Hybridation des métaheuristiques pour la résolution de problème d'ordonnancement multi-objectif dans un atelier flow-shop

JEBARI Hakim*, RAHALI EL AZZOUZI Saida, SAMADI Hassan
Laboratoire des technologies de l'information et de la communication (LABTIC)
Ecole nationale des sciences appliquées de Tanger (ENSAT)
Tanger, Maroc
hjebari1@yahoo.fr, rahali_elazzouzi@yahoo.fr, ha.samadi@gmail.com

***Résumé*— Cet article traite la résolution des problèmes d'ordonnancement multi-objectifs dans un atelier complexe de type flow-shop par la construction des métaheuristiques hybridées séquentiellement et des métaheuristiques hybridées parallèlement synchrone basée sur la pondération de fonction objective : l'algorithme génétique avec la recherche tabou, l'algorithme génétique avec le recuit simulé et l'algorithme génétique avec l'algorithme génétique. Ensuite une étude comparative est menée sur les résultats de chaque type d'hybridation d'une part, d'autre part entre les résultats de ces deux types d'hybridation, afin d'identifier la méthode et le type d'hybridation qui fournit la meilleure solution pour l'ordonnancement multi-objectif.**

***Mots clés*— Ordonnancement ; métaheuristiques; algorithmes génétiques; recherche tabou; recuit simulé; hybridation; flow-shop; multi-objectifs.**

I. INTRODUCTION

Dans les dernières années, le développement rapide de la technologie de l'information avec l'intensification de la tendance de la mondialisation économique, fait que les systèmes de production doivent faire face de plus en plus à la compétition au niveau du délai d'achèvement de nouveaux produits.

Les décisions des entreprises ont un impact significatif sur la fabrication, le coût du produit, le délai de livraison, la qualité… etc., pour les améliorés, elles doivent optimiser la planification et l'ordonnancement de la production, la tâche la plus difficiles. D'une manière générale, on distingue plusieurs classes d'objectifs concernant un ordonnancement [1] :

- Les objectifs liés au temps : minimisation du temps total d'exécution, du temps moyen d'achèvement, des durées totales de réglage ou des retards par rapport aux dates de livraison.
- Les objectifs liés aux ressources : maximiser la charge d'une ressource ou minimiser le nombre de ressources nécessaires pour réaliser un ensemble de tâches.
- Les objectifs liés au coût : minimiser les coûts de lancement, de production, de stockage, de transport, etc.

La satisfaction de tous les critères à la fois est souvent délicate, car elle conduit souvent à des situations contradictoires et à la recherche de solutions à des problèmes complexes d'optimisation [2].

Dans cet article, les méthodes de résolution des problèmes d'ordonnancement sont basées sur l'hybridation séquentielle des Métaheuristiques :

- Algorithme génétique avec la recherche tabou.
- Algorithme génétique avec le recuit simule.
- Algorithme génétique avec l'algorithme génétique.

Une méthode hybride est une méthode de recherche constituée d'au moins de deux méthodes de recherche distinctes. Elle consiste à exploiter les avantages respectifs de plusieurs méthodes en combinant leurs algorithmes suivant une approche synergétique [3].

Une méthode hybride peut être mauvaise ou bonne selon le choix et les rôles de ses composants. Pour définir une méthode hybride efficace, il faut savoir caractériser les avantages et les limites de chaque méthode.

Les origines des algorithmes hybrides remontent probablement aux travaux décrits dans [4] [5].

Actuellement, les approches hybrides gagnent en popularité car ce type d'algorithme produit généralement les meilleurs résultats pour plusieurs problèmes d'optimisation combinatoire [6]. En effet, selon [7], les approches hybrides ont permis d'obtenir de bons résultats dans une grande variété de problèmes théoriques d'optimisation combinatoire.

II. MÉTHODES DE RESOLUTION

A. Les algorithmes hybrides

Le développement et l'application des méta-heuristiques hybrides est de plus en plus susciter l'intérêt académique, ces les méthodes hybrides combiner différents concepts ou des composants de divers méta-heuristiques [7] et à cette fin, ils tentent de fusionner les points forts et éliminent les faiblesses

Xème Conférence Internationale : Conception et Production Intégrées, CPI 2015, 2-4 Décembre 2015, Tanger - Maroc.

Xth International Conference on Integrated Design and Production, CPI 2015, December 2-4, 2015, Tangier - Morocco.

des différents concepts de méta-heuristiques. Par conséquent, l'efficacité de l'espace de solution rechercher peut être encore améliorée et de nouvelles opportunités émergé ce qui peut conduire à des méthodes de recherche encore plus puissante et plus flexible. Talbi [8] a proposé une taxonomie pour les méta-heuristiques hybrides, [9], propose une classification des techniques d'hybridation en les classifiant en trois catégories selon leur architecture :

- Hybridation séquentielle
- Hybridation parallèle synchrone
- Hybridation parallèle asynchrone

1) Hybridation séquentielle : c'est le type d'hybridation le plus populaire. Elle consiste à appliquer plusieurs méthodes de telle manière que le (ou les) résultat(s) d'une méthode serve(nt) de solution(s) initiale(s) à la suivante.

2) Hybridation parallèle synchrone : cette hybridation est réalisée par incorporation d'une méthode de recherche particulière dans un opérateur. Elle est plus complexe à mettre en œuvre que la précédente. L'objectif est de combiner une recherche locale avec une recherche globale dans le but d'améliorer la convergence.

3) Hybridation parallèle asynchrone (coopérative) : les méthodes hybrides appartenant à cette classe sont caractérisées par une architecture telle que deux algorithmes A et sont impliqués simultanément et chacun ajuste l'autre. Les Algorithmes A et B partagent et échangent de l'information tout au long du processus de recherche.

B. Méthodes multi objectifs

L'optimisation multi objectif est un domaine fondamental de l'aide à la décision multicritère, nécessaire aux nombreux milieux scientifiques et industriels. Au cours des deux dernières décennies, un très grand nombre de travaux, à la fois théoriques et appliqués, ont été publiés dans ce domaine.

La résolution d'un problème d'optimisation multi objectif consiste à déterminer la solution correspondante au mieux aux préférences du décideur parmi les solutions de bon compromis. L'une des questions les plus difficiles est donc liée à l'identification de l'ensemble Pareto optimal, ou d'une approximation de celui-ci pour des problèmes complexes. En particulier, beaucoup de problèmes de mécanique rencontrés dans l'industrie sont de nature multi objectif.

La classification des méthodes d'optimisation multi objectif s'articule autour des notions de transformation et d'optimum de Pareto :

• Les méthodes agrégées qui sont La méthode de pondération, la méthode de programmation par but, la méthode du min-max et la méthode du ε contrainte.

• Les méthodes fondées sur Pareto : proposée par

[10] pour résoudre les problèmes proposés par [11], qui sont l'algorithme génétique à plusieurs objectifs (MOGA), l'algorithme génétique de tri non dominé (NSGA II), l'algorithme SPEA II et l'algorithme PAES.

• Les méthodes non agrégées et non Pareto comme l'algorithme VEGA.

Les méthodes agrégées ou l'utilisation de la dominance de Pareto traitent les objectifs simultanément, alors que, les méthodes dites non agrégées et non Pareto possèdent un processus de recherche qui traite séparément les objectifs.

En ce qui concerne le problème multi objectif, trois techniques différentes de résolution sont proposées.

L'agrégation en un seul objectif, les méthodes basées sur des approches non Pareto et les méthodes basées sur la dominance de Pareto. Les algorithmes MOGA [12], NPGA [13], SPEA-II [14], NSGA-II [15] sont quelques méthodes proposées sur la dominance de Pareto pour résoudre des problèmes multi objectifs.

III. RESOLUTION DU PROBLÈME DE L'ORDONNANCEMENT EN INDUSTRIE AUTOMOBILE

Les industries doivent maintenir un taux de productivité en constante évolution pour faire face à la concurrence ; des contraints peuvent toutefois apparaître au niveau du système de production qui engendre des coûts de production ainsi que des coûts de non utilisation relativement élevés.

Nous intéressons dans l'article au cas d'usine de fabrication des câbles de banche de bord des voitures pour l'industrie automobile, un cas similaire est déjà traité dans [16].

L'atelier peut contenir une ou plusieurs chaînes de production, chaque chaîne est composée de 4 Machines :

- M1 : Machine de coupe et de torsion des fils simple.
- M2 : Machine d'assemblage du câble.
- M3 : Machine de test électrique pour tester la fonctionnalité de produit fini.
- M4 : Machine d'emballage des produits finis.

L'atelier étudié comporte 2 chaînes de production.

Les produits passent par les quatre machines en série dans le même ordre. Il s'agit donc d'un atelier de type flow-shop.

Plusieurs opérations de changement des outils ou de réglages sont à gérer sur les postes de l'atelier, conjointement aux opérations de production. Les temps improductifs générés par ces opérations sont assez importants, compte tenu du fait que le temps de lancement de la fabrication d'un produit dépend de celui qui l'a précédé. Tous les arrêts et les changements survenant au cours d'une opération doivent être pris en compte pour permettre un bon ordonnancement de l'atelier.

A. Problèmes d'ordonnancement de type flow-shop

Dans le problème d'ordonnancement de systèmes de type flow shop chaque machine ne peut effectuer qu'une seule opération à la fois et chaque job ne peut avoir qu'une seule opération en cours de réalisation simultanément. La capacité de stockage inter-machines est définie et la préemption d'opérations n'est pas autorisée. Les problèmes

d'ordonnancement d'ateliers de type flow-shop, ou ateliers à cheminements uniques, sont parmi les problèmes les plus connus dans le domaine de l'ordonnancement. Ils ont fait l'objet de nombreuses études dont l'objectif principal est la minimisation de la date de fin d'exécution [17] [18] [19].

La résolution des problèmes d'ordonnancement de type flow-shop a connu plusieurs étapes.

Depuis les travaux de Johnson [20], l'objectif le plus visé, dans la résolution de ces problèmes d'ordonnancement est de réduire au maximum le Makespan ou C_{max}, temps de fin de fabrication du dernier produit.

Diverses approches heuristiques [21] [22] [23] [24] ont été proposées pour réduire le Makespan [25].

Certain auteurs sont concentrés sur la résolution d'un seul objectif [26] [27] [28] [29] [30] [31] [32], Cependant, la plupart des problèmes réels impliquant des objectifs multiples.

D'autres auteurs ont abordés les problèmes d'ordonnancement multi-objectifs de type flow-shop [33] [34] [35] [36] [37] [38] [39] [40] [41] [42] [43] [44] [45] [46] [47].

Nous nous proposons de résoudre le problème d'ordonnancement en industries automobile d'un point de vue multi-objectif tout en énonçant les différents critères et contraintes s'y rattachant et en utilisant les algorithmes génétiques comme méthode de résolution.

B. Optimisation multi-objectifs

D'un point de vue mathématique, un problème d'optimisation multi-objectif, se présente, dans le cas où le vecteur f regroupe k fonctions objectif, de la façon suivante [7] :

$$\begin{cases} \text{minimiser} & \vec{f}(\vec{x}) \quad \vec{x} \in R^m \quad , \vec{f}(\vec{x}) \in \Re^k \\ \text{avec} & \vec{g}(\vec{x}) \leq 0 \quad \vec{g}(\vec{x}) \in \Re^m \\ \text{et} & \vec{h}(\vec{x}) = 0 \quad \vec{h}(\vec{x}) \in \Re^p \end{cases}$$

Pour mieux résoudre ces problèmes d'optimisation multi-objectifs, il est nécessaire de simplifier les différentes fonctions objectives pour faciliter leur traitement. Parmi les différentes méthodes utilisées, la méthode de pondération des fonctions objectives qui se présente est comme suit : à chacune des fonctions objectives, est appliqué un coefficient de pondération et la somme pondérée de fonctions objectives est effectuée. Une nouvelle fonction objective est ainsi obtenue [48].

Formulation du problème avec la méthode de pondération des fonctions objectives :

$$\begin{cases} \text{miminiser} & f_{eq}(\vec{x}) = \sum_{i=1}^{k} w_i f_i(\vec{x}) \quad \vec{x} \in \Re^n \\ w_i \geq 0 \ \forall i \in \{1, \ldots, k\} & \text{et} \quad \sum_{i=1}^{k} w_i = 1 \\ \text{avec} & \vec{g}(\vec{x}) \leq 0 \quad \vec{g}(\vec{x}) \in \Re^m \\ \text{et} & \vec{h}(\vec{x}) = 0 \quad \vec{h}(\vec{x}) \in \Re^p \end{cases}$$

Cette approche est très efficace algorithmiquement par rapport aux autres méthodes compte tenu du fait qu'elle permet de retrouver la surface de compromis, en faisant varier les coefficients de pondération par le décideur qui dispose des connaissances approfondies du problème étudié, en plus elle est aussi simple à utiliser en effet, le problème multi-objectif est transformé en problème mono-objectif.

L'hybridation séquentielle des deux métaheuristiques est utilisée aussi dans cet article pour faire l'optimisation multi-objective, la première pris en compte le critère C1 et la seconde pris en compte le critère C2.

L'atelier étudié possède deux chaînes Ch1 et Ch2 fonctionnant en flow-shop. L'optimisation du passage des produits dans les chaînes consiste à minimiser les temps de production (en minimisant les temps d'arrêt, de nettoyage et de non utilisation des chaînes).

Nous proposons de mettre en œuvre une méthode d'optimisation multi-objective basée sur l'hybridation des métaheurstiques pour la résolution du problème d'ordonnancement des produits.

C. Formulation du problème

- F_{ik}: opération de fabrication du produit i sur la ligne Ch_k.
- P_i : produit fini après l'opération F_{ik}.
- P_{ik} : temps de fabrication de l'opération F_{ik}.
- CP_{ik}: temps de fin d'exécution de P_i sur la chaîne Ch_k.
- CP_i^{stk}: coût de stockage par unité de temps du produit P_i.
- tp_{ik} : temps de préparation de la chaîne Ch_k avant l'opération F_{ik}.
- tp_{ik}^{arr} : Temps d'arrêt durant l'opération F_{ik} sur la chaîne Ch_k.
- tp_{ik}^{nu} : Temps de non utilisation de la chaîne Ch_k avant l'opération F_{ik}.
- C_{prod}^{tot} : Coût total de production.
- C_k^{ui} : Coût unitaire de production du produit i sur la chaîne Ch_k.
- D_{ik}^{nett} : Durée des opérations de nettoyage sur la chaîne Ch_k.
- D_{ik}^{chf} : Durée des changements de format sur la chaîne Ch_k.
- $C_{arr\,k}$: Coûts d'arrêt et de non utilisation de la chaîne Ch_k par unité de temps.
- C_{arr}^{tot} : Coût total d'arrêt et de non utilisation des chaînes par unité de temps.
- $trop\_Ch_i$: Les temps de production exprimés en unité de temps.
- $tarr\_Ch_i$: Les temps d'arrêt exprimés en unité de temps.
- $tnett\_Ch_i$: Les temps de nettoyage exprimés en unité de temps.

Les dates sont calculées à partir d'un temps initial t_0 :

- $Cpro\_Ch_i$: Les coûts de production.
- $Cnou\_Ch_i$: Les coûts de non utilisation des chaines.

D. Critères à minimiser

Les critères à minimiser sont ceux relatifs aux coûts de fabrication et aux coûts des temps engendrés par les opérations de nettoyage ainsi que des temps d'arrêt et de non utilisation des chaînes.

Ils s'expriment respectivement par les deux fonctions objectives suivantes : F1 et F2, correspondant respectivement au coût total de production et au coût d'arrêt et de non utilisation des chaînes.

$$F_1 = C_{prod}^{tot} = \sum_k \sum_i W_{ik}\ P_{ik}\ C_k^{ui},$$

$$W_{ik} = \begin{cases} 1 : \text{si le produit est fabriquer dans la chaine} \\ 0 : \text{sinon} \end{cases} \quad (1)$$

$$F_2 = C_{arr}^{tot} = \sum_k C_{arr\,k} \sum_i W_{ik}\ tp_{ik}^{arr} + tp_{ik}^{nu} \text{ tel que } tp_{ik}^{arr} = D_{ik}^{nett} + D_{ik}^{chf},$$

$$W_{ik} = \begin{cases} 1 : \text{si le produit est fabriquer dans la chaine} \\ 0 : \text{sinon} \end{cases} \quad (2)$$

E. Fonction fitness à optimiser

Etant donné que la minimisation du coût de production est plus importante que la minimisation des coûts de non utilisation des chaînes, la fonction fitness F à minimiser dans le premier type de l'hybridation correspond à la somme pondérée des deux fonctions objectifs F_1 et F_2, avec des poids β_1 et β_2 définis en fonction de l'importance de ces deux critères :

$$F = F_1\,\beta_1 + F_2\,\beta_2 \ (3),\quad \beta_1 + \beta_2 = 1,\quad \beta_1 > 0,\quad \beta_2 > 0 \quad (3)$$

$$F = \beta_1\ C_{prod}^{tot} + \beta_2\ C_{arr}^{tot},\quad \beta_1 + \beta_2 = 1,\quad \beta_1 > 0, \beta_2 > 0 \quad (4)$$

$$F = \beta_1 \left(\sum_k \sum_i W_{ik}\ P_{ik}\ C_k^{ui} \right) + \beta_2 \left(\sum_k C_{arr\,k} \sum_i W_{ik}\ tp_{ik}^{arr} + tp_{ik}^{nu} \right),$$

$$\beta_1 + \beta_2 = 1,\quad \beta_1 > 0,\quad \beta_2 > 0 \quad (5)$$

Pour le deuxième type de l'hybridation correspond à la fonction objective F1 pour la première méta-heuristique et la fonction objective F2 pour la deuxième méta-heuristique.

Les coûts sont calculés en fonction des formules (1) et (2) et la fonction fitness déduite à partir de la formule (3), (4) et (5).

F. Adaptation d'algorithme génétique :

Les algorithmes génétiques sont des algorithmes d'optimisation inspirés de la théorie de l'évolution des espèces de Charles Darwin. Les premier travaux de John Holland remontent aux années 1960 et ont trouvé un premier aboutissement en 1975 avec la publication de [49]. C'est cependant l'ouvrage de David Goldberg qui a largement contribué à développer les algorithmes génétiques [50]. Un algorithme génétique est basé sur une population d'individus dont chacun est une solution candidate du problème. Chaque solution doit être codée. Cette représentation codée est appelée chromosome, et est composée de gènes. Le degré d'adaptation d'un individu à l'environnement est exprimé par la valeur de la fonction objective correspondante. La taille de la population reste constante tout au long de l'algorithme génétique. La recherche de la solution est réglée par trois opérateurs qui sont appliqués successivement. La phase de coopération est gouvernée par un opérateur de sélection et un opérateur de croisement alors que la phase d'adaptation individuelle fait appel à un opérateur de mutation. La création d'une nouvelle génération est obtenue par itération de l'algorithme génétique qui va créer de nouveaux individus et en détruire d'autres ce qui permet le renouvellement de la population. L'exploration de l'espace de recherche est alors réalisée par les opérateurs de mutation et assure la diversification des individus de la population. L'exploitation, quant à elle, est assurée par les opérateurs de croisement, qui recombinent les solutions, afin de les améliorer en conservant leurs meilleures caractéristiques.

Les algorithmes génétiques présentent plusieurs avantages tels que :

- La simplicité de l'approche ;
- La possibilité de paralléliser l'algorithme ;
- La facilité d'implémentation ;
- La flexibilité : peut être facilement modifié pour d'autres problèmes ;
- Il gère les problèmes d'optimisation multi-objectifs et multimodales ;
- Il permet une bonne exploration de l'espace de recherche ;

D'autre part, il existe des limites pour cet algorithme tels que :

- Le problème de représentation de la solution.
- L'ajustement de différents paramètres : taille de population, taux de mutation,…etc.
- Son exécution qui est lente par rapport à d'autres méthodes.
- Sa convergence prématurée.
- Il ne peut pas garantir des temps de réponse constants.

Pour utiliser un algorithme génétique pour un problème particulier, on doit donc disposer des cinq éléments suivants [51] :

- Le codage des solutions : associe à chacun des points de l'espace d'états une structure de données. Elle vient généralement après une phase de modélisation mathématique du problème traité. La qualité du codage conditionne le succès de l'algorithme.
- Une méthode de génération de la population initiale : la population d'individus produite initialement qui servira de base pour les générations futures doit être non homogène.
- Une fonction à optimiser : Cette fonction retourne une valeur réelle appelée fitness, qui va permettre de déterminer la probabilité de sélection d'un individu [52].
- Les opérateurs génétiques : permettent de diversifier la population au cours des générations et d'explorer l'espace d'états. Le croisement recompose les gènes d'individus de la population. La mutation entraîne des

altérations minimes sur les individus pour éviter la convergence rapide de la population. La sélection favorise les meilleurs individus.

- Les paramètres de dimensionnement : taille de la population, nombre total de générations ou critère d'arrêt, probabilités d'application des opérateurs de croisement et de mutation.

Le logigramme de l'algorithme génétique adapté est décrit comme suit :

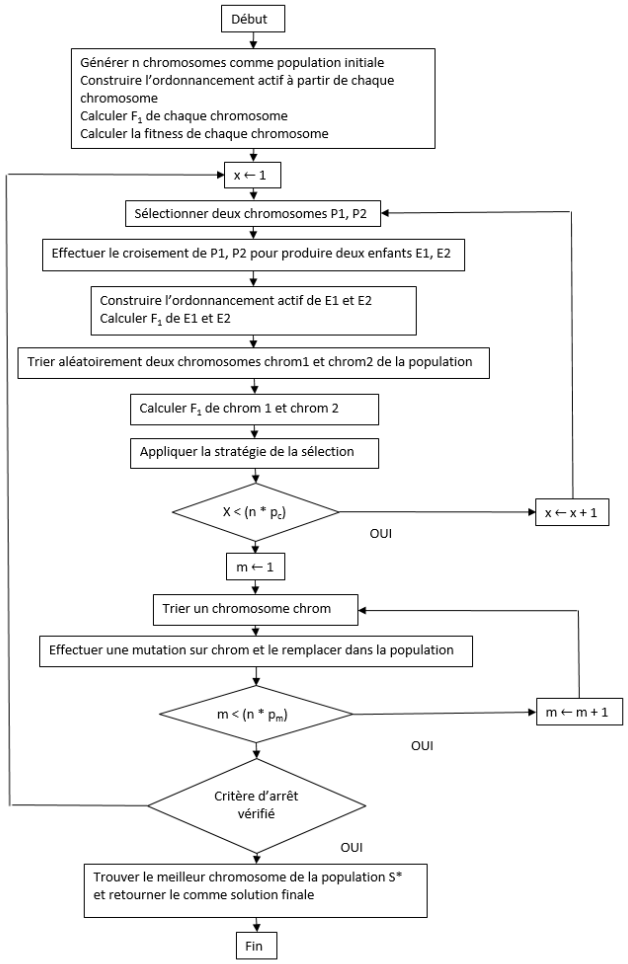


Fig 1. Logigramme de l'algorithme génétique adapté

G. Adaptation du recuit simulé :

Dans le recuit simulé (SA), un système est initialisé à une température T avec une configuration dont l'énergie est évaluée comme [53], [54], [55]. Une nouvelle configuration est réalisée en appliquant une variation aléatoire, et la variation de l'énergie ΔE est calculée. La nouvelle configuration est inconditionnellement acceptée si elle diminue l'énergie du système ΔE. Si l'énergie du système est augmentée par le changement, la nouvelle configuration est acceptée avec une certaine probabilité aléatoire. Dans le schéma original de Metropolis, la probabilité est donnée par le facteur de Boltzmann $e^{-\Delta E/KT}$. Ce processus est répété plusieurs fois à la température actuelle afin de parcourir efficacement l'espace de recherche, puis la température est abaissée. Le procédé est répété à des températures successivement plus basses jusqu'à ce que l'état congelé soit atteint. Cette procédure permet au système de se déplacer à la baisse des états d'énergie, tout en sautant hors des minima locaux (en particulier à des températures plus élevées) en raison de l'acceptation probabiliste de quelques mouvements à la hausse.

Le logigramme du recuit simulé adapté est décrit comme suit :

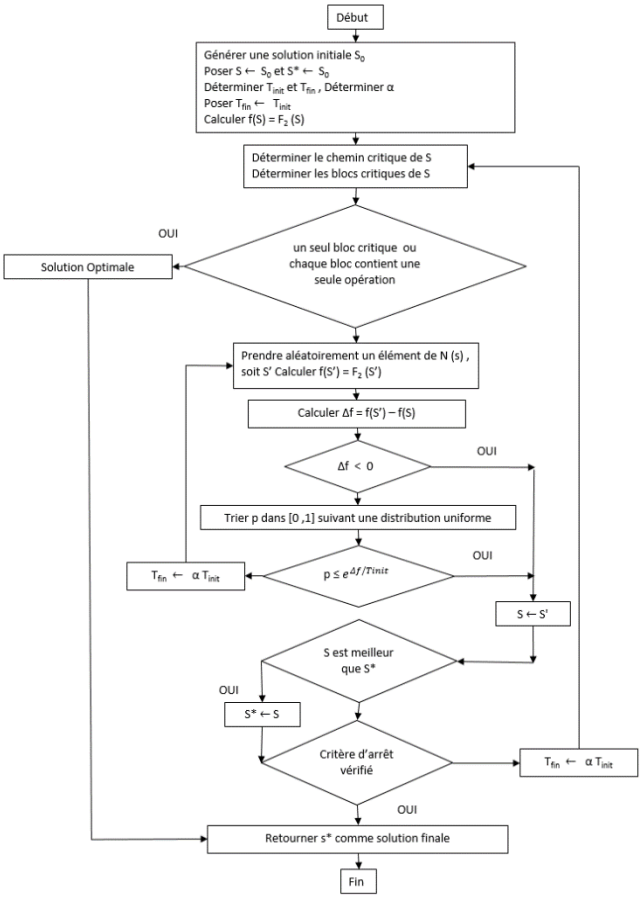


Fig 2. Logigramme de la recuit simulé adapté

H. Adaptation de la méthode Tabou

La recherche tabou (TS) est une méthode d'optimisation mathématique, appartenant à la classe des techniques de recherche locale [56], [57], [58]. La recherche tabou améliore les performances d'une méthode de recherche locale en utilisant des structures adaptées de mémoire : une fois une solution potentielle a été déterminée, elle est marquée comme taboue afin que l'algorithme ne visite pas cette possibilité à plusieurs reprises. Pour explorer les régions de l'espace de recherche qui seraient laissées inexplorées par la procédure de recherche locale, la recherche tabou modifie la structure du voisinage de chaque solution au fur et à mesure que la recherche progresse. Les solutions admises sont déterminées par l'utilisation de structure de mémoire.

Dans sa forme la plus simple, une liste taboue est une mémoire à court terme qui contient les solutions qui ont été

visitées dans le passé récent. C'est le type le plus important de la structure de la mémoire utilisée pour déterminer les solutions admises pour la liste taboue.

Les concepts de base de la recherche tabou sont :

- Liste tabou : utilisation d'une structure de mémoires flexibles pour mémoriser les configurations ou bien les zones visitées et inclure des mécanismes permettant d'interdire à l'algorithme certains mouvements pour ne pas retourner trop rapidement (temporairement) vers les zones visitées.
- Critère d'aspiration : (assouplissement du mécanisme de la liste taboue) autorisation et acceptation de certains mouvements tabous.
- L'alternance entre ces deux premiers concepts dans le processus est réalisée à l'aide d'un mécanisme de contrôle.
- Stratégie d'intensification : exploitation plus approfondie des zones prometteuses trouvées (visitées) récemment et renforcement de la recherche dans ces régions.
- Stratégie de diversification : exploration des zones qui n'ont pas encore été visitées.

Le logigramme de la recherche Tabou adaptée est décrit comme suit :

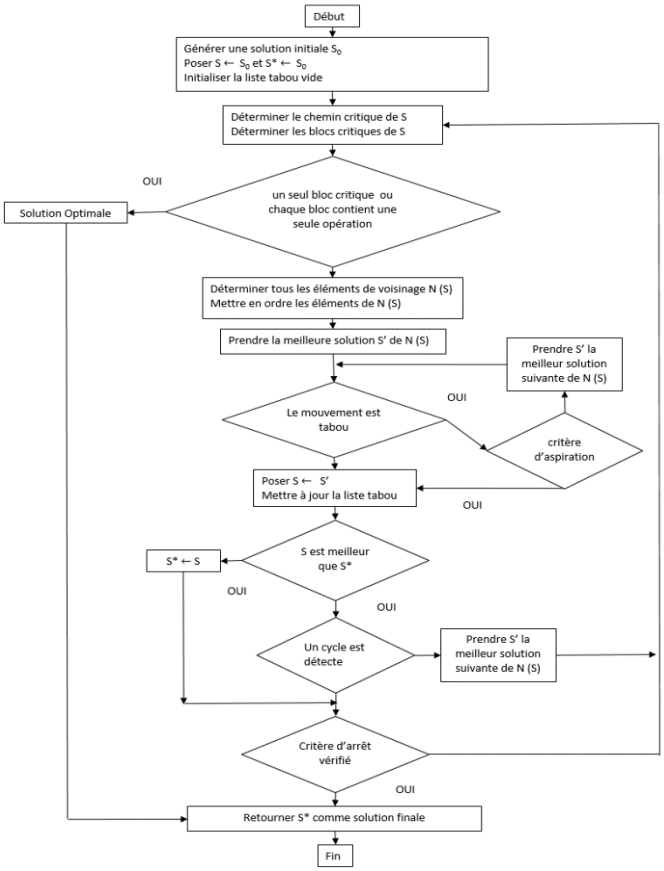


Fig 3. Logigramme de la recherche Tabou adaptée

I. Application de la 1er approche de l'hybridation des méta heuristiques

L'hybridation séquentielle consiste à appliquer plusieurs méthodes de telle façon que les résultats d'une méthode servent de solutions initiales à la suivante.

L'algorithme génétique possède une connaissance globale explore l'espace de recherche tandis que la recherche tabou détient une connaissance locale et exploite le voisinage. La résolution par l'approche d'hybridation séquentielle se fait en deux étapes :

La première étape est une application directe de l'algorithme génétique pour la prise en compte de critère F_1. Dans cette étape la population initiale est générée aléatoirement, l'opérateur de sélection permet de choisir les meilleurs individus de la population courante, les opérateurs de croisement et de mutation permettent la génération de nouvelles solutions, à la fin de cette étape, une population d'individus qui satisfait le critère F_1 est obtenue.

Dans une deuxième étape, on a recours à la recherche tabou pour optimiser le critère F2. Dans ce cas, un individu quelconque de la population finale, obtenue dans la première étape, est choisi comme solution initiale. Cet individu servira de base pour la détermination des solutions ultérieures, une fonction voisine permet de générer des nouvelles solutions à partir de la solution courante. La liste tabou permet de conserver la trace des dernières solutions déjà visitées. Le critère d'aspiration permet de revenir à une solution déjà visitée et de redémarrer la recherche dans une autre direction. A la fin de cette deuxième étape, on obtient une solution qui satisfait le deuxième critère. Finalement, on obtient une solution qui satisfait les deux critères. La technique d'hybridation adoptée dans notre travail est celle décrite par figure 4.

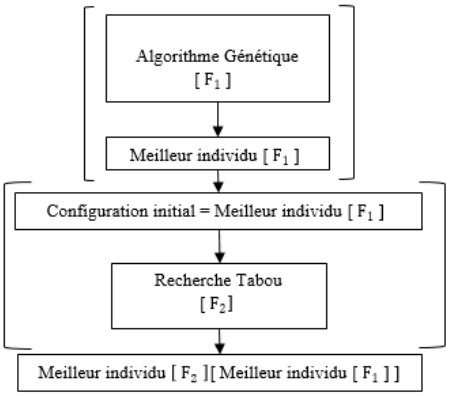


Fig 4. Hybridation sequentiel de l'algorithme génétique avec la recherche tabou

De la même manière on hybride l'algorithme génétique avec le recuit simulé et l'algorithme génétique avec l'algorithme génétique comme le montre la figure 5 et la figure 6.

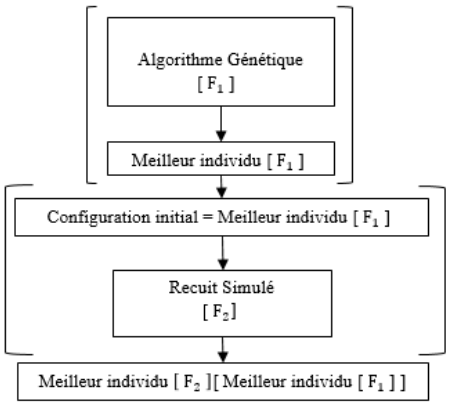


Fig 5. Hybridation sequentiel de l'algorithme génétique avec la recuit simulé

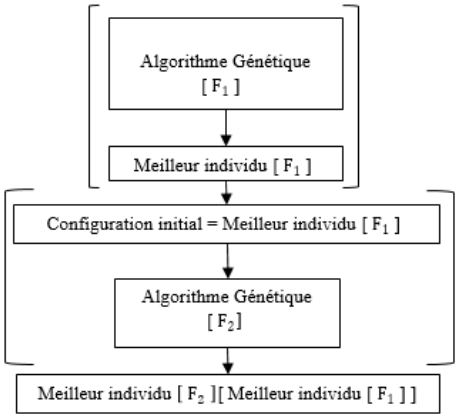


Fig 6. Hybridation sequentiel de l'algorithme génétique avec la l'algorithme génétique

J. Application de la 2 ème approche de l'hybridation des méta heuristiques

L'algorithme se compose de deux phases qui alternent régulièrement les progrès génétiques. Dans la première, seule la partie pure de l'AG est active à objectif d'explorer l'espace des solutions pour détecter les zones d'intérêt où les solutions peuvent être réglées. La deuxième phase est l'endroit où l'AG est hybride (l'opérateur de mutation qui est remplacé par d'autre algorithme).

Certaines régions prometteuses de l'espace de recherche vont être atteintes et donc vont être exploitées par le recuit simulé [59] et la recherche tabou. L'équilibre entre l'exploration et l'exploitation va être optimisé.

La technique d'hybridation adoptée dans notre travail est celle décrite par la figure 7 :

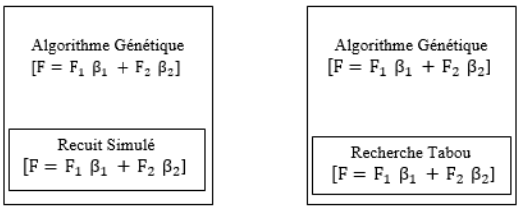


Fig 7. Hybridation parallèle synchrone de l'algorithme génétique avec la recuit simulé et avec la recherche tabou

IV. SIMULATION ET ANALYSE

Tous les méthodes décrit dans cet article ont été programmées en Java et exécutées dans un Core ™ i5 CPU avec 1.9 GHz 2.5 GHz et 4 Go de RAM.

Les valeurs de $\mathrm{tarr\_Ch}_i$ et $\mathrm{tnett\_Ch}_i$ de chaque produit sur chaque chaîne sont attribuées aléatoirement entre 1 et 10.

$$\forall\, i = \{1,2\} \quad \mathrm{tarr\_Ch_i}(P_i) = \mathrm{Random}[1,5]$$

$$\forall\, i = \{1,2\} \quad \mathrm{tnett\_Ch}_i(P_i) = \mathrm{Random}[1,5]$$

La valeur de $\mathrm{trop\_Ch_i}$ de chaque produit sur chaque chaîne est attribuée aléatoirement entre 20 et 90.

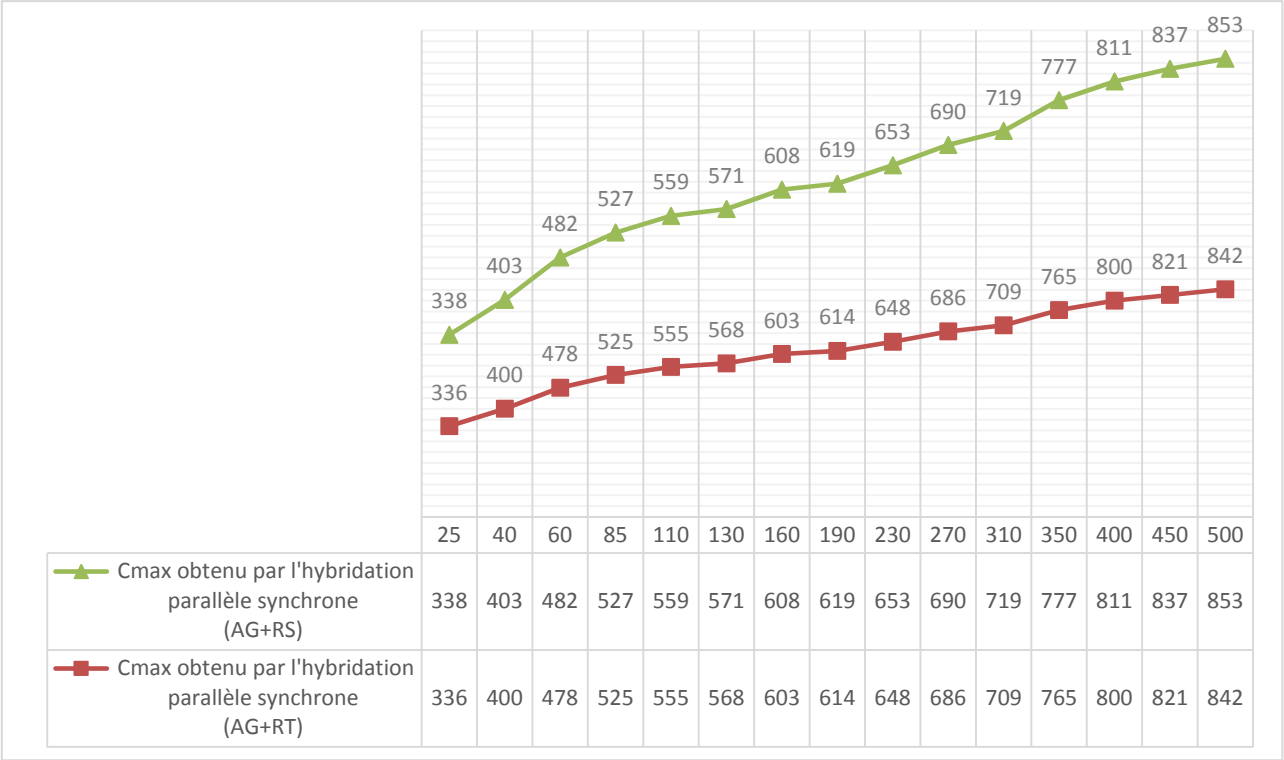
$$\forall\, i = \{1,2\} \quad \mathrm{trop\_Ch_i}(P_i) = \mathrm{Random}[20,90]$$

Les valeurs de $\mathrm{Cpro_Ch}_i$et $\mathrm{Cnou\_Ch}_i$ de chaque produit sur chaque chaîne calculées en fonction des formules (1) et (2) et la fonction fitness déduite à partir des formules (3) et (4).

Ces valeurs sont ensuite stockées dans un fichier pour leur utilisation dans tous les programmes, ils sont exprimés en minute.

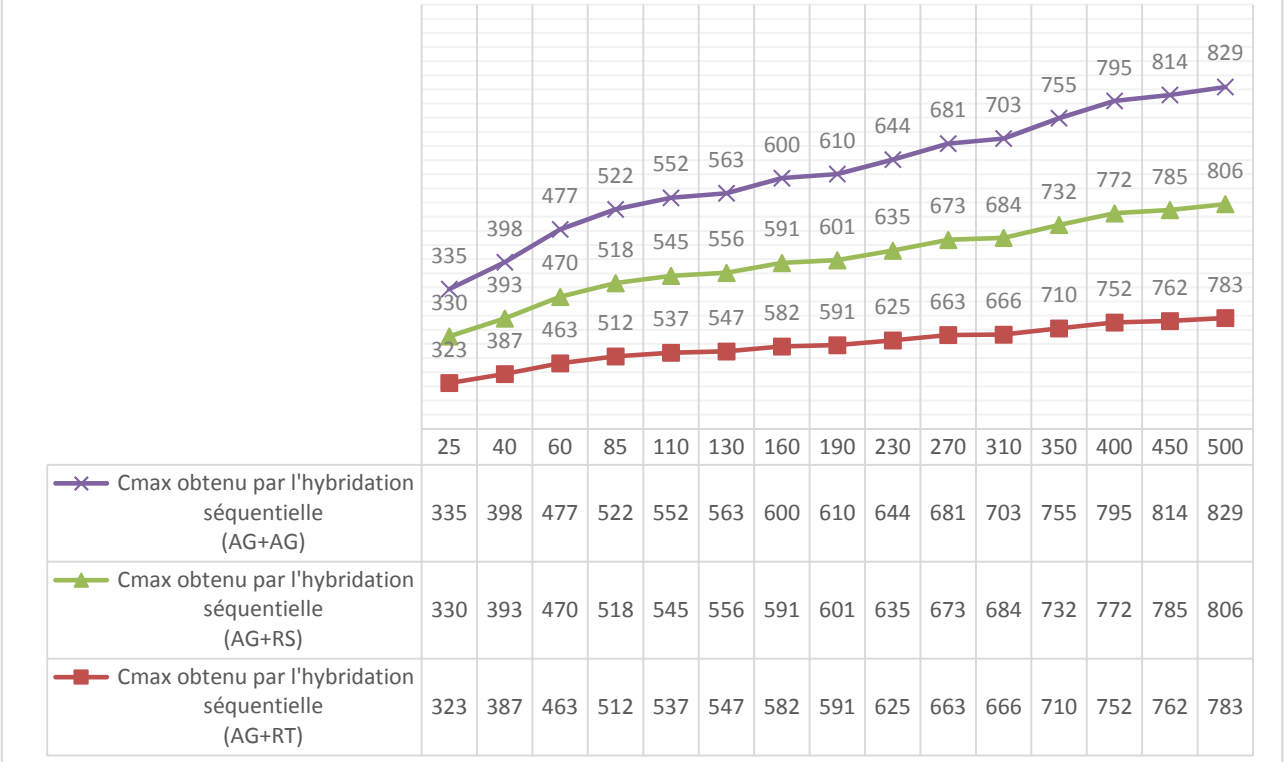
Les paramètres de la configuration de l'algorithme génétique, la recherche tabou et le recuit simulé sont les mêmes dans les deux types d'hybridation.

Après la construction des ordonnancements qui satisfaits les critères citées au-dessus en utilisent les deux types d'hybridation, la première est l'hybridation parallèle synchrone avec l'utilisation de la fonction de pondération (de fonction fitness F), et la deuxième est l'hybridation séquentielle (de fontion fitness F_1 pour la 1er métaheuristique et F_2 pour la 2éme métaheuristique), on obtient les résultats montrer dans les figures 8 et 9 respectivement.



	25	40	60	85	110	130	160	190	230	270	310	350	400	450	500
Cmax obtenu par l'hybridation parallèle synchrone (AG+RS)	338	403	482	527	559	571	608	619	653	690	719	777	811	837	853
Cmax obtenu par l'hybridation parallèle synchrone (AG+RT)	336	400	478	525	555	568	603	614	648	686	709	765	800	821	842

Fig 8. Hybridation parallèle synchrone de l'algorithme génétique avec le recuit simulé et avec la recherche tabou en utilisant la fonction de pondération



	25	40	60	85	110	130	160	190	230	270	310	350	400	450	500
Cmax obtenu par l'hybridation séquentielle (AG+AG)	335	398	477	522	552	563	600	610	644	681	703	755	795	814	829
Cmax obtenu par l'hybridation séquentielle (AG+RS)	330	393	470	518	545	556	591	601	635	673	684	732	772	785	806
Cmax obtenu par l'hybridation séquentielle (AG+RT)	323	387	463	512	537	547	582	591	625	663	666	710	752	762	783

Fig 9. Hybridation séquentielle de l'algorithme génétique avec le recuit simulé, avec la recherche tabou et avec l'algorithme génétique

L'étude comparative des solutions obtenues par l'hybridation parallèle synchrone de l'algorithme génétique avec le recuit simulé et de l'algorithme génétique avec la recherche tabou en utilisant la fonction de pondération preuve que cette dernière méthode fournit des meilleures solutions comme le montre la figure 8.

Concernant l'hybridation séquentielle de l'algorithme génétique avec le recuit simulé, de l'algorithme génétique avec la recherche tabou et de l'algorithme génétique avec l'algorithme génétique leur étude comparative des solutions obtenues fournit le classement suivant au niveau de la qualité comme le montre la figure 9 :

L'hybridation séquentielle de l'algorithme génétique avec la recherche tabou puis l'hybridation séquentielle de l'algorithme génétique avec le recuit simulé enfin 'hybridation séquentielle de l'algorithme génétique avec l'algorithme génétique.

Cette étude montre que l'hybridation séquentielle de ces trois métaheuristiques donnes des meilleures solutions par rapport à l'hybridation parallèle synchrone des mêmes métaheuristiques (en utilisant la fonction de pondération) pour résoudre le problème traiter dans cet article.

V. CONCLUSION

Le problème d'ordonnancement multi-objectif dans un atelier flow-shop est un problème extrêmement difficile à résoudre et pour améliorer les solutions qui fournissent les métaheuristiques, nous avons appliqué deux approches d'hybridation séquentielle et parallèle synchrone (en utilisent la méthode de pondération des fonctions objectives) entre l'algorithme génétique et les métaheuristiques à solution unique (la recherche tabou et le recuit simulé). Les résultats obtenus par la 1er méthode sont nettement meilleurs que ceux obtenus par la 2éme méthode.

Ainsi, l'hybridation séquentielle de l'algorithme génétique avec la recherche tabou a amélioré la qualité des solutions obtenues.

Parmi les perspectives de ce travail est de généraliser cette démarche pour la résolution de problème d'ordonnancement multi-objectif sur d'autre type d'atelier de production.

D'autres part notre travail peut s'étaler à étudier, à développer, à exploiter et à hybrider d'autres métaheuristiques à population de solutions (la méthode de la colonie de fourmis, la recherche par dispersion, l'algorithme à essaim de particules, la recherche par dispersion, le système immunitaire artificiel …etc) séquentiellement avec les métaheuristiques à solution unique et à comparer leurs résultats pour trouver ceux qui fournissement les meilleurs solutions.

Références

[1] Esquirol P. et Lopez P., L'ordonnancement. Economica, 1999.

[2] Souier, M., Métaheuristiques pour la manipulation de routages alternatifs en temps réel dans un Job Shop, Mémoire de Magister,Université Abou Bakr Belkaid, Tlemcen, 2009.

[3] Günther R. Raidl , A Unified View on Hybrid Metaheuristics, Institute of Computer Graphics and Algorithms Vienna University of Technology, Vienna, Austria, 2006.

[4] H. Mühlenbein, M. Gorges-Schleuter & O. Krämer, Evolution algorithms in combinatorial optimization. Parallel Computing 7 : 65-85, 1988.

[5] J.J. Grefenstette, Incorporating problem specific knowledge into genetic algorithms, L.Davis (Ed.) Genetic Algorithms and Simulated Annealing, Morgan Kaufmann Publishers, p.42-60, 1987.

[6] Talbi, E.G., S. Cahon & N. Melab, Designing cellular networks using a parallel hybrid metaheuristic, Journal of Computer Communications 30(4): 698-713, 2007.

[7] Talbi, E.-G. Metaheuristics: From Design to Implementation, Wiley 2009.

[8] Talbi E.G., A taxonomy of hybrid metaheuristics. Journal of Heuristics 8(5), pp.541-565, 2002.

[9] D. Duvidier, Etude de l'hybridation des méta-heuristiques, application à un problème d'ordonnancement de type jobshop, Thèse de Doctorat, université du littoral France, décembre 2000.

[10] D.E. Goldberg. Genetic algorithms for search, optimization, and machine learning. In : Addison-Wesley, MA : (ed), Reading, 1989.

[11] D. Schaffer. Multiple objective optimisation with vector evaluated genetic algorithm. In genetic Algorithm and their Applications : Proceedings of the First International Conference on Genetic Algonthm, pages 93–100, 1985.

[12] C. M. Fonseca and P. J. Fleming. Genetic algorithm for multiobjective optimization formulation, discussion and generalization. In Proceedings of the Fifth International Conference on Genetic Algorithms, San Mateo. California, pages 416–423, 1993.

[13] Horn, J., Nafpliotis, N., and Goldberg, D. (1994). A niched Pareto genetic algorithm for multiobjective optmization. In First IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence, Volume 1.

[14] Zitzler, E. and Thiele, L. (1999). Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach. *Evolutionary Computation, IEEE*, 3(4):257–271.

[15] Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm : NSGA-II. *Evolutionary Computation, IEEE Transactions on*, 6(2):182–197.

[16] H. Jebari, S. Rahali El azzouzi, H. Samadi. Algorithmes évolutionnaires pour la résolution de problèmes d'ordonnancement multi-objectifs dans un atelier Flow-Shop. SITA14, 9th International Conference on Intelligent Systems: Theories and Applications, 07-08 May 2014, Rabat, Morocco.

[17] E. G. Negenman, « Local search algorithms for the multiprocessor flow-shop scheduling problem ». European Journal of Operational Research, vol. 128, pp. 147-158, 2001.

[18] J. Breit, « A polynomial-time approximation scheme for the two-machine flowshop scheduling problem with an availability constraint ». Computers & Operations Research, vol. 33, pp. 2143–2153, 2006.

[19] R. Ruiz, C. Maroto et J. Alcaraz, « Two new robust genetic algorithms for the flow-shop scheduling problem ». Omega, vol. 34, pp. 461 – 476, 2006.

[20] S.M. Johnson, « Optimal two and three stage production schedules with setup times included ». Naval Research and Logistics Quarterly, vol. 1, 1954.

[21] D. G. Dannenbring, « An evaluation of flow-shop sequencing heuristics ». Management Science, vol. 23, pp. 1174-1182, 1977.

[22] M. Nawaz, E.E. Enscore et I. Ham, « A heuristic algorithm for m-machine, n-job flow-shop sequencing problem ». OMEGA, vol. 11, pp. 91 98, 1983.

[23] I.H. Osman et C.N. Potts, « Simulated annealing for permutation flow-shop scheduling ». OMEGA, vol. 17, pp. 551-557, 1989.

[24] M. Widmer et A. Hertz, « A new heuristic method for the flow-shop sequencing problem ». European Journal of Operational Research, vol. 4, pp. 186-193.1990.

[25] T. Murata, H. Ishibuchi et H. Tanaka, « Multi-objective genetic algorithm and its applications to flow-shop scheduling ». Computers Industrial Engineering, vol. 30, n° 4, pp. 957-968, 1996.

[26] Pan JC-H, Chen J-S, Chao C-M. Minimizing tardiness in a two-machine flow shop. Computers and Operations Research 2002; 29(7):869–85.

[27] Fink A, Vob S. Solving the continuous flow shop scheduling problem by metaheuristics. European Journal of Operational Research 2003; 151(2):400–14.

[28] Bulfin RL, M'Hallah R. Minimizing the weighted number of tardy jobs on a two-machine flow shop. Computers and Operations Research 2003; 30(12):1887–900.

[29] Blazewicz J, Pesch E, Sterna M,Werner F. A comparison of solution procedures for two-machine flow shop scheduling with late work criterion. Computers and Industrial Engineering 2005; 49(4):611–24.

[30] Choi B-C, Yoon S-H, Chung S-J. Minimizing maximum completion time in a proportionate flow shop with one machine of different speed. European Journal of Operational Research 2007; 176 (2): 964–74.

[31] Grabowski J, Pempera J. Some local search algorithms for no-wait flow shop problem with makespan criterion. Computers and Operations Research 2005; 32(8):2197–212.

[32] Wang J-B, Daniel Ng.CT, Cheng TCE, Li-Li Liu. Minimizing total completion time in a two machine flow shop with deteriorating jobs. Applied Mathematics and Computation 2006;180(1):185–93.

[33] J.C. Ho et Y.L. Chang, « A new heuristic for the n-job, m-machine flow-shop problem ». European Journal of Operational Research, vol.52, pp. 194-202, 1991.

[34] R. Gangadharan et C. Rajendran, « A simulated annealing heuristic for scheduling in a flow-shop with bi-criteria ». 16th International Conference on Computers Industrial Engineering, pp. 345-348, 1994.

[35] M. Mabed, M. Rahoual, E. Talbi et C. Dhaenens, « Algorithmes génétiques multicritères pour les problèmes de flow-shop ». 3e Conférence Francophone de Modélisation et SIMulation MOSIM'01 – du 25 au 27 avril 2001 – Troyes.

[36] S. Bertel et J. C. Billaut, « Problème d'ordonnancement multicritère dans un flow-shop hybride avec recirculation ». 3ème Conférence Francophone de Modélisation et SIMulation, MOSIM'01, 25-27 avril 2001, Troyes.

[37] K. Neumann, C. Schwindt et N. Trautmann, « Scheduling of continuous and discontinuous material flows with intermediate storage restrictions ». European Journal of Operational Research, vol. 165, pp. 495–509, 2005.

[38] G. Onwubolu et D. Davendra, « Scheduling flow shops using differential evolution algorithm ». European Journal of Operational Research, vol. 171, pp. 674–692, 2006.

[39] Toktas B, Azizoglu M, Koksalan S.K. Two-machine flow shop scheduling with two criteria: maximum earliness and makespan. European Journal of Operation Research 2004;157(2):286- 95.

[40] Arroyo JEC, Armentano VA. Genetic local search for multi-objective flowshop scheduling problems. European Journal of Operational Research 2005; 167 (3): 717–38.

[41] Noorul Haq A, Radha Ramanan T. A bicriterian flow shops scheduling using artificial neural network. The International Journal of Advanced Manufacturing Technology 2006; 30(11- 12):1132-8.

[42] Rahimi-Vahed R, Mirghorbani S.M. A multi-objective particle swarm for a flow shop scheduling problem. Journal of Combinatorial Optimization 2007;13(1):79-102.

[43] Yagmahan B, Yenisey M.M. A multi-objective ant colony system algorithm for flow shop scheduling problem. Expert Systems with Applications 2010;37(2):1361–1368.

[44] Shahul Hamid Khan, B., Prabhaharan, G., & Asokan, P. (2007). A grasp algorithm for m-machine flowshop scheduling problem with bicriteria of makespan and maximum tardiness. International Journal of Computer Mathematics, 84(12), 1731–1741.

[45] Arroyo, J. E. C., & de Souza Pereira, A. A. (2010). A GRASP heuristic for the multi-objective permutation flowshop scheduling problem. The International Journal of Advanced Manufacturing Technology, 55(5-8), 741–753.

[46] Dubois-Lacoste, J., López-Ibáñez, M., & Stützle, T. (2011). A hybrid TP+PLS algorithm for bi-objective flow-shop scheduling problems. Computers & Operations Research, 38(8), 1219–1236.

[47] Ciavotta, M., Minella, G., & Ruiz, R. (2013). Multi-objective sequence dependent setup times permutation flowshop: A new algorithm and a comprehensive study. European Journal of Operational Research, 227(2), 301–313.

[48] Y. Collette et P. Siarry, « Optimisation Multiobjectif ». Editions Eyrolles, Paris, 2002.

[49] Holland J. H., Genetic algorithms and the optimal allocation of trials, SIAM Journal on Computing 2, 1973

[50] D. E. Goldberg, Genetic Algorithms in Search, Optimization, and Machine Learning, Addison-Wesley, Reading, MA, 1989

[51] Randy L. Haupt & Sue Ellen Haupt, practical genetic algorithms, 2nd ed. John Wiley & Sons, Inc., New Jersey 2004.

[52] D. Duvivier, Ph. Preux, C. Fonlupt, D. Robilliard, E-G. Talbi, The fitness.function and its impact on Local Search Methods, IEEE Systems, Man. and Cybernetics (IEEE SMC'98), pp. 2478-2483, San Diego, USA, 1998.

[53] S. Kirkpatrick, C.D. Gelatt, M.P. Vecchi, Optimation by Simulated Annealing, Science 220 (1983) pp. 671–680

[54] Bohachevsky I., Jonhson M. E., Stein M.L., Generalized simulated annealing for function optimization, Technometrics 28 (1986) pp. 209–217

[55] Dimitris Bertsimas, John Tsitsiklis, Simulated annealing, Statistical Science, vol. 8, No. 1 (1993) pp. 10–15

[56] Glover F., Tabu search—part I, ORSA Journal on Computing, 1(3) (1989) pp. 190–206

[57] Glover F., Tabu search—part II, ORSA Journal on Computing, 2(1) (1990) pp. 4–32

[58] Tai-Hsi Wu, Jinn-Yi Yeh, Chin-Chih Chang, A hybrid Tabu Search Algorithm to Cell Formation Problem and its Variants, World Academy of Science, Engineering and Technology 53 (2009) pp. 1090–1094

[59] H. Jebari, S. Rahali El azzouzi, H. Samadi. The Hybrid Genetic Algorithm for Solving Scheduling Problems in a Flexible Production System. International Journal of Computer Applications 110(12):22-29, January 2015.