7e Conférence Internationale de MOdélisation et SIMulation - MOSIM'08 - du 31 mars au 2 avril 2008 – Paris- France

« Modélisation, Optimisation et Simulation des Systèmes : Communication, Coopération et Coordination.».

UNE APPROCHE EVOLUTIONNAIRE BI-OBJECTIF POUR L'ORDONNANCEMENT CONJOINT PRODUCTION- MAINTENANCE : CAS DE MACHINES PARALLELES

A. BERRICHI, M. MEZGHICHE

Université M'hamed Bouguara de Boumerdès
LIFAB, Avenue de l'indépendance, 35000,
Boumerdès, Algérie.
{aberrichi, mohamed.mezghiche}@umbb.dz

L. AMODEO, F. YALAOUI, E. CHÂTELET

Université de Technologie de Troyes
ICD, Rue Marie Curie, BP 2060, 10010,
TROYES, France
{lionel.amodeo, farouk.yalaoui, eric.chatelet}@utt.fr

RESUME : *Cet article traite du problème de l'ordonnancement conjoint de la production et de la maintenance préventive selon une nouvelle approche Pareto permettant de trouver des compromis entre les objectifs de la production et ceux de la maintenance. Des modèles de fiabilité sont appliqués pour modéliser l'aspect maintenance du problème. L'objectif est de minimiser deux critères : le makespan pour l'aspect production et l'indisponibilité pour l'aspect maintenance préventive. Deux types de décisions doivent être prises simultanément : trouver la meilleure affectation des tâches de production aux machines pour minimiser le makespan et décider à quels moments réaliser les actions de maintenance préventive pour minimiser l'indisponibilité du système, leur nombre n'est pas fixé à l'avance. Une étude expérimentale a été conduite pour le réglage des paramètres de l'algorithme et plusieurs extensions sont proposées.*

MOTS-CLES : *ordonnancement, production, maintenance préventive, optimisation multi- objectifs, indisponibilité, algorithmes génétiques.*

1. INTRODUCTION

Dans les systèmes de production manufacturière, les machines sont parmi les ressources primordiales pour réaliser le plan de production. Les services qui interviennent directement sur les machines sont le service ordonnancement et le service maintenance.
Le problème du service ordonnancement est de trouver un ordonnancement prévisionnel admissible des tâches de production satisfaisant au mieux un ou plusieurs critères, par exemple satisfaire les délais fixés avec les clients en minimisant les retards ou terminer le plutôt possible en minimisant le makespan. La résolution de ce problème a fait, et fait encore l'objet de plusieurs études, en fonction de la configuration de l'atelier de production (une machine, machines parallèles, flow shop, job shop et open shop), du critère (ou critères) à optimiser et des contraintes à considérer (préemption, setups, etc.). La majorité de ces problèmes, même avec une seule machine, sont réputés être des problèmes NP –difficiles (Garey and Johnson, 1979).
Pour le service maintenance, l'une de ses missions principales est la détermination d'un meilleur planning des périodes de maintenance préventive (MP) dont l'objectif est d'assurer aux machines un bon état de fonctionnement à tout instant (une disponibilité maximale) et éviter ainsi les pannes. Plusieurs travaux de recherche dans ce domaine ont été réalisés pour résoudre ce problème.

Historiquement, la relation entre la production et la maintenance a été de nature conflictuelle. Cette attitude est perpétuée par le manque de communication entre les deux services (Weinstein and Chung, 1999). Ces conflits ont comme conséquence la non satisfaction des clients due aux interruptions causées par les actions de MP ou la panne de machines due au non respect, par le service ordonnancement, des périodes prévues pour la maintenance. Pour éviter ces conflits, nous proposons dans ce papier un modèle d'optimisation multi objectif intégré, prenant en compte la fiabilité des machines pour l'aspect maintenance. Ce modèle permet de générer des solutions "compromis" satisfaisant au mieux deux critères, l'un lié à la production et l'autre lié à la maintenance. Le reste du papier est organisé comme suit: la section 2 donne un état de l'art des travaux d'ordonnancement prenant en compte la maintenance préventive. La section 3 décrit la formulation du problème intégré d'ordonnancement de la production et de planification de la maintenance et le mode d'évaluation de la disponibilité (ou l'indisponibilité) du système. La section 4 présente les procédures de l'algorithme évolutionnaire proposé. Les résultats expérimentaux apparaissent dans la section suivante. La conclusion et les extensions du problème sont résumées dans la section 6.

2. ETAT DE L'ART

Les travaux de recherche qui prennent en compte la maintenance dans l'ordonnancement de la production peuvent être divisés en deux catégories: l'approche déterministe (ou séquentielle) et l'approche stochastique (ou intégrée).
Dans l'approche déterministe (séquentielle), les périodes des tâches de MP ainsi que leur nombre sont connus et

fixés à l'avance et elles constituent une contrainte pour optimiser l'ordonnancement des tâches de production. La plupart des travaux de recherche prenant en considération la maintenance adoptent cette approche et sont communément appelés "scheduling with machine availability constraints". (Schmidt, 2000) et (Lee, 1996) ont fait des investigations détaillées sur ce problème en analysant plusieurs configurations pour différentes mesures de performance et de contraintes.

Dans l'approche stochastique (intégrée), les dates de début des actions de maintenance sont considérées comme des variables de décision du problème. Les tâches de production et de maintenance sont ordonnancées simultanément. Les travaux traitant le problème selon cette approche sont rares. (Cassady and Kutanoglu ,2003) ont proposé un modèle intégré dans le cas d'une seule machine. L'approche par énumération totale est utilisée pour minimiser le retard total pondéré de la production, ce qui n'est pas pratique. (Ruiz et *al*., 2007) ont considéré le problème dans le cas d'un flow shop de permutation pour minimiser le makespan. Les auteurs ont utilisé et comparé plusieurs heuristiques. Ils ont utilisé des modèles de fiabilité pour déterminer les meilleurs intervalles des périodes de maintenance en se basant sur le maintien d'un niveau minimal de fiabilité. Cependant, ces intervalles de maintenance sont déterminés en ignorant les exigences de production. Ils sont fixés à l'avance pour chaque machine séparément, donc les fiabilités des machines sont considérées comme contraintes pour optimiser un seul critère lié à la production. (Kaabi *et al.*, 2002) (*resp.* (Kaabi *et al*., 2003)) ont étudié le problème à une machine (*resp.* de flow shop) où les périodes de maintenance doivent se dérouler dans un intervalle de tolérance sans faire intervenir des modèles de fiabilité dans leur modèle. Des heuristiques basées sur le Branch & Bound et la recherche locale (*resp*. les algorithmes génétiques) ont été développés pour le problème à une seule machine (*resp*. de flow shop). La fonction objectif est la somme pondérée de deux critères: le retard total pour l'aspect production et la somme des retards et des avances pour l'aspect maintenance préventive. C'est le seul travail ayant tenté de considérer conjointement un critère lié à la production et un autre lié à la maintenance. Le modèle adopté pour évaluer une solution est la somme pondérée des critères. Cette méthode est facile à implémenter en transformant le problème en un problème d'optimisation monocritère. Malheureusement, le modèle tel qu'il est appliqué dans ces deux études (en particulier, les poids sont fixes durant l'exécution de l'algorithme) peut présenter plusieurs inconvénients: la seule solution obtenue dépendra des poids des critères, les régions concaves du front Pareto restent inaccessibles même si on change les valeurs des poids, etc.

A notre connaissance, il n'existe pas de travaux pour le cas de machines parallèles, encore moins traitant le problème selon une approche évolutionnaire prenant en considération la fiabilité des machines comme critère de performance du système de production.

Dans cet article, nous proposons un modèle intégré bi-objectif pour machines parallèles utilisant la fiabilité du système de production comme objectif pour l'aspect maintenance. Un algorithme Pareto évolutionnaire sera appliqué pour générer un ensemble de solutions de compromis minimisant à la fois deux critères importants dans les systèmes de production : le makespan et l'indisponibilité du système au sens fiabilité. Les intervalles des périodes de maintenance sont optimisés au cours du processus d'optimisation global au même titre que la séquence de production et l'indisponibilité du système. Au lieu de fixer un seuil de fiabilité pour chaque machine séparément, c'est la disponibilité du système qui est considérée.

3. MODELISATION DU PROBLEME INTEGRE

Cette section décrit la formulation du problème d'ordonnancement de la production et du problème de planification de la maintenance préventive systématique séparément, ensuite le problème intégré bi- objectif.

3.1. Le problème d'ordonnancement de la production

De point de vue production, nous considérons le problème d'ordonnancement sur machines parallèles avec minimisation du makespan (le makespan est le temps de fin de la tâche qui se termine la dernière). Nous supposons que les tâches sont disponibles au début de la période de production et leur préemption n'est pas permise. Ce problème est classé NP- difficile (Garey and Johnson, 1979).

3.2. Le problème de planification de MP

Dans les systèmes manufacturiers, les actions de maintenance préventive aident à maintenir les outils de production en bon état de fonctionnement (elles accroîssent la disponibilité) et permettent de réduire les coûts en évitant des pannes fortuites. La *disponibilité* d'une entité *E* est définie comme son "*aptitude à être en état d'accomplir une fonction requise dans des conditions données, à un instant donné ou pendant un intervalle de temps donné, en supposant que la fourniture de moyens extérieurs nécessaires soit assurée*" [Norme NF X 60-500]. Elle est en général définie en terme probabiliste comme suit à un instant t :

$$A(t) = P(E \text{ non défaillant à l'instant } t) \quad (1)$$

L'aptitude contraire de la disponibilité est l'*indisponibilité* définie par :

$$\bar{A}(t) = 1 - A(t) \quad (2)$$

L'expression de la disponibilité d'une machine M_i dépend de son taux de défaillance λ_i et de son taux de réparation μ_i. Nous nous limiterons ici à des machines dont le taux de défaillance λ_i et le taux de réparation μ_i sont cons-

tants. Autrement dit, nous supposons que le temps jusqu'à défaillance (*resp*. du temps jusqu'à réparation) de la machine M_i est gouverné par la loi de probabilité exponentielle de paramètre λ_i (*resp*. μ_i). Nous supposons aussi qu'après intervention la machine devient neuve ("as good as new"). A partir de ces hypothèses, à partir de l'état initial t=0, la disponibilité de la machine M_i à un instant t est donnée par l'expression suivante (Ebeling, 1997; Villemeur, 1988):

$$A_i(t) = \frac{\mu_i}{\lambda_i + \mu_i} + \frac{\lambda_i}{\lambda_i + \mu_i} \exp[-(\lambda_i + \mu_i)t] \qquad (3)$$

La disponibilité $A_i(t)$ d'une machine M_i étant une fonction décroissante du temps, l'indisponibilité $1 - A_i(t)$ est donc une fonction croissante. Par conséquent, si aucune action de MP n'est effectuée sur M_i, son indisponibilité va croître. Si T est la date de fin de l'action de MP sur la machine M_i, sa disponibilité $A_i(t)$ à l'instant t est donnée par l'expression suivante (Ebeling, 1997; Villemeur, 1988):

$$A_i(t) = \frac{\mu_i}{\lambda_i + \mu_i} + \frac{\lambda_i}{\lambda_i + \mu_i} \exp[-(\lambda_i + \mu_i)(t - T)] \qquad (4)$$

La disponibilité d'un système dépend de sa propre structure (parallèle, série, hybride) mais aussi des caractéristiques de ses éléments. Pour m composants indépendants disposés en parallèle, chacun ayant une disponibilité $A_i(t)$ à l'instant t, la disponibilité du système au temps t est donnée par l'expression (Ebeling, 1997; Villemeur, 1988) :

$$A_S(t) = 1 - \prod_{i=1}^{i=m} [1 - A_i(t)] \qquad (5)$$

Par conséquent, l'indisponibilité du système est :

$$\overline{A}_S(t) = 1 - A_S(t) = \prod_{i=1}^{i=m} [1 - A_i(t)] \qquad (6)$$

3.3. Le problème intégré bi- objectif

Deux objectifs sont à minimiser simultanément dans notre étude : la minimisation du makespan pour l'aspect production et la minimisation de l'indisponibilité du système de production pour l'aspect maintenance, sous les conditions définies dans les sections 2.1 et 2.2. Donc, deux décisions doivent être prises en même temps. La première est de trouver la meilleure affectation des n tâches de production aux machines dans l'objectif de minimiser le makespan. La seconde est de décider quand il faut effectuer les interventions de MP sur les machines, dont leur nombre n'est pas fixé à l'avance, dans l'objectif de minimiser l'indisponibilité du système.

Soit C_i la date de fin de la tâche de production numéro i.
Soit C_{max} la date de fin de la dernière tâche de production (le makespan).

$$C_{\max} = max_{i=1,n}\{C_i\}$$

Soit $T = \{0, t_1, t_2, \ldots, t_s, C_{max}\}$ où $t_1, t_2, \ldots, t_s$ sont les dates de début des actions de MP sur toutes les machines. Puisque l'indisponibilité est une fonction croissante dans chaque intervalle $[t_i, t_{i+1}]$, i=0, … , s, avec t_0=0 et t_{s+1}= C_{max},et comme nous avons supposé qu'une machine devient aussi bonne que neuve("as good as new") à la fin de chaque action de MP, l'indisponibilité du système est calculée uniquement aux instants $t_1, t_2, \ldots, t_{s+1}$.
Les deux fonctions objectif à minimiser pour m machines parallèles sont :

$F_1 = C_{\max}$, qui est le Makespan.

$F_2 = \max_{t \in T} \{\overline{A}_S(t) = \prod_{i=1}^{i=m} [1 - A_i(t)]\}$, qui est l'indisponibilité du système.

La durée d'une tâche de MP sur une machine M_i est supposée égale au temps moyen d'une action de MP dont la valeur est équivalente à $1/\mu_i$ (Adzakpa *et al*., 2004) .

4. METHODE DE RESOLUTION

Pour la méthode de résolution, nous avons adopté un algorithme évolutionnaire élitiste de type Pareto basé sur NSGA-II (Non dominated Sorting Genetic Algorithm). Ce choix est justifié par le fait que plusieurs études dans la littérature classent NSGA-II comme l'un des algorithmes évolutionnaires les plus compétitifs en optimisation multicritère et il a été utilisé dans divers domaines en pratique.

4.1. L'algorithme NSGA-II

NSGA-II est un algorithme évolutionnaire élitiste de type Pareto. Il calcule une approximation de l'ensemble des solutions non dominées appelé aussi *front Pareto* ou surface de compromis, en se basant sur le concept de non dominance.

Principe général
Au départ, on génère aléatoirement la population initiale P_0 de taille N. On trie P_0 en plusieurs fronts F_i (*le ranking*) selon le critère de non dominance. A chaque solution est assignée une valeur d'adaptation égale à son niveau de non dominance (1 est le meilleur niveau). La sélection par tournoi binaire, les opérateurs de croisement et de mutation sont ensuite utilisés pour créer une population enfant Q_0 de taille N. Pour une génération $t \geq 1$, la procédure change.
La première phase consiste à créer la population $R_t = P_t \cup Q_t$ de taille $2N$ et à appliquer la procédure de *ranking* pour identifier la liste des fronts F_i de solu-

tions non dominées. La deuxième phase consiste à construire une nouvelle population parente P_{t+1} contenant les N meilleures solutions de R_t en y incluant les meilleurs fronts tant que le nombre de solutions dans P_{t+1} est inférieur à N. Pour compléter P_{t+1} avec les N-| P_{t+1} | solutions restant à inclure dans P_{t+1}, la procédure de *crowding* est appliquée au premier front F_i non inclus. La population P_{t+1} est ensuite utilisée pour la sélection, le croisement et la mutation afin de créer une nouvelle population Q_{t+1} de taille N. On peut trouver Plus de détails sur les différentes procédures dans (Deb *et al*., 2000).

4.2. Composants de NSGA-II pour notre problème

4.2.1 Représentation d'une solution

Une solution (un chromosome) se compose de deux segments: la partie production et la partie maintenance. La partie production est une séquence des numéros des tâches. La partie maintenance est une suite de nombres représentant les périodes de MP des machines.

4.2.2 Evaluation d'une solution

L'évaluation d'un chromosome se fait par l'algorithme 4 suivant la stratégie list scheduling: affecter la tâche suivante à la première machine disponible (libre).

Algorithme 4 : EvaluerSolution

1. Affecter les tâches de production aux machines (selon list scheduling)
2. Calculer le *makespan* correspondant C_{max}.
3. Insérer les tâches de MP dans l'ordonnancement de production en évitant la préemption des tâches de production, avec mise à jour des dates des tâches de production.
4. Réévaluer le *makespan*.
5. Evaluer l'indisponibilité du système, égale à la plus grande valeur d'indisponibilité du système rencontrée durant l'horizon d'ordonnancement.

Concernant les actions de MP, si la date d'intervention prévue ne coïncide pas avec la date de fin d'une tâche de production, alors l'action de MP est retardée à la fin de cette tâche de production pour éviter la préemption. L'exemple suivant illustrera l'évaluation d'un chromosome.

Exemple

Soit le système de production composé de deux machines parallèles sur lesquelles huit tâches doivent être ordonnancées. Les caractéristiques des tâches de production sont :

Tâche n°	1	2	3	4	5	6	7	8
Durée	4	6	8	10	12	14	16	18

On suppose que les deux machines ont les mêmes caractéristiques : $\lambda_1 = \lambda_2 = 0.1$, $\mu_1 = \mu_2 = 0.5$. Donc, la durée des actions de MP sur les deux machines est égale à 2 unités de temps.

Soit le chromosome suivant à évaluer :

5	4	6	8	7	3	1	2	**10**	**18**

Les huit premiers allèles déterminent l'ordonnancement des tâches de production. Les deux derniers allèles (10,18) représentent la partie maintenance qui indiquent qu'une intervention sur la machine M_1 doit être effectuée à priori chaque 10 unités de temps et sur la deuxième machine M_2 chaque 18 unités de temps.

L'application de l'algorithme 4 au chromosome de l'exemple suit les étapes suivantes :

Etape1. *Ordonnancer les tâches de production.*

L'heuristique list scheduling donne l'ordonnancement montré figure 1.

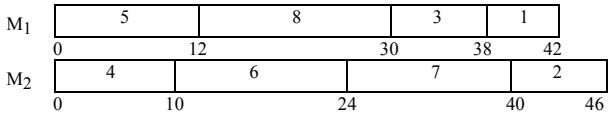

Figure1. Ordonnancement de la production sans les actions de PM

Etape2. *Calcul du makespan: $C_{max} = 46$.*

Etape3. *Insérer les actions de MP et mettre à jour les dates de production* (voir figure 2).

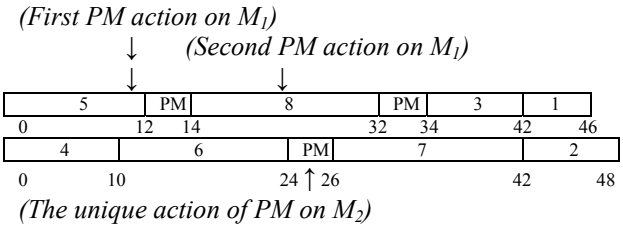

Figure 2. Ordonnancement simultané des tâches de production et des tâches de PM

Les actions de maintenance débutent aux dates 12 et 32 pour la machine M_1 et à la date 24, pour la machine M_2.

Etape4. *Recalculer le makespan: $C_{max} = 48$.*
Etape5. *Calculer l'indisponibilité du système aux instants :12, 24, 32 et C_{max}.*

L'indisponibilité du système correspond à la plus grande valeur parmi celles obtenues aux quatre instants. Tableau 1 (*resp.* Tableau 2) donne l'indisponibilité (*resp*. la disponibilité) de chaque machine aux quatre instants. L'indisponibilité (*resp*. la disponibilité) du système est calculée selon la formule (6) (*resp*. (5)). Donc, l'indisponibilité (*resp*. la disponibilité) du système est égale à 0.0809 (*resp.* à 0.9190).

t	$\bar{A}_1(t)$	$\bar{A}_2(t)$	$\bar{A}_S(t)$
12	0.2814	0.2814	0.0792
24	0.2770	0.2856	0.0791
32	0.2851	0.2507	0.0715
48	0.2835	0.2855	0.0809

Tableau 1. Indisponibilités des machines et du système

t	$A_1(t)$	$A_2(t)$	$A_S(t)$
12	0.7185	0.7185	0.9207
24	0.7229	0.7143	0.9208
32	0.7148	0.7492	0.9284
48	0.7164	0.7144	0.9190

Tableau 2. Disponibilités des machines et du système

Le vecteur des fonctions objectif pour cette solution est $(F_1, F_2) = (48, 0.0809)$.

4.2.3 Stratégies de reproduction

La population initiale est générée aléatoirement. Les procédures de *ranking* et de *crowding* sont utilisées par NSGA-II à différents niveaux de l'algorithme pour assurer l'élitisme et la diversité au cours de la reproduction. La condition d'arrêt est le nombre de générations fixé.

4.2.3.1 Opérateur de croisement

La méthode de croisement à deux points est implémentée comme proposé dans (Ishibuchi and Murata, 1998). Le premier (*resp.* le second) fils est généré en conservant les extrémités de chromosome du premier parent et en complétant avec les tâches manquantes suivant leur ordre d'apparition dans le deuxième (*resp.* le premier) parent, tel qu'il est illustré sur figure 3. Pour la partie maintenance, on a opté pour le croisement à un point.

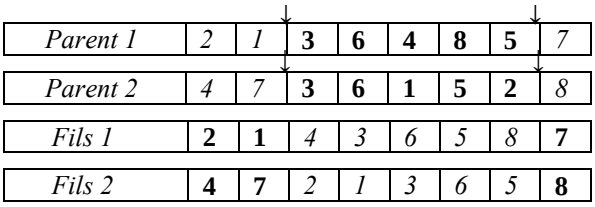

Figure 3. Croisement à deux points

4.2.3.2 Opérateur de mutation

Pour la partie maintenance du chromosome, un gène est choisi au hasard et sa valeur est remplacée par une valeur générée aléatoirement. Pour la partie production, la mutation- décalage à droite est utilisée comme illustré figure 4, où une tâche choisie aléatoirement est déplacée et insérée à une position sélectionnée aléatoirement. La structure de voisinage définie par la mutation- décalage a été souvent utilisée pour les algorithmes de recherche tabou et de recuit simulé (Ishibuchi and Murata, 1998).

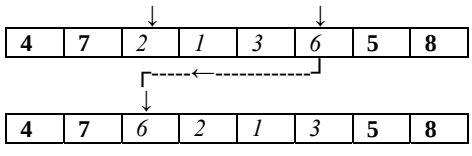

Figure 4. Mutation décalage- insertion

5. TESTS ET RESULTATS

5.1. Mesures de performance

Nous avons utilisé deux métriques pour distinguer entre deux fronts: la métrique d'hyper volume *H* et la métrique *C* introduites par (Zitzler, 1999). La métrique *H* calcule une approximation du volume compris sous la courbe formée par le front à évaluer. Lorsque les surfaces de compromis comportent le même nombre de points, plus la valeur d'hyper volume est petite, meilleure sera la surface de compromis. Dans le cas bidimensionnel (bi- objectif), le volume *H* est l'union des rectangles définis par les points (0, 0) et $(f_1(x_i), f_2(x_i))$, pour un vecteur de décision x_i (i= 1, n). La métrique *C* est une mesure relative qui permet de différencier nettement deux fronts *A* et *B* lorsque d'autres métriques (l'hyper volume par exemple) donnent des évaluations trop proches. La valeur *C (A, B)* correspond au pourcentage d'éléments de *B* faiblement dominés par au moins un des éléments de *A*. Elle est calculée à l'aide de la formule (7):

$$C(A,B) = \frac{\left|\{b \in B / \exists a \in A : a \prec b\}\right|}{|B|} \qquad (7)$$

5.2. Choix des paramètres de NSGA-II

Un choix judicieux des paramètres de l'algorithme génétique, tels que la taille de la population, les taux de croisement et de mutation ainsi que le nombre de générations permet d'obtenir des solutions de qualité. Il n'existe pas de valeurs typiques acceptées universellement pour tous ces paramètres, mais la majorité des études utilisant les algorithmes génétiques montrent une certaine tendance pour certaines valeurs sur la base d'expérimentations. Par exemple, le taux de croisement est souvent choisi supérieur à 70 % et le taux de mutation est généralement choisi inférieur à 10 % par bit et moins de 50 % par chromosome. Pour nos expériences, nous fixons le taux de croisement à 0.8 comme dans (Deb *et al*., 2000) et (Amodeo *et al*., 2007). Le taux de mutation est fixé à 0.3 pour la production et à 0.01 pour la partie maintenance comme dans la plupart des applications des AGs aux problèmes de permutation (Ishibuchi and Murata, 1998). Il reste à déterminer des valeurs appropriées pour la taille de la population et le nombre de générations qui donneront de bonnes solutions en un temps raisonnable.

Pour ce faire, les deux métriques *H* et *C* ont été utilisées. La métrique *H* étant une mesure absolue, nous avons étudié son évolution en fonction du nombre de générations, la taille de la population étant fixée (figure 5). Par contre, puisque la métrique *C* est une mesure relative, elle a été utilisée pour mesurer la contribution marginale de *50* générations supplémentaires. Autrement dit, dans le calcul de *C (A, B), A* est le front obtenu jusqu'à une génération *t* et *B* est celui obtenu à la génération t+50, en utilisant la même population initiale (figure 6). La figure

7 montre l'évolution de la métrique *C* en fixant *A* à 20 et faisant varier *B* à pas de 50, pour différentes valeurs de la taille de la population.

Nous avons utilisé les diagrammes de dispersion (*box plots*) pour visualiser la distribution des données d'un échantillon. Ces diagrammes permettent de suivre et de visualiser simultanément plusieurs caractéristiques et comparer plusieurs échantillons de données : la médiane Q_2 de la série de données (la barre à l'intérieur du rectangle), le premier quartile Q_1 (la barre inférieure du rectangle), le troisième quartile Q_3 (la barre supérieur du rectangle), l'intervalle inter- quartile Q_3 - Q_1, la valeur maximale non aberrante de la série (la ligne fine en haut), la valeur minimale non aberrante de la série (la ligne fine en bas) et les valeurs aberrantes qui apparaissent ici en astérisques et en cercles. Dans notre étude expérimentale, nous nous sommes intéressés à la tendance générale des métriques *H* et *C* comme variables en fonction du nombre de générations et de la taille de la population. Comme dans (Zitzler, 1999), 30 exécutions indépendantes ont été effectuées.

Les différents algorithmes utilisés ont été codés en C++ sous Windows et exécutés sur un Pentium 4 (avec 2.93 GHz et 256 Mo de RAM).

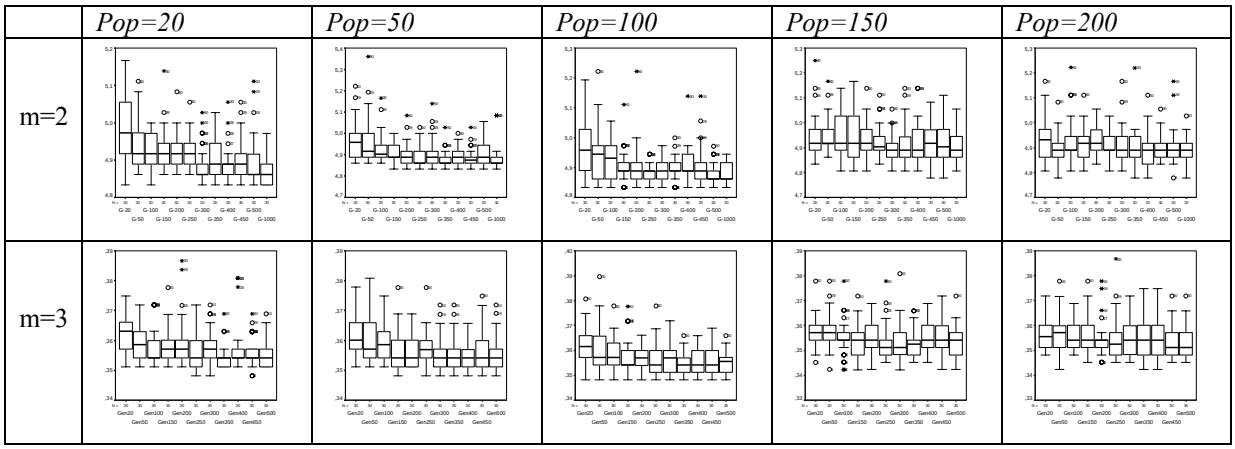

Figure 5. Evolution du volume H selon le nombre de générations (20, 50, 100, 150, 200, 250, 300,350, 400, 450, 500, 1000) pour différentes tailles de la population (20, 50, 100, 150 et 200), et le nombre de machines m égale à 2 et3

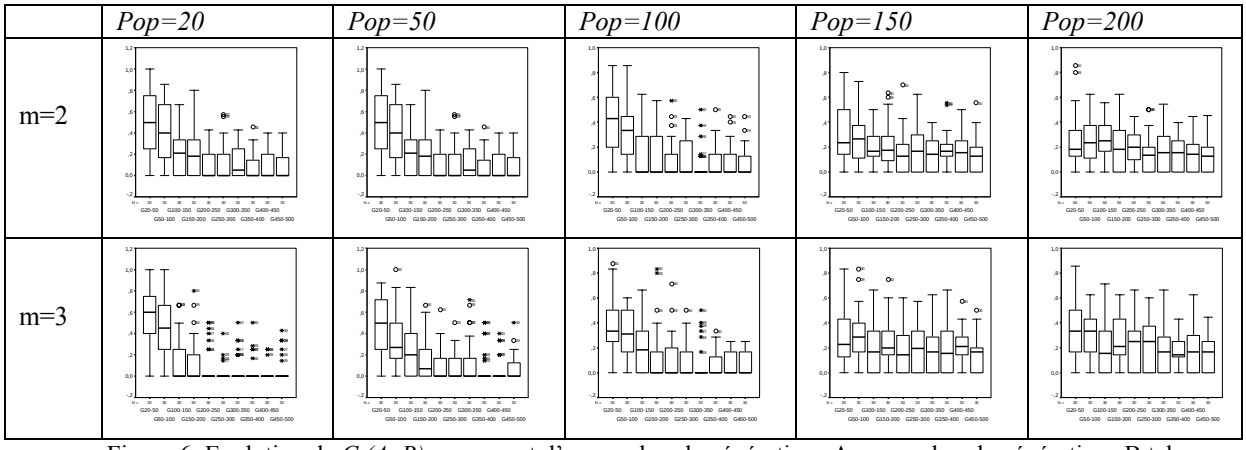

Figure 6. Evolution de *C (A, B)* en passant d'un nombre de générations A au nombre de générations B tels que (A,B)∈ {(20,50), (50,100), (100,150), (150,200), (200,250), (250,300), (300,350), (350,400), (400,450), (450,500)} pour différentes tailles de la population (20, 50, 100, 150 et 200), et le nombre de machines m égale 2 et3

La figure 5 montre qu'à partir de la taille *n*=150, la métrique *H* commence à converger et à se stabiliser et devient encore plus stable pour *n*= 200, que ce soit en termes de décroissance ou de dispersion. A partir de la figure 6, on peut remarquer qu'à partir de la génération *t*=150, il n'y a pratiquement plus d'amélioration en termes de la métrique *C*, pour les différentes tailles de la population.

Nous remarquons aussi, d'après la figure7, que la mesure *C* ne commence à se stabiliser qu'en passant de la génération 20 à la génération 150. Ainsi, pour obtenir des résultats de meilleure qualité, une taille de 200 pour la population et un nombre de générations de 200 semblent appropriés.

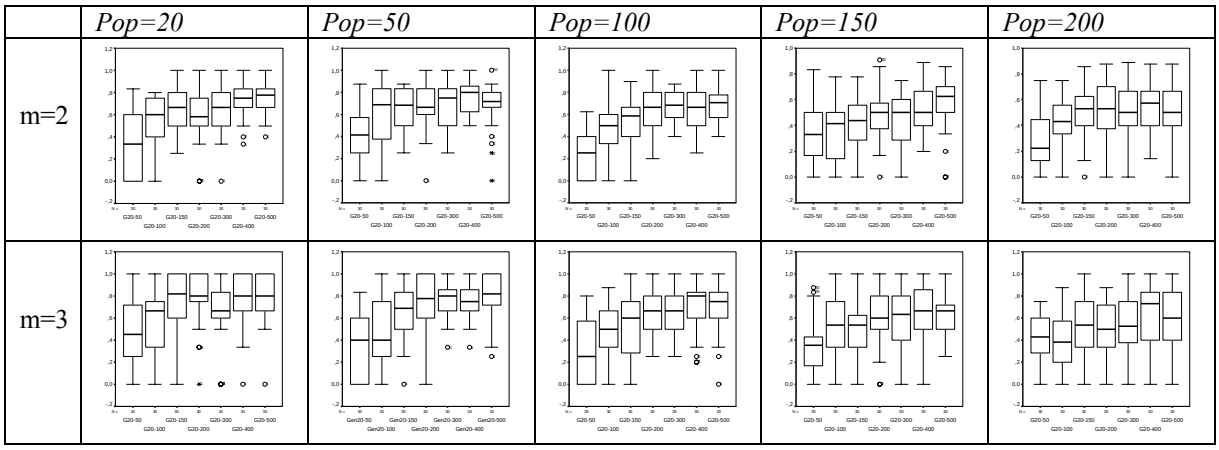

Figure 7. Evolution de C (A, B) de A=20 à B =50, 100, 150, 200, 300, 400 et 500 générations pour différentes tailles de la population (20, 50, 100, 150 et 200), et le nombre de machines m égale à 2 et3

6. CONCLUSION ET PERSPECTIVES

Dans cet article nous avons étudié le problème d'ordonnancement conjoint de la production et de la maintenance selon une approche évolutionnaire bi-objectif de type Pareto, dans le cas de machines parallèles, optimisant deux critères à la fois. Le premier critère est lié aux activités de production, qui est le makespan, et l'autre objectif est lié à la maintenance préventive systématique, qui est l'indisponibilité du système de production. Des modèles de fiabilité ont été utilisés pour évaluer l'indisponibilité du système. Nous avons proposé un algorithme génétique basé sur NSGA-II adapté à notre problème. Afin d'assurer de meilleurs résultats, une étude expérimentale a été menée pour sélectionner les meilleurs valeurs des paramètres de l'algorithme génétique. La méthode permet au décideur d'avoir entre les mains un ensemble de solutions compromis efficaces. Dans le futur proche, nous pensons à l'étude du problème en prenant en considération d'autres critères, liés à la production ou la maintenance. Nous comptons aussi inclure dans notre modèle des contraintes liées aux deux aspects. D'autres structures de systèmes de production plus complexes (flow shop hybride ou job shop) peuvent être envisagés. On pense aussi à l'hybridation de NSGA II avec d'autres heuristiques.

REFERENCES

Adzakpa, K.P., K.H. Adjallah and F. Yalaoui, 2004. On-line maintenance job scheduling and assignment to resources in distributed systems by heuristic-based optimization. *Journal of intelligent manufacturing*, 15, p.131-140.

Amodeo, L., H. Chen and A. El Hadji, 2007. Multi-objective Supply Chain Optimization: An Industrial Case Study. Springer-Verlag, EvoWorkshops, LNCS 4448, 732--741.

Cassady, C. R., and E. Kutanoglu, 2003. Minimizing job tardiness using integrated preventive maintenance planning and production scheduling, *IIE Transactions,* 35, p. 503-513.

Deb, K. S. Agrawal, A. Pratab, and T. Meyarivan, 2000. A Fast Elitist Non Dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II. KanGAL report 200001, Indian Institute of Technology, Kanpur, India.

Ebeling, C. E., 1997. *An introduction to reliability and maintainability engineering*. McGraw-Hill publisher, USA.

Garey, M. R, and D. S Johnson, 1979. *Computers and Intractability: A guide to the theory of NP-Completeness.* W.H. Freeman and Company publishers, San Francisco, California.

Goldberg, D.E., 1989. *Genetic algorithms in search, optimization and machine learning.* Addison-Wesley Publishers.

Ishibuchi, H., and T. Murata, 1998. A multi-objective genetic local search algorithm and its application to flow shop scheduling. *IEEE Transactions on Systems, Man and Cybernetics*, 28, p.392-403.

Kaabi, J., C. Varnier, and N. Zerhouni, 2002. Heuristics for scheduling maintenance and production on a single machine. *IEEE Conference on Systems, Man and Cybernetics.* October 6-9 Hammamet, Tunisia.

Kaabi, J., C. Varnier, and N. Zerhouni, 2003. *Genetic algorithm for scheduling production and maintenance in a Flow Shop*. Laboratoire d'automatique de Besançon. France.

Lee, C-Y., 1996. Machine scheduling with an availability constraint. *Journal of Global Optimization,* 9, p.395-416.

Ruiz, R., J. Carlos Garcia-Diaz, and C. Maroto, 2007. Considering scheduling and preventive maintenance in the flow shop sequencing problem. *Computers and Operations Research*, 34(11), p.3314-3330.

Schmidt, G., 2000. Scheduling with limited machine availability. *European Journal of Operational Research*, 121, p.1-15.

Villemeur, A., 1988. *Sûreté de fonctionnement des Systèmes industriels*, Eyrolles, Paris.

Weinstein, L., and C-H. Chung, 1999. Integrated maintenance and production decisions in hierarchical production planning environment. *Computers and Operations Research*, 26, p.1059-1074.

Zitzler, E., M. Laumanns and L. Thiele, 2002. *SPEA2: improving the strength pareto evolutionary algorithm for multi-objective optimization*. Swiss Federal Institute of Technology, Zurich.

Zitzler, E., 1999. *Evolutionary algorithms for multi-objective optimization: methods and applications.* PhD thesis, Swiss Federal Institute of Technology, Zurich.