7e *Conférence Francophone 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»

ORDONNANCEMENT DE TÂCHES DE RÉGLAGE SUR DES OPÉRATEURS MULTI-COMPÉTENTS: VALIDATION DE MODÈLE PAR SIMULATION

Cédric Pessan (1, 2) **Emmanuel Néron (1)**

(1) Laboratoire d'Informatique de l'Université de Tours
Polytech'Tours
64 av. Jean Portalis
37200 Tours
cedric.pessan@univ-tours.fr
emmanuel.neron@univ-tours.fr

(2) SKF France SA
Industrial Division / MDGBB
64 av. Charles de Gaulle
37540 Saint-Cyr-sur-Loire

RÉSUMÉ : *L'optimisation de changements de série est un élément clé de la flexibilité de la production et en particulier pour les chaînes de production du site SKF de Saint-Cyr-sur-Loire sur lesquels notre étude est basée. Une manière d'optimiser les changements de série est d'ordonnancer optimalement les tâches de réglage de toutes les machines d'une ligne de production sur les opérateurs. Ces tâches de réglage nécessitent un haut niveau de compétences pour pouvoir être terminées rapidement. C'est pourquoi un opérateur en fonction de son expérience prendra plus ou moins de temps pour effectuer une tâche. Dans notre modélisation du problème d'optimisation des changements de série de l'usine SKF, nous avons modélisé les temps opératoires en utilisant les temps moyens de réglage de chaque opérateur à partir de données observées sur plusieurs mois. D'autre part, pour résoudre ce problème nous avons développé un algorithme mémétique. Dans ce papier, nous montrons à l'aide d'une approche par simulation que l'hypothèse faite sur les temps opératoires est valide sur les ordonnancements obtenus par notre méthode de résolution.*

MOTS-CLÉS : *Ordonnancement, machines parallèles non reliées, validation par simulation, algorithme mémétique*

1. INTRODUCTION

L'usine SKF de Saint-Cyr-sur-Loire produit des roulements à bille en grande série, chaque série de roulements ayant ses caractèristiques propres telles que: le diamètre des roulements, l'épaisseur... Il est donc nécessaire de changer l'outillage des machines lorsque l'on veut passer d'une série à l'autre. Or, la flexibilité de la production est un des enjeux majeurs de la production puisqu'elle permet de réduire la taille des stocks et de répondre plus rapidement aux demandes des clients. Pour obtenir cette flexibilité, il est notamment nécessaire de réduire au maximum les pertes dues aux changements de séries nécessaires lorsque l'on passe d'un batch à l'autre sur une ligne de production. Or, un changement de série est composé de plusieurs opérations de réglage sur chaque machine qui composent une ligne de production.

Deux approches complémentaires peuvent être considérées pour réduire le temps de changement de série. Soit, on réduit les temps opératoires de chaque opération, soit on essaie d'ordonnancer optimalement les opérations de réglage sur les opérateurs de manière à utiliser au mieux leurs compétences tout en respectant leurs périodes de disponibilité. La réduction des temps opératoire des opérations de réglage a été largement étudiée et formalisée dans la méthode SMED: Single Minute Exchange of Die (Shingo 1985). Dans notre approche, nous supposons donc que la méthode SMED est déjà appliquée et nous nous proposons d'optimiser l'affectation des tâches de réglage sur les opérateurs de sorte à organiser au mieux le changement de série.

De nombreux problèmes d'ordonnancement prennent en compte des temps de réglage des moyens de production dépendants ou non de la séquence des séries, on peut se réferrer notamment à l'état de l'art sur ces problèmes proposé par (Allahverdi, Gupta & Aldowaisan 1999). Certains de ces problèmes considèrent que les temps de réglage nécessitent l'utilisation

d'un serveur qui peut être par exemple un opérateur (Kravchenko & Werner 1997, Brucker, Knust & Wang 2005). Le problème que nous traitons consiste à ordonnancer plus précisément les tâches de réglage en elles-mêmes à l'intérieur d'une période de réglage: non seulement nous considérons qu'il y a plusieurs ressources nécessaires pour passer d'une série à une autre, mais nous décomposons cette opération en tâches qui doivent être ordonnancées sur des opérateurs.

Les tâches de réglage sont des opérations relativement délicates à réaliser nécessitant des compétences particulières. Le temps de réglage d'une machine dépend donc beaucoup de l'expérience de l'opérateur. C'est pourquoi, dans la modélisation que nous proposons de ce problème d'affectation de tâches de réglage, nous avons considéré que les temps opératoires d'un opérateur étaient les moyennes de ses temps de réglage observés sur les derniers mois sur chaque machine.

Dans cette étude, nous validons à l'aide de la simulation l'hypothèse que nous avons faite sur les temps opératoires. Nous cherchons en particulier à vérifier si le critère ne varie pas significativement lorsque les temps opératoires s'écartent de la moyenne. De telles approches ont déjà été utilisées avec succès pour valider la robustesse d'ordonnancements obtenus par des approches proactives (Billaut, Moukrim & Sanlaville 2005). Comme évoqué précedemment, le problème que nous étudions est tiré d'un cas réel rencontré dans les usines SKF. C'est pourquoi nous avions la possibilité d'utiliser des données réelles pour générer des aléas, utilisés en simulation, et ainsi confronter les ordonnancements obtenus par nos méthodes à la présence d'aléas réalistes.

Dans la section 2, nous définirons le problème industriel puis la modélisation que nous proposons. Dans la section 3, nous présentons une méthode de résolution pour ce problème que nous avons développée dans (Pessan, Néron & Bellenguez-Morineau 2006). Dans la section 4, nous présentons la simulation qui nous permettra de valider la modélisation du problème sur les solutions proposées par cette méthode et dans la section 5, l'application permettant de simuler les aléas est présentée.

2. PRÉSENTATION DU PROBLÈME

Dans cette section, nous commençons par présenter le problème tel qu'il est rencontré sur les lignes de production du groupe SKF. Ensuite, nous modélisons ce problème sous forme d'un problème d'ordonnancement à machines parallèles non reliées et enfin, nous présentons les hypothèses faites sur le problème nous permettant de considérer celui-ci comme un problème d'affectation

2.1. Cas industriel: l'exemple SKF

Une ligne de production est un agencement de machines dans une structure série parallèle. En effet, une ligne est composée de plusieurs étages en série par lesquels chaque pièce doit passer et certains étages comportent plusieurs machines en parallèle pour augmenter la productivité de l'étage. Il est donc nécessaire d'avoir une machine de chaque étage en fonctionnement pour pouvoir produire (cf. diagramme du haut de la figure 1.

Pendant un changement de série, il est nécessaire de stopper chaque machine pendant la durée du réglage à effectuer en vue de poduire la nouvelle série. L'entreprise cherche donc à minimiser la perte de production provoquée par ces arrêts machines.

Lorsque l'on redémarre les machines, il est possible de produire si au moins une machine de chaque étage est réglée. Il est donc possible d'augmenter la productivité au fur et à mesure que les machines en double sur les étages les plus lents sont réglées. Dans ce contexte, l'objectif n'est donc pas de régler toutes le machines au plus tôt mais de produire un maximum de pièces pendant une période de temps qui contient un changement de série.

Pour pouvoir régler une machine, il faut que la dernière pièce du lot courant soit déjà passée par l'étage de cette machine. Autrement dit, il ne doit plus y avoir dans la ligne de pièces susceptibles d'être traitées par la machine.

De plus, la productivité est mesurée à la fin de la ligne: pour qu'une machine nouvellement réglée puisse avoir un impact sur la productivité, il faut au moins que la première pièce traitée par cette machine ait eu le temps d'arriver au bout de la ligne.

Les tâches de réglage des machines doivent verifier les contraintes suivantes:

- il y a une et une seule tâche de réglage par machine
- un seul opérateur est nécessaire pour effectuer une tâche
- une tâche commencée ne peut pas être interrompue (pas de préemption)
- un opérateur ne peut effectuer qu'une seule tâche à la fois.

De plus, les opérateurs ne sont pas tous disponibles

en même temps, on ne peut donc leur affecter une tâche que pendant leurs périodes de disponibilité.

Dans la section suivante, nous allons voir comment ce problème peut être modélisé par un problème à machines parallèles non reliées.

2.2. Problème à machines parallèles

Le problème peut s'identifier comme un problème à machines parallèles non reliées qui peuvent exécuter divers types de tâches. Les machines au sens des ressources disjonctives du problème d'ordonnancement représentent les opérateurs et les tâches les machines à régler. De plus, toutes les machines n'ont pas la même période de disponibilité.

Nous notons n le nombre de tâches: il y a une tâche par machine $\mathcal{M}_i, i \in [1, n]$ à régler. Nous définissions également $n_1 \leq n$ le nombre de machines prioritaires nécessaires pour démarrer la production. Les machines indexées de 1 à n_1 sont les machines prioritaires, les autres machines étant indexées de $n_1 + 1$ à n.

r_i est la distance en temps que met un roulement pour passer de $\mathcal{M}_1$ à $\mathcal{M}_i$. C'est aussi la date de début au plus tôt de $\mathcal{M}_i$.

q_i est la distance en temps que met un roulement pour passer de $\mathcal{M}_i$ à $\mathcal{M}_{n_1}$. C'est le temps nécessaire pour que la fin de réglage d'une machine ait un impact sur la production.

De plus, pour chaque machine $\mathcal{M}_i$, nous connaissons ρ_i, le nombre de pièces que la machine est capable de traiter par unité de temps.

L'exemple de la figure 1 illustre la relation entre la structure de la ligne, le réglage de tâches et l'évolution de la productivité. Les machines ne peuvent être réglées qu'après leur date de début au plus tôt. La production est toujours limitée par l'étage le plus lent. Par exemple, avant la fin du réglage de $\mathcal{M}_6$, la machine limitante est $\mathcal{M}_3$, après son réglage, il faut attendre un temps q_6 pour voir la production évoluer: la nouvelle machine limitante (i.e. "bottleneck") est $\mathcal{M}_1$.

Pour être exécutée, une tâche de réglage nécessite l'intervention d'un opérateur. Nous notons λ, le nombre d'opérateurs. Chaque opérateur $O_h, h \in [1..\lambda]$ est plus ou moins apte à effectuer une tâche $\mathcal{M}_i$ en fonction de son expérience et par hypothèse, nous connaissons le temps moyen qu'il a mis pour effectuer la tâche sur les anciens changements de série. Nous noterons $p_{i,h}$ ce temps. Pour construire l'ordonnancement, nous considérons que l'opérateur mettra ce temps $p_{i,h}$ pour réaliser la tâche. Si l'opérateur O_h n'est pas

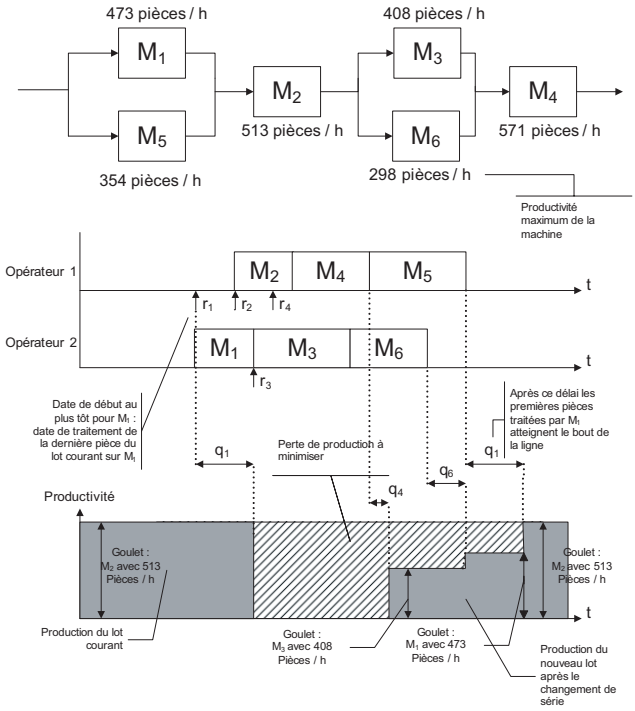


Figure 1: Exemple d'instance à 6 machines et 2 opérateurs

qualifié pour intervenir sur la machine $\mathcal{M}_i$, nous considérons que $p_{i,h} = +\infty$. C'est cette donnée que nous allons chercher à valider par le moteur de simulation (cf. section 4).

Pour chaque opérateur O_h nous connaissons ses périodes de disponibilité notées $A(h,t)$. On aura $A(h,t) = 1$ si l'opérateur peut intervenir sur un changement de série à la date t. Il est important de noter que dans le cas de SKF, les périodes de disponibilités sont continues : elles ont un début et une fin et il n'y a aucune pause au milieu.

En utilisant la notation classique à 3 champs des problèmes d'ordonnancement, si $f(C_i, q_i)$ est la fonction donnant le nombre de roulements produit, ce problème peut se noter:

$$R, MPM|r_i, q_i|f(C_i, q_i)$$

Les opérateurs multi-compétents sont ici modélisés par des machines à usage multiple (Jurisch 1992, Dauzère-Pérès, Roux & Laserre 1998).

Un cas particulier de ce problème est le $P|r_i, q_i|C_{max}$ connu comme étant $NP - difficile$ au sens fort (Garey & Johnson 1978). Des méthodes de résoluton exactes pour ce problème ont été proposées dans (Gharbi & Haouari 2002) par exemple. Mais

l'originalité de notre problème se situe dans le critère qui ne s'intéresse pas aux dates de fin des tâches mais à la production de la ligne. De plus, le fait que nous devions utiliser les méthodes que nous proposons dans un logiciel utilisé quotidiennement nous conduit à proposer des heuristiques permettant d'obtenir rapidement de bonnes solutions.

2.3. Problème à séquence fixée

Les contraintes industrielles nous conduisent à considérer qu'il existe une régle de priorité que chaque opérateur doit respecter:

1. Les tâches prioritaires sont classées dans l'ordre croissant des r_i.
2. Les tâches non prioritaires sont triées dans l'ordre où elles sont les plus critiques: tout d'abord la tâche non prioritaire sur l'étage le plus limitant, puis parmis les tâches restantes, la tâche non prioritaire sur l'étage le plus limitant...

Sur l'exemple de la figure 1, la liste de priorité est: $\mathcal{M}_1, \mathcal{M}_2, \mathcal{M}_3, \mathcal{M}_4, \mathcal{M}_6, \mathcal{M}_5$. En effet, les 4 machines prioritaires sont triées par r_i croissants et $\mathcal{M}_6$ est plus prioritaire que $\mathcal{M}_5$ car $\mathcal{M}_3$ est plus limitant que $\mathcal{M}_1$. L'ordonnancement proposé respecte cette liste de priorité sur chaque opérateur.

On peut montrer (Pessan, Bouquard & Néron 2006) que sur une ligne série cet ordre est optimal. De plus, pour les machines en double, il est en pratique impossible de gagner du temps en commençant une machine $\mathcal{M}_i$ qui est sur un étage non limitant avant $\mathcal{M}_j$ qui est sur un étage limitant car q_i devrait être supérieur à $p_j + q_j$ pour qu'il y a ait éventuellement un gain, ce qui n'arrive pas en pratique. On peut donc considérer cet ordre comme optimal. Le problème revient donc à chercher l'affectation optimale.

2.4. Critère

L'objectif est de maximiser le nombre de pièces produites pendant le changement de série sachant que la quantité de pièces à produire pour la série amont est connue à l'avance. Cela revient donc à maximiser le nombre de pièces de la série aval que l'on peut produire et c'est cette quantité que le critère $f(C_i)$ exprime.

À un instant donné, suivant la configuration dans laquelle on se trouve, i.e. quelles machines ont déjà été réglées pour la série aval, on peut déterminer quelle est la productivité de la ligne. Au cours du changement de série, cette configuration va bien sûr évoluer, c'est pourquoi, une manière d'exprimer $f(C_i)$ consiste à examiner pour chaque unité de temps, la quantité de pièces qui pourra être produite.

On sait qu'avant d'avoir toutes les machines prioritaires réglées, il sera impossible de produire la moindre pièce. Il est donc inutile d'examiner un intervalle avant que toutes les machines prioritaires n'aient été réglées.

Soit:

- T_c : Horizon de calcul de la production pour le critère
- T_g : Date de sortie de la première pièce de la chaîne, c'est à dire la date de fin de réglage de la dernière machine prioritaire à laquelle on ajoute le temps de latence de la machine :

$$T_g = \max_{i \in [1..n_1]} (C_i + q_i)$$

- $Prod(t)$: Quantité de pièces produites entre t et $t+1$

Le critère s'exprime par :

$$f(C_i) = \sum_{t=T_g}^{T_c} Prod(t)$$

$Prod(t)$ est déterminé par l'étage le plus lent de la ligne. Soit $ProdEt(i,t)$ la quantité de pièces que peut produire l'étage de la machine i entre t et $t+1$. Si on considère que chaque étage a exactement une machine prioritaire, on obtient:

$$Prod(t) = \min_{i \in [1..n_1]} ProdEt(i,t)$$

Il reste à exprimer $ProdEt(i,t)$, soit:

- $Et(i,j) = \begin{cases} 1 & \text{si les machines i et j font} \\ & \text{partie du même étage} \\ 0 & \text{sinon} \end{cases}$
- $Fini(i,t) = \begin{cases} 1 & \text{si } t > C_i + q_i \\ 0 & \text{sinon} \end{cases}$
  indique si une machine est déjà réglée et que les pièces qui passent par celle-ci commencent à sortir de la ligne (ou sont en attente de la fin du réglage d'une machine prioritaire qui bloque la production).

La capacité de production d'un étage est la somme des capacités de production de chacune des machines de cet étage qui sont déjà réglées. Ce qui s'exprime par:

$$ProdEt(i,t) = \sum_{j=1}^{n} Et(i,j) * Fini(j,t) * \rho_j$$

Il faut maintenant généraliser cette expression au cas (rencontré à SKF) où l'on a dans la ligne de production plusieurs groupes de machines séries en parallèle.

Soit $L(i)$ la longueur de la branche à laquelle appartient la machine i. On parle ici de branches dans un étage: s'il s'agit d'un étage à machines parallèles on aura $L(i) = 1$, s'il s'agit d'un étage où l'on a plusieurs couples de machines en parallèle (couples ébauche/finition par exemple) on aura $L(i) = 2 \ldots$

Soit $P(i)$ un booléen indiquant s'il s'agit de la première machine d'une branche.

De plus on considère que toutes les machines d'une branche se suivent dans leurs numéros d'indexation.

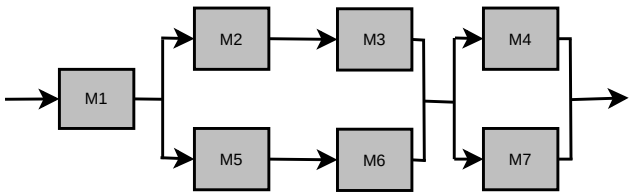


Figure 2: Exemple de ligne avec des branches de longueur 2

Sur la ligne de la figure 2, on a par exemple: $L(1) = 1$, $P(1) = 1$, $L(2) = 2$, $P(2) = 1$, $L(6) = 2$, $P(6) = 0$, $L(7) = 1$, $P(7) = 1 \ldots$

L'expression de $ProdEt(i,t)$ devient alors:

$$ProdEt(i,t) = \sum_{j=1}^{n} Et(i,j) * \prod_{k=j}^{j+L(i)-1} Fini(k,t) * P(i) * \min_{k\in[j,j+L(i)-1]} \rho_k$$

Cette expression tient compte du fait que la productivité de chaque branche est limitée par la machine la plus lente de la branche.

Pour calculer efficacement ce critère, il ne faut tenir compte que des dates de fin de réglage des machines. Il est alors possible de calculer la valeur du critère en O(n) à l'aide de algorithme de programmation dynamique présenté ci-dessous.

Soit:

- L la liste des tâches triées de la machine la plus limitante en production à la moins limitante (i.e., l'ordre dans lequel la productivité peut être augmentée au fur et à mesure que les tâches se terminent).
- PL(i) la productivité de la ligne lorsque toutes les machines $\mathcal{M}_j, j \in [1..i]$ sont réglées.
- CL(i) la date de fin maximale des i premières tâches de L

Alors, nous pouvons définir:

$$CL(i) = \begin{cases} 0 & \text{si i=0} \\ max(CL(i-1), C_{L_i}) & \forall i \in [1..n] \end{cases}$$

Et le critère peut s'exprimer par:

$$f(C_{L_i\in[1..k]}) = \begin{cases} 0 & \text{si k=0} \\ f(C_{L_i\in[1..k-1]})+ & \\ PL(k) * (CL(k+1) - CL(k)) & \forall k \in [1..n-1] \\ f(C_{L_i\in[1..k-1]})+ & \\ PL(k) * (T_c - CL(k)) & \text{si k=n} \end{cases}$$

On peut donc calculer le critère en O(n) une fois PL(i) calculé, ce qui peut être précalculé avant l'execution de la méthode d'optimisation.

3. MÉTHODE DE RÉSOLUTION

3.1. Algorithme mémétique

Un algorithme mémétique (Moscato 1989, Moscato 1999) est un algorithme mettant en oeuvre à la fois un algorithme génétique et un algorithme de voisinage. Dans notre cas, l'algorithme de voisinage utilisé est une descente locale utilisant un voisinage représentant des échanges multiples de tâches entre opérateurs.

Un algorithme génétique (Holland 1975) est une méthode d'optimisation qui met en œuvre les mécanismes de l'évolution sur une population de solutions au problème. L'avantage d'un tel algorithme est qu'il peut généralement trouver plusieurs bonnes solutions rapidement en cherchant dans tout l'espace des solutions. D'un autre coté, la méthode de descente locale permet de focaliser la recherche dans un endroit précis de l'espace de recherche. C'est pourquoi ces deux méthodes utilisées conjointement permettent souvent d'obtenir des heuristiques très performantes.

Dans cette section, nous définissons la méthode de codage des solutions, et les différents opérateurs de l'algorithme mémétique.

3.1.1 Méthode de codage des solutions

Comme nous avons pu le voir dans la section 2.3, le problème d'ordonnancement des tâches de réglage peut se ramener à un problème d'affectation dans le cas de l'usine SKF. Il suffit donc pour encoder les solutions d'encoder l'affectation des tâches sur les opérateurs.

Chaque individu sera encodé en utilisant un tableau de n entiers (un pour chaque tâche). Les entiers sont les numéros d'opérateurs affectés à chaque tâche.

Exemple:

1	2	1	3	2	2	4
			↑			

La machine 4 est réglée par l'opérateur 3.

Les opérateurs autorisés pour chaque tâche sont ceux qui ont la compétence pour l'effectuer: si k_i est l'opérateur affecté à la tâche i, $k_i \in \{[0..\lambda], i \in [0..n], p_{i,k_i} \neq +\infty\}$

3.1.2 Opérateur de croisement

L'opérateur de croisement de notre algorithme mémétique est un classique opérateur de croisement à deux points dont l'efficacité a été montré par (Beasley, Bull & Martin 1993). L'opérateur est utilisé pour générer γ nouveaux individus à chaque itération, les parents étant sélectionnés à l'aide d'une sélection par tournoi binaire.

Exemple:

chromosome 1:

1	**2**	**1**	**3**	**2**	**2**	**4**

chromosome 2:

3	*2*	*1*	*3*	*3*	*3*	*3*
		↑			↑	
		p_1			p_2	

fils 1:

1	**2**	*1*	*3*	*3*	*3*	**4**

fils 2:

3	*2*	**1**	**3**	**2**	**2**	*3*

Cet opérateur a la propriété de ne générer que des individus contenant des affectations valides vis-à-vis des compétences des opérateurs.

3.1.3 Opérateur de mutation

L'opérateur de mutation modifie une affectation en choisissant aléatoirement un nouvel opérateur. Cet opérateur a un rôle de diversification important puisque c'est par cet opérateur que l'on peut introduire de nouvelles affectations dans la population de l'algorithme mémétique. Dans notre algorithme, la probabilité de mutation d'un nouvel individu est dynamique: elle augmente progressivement à partir d'une valeur de base pm tant que le critère n'est pas amélioré et est réinitialisé à pm lorsqu'une solution améliorant le critère est trouvée.

3.1.4 Diversification

Dans un algorithme génétique, la diversification a pour but d'éviter de rester dans un optimum local et explorer de nouvelles régions de l'espace de recherche. Outre la probabilité de mutation dynamique, nous utilisons 2 mécanismes permettant de diversifier la recherche:

- Remplacement de l'opérateur de mutation par la génération aléatoire d'un individu.
- Regénération aléatoire de toute la population en ne gardant que le meilleur individu de l'ancienne.

Ces mécanismes de diversification ne sont utilisés qu'après un nombre d'itération D sans amélioration du critère.

3.1.5 Selection

Comme la taille de la population doit rester constante, il est nécessaire de sélectionner les individus qui survivent d'une itération à l'autre de l'algorithme mémétique. Pour cela, nous utilisons une sélection par tournoi (Miller & Goldberg n.d.) permettant d'éliminer des individus tant que l'on n'a pas atteint la taille de population désirée.

3.1.6 Opérateur d'intensification

L'opérateur d'intensification est utilisé pour focaliser la recherche autour d'une solution donnée. Dans notre algorithme, nous définissons une probabilité faible p_{int} d'utiliser cet opérateur à chaque itération. Lorsque cet opérateur est utilisé, un individu est sélectionné au hasard dans la population et une descente locale utilisant le 3-voisinage est utilisée (voir section 3.2).

3.2. Descente locale

Le voisinage de notre algorithme de descente locale utilise un graphe permettant de recherche des chaînes de k réaffectations de tâches, k étant un paramétre du voisinage: dans ce papier nous le nommons k-voisinage.

Le graphe est composé de n sommets tâche et λ sommets opérateur ainsi que de deux sommets additionnels nommés S et P. Les arcs du graphe sont les suivants:

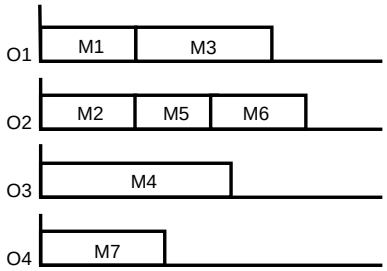
- Un arc d'un sommet opérateur vers un sommet tâche si la tâche est affectée à cet opérateur dans la solution courante.
- Un arc d'un sommet tâche vers un sommet opérateur lorsque l'opérateur a la compétence.
- Un arc de chaque sommet opérateur à P.
- Un arc de S à chaque opérateur qui possède une tâche critique. Ce sont les tâches permettant d'augmenter la production lorsqu'elles sont terminées dans la solution courante: il s'agit d'une part de la tâche prioritaire se terminant le plus tard (celle qui conditionne le début de la production) et d'autre part de chaque tâche non prioritaire qui n'a aucune tâche plus prioritaire (suivant L_p) se terminant après.

Les solutions voisines vont alors être représentées par des chemins de S à P dans ce graphe:

- Lorsque l'on passe d'un sommet opérateur vers un sommet tâche, il s'agit d'une tâche que l'on enlève d'un opérateur.
- Lorsque l'on passe d'un sommet tâche vers un sommet opérateur, il s'agit d'une tâche que l'on affecte à un opérateur.

L'algorithme de descente locale recherche un chemin de S à P passant par $2k + 2$ arcs conduisant à une solution meilleure que la solution courante.

La figure 3 montre un exemple de graphe construit à partir d'un diagramme de Gantt (la solution courante) et d'une liste de compétences de chaque opérateur. Si on prend par exemple le chemin {S,O1,M1,O3,P} dans le 2-voisinage, la solution représentée par ce graphe est celle où l'on enlève la tâche M1 de l'opérateur O1 et où l'on ajoute cette même tâche M1 à l'opérateur O3.



opérateur:	compétences
O1	M1,M3
O2	M2,M5,M6
O3	M1,M4,M5,M6,M7
O4	M7

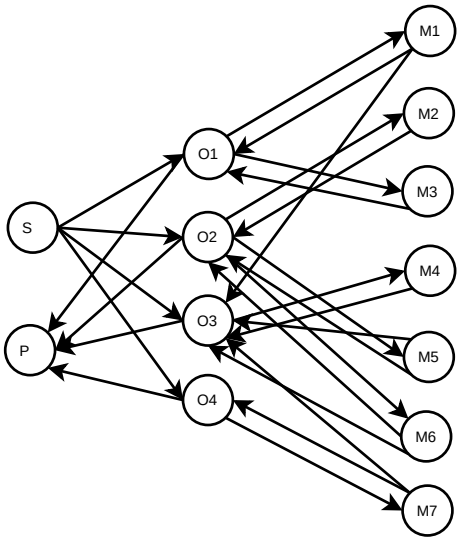


Figure 3: exemple de graphe de voisinage

4. SIMULATION POUR LA VALIDATION DES HYPOTHÈSES

Le but de l'approche de validation des résultats par la simulation est la validation des hypothèses utilisées pour le calcul des solutions par les méthodes précédentes. L'hypothèse la plus forte consiste à construire une solution à partir des moyennes des durées observées sur les temps opératoires par opérateur et par machine. Nous cherchons ici à vérifier si les solutions ainsi construites restent valides lorsque les durées opératoires sont des variables aléatoires correspondant au mieux aux valeurs observées sur les derniers mois.

4.1. Démarche générale

Dans cette partie, nous appellerons *solution de base* la solution produite par les méthodes décrites précédemment (descente locale, algorithme génétique, algorithme mémétique). Il s'agit donc de vérifier le comportement de la solution de base en présence d'aléas sur les durées opératoires. Ces aléas sont générés de telle sorte à correspondre au mieux aux valeurs observées par couple opérateur/machine. La validation des résultats obtenus par la simulation se fait en quatre étape

1. identifier les variables aléatoires les plus proba-

bles permettant de représenter par couple opérateur/machine, les temps opératoires observés. (cf 4.2)

2. procéder à un tirage des variables aléatoires, et reconstruire la solution à partir de ces valeurs en respectant l'affectation des opérateurs aux machines et la séquence sur chaque opérateur définis dans la solution de base. En déduire la valeur du critère associé. Cette phase est répétée autant de fois que nécessaire.

3. à chaque itération de l'étape précédente, on obtient une valeur du critère. A partir de cet échantillon de valeurs du critère, il est possible de calculer une moyenne et un intervalle de confiance à α pourcent. Notre but ici est de vérifier pour une précision du résultat souhaitée, i.e, la demi largeur de l'intervalle de confiance fixée, et une couverture (α également donnée), si la valeur du critère correspondant à la solution de base (celle calculée à partir des moyennes des temps opératoires) est dans l'intervalle de confiance. Dans ce cas, la valeur du critère de la solution de base est une information pertinente pour prédire la valeur effective du critère lors du changement de série dans un contexte réel c'est-à-dire soumis à des aléas. En d'autres termes, l'hypothèse utilisée pour construire cette solution de base peut alors être considérée comme pertinente. (cf. 4.2)

4.2. Choix des V.A. en entrée de la simulation

Il s'agit ici d'identifier pour chaque couple opérateur/machine la variable aléatoire qui correspond le mieux à l'échantillon donné, i.e. les durées opératoires observées. L'utilité de ces variables aléatoires est de permettre de simuler le comportement de notre solution de base, en présence d'aléas, autant de fois que nécessaire jusqu'à l'obtention de résultats ayant la précision souhaitée.

Le choix des variables aléatoires en entrée suit un mécanisme classique (Law & Kelton 2000). Dans un premier temps il peut être utile de vérifier l'indépendance des échantillons, e.g, en utilisant un diagramme de dispersion. Cependant pour cette étude cette étape n'est pas cruciale puisque les différents échantillons correspondent à des temps observés à des dates différentes, espacées de plusieurs jours voir de plusieurs semaines, il n'y a aucune raisons que ceux-ci soient corrélés. Ensuite des indicateurs préliminaires (moyenne estimée, variance estimée, coefficient de symétrie estimé, coefficient de variation estimée) sont calculés permettant éventuellement d'éliminer certaines lois usuelles (par exemple les lois symétriques si le coefficient de symétrie estimé est largement différent de 0). L'étape suivante est la construction d'un histogramme à partir de l'échantillon de départ auquel les densités de probabilité des lois usuelles, non encore écartées, peuvent être comparées. Pour les lois usuelles restantes, il est nécessaire d'estimer son ou ses paramètres, e.g., par le calcul du maximum de vraisemblance. Enfin pour chaque couple loi/paramètre(s), on effectue un test d'adéquation, e.g., χ^2, permettant d'identifier avec quelle probabilité le couple loi/paramètre 'colle' à l'échantillon donné. A l'issue de cette étape soit une loi est retenue, soit une des lois éliminées aux étapes 2 et 3 est reintroduite dans les lois usuelles possibles.

Le problème éventuellement rencontré lors de l'identification des variables aléatoires concerne les couples opérateur/machine pour lesquels peu de mesures observées sont à disposition. Dans ce cas, une loi triangulaire ou une loi uniforme est choisie sur l'intervalle des valeurs observées.

4.3. Interprétation des sorties

Une réplication de la simulation correspond à (1) le tirage d'une variable aléatoire suivant la loi identifiée à l'étape précédente pour chaque couple opérateur/machine intervenant dans la solution de base et (2) la construction d'une solution en respectant l'affectation et le séquencement des opérations sur les opérateurs de la solution de base mais en prenant en compte les tirages réalisés des durées. La valeur du critère correspondant à cette nouvelle solution est calculée. Ainsi à n réplications correspondent n valeurs du critère indépendantes et identiquement distribuées. A partir de ces n échantillons une moyenne estimée ($\bar{X}(n)$), et une variance estimée ($S^2(n)$) sont calculées. On en déduit un intervalle de confiance :

$$\bar{X}(n) \pm z_\alpha \sqrt{\frac{S^2(n)}{n}} \quad (1)$$

.

La particularité ici consiste dans le fait que la précision absolue (en nombre de roulement) ou relative (en % de la moyenne) est une donnée. Par exemple on souhaite savoir si etant donnée une moyenne calculée en prenant en compte les aléas, i.e., par la simulation, et une marge d'erreur de 5% sur cette moyenne, si la valeur de la solution de base est dans cette marge d'erreur, validant alors l'hypothèse de base de notre étude qui est de construire la solution de base à partir des moyennes des durées observées. Or en appliquant la formule classique de calcul de l'intervalle de confiance, la demi-largeur de l'intervalle est un resultat du calcul et non une donnée. Si la demi-largeur obtenue ne correspond pas à la précision souhaitée il est alors

necessaire de diminuer cette demi-largeur. D'autres replications de la simulation sont effectuées, augmentant ainsi le nombre de valeurs à partir desquelles cette demi-largeur est calculée. Le nombre exact de replications nécessaires pour obtenir la precision souhaitée, qu'elle soit absolue ou relative, peut être detérminée analytiquement (Law & Kelton 2000), en faisant l'hypothèse que notre estimateur de variance est de bonne qualité.

5. APPLICATION

Le moteur de simulation a été encapsulé dans une application prenant en entrée des ordonnancements et fournissant en sortie les différents resultats en terme de production pour chaque réplication. Ces résultats sont ensuite transmis à un module permettant de calculer l'intervalle de confiance et l'afficher graphiquement.

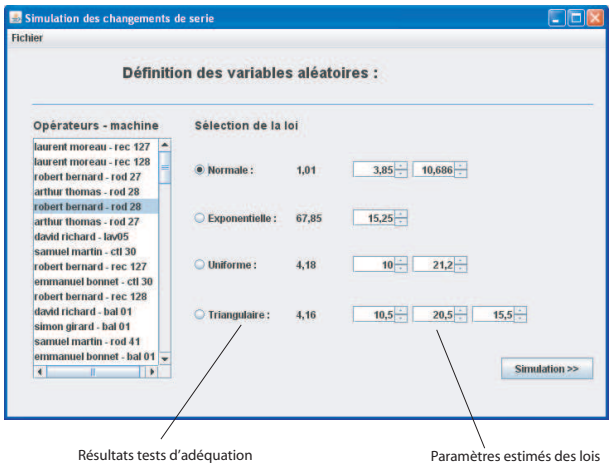


Figure 4: Assistant de définition des variables aléatoires

La figure 4 présente l'interface utilisée pour faciliter la sélection des lois correspondant aux différents couple machine/opérateur. Nous avons retenus 4 lois: normale, exponentielle, uniforme et triangulaire. Le logiciel estime les paramètres de chacunes de ces lois et propose ces valeurs par défaut. Le résultat du test du χ^2 est également présenté permettant de préselectionner la loi qui ressemble le plus probablement à l'échantillon.

L'interface présentée sur la figure 5 permet de comparer l'évolution de la production lors des différentes réplications. Cette interface permettra notamment d'exploiter le moteur de simulation dans l'entreprise: il est important de se rendre compte des cas pénalisant la production pour savoir où renforcer la formation des opérateurs.

La figure 6 représente l'intervalle de confiance calculé

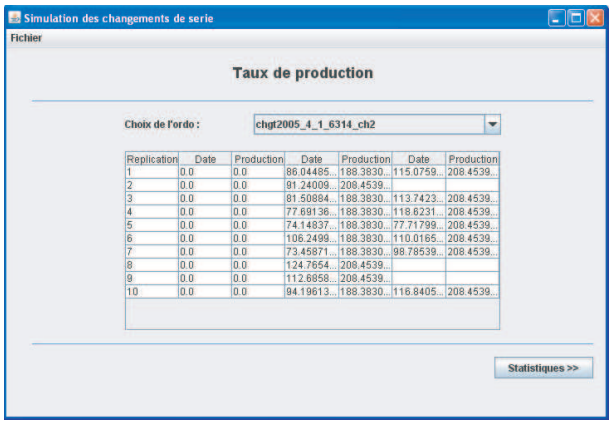


Figure 5: Interface de comparaison des réplications

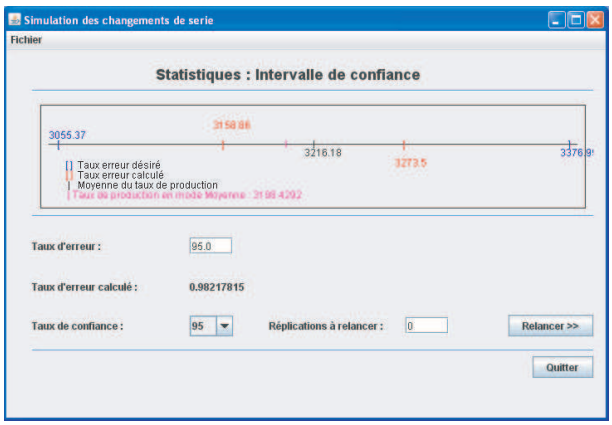


Figure 6: Intervalle de confiance calculé

à partir des réplications faites par le moteur de simulation. Sur l'exemple présenté, la solution calculée par notre méthode prenant uniquement en compte les temps de réglage moyens est de 3198 pièces produites. De plus, on peut voir que la moyenne des critères trouvés lors des différentes réplications est de 3216 situé dans un intervalle de confiance à 95% délimité par 3158 et 3273, soit une précision absolue de 160 unités produites autour de la moyenne (précision relative de 95%). Or, la solution trouvée par l'algorithme mémétique se situe bien dans cet intervalle. On peut donc dire que pour cet exemple, l'ordonnancement trouvé est valide vis-à-vis des temps opératoires observés sur les derniers mois.

6. CONCLUSION

Dans cette étude, nous avons montré comment les ordonnancements trouvés par des méthodes reposant sur des hypothèses faites sur les temps opératoires peuvent être validés par une approche par simulation.

En effet, le problème d'optimisation des changements de série que nous avons à résoudre dans l'usine SKF a comme ressources des opérateurs qui n'effectuent pas toujours leurs tâches à la même vitesse. Nous avons donc mis au point des méthodes de résolutions (Pessan, Néron & Bellenguez-Morineau 2006) qui utilisent comme temps opératoire la moyenne des temps constatés récemment pour chaque opérateur. Le moteur de simulation nous permet à partir des échantillons de temps de réglage dont nous disposons de vérifier que cette hypothèse est réaliste et que notre méthode conduit à des ordonnancements valides. Les résultats expérimentaux obtenus par ce moteur seront présentés lors de la conférence.

References

Allahverdi, A., Gupta, J. & Aldowaisan, T. (1999). A review of scheduling research involving setup considerations, *Omega, Int. J. Mgmt. Sci.* **2**(27): 219–239.

Beasley, D., Bull, D. & Martin, R. (1993). An overview of genetic algorithms: Part 2, research topics, *University Computing* **4**(15): 170–181.

Billaut, J., Moukrim, A. & Sanlaville, E. (2005). *Flexibilité et robustesse en ordonnancement*, Hermès Science Publications.

Brucker, P., Knust, S. & Wang, G. (2005). Complexity results for flow-shop problems with a single server, *European Journal of Operational Research* (165): 398–407.

Dauzère-Pérès, S., Roux, W. & Laserre, J. (1998). Mult-resource shop scheduling with resource flexibility, *European Journal of Operational Research* (107): 289–305.

Garey, M. & Johnson, D. (1978). Strong np-completeness results: motivations, examples, and implications, *Journal of the ACM* **3**(5): 499–508.

Gharbi, A. & Haouari, M. (2002). Minimizing makespan on parallel machines subject to release dates and delivery times, *Journal of Scheduling* (5): 329–355.

Holland, J. (1975). *Adaptation in natural and artificial systems*, University of Michigan Press.

Jurisch, B. (1992). *Scheduling jobs in shops with multi-purpose machines*, PhD thesis, University of Osnabrük, Germany.

Kravchenko, S. & Werner, F. (1997). Parallel machine scheduling problems with a single server, *Mathematical and Computer Modelling* **12**(26): 1–11.

Law, A. M. & Kelton, W. D. (2000). *Simulation Modeling and Analysis, Third Edition*, MC Graw Hill.

Miller, B. & Goldberg, D. (n.d.). Genetic algorithms, tournament selection and the effects of noise, *Technical Report 95006*, University of Illinois at Urbana-Champaign.

Moscato, P. (1989). On evolution, search optimization, genetic algorithms and martial arts: Towards memetic algorithms, *Technical Report C3P 826, Caltech concurrent computation program* .

Moscato, P. (1999). *New ideas in Optimization*, McGraw-Hill, New York, chapter Memetic algorithms: A short introduction, pp. 219–234.

Pessan, C., Bouquard, J. & Néron, E. (2006). An unrelated parallel machines model for production resetting optimization, *SSSM, 2006 International Conference*, Troyes, France, pp. 1178–1182.

Pessan, C., Néron, E. & Bellenguez-Morineau, O. (2006). Modélisation et planification des opérations de réglage de machines lors de changements de série, *in* F. R. M. Gourgand (ed.), *MOSIM'06: Défis et Opportunités*, Rabat, Maroc, pp. 1545–1554.

Shingo, S. (1985). *A revolution in Manufacturing : the SMED system*, Cambridge, MA: Productivity Press.