermont – Ferrand II). 06/07/2005 : N° : 194
[52] Le Bail J., Alla H., David R. (1991). *Hybrid Petri Nets,* European Control Conference, p.1472-1477,Grenoble (France).
[53] LE BAIL J., ALLA H., DAVID R., «Hybrid Petri Nets», European Control Conference, Gronoble, p. 141472-1477, Juillet 1991.
[54] S. A. Melnick. P. L. Carter. « Production Activity Control ». The business ONE. Irwib/APICS Series in Production Management. Richard D. Irwin Inc., 1987.
[55] F. Messine, B. Nogarède, J. L. Langounaelle « Optimal Design of Electro-Mechanical Actuators : A New Method Based on Global Optimization » IEEE Tans. Magn. Vol. 34, N°.1, pp.299-308, 1998.
[56] W. METROPOLIS, A. ROENBLUTH, M. ROSENBLUTH, A. TELLER, E.TELLER, Equation of the state calculations by fast computing machines. *Journal of Chemical Physics* 21 : 1087-1092, 1953.

[57] Z. Michalewicz « Genetic Algorithms + Data Structures = Evolution Programs » Springer-Verlag, 1994
[58] R. Milne, « Strategies for diagnosis ». IEEE Transactions on Systems, Man and Cybernetics, Vol.17, N°3, p 333-339, 1987.
[59] Michel Minoux. Programmation Mathématique : Théorie et Algorithmes, Dunod, vol. 1, Paris, 1983.
[60] MOODY J. et ANTSAKLIS P., Supervisory control of discrete event systems using Petri nets, Kluwer, Boston (USA) : 186 p., 1998.
[61] J. A. Nelder and R. Mead. A simplex Method for Function Minimization, Comput . , vol. 7, p. 308-313, 1965
[62] Omessaad HAJJI, Contribution au développement de méthodes d'optimisation stochastiques. Application a la conception des dispositifs électrotechniques. N° d'ordre : 3369 Ecole centrale de Lille, Thèse de Doctorat, Soutenue le 03/12/2003
[63] L. Painton, U. M. Diwekar « Synthesizing Optimal Design Configurations for a Brayton Cycle Power Plant » Comput. Chem. Eng., Vol.18, pp.369-381, 1994
[64] PETRI C.A., Kommunikation mit Automaten, Ph.D., Institut für Instrumentelle Mathematik, University of Bonn (Allemagne) : 1962.
[65] F. Pezzella, G. Morganti, G. Ciaschetti, *A genetic algorithm for the flexible job-shop scheduling problem*, Science Direct, Computers & Operations Research, Vol. 35, 2008, p.3202-3212.
[66] Pierre CASTAGNA, Contribution à la modélisation, la simulation et la commande de systèmes de production et de transitique,Université de Nantes, UMR CNRS 6597
[67] Philippe VELLEMANS, Contribution à la conception d'un outil adapté à l'Opérateur Humain pour la reconstruction d'une traçabilité « approchée », Thèse de Doctorat De l'université de Reims Champagne-Ardenne, Soutenue le 14 décembre 2006
[68] M. J. D. Powell, « An efficient method for finding the minimum of a function of several variables without calculating derivation » Computer Journal, Vol. 7, pp. 155-162, 1965
[69] Y. Rahmat-Sami, E. Michielssen « Electromagnetic Optimisation by Genetic Algorithms » John Wiley & Sons, 1999
[70] M. P. Rakotoson « Synthèse des caractéristiques et techniques de développement de la commande des systèmes discontinus : application aux systèmes de production flexibles ». Thèse de Doctorat. Université des sciences et Techniques de Lille, juillet 1993.
[71] C. Ramchandani « Analysis of Asynchronous Concurrent Systems by Timed Petri Nets », Technical Report n°120, Laboratory for Computer Science, MIT Cambridge, MA, 1974
[72] René David, Hassane Alla, Discret Event Dynamic Systems : Theory and Application, 11, 9-40, 2001, 2001 Kluwer Academic Publishers, Boston
[73] Rodde G. - *Les systèmes de production*, Ed. Hermès, Paris, 1989.
[74] Roy B. « Un chaînon manquant en Recherche Opérationnelle et Aide à la Décision : les conclusions robustes ». Cahier du Laboratoire d'Analyse et de Modélisation de Systèmes pour l'Aide à la décision, N°144, Université de Paris Dauphine 1997.
[75] P. Siarry, Y. Collette. Optimisation multiobjectif. Eyrolles ISBN: 2-212-11168-1.2002
[76] J. Sifakis. « Use of Petri Nets for Performance Evaluation ». In : Beilner, H., Gelenbe, E. : Modelling and Performance Evaluation of Computer Systems, Measuring, Modelling and Evauating Computer Systems, pp. 75-93. Amsterdam : North Holland, 1977.
[77] Robert Valette, « Les réseaux de Petri ». Support de cours, France 1992.
[78] R. Valette, R. Champagnat, D. Andreu, H. Pingaud, « Modélisation des systèmes de production hybrides » AGIS'97, colloque de recherche doctorale, Angers, France, 1997.

[79] D. Vanderbilt, S. G. Louie « A Monte Carlo Simulated Annealing Approach ti Optimization over Continuous Variables », Journal Of Computational, Physics, 56, pp.259-271, 1984
[80] J. Zaytoon, «Systèmes dynamiques hybrides», ouvrage collectif sous la direction de, collection Hermès Science, ISBN 2-7462-0247-6, Paris 2001.
[81] Christian Ziegler : « Sûreté de fonctionnement d'architectures informatiques embarquées sur automobile ». Thèse de Doctorat, LAAS, 12 juillet 1996.

ANNEXES

Le réseau de pétri lot

Introduction [17]

Les réseaux de Petri lots déterministes et stochastiques (BDSPN) dédié spécialement pour les SED comportant des phénomènes lots notamment, les chaînes logistiques, les systèmes de stockage et les systèmes de production par lots. en lui intégrant de nouveaux composants lots: places lots, jetons lots et transitions lots pouvant représenter explicitement les flux lots (jetons lots) et les opérations lots (transitions lots) qui caractérisent les processus lots.

L'aspect lot a été déjà intégré dans les réseaux de Petri hybrides pour la modélisation des phénomènes d'accumulation dans les systèmes de production. Notons qu'il n'y a aucune similitude dans le fonctionnement et dans le contexte d'application avec les BDSPNs.

2.1 Définitions [17]

Définition 1 [17]

Le réseau de Petri lot déterministe et stochastique noté BDSPN, est un 9-uplets: BDSPN = (P, T, I, O, V, W, Π, D, μ_0) composé de deux types de places et de trois types de transitions tels que :

$P = P_d \cup P_b$ est l'ensemble des places composé de l'ensemble des places discrètes P_d et de l'ensemble des places lots P_b.

$T = T_i \cup T_d \cup T_e$ est l'ensemble des transitions composé de l'ensemble des transitions immédiates T_i, de l'ensemble des transitions déterministes T_d, et de l'ensemble des transitions stochastiques T_e (avec un temps de franchissement exponentiel).

$I \subseteq (P \times T)$ est l'ensemble des arcs d'entrée des transitions, $O \subseteq (T \times P)$ est l'ensemble des arcs de sortie des transitions, $V \subseteq (T \times P)$ est l'ensemble des arcs inhibiteurs du réseau. ($V \cap I = \emptyset$).

W est une fonction définissant le poids des arcs pouvant être constants ou variables en fonction du marquage M du réseau.

$\Pi : T \rightarrow IN$ est une fonction de priorité entre les transitions telle que $\Pi(t) = 0$ si $t \in T_d \cup T_e$ et $\Pi(t) \geq 0$ si $t \in T_i$.

$D : (T_i \times T_d \times T_e) \times IN^{|P|} \rightarrow (0,\infty)$ est une fonction qui définit le temps de franchissement des transitions telle que : $D(t) = 0$ si $t \in T_i$, $D(t) \in IR^+$ si $t \in T_d$, et D(t) est généré aléatoirement suivant une loi exponentielle si $t \in T_e$.

μ_0 est le marquage initial du réseau tel que : $\mu_0 : P \rightarrow IN \cup 2^{IN}$, $\mu_0 \in IN$, si $p \in P_d$, $\mu_0 \in 2^{IN}$, si $p \in P_b$.

Définition 2 [17]

Le μ- marquage d'une place discrète est un nombre entier (positif ou nul). Le μ- marquage d'une place lot est un ensemble de nombres entiers positifs non nuls.

Définition 3 [17]

Le M- marquage d'une place lot est la somme de l'ensemble des éléments de son μ-marquage. Le M- marquage d'une place discrète est égal à son μ-marquage.

Définition 4 [17]

Une transition t est dite une transition discrète si et seulement si elle ne présente aucune place lot dans l'ensemble de ses places d'entrée $^\bullet t$.

On réseau tel que : note par T_D l'ensemble des transitions discrètes du réseau tel que: $T_D = \{t \in T / {}^\bullet t \cap P_b = \emptyset\}$

Une transition t est dite une transition lot si et seulement si elle a au moins une place lot à son entrée. On note par T_B l'ensemble des transitions lots du réseau tel que:

$T_B = \{t \in T / {}^\bullet t \cap P_b \neq \emptyset\}$

2.2 Règles de Fonctionnement [17]

Deux types de franchissement pour un BDSPN: franchissement lot et franchissement discret dont les règles de validation dépendent du type de la transition à franchir. Elles permettent de synchroniser le flux des jetons lots et des jetons discrets.

- **Cas 1**

Franchissement discret ($t \in T_D$)

Une transition discrète t est validée pour un µ-marquage µ (M son M-marquage correspondant) si et seulement si :

$\forall p \in {}^\bullet t, \quad M(p) \geq W(p, t)$ (1)

$\forall p \in {}^o t, \quad M(p) < W(p, t)$ (2)

Le franchissement de la transition t fait évoluer le µ-marquage µ à µ' tel que :

$\forall p \in {}^\bullet t : \mu'(p) = \mu(p) - W(p, t)$ (3)

$\forall p \in t^\bullet \cap P_d : \mu'(p) = \mu(p) + W(t, p)$ (4)

$\forall p \in t^\bullet \cap P_b : \mu'(p) = \mu(p) + \{W(t, p)\}$ (5)

Dans ce cas, les règles de validation d'une transition discrète t sont similaires à celles des réseaux de Petri ordinaires. Le franchissement discret d'une transition discrète consiste à :

- ♦ Retirer W (p, t) jetons discrets dans toute place (discrète) p d'entrée de t.
- ♦ Ajouter W(t, p) jetons discrets dans toute place discrète de sortie de t.
- ♦ Ajouter un jeton lot de taille W(t,p) dans toute place lot de sortie de t.

- **Cas 2**

Franchissement lot ($t \in T_b$)

Une transition lot t est validée pour un µ-marquage µ (M son M-marquage correspondant) si et seulement si :

il existe un indice de franchissement lot $q \in IN^*$ tel que :

$\forall p \in {}^\bullet t \cap P_b, \exists b \in \mu(p) : q = b / W(p, t)$ (6)

$\forall p \in {}^\bullet t \cap P_d, \quad M(p) \geq q \times W(p, t)$ (7)

$\forall p \in {}^o t, \quad M(p) < W(p, t)$ (8)

Le franchissement de la transition t avec l'indice q fait évoluer le µ-marquage µ à µ' tel que:

$\forall p \in {}^\bullet t \cap P_d, \quad \mu'(p) = \mu(p) - q \times W(p, t)$ (9)

$\forall p \in {}^\bullet t \cap P_b, \quad \mu'(p) = \mu(p) - \{q \times W(p, t)\}$ (10)

$\forall p \in t^\bullet \cap P_d, \quad \mu'(p) = \mu(p) + q \times W(t, p)$ (11)

$\forall p \in t^\bullet \cap P_b, \quad \mu'(p) = \mu(p) + \{q \times W(t, p)\}$ (12)

Les conditions de validation (7) et (8) sont analogues aux règles de q-validation d'une transition donnée dans un réseau de Petri standard. Dans ces réseaux de Petri, une transition q validée est franchie q fois simultanément. Dans les BDSPN, l'indice q, qu'on appelle indice de franchissement lot est déterminé par la taille du jeton lot qui franchira la transition lot t dans chaque place lot d'entrée de t (6). Le franchissement lot d'une transition lot consiste à :

- ♦ Retirer q× W (p, t) jetons discrets dans toute place discrète p d'entrée de t.
- ♦ Retirer un jeton lot b correspondant à l'indice de franchissement q dans toute place lot p de t.
- ♦ Ajouter q×W(t, p) jetons discrets dans toute place discrète p de sortie de t.
- ♦ Ajouter un jeton lot de taille q× W(t, p) dans toute place lot p de sortie de t.

Notons que le franchissement discret (cas 1) est un cas particulier du franchissement lot (cas 2) en considérant q = 1.
Le modèle BDSPN a permis de modéliser le flux lot et les opérations lots correctement alors que le modèle standard nous donne un comportement qui ne correspond pas au système réel.
La politique de franchissement des transitions d'un BDSPN est similaire à celle d'un réseau de Petri stochastique ordinaire.
Le temps d'exécution d'une opération lot peut dépendre de la taille du lot correspondant. Par exemple, le temps nécessaire pour usiner un lot de b pièces sur une machine (représenté comme un jeton lot de taille b) dépend du nombre de pièces du lot en question. En général, ce temps d'exécution est proportionnel à la taille du lot. En d'autres termes, si le temps nécessaire pour usiner une pièce est de d_k, alors le temps nécessaire pour usiner un lot de b pièces est égal à $b \times d_k$.
L'équation d'état qui gouverne simultanément le comportement dynamique du flux des jetons discrets et des jetons lots dans un BDSPN.

Cas 1
Equation de Franchissement d'une seule transition.
Le franchissement d'une transition $t_j \in T$ validée pour un µ-maquage μ_k conduit le BDSPN à un nouveau µ- marquage μ_{k+1} qui peut être obtenu à l'aide de l'équation de changement d'état suivante :

$$\mu_{k+1} = \mu_k + W \times F$$

$$F[i]_{|T|\times 1} = \begin{cases} 0 & if\ i \neq j \\ q & if\ i = j \end{cases} \qquad i = 1\ to\ |T|$$

Avec:
- W est matrice d'incidence du BDSPN similaire à un réseau de Petri standard.
- F est le vecteur caractéristique de franchissement de la transition t de dimension $|T| \times 1$.
- q est l'indice de franchissement de la transition t_j à partir du µ-marquage µ tel que: $^\bullet t_j$

$$q = \begin{cases} 1 & si\ \bullet tj \cap Pd \neq \emptyset \\ \frac{b}{w_{(p,tj)}},\ b \in \mu(p)\ si\ \bullet tj \cap Pb \neq \emptyset \end{cases} \qquad (15)$$

Cas 2 Equation d'un Franchissement multiple (séquence de transition)
Soit S une séquence de franchissement faisable (ensemble de transitions à franchir successivement) d'un M-marquage M_0 à un nouveau M-marquage M_d. Le franchissement de la séquence S est formalisé comme suit :

$$M_d = M_0 + W \times F \qquad (16)$$

$$F[j]_{|T|\times 1} = \begin{cases} Nombre\ de\ franchissement\ de\ tj \in S, & si\ tj \in TD \\ \Sigma q\ de\ tj \in S, & si\ tj \in TB \end{cases}$$

Contrairement à l'équation caractérisant le changement d'état par le franchissement d'une seule transition, l'équation (16) est en fonction du M-marquage du BDSPN et pas de son µ-marquage. Ceci est dû au fait que franchir un ensemble de transitions au même temps ne nous permet pas de garder l'information sur la taille des jetons lots issus de chaque franchissement d'une transition de la séquence S. Cette équation n'est pas aussi applicable dans le cas où on a un ou plusieurs poids des arcs variables, car la matrice W est constante dans l'équation.