CIGI 2011
Job shop sous contraintes de disponibilité des ressources : modèle mathématique et heuristiques

SADIA AZEM[1], RIAD AGGOUNE[2], STÉPHANE DAUZERE-PERES[1]

[1] Département Sciences de la Fabrication et Logistique, CMP, Ecole des Mines de Saint-Etienne,
880 AVENUE DE MIMET, F-13541 GARDANNE, FRANCE.
sadia.azem@gmail.com
dauzere-peres@emse.fr

[2] Centre de Recherche Public Henri Tudor
6 rue de Luxembourg, L-4002 Esch-sur-Alzette, Luxembourg.
riad.aggoune@tudor.lu

Résumé – La majeure partie des études des problèmes d'ordonnancement se placent dans le contexte où les ressources sont disponibles en permanence. Ce qui en réalité n'est pas toujours le cas. Nous traitons le contexte d'indisponibilités connues à l'avance ; nous sommes particulièrement intéressés par le problème d'ordonnancement dans un atelier de type job shop, dans lequel les opérations des jobs (ou tâches) peuvent être interrompues par des périodes d'indisponibilité, et les périodes d'indisponibilité peuvent être déplacées dans des fenêtres de temps. Intégrer ces contraintes augmente la complexité des problèmes d'ordonnancement. Dans cet article, nous proposons un modèle mathématique et des méthodes approchées pour le problème. En plus de la résolution des problèmes considérés, le but de cette modélisation est d'analyser l'impact des contraintes d'indisponibilité des ressources et d'évaluer la qualité des méthodes approchées. Ces dernières sont des heuristiques de construction qui élaborent rapidement un ordonnancement sur la base de stratégies de décision. Plusieurs expérimentations ont été effectuées pour valider les méthodes proposées.

Abstract - In most of machine scheduling literature, resources are assumed to be continuously available which is not always true in practice. We deal with the context of unavailability periods known a priori; we are particularly interested in job shop problem where the job operations may be interrupted by resource availability periods; and the unavailability periods may be moved in time windows. Integrating these constraints increase the complexity of the scheduling problems. In this paper, we propose a mathematical model and approximation methods for the problem. In addition to the resolution of the considered problems, the aim of this modeling is to support the analysis of the impact of resource unavailability constraints and evaluate the quality of the approximation methods. These methods are construction heuristics that quickly determine a schedule based on decision strategies. Various experiments have been performed to validate the proposed methods.

***Mots clés* – ordonnancement, contraintes de disponibilité, systèmes industriels, modélisation mathématique, heuristiques de construction.**
***Keywords* - Scheduling, availability constraints, industrial systems, mathematical modeling, construction heuristics.**

1 INTRODUCTION

Dans les systèmes de production, les différentes ressources aussi bien matérielles qu'humaines peuvent être indisponibles pour diverses raisons : congés ou défections du personnel, activités de maintenance ou pannes des machines, etc... Ceci influence les plans de production de façon significative : une ressource additionnelle pour absorber cette charge de travail n'est pas forcément disponible ; d'où la nécessité de trouver la meilleure façon de répartir la charge de travail entre les ressources en prenant en compte les périodes d'indisponibilité, le type des opérations des jobs que les ressources peuvent effectuer, l'ordre entre les opérations.

Nous présentons, dans cet article, un modèle mathématique et des heuristiques de construction pour résoudre le problème d'ordonnancement dans un atelier de production de type job shop (appelé aussi atelier multigamme) lorsque les ressources ne sont pas disponibles en continu ; et ce pour mieux modéliser la réalité industrielle. Le contexte d'étude est celui d'indisponibilités connues à l'avance ou lorsqu'il est possible de refaire l'ordonnancement une fois les indisponibilités connues. Des aspects liés à la flexibilité des opérations des jobs (appelés auusi commandes) et des périodes d'indisponibilité des ressources sont abordés ; notamment, l'autorisation de la préemption d'une opération en présence d'une période d'indisponibilité, i.e, une opération peut être

interrompue par une période d'indisponibilité (mais non par une autre opération), ensuite reprise avec une éventuelle pénalité, dès que la ressource est à nouveau disponible (ceci peut être le cas de produits regroupés en lots) ; le déplacement d'une période d'indisponibilité dans une fenêtre de temps (pour créer un temps libre sur la ressource pour réaliser des opérations plus tôt). La modélisation mathématique est souvent négligée par les chercheurs à cause de la forte complexité (NP-difficile) du problème. Nous proposons cette approche car elle permet de résoudre certaines tailles de problèmes, elle nous instruit sur la façon de traiter les contraintes de disponibilité des ressources et permet d'évaluer la qualité des solutions obtenues par des heuristiques. Ces dernières visent à obtenir de bonnes solutions rapidement. Elles construisent un ordonnancement en se basant sur des règles de priorité. L'originalité des heuristiques que nous proposons réside notamment dans l'intégration de la flexibilité des opérations et des périodes d'indisponibilité au moment de la construction de l'ordonnancement. Ces méthodes, qui constituent des blocs de construction, peuvent être intégrées dans d'autres méthodes approchées pour améliorer les résultats obtenus. Comme critère objectif à minimiser, nous considérons soit le makespan (date d'achèvement de l'ordonnancement), soit la somme des dates de fin des jobs (dates de fin des dernières opérations des jobs) ; il n'existe pas de hiérarchie entre ces critères. Ces approches peuvent naturellement traiter d'autres critères ou contraintes sur les jobs. Il est aussi facile de les étendre pour traiter le problème d'ordonnancement d'un atelier de type job shop flexible.

L'article est structuré comme suit. Le problème traité est défini dans la section 2. La section 3 est consacrée à la modélisation mathématique, et la section 4 aux heuristiques. Les résultats des expérimentations numériques sont présentés dans la section 5.

2 DEFINITION DU PROBLEME

Le problème du job shop avec contraintes de disponibilité des ressources peut être défini par un ensemble de n jobs $\mathcal{J}=\{J_1,J_2,..,\ J_n\}$ à réaliser sur un ensemble de m machines $\mathcal{M}=\{M_1,M_2,..,\ M_m\}$. Chaque job J_i est composé d'une séquence linéaire de n_i opérations $\{O_{i1},O_{i2},..,O_{ij},..,O_{in_i}\}$. Chaque machine peut effectuer une seule opération à la fois et chaque opération O_{ij} requiert une machine uniquement pour sa réalisation, et ce pendant p_{ij} unités de temps ; sa date de début est t_{ij} et sa date de fin est C_{ij}. Il y a m_r périodes d'indisponibilité $\{h_{r1},h_{r2},..,h_{rk},..,h_{rm_r}\}$ sur chaque machine M_r ; on suppose qu'elles ne sont pas trop proches. La date de début S_{rk} de la période d'indisponibilité h_{rk} de durée p'_{rk} est connue à l'avance. Dans cet article, on dira que h_{rk} est flexible (notion introduite dans [Aggoune, 2004]) lorsqu'on associe à S_{rk} une fenêtre de temps $[ES_{rk},LS_{rk}]$ (ES_{rk} et LS_{rk} sont respectivement les dates de début au plus tôt et au plus tard de h_{rk}) ; l'idée étant de créer un temps libre sur la machine pour exécuter une opération soit avant la période d'indisponibilité soit après ce qui permet à l'opération de s'achever plus tôt (Figure 1). Chaque job passe sur les machines dans un ordre appelé gamme opératoire définie à priori. On désigne par mr_{ij} la machine sur laquelle l'opération O_{ij} est réalisée.

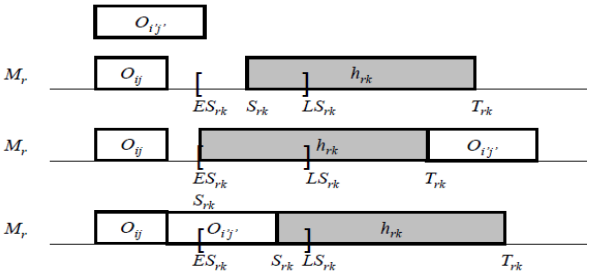

Figure 1. Fenêtre de temps pour la date de début d'une période d'indisponibilité

Par ailleurs, dans la littérature scientifique, il existe quatre cas pour l'interruption d'une opération par une période d'indisponibilité (Figure 2) : opérations strictement non-préemptive, sécable, non-sécable et semi-sécable. Le premier est dû à [Aggoune, 2004] et les trois autres à [Lee, 1996, 1997, 1999]. Une opération est dite *strictement non-préemptive* lorsqu'elle ne peut être interrompue ni par une autre opération ni par une période d'indisponibilité. Une opération interrompue par une période d'indisponibilité est dite *sécable* si son exécution peut continuer dès que la machine qui l'exécute est de nouveau disponible. Elle est dite *non-sécable* si elle doit recommencer complètement. Il est important de noter que ce cas est différent du cas non-préemptif du fait que, dans ce dernier, lorsque l'opération ne peut être effectuée avant la période d'indisponibilité, elle doit commencer et se terminer après. Une opération est dite *semi-sécable* si elle doit partiellement recommencer lorsque la machine est de nouveau disponible. Noter que l'étude du cas semi-sécable inclue les cas sécable et non-sécable. Cependant, l'étude du cas non-sécable est moins pertinente à considérer que les autres cas ; car nous étudions des périodes d'indisponibilité prévues.

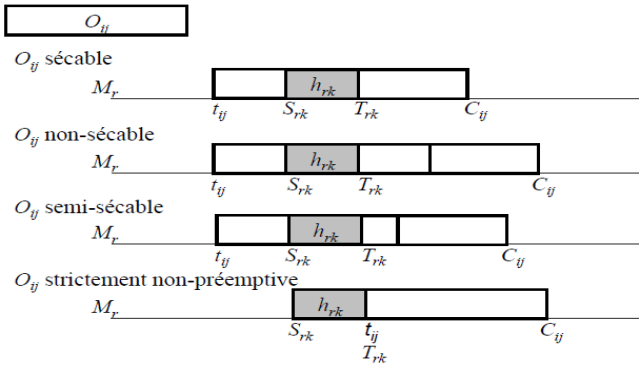

Figure 2. Les différents cas d'interruption d'une opération

Nous associons à chaque opération O_{ij} un *coefficient de pénalité sur la préemption* α_{ij} qui représente la partie de l'opération O_{ij} à refaire après la période d'indisponibilité l'interrompant ; il représente son caractère sécable. Ainsi, α_{ij}=0 (resp. α_{ij} = 1) si O_{ij} est sécable (resp. non-sécable) et $0 \le \alpha_{ij} \le 1$ si O_{ij} est semi-sécable.

Il existe une autre terminologie introduite dans [Mauguière *et al.*, 2005]. Elle concerne les périodes d'indisponibilité permettant l'interruption d'opérations : périodes d'indisponibilité traversable et non-traversable (Figure 3). Ainsi, une période d'indisponibilité est dite *traversable* si elle permet l'interruption d'une opération ; bien entendu, cette interruption ne se fera pas si l'opération est non-préemptive. Une période d'indisponibilité qui ne permet pas l'interruption d'une opération est dite *non-traversable* ; dans ce cas, l'opération ne sera pas interrompue même si elle est préemptive.

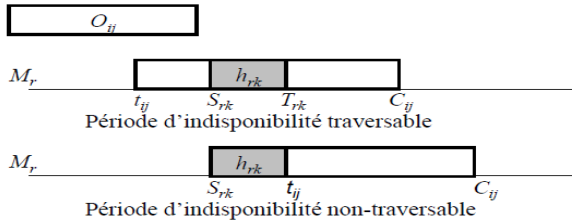

Figure 3. Période d'indisponibilité permettant l'interruption d'une opération

Nous associons un *coefficient de préemption* β_{ijk} à une opération O_{ij} qui doit être effectuée sur la machine M_r et qui peut être interrompue par une période d'indisponibilité h_{rk}. Cela représente le caractère préemptif de O_{ij} ou le caractère traversable de h_{rk}. Ainsi, $\beta_{ijk} = 0$ si h_{rk} est non-traversable ou O_{ij} est non-préemptive. Et $\beta_{ijk} = 1$ si h_{rk} est traversable et O_{ij} est préemptive. Dans ce cas, la position de l'opération O_{ij} par rapport à la période d'indisponibilité h_{rk}, dépend de son caractère sécable.

Exemple d'illustration : Le système de production est constitué de trois machines M_1,..,M_3. Quatre jobs J_1,..,J_4 doivent être effectués par ces machines. Les gammes opératoires des jobs sont les suivantes (les nombres mis entre parenthèses représentent les durées opératoires des jobs sur les machines associées) :

J_1 : M_1 (1) M_2 (2) M_3 (3)
J_2 : M_2 (1) M_1 (2) M_3 (3)
J_3 : M_3 (2) M_2 (1) M_1 (3)
J_4 : M_1 (4) M_3 (1) M_2 (1)

Une solution réalisable du problème sans périodes d'indisponibilité sur les machines est représentée sur le diagramme de Gantt de la Figure 4.a. Notons que le diagramme de Gantt se compose de lignes horizontales désignant les machines ; les opérations y sont représentées, en fonction des machines correspondantes, à partir de leur dates de début d'exécution, sous forme de barres ayant des longueurs proportionnelles à leur durées opératoires.

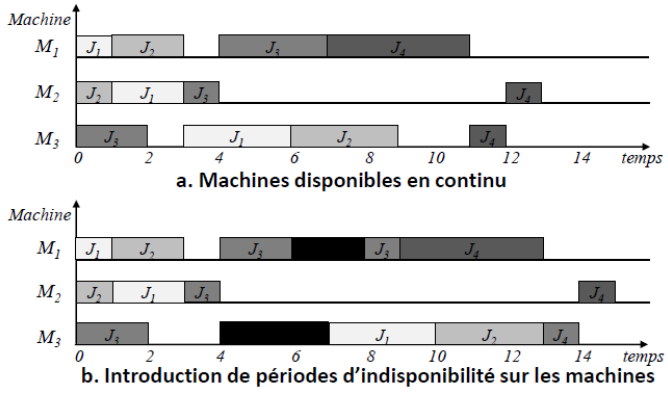

Figure 4. Exemple de système de production

Nous introduisons des périodes d'indisponibilité sur les machines M_1 et M_3 aux ¨périodes [6,8] et [4,7] respectivement. Ceci a un impact sur les dates de début et de fin de certaines opérations (Figure 4.b) ; par exemple : l'opération O_{33} est sécable, elle se termine donc à l'instant 9 au lieu de 7 ; l'opération O_{31} est non-préemptive, elle commence à l'instant 7 au lieu de 3 et se termine à l'instant 10 au lieu de 6. Quant à l'ordonnancement, il se termine donc à l'instant 15 au lieu de 13 ; la qualité de la solution est ainsi dégradée.

Le but est de déterminer les dates de début et de fin de chaque opération afin de minimiser le makespan C_{max} ou la somme des dates de fin des jobs $\sum_{i=1}^{n} C_{in_i}$ (que l'on dénote $\Sigma_i C_i$ dans la suite de l'article).

Le problème d'ordonnancement dans un atelier de type job-shop avec contraintes de disponibilité des ressources est NP-difficile au sens fort du fait que le problème sans périodes d'indisponibilité est fortement NP-difficile.

3 MODELISATION MATHEMATIQUE

La formulation que nous proposons est déduite de celle de [Applegate et Cook, 1991]. Dans [Azem, 2010], l'étude du cas non-préemptif montre que cette formulation donne de meilleurs résultats que celle utilisant la discrétisation du temps ; car cette dernière engendre un nombre considérable de variables et de contraintes. Par ailleurs, à notre connaissance, c'est le seul modèle qui inclut tous les problèmes présents dans la littérature définis par l'indisponibilité des ressources ; et ce grâce à l'introduction des coefficients de pénalité sur la préemption et les coefficients de préemption.

Les variables additionnelles suivantes sont introduites :

- $X_{ij,i'j'}$: Variable binaire servant à définir laquelle des opérations O_{ij} et $O_{i'j'}$ est effectuée avant l'autre. Elle vaut 1 si O_{ij} est effectuée avant $O_{i'j'}$ et 0 sinon,
- $Y_{ij,rk}$: Variable binaire pour déterminer si l'opération O_{ij} commence avant ou après la période d'indisponibilité h_{rk}. Elle est égale à 1 si O_{ij} commence avant h_{rk} et 0 sinon.
- $Z_{ij,rk}$: Variable binaire qui est égale à 1 si l'opération O_{ij} commence avant (i.e. $Y_{ij,rk}$ vaut 1) et finit après la période d'indisponibilité h_{rk} , et est égale à 0 sinon.

La formulation qui suit, où M est un nombre très grand, modélise la minimisation du makespan.

Ainsi, La fonction objectif (1) est la minimisation du makespan C_{max}. La contrainte (2) assure qu'une opération $O_{i'j'}$ qui succède à une opération O_{ij} dans la gamme opératoire du job ne peut commencer avant la fin de O_{ij}. Les contraintes (3) et (4) garantissent le non chevauchement des opérations O_{ij} et $O_{i'j'}$ devant s'effectuer sur la même machine, i.e. l'opération O_{ij} est exécutée avant ou après l'opération $O_{i'j'}$. La contrainte (5) assure que si $Y_{ij,rk} = 0$, alors l'opération O_{ij} doit être réalisée après la période d'indisponibilité h_{rk}. La contrainte (6) signifie que lorsque $Y_{ij,rk} = 1$, l'opération O_{ij} doit commencer avant la date de début de la période d'indisponibilité h_{rk}. Cependant, en fonction des valeurs de β_{ijk} et $Z_{ij,rk}$, elle peut soit se terminer avant h_{rk} ou après. Lorsque $\beta_{ijk} = 0$ (préemption non-autorisée), O_{ij} commence et finit avant le début de h_{rk} quelque soit la valeur de $Z_{ij,rk}$. Cependant, lorsque $\beta_{ijk} = 1$ (préemption autorisée), la fin de O_{ij} dépend de la valeur de $Z_{ij,rk}$. Lorsque $Z_{ij,rk} = 0$, O_{ij} commence et finit avant h_{rk} ; alors que lorsque $Z_{ij,rk} = 1$, O_{ij} commence avant h_{rk} et se termine après. Notons que lorsque $Y_{ij,rk} = 0$, la contrainte (6) est toujours satisfaite. La contrainte (7) traduit la possibilité d'interrompre O_{ij} lorsqu'elle commence avant h_{rk} (Figure 5). Les contraintes (8) et (9) fournissent des bornes pour $C_{ij} - t_{ij}$ qui correspondent à la durée de l'opération O_{ij} sur la machine M_r. En effet, lorsque O_{ij} n'est pas interrompue par h_{rk}, cette durée est au moins égale (en fait est égale) à la durée opératoire p_{ij}. Cependant, lorsque O_{ij} est interrompue par h_{rk}, cette durée est au moins égale (en fait est égale) à p_{ij} à laquelle on ajoute la durée p'_{rk} de la période d'indisponibilité et la proportion de l'opération à refaire ($p_{ij}+p'_{rk}+\alpha_{ij}(S_{rk}-t_{ij})$). Ceci est du au fait que :

$$C_{ij} = t_{ij} + (S_{rk} - t_{ij}) + p'_{rk} + (p_{ij} - (S_{rk} - t_{ij})) + \alpha_{ij}(S_{rk} - t_{ij})$$

$$Min\ C_{\max} \quad (1)$$

$$t_{i(j+1)} \geq C_{ij} \quad i=1,..,n\ ;\ j=1,..,n_i-1 \quad (2)$$

$$t_{ij} + M\ X_{ij,i'j'} \geq C_{i'j'} \quad \forall\ (O_{ij} \neq O_{i'j'})\ s.t.\ mr_{ij} = mr_{i'j'} \quad (3)$$

$$t_{i'j'} + M\ (1 - X_{ij,i'j'}) \geq C_{ij} \quad \forall\ (O_{ij} \neq O_{i'j'})\ s.t.\ mr_{ij} = mr_{i'j'} \quad (4)$$

$$t_{ij} - S_{rk} + M\ Y_{ij,rk} \geq p'_{rk} \quad \forall\ (h_{rk}, O_{ij})\ s.t.\ mr_{ij} = M_r \quad (5)$$

$$S_{rk} - t_{ij} + M\ (1 - Y_{ij,rk}) \geq (1 - \beta_{ijk}\ Z_{ij,rk})\ p_{ij} + \varepsilon\ \beta_{ijk}\ Z_{ij,rk} \quad \forall\ (h_{rk}, O_{ij})\ s.t.\ mr_{ij} = M_r \quad (6)$$

$$\beta_{ijk}\ Z_{ij,rk} \leq \beta_{ijk} Y_{ij,rk} \quad \forall\ (h_{rk}, O_{ij})\ s.t.\ mr_{ij} = M_r \quad (7)$$

$$C_{ij} - t_{ij} + \beta_{ijk} M Z_{ij,rk} \geq p_{ij} \quad \forall\ (h_{rk}, O_{ij})\ s.t.\ mr_{ij} = M_r \quad (8)$$

$$\beta_{ijk}[C_{ij} - t_{ij} + M(1 - Z_{ij,rk})] \geq \beta_{ijk}[\ p_{ij} + p'_{rk} + \alpha_{ij}(S_{rk} - t_{ij})] \quad \forall\ (h_{rk}, O_{ij})\ s.t.\ mr_{ij} = M_r \quad (9)$$

$$\beta_{ijk}[-t_{ij} + S_{rk} - M(1 - Z_{ij,rk})] \leq \beta_{ijk}(\ p_{ij} - \varepsilon) \quad \forall\ (h_{rk}, O_{ij})\ s.t.\ mr_{ij} = M_r \quad (10)$$

$$Y_{ij,r(k+1)} - Y_{ij,rk} \geq 0 \quad \forall\ (h_{rk}, O_{ij})\ s.t.\ mr_{ij} = M_r \quad (11)$$

$$C_{\max} \geq C_{in_i} \quad i=1,..,n \quad (12)$$

$$C_{ij} \geq t_{ij} + p_{ij} \quad i=1,..,n\ ;\ j=1,..,n_i \quad (13)$$

$$t_{ij} \geq 0 \quad i=1,..,n\ ;\ j=1,..,n_i \quad (14)$$

$$C_{ij} \geq 0 \quad i=1,..,n\ ;\ j=1,..,n_i \quad (15)$$

$$S_{rk} \geq ES_{rk} \quad \forall h_{rk} \quad (16)$$

$$S_{rk} \leq LS_{rk} \quad \forall h_{rk} \quad (17)$$

$$X_{ij,i'j'} \in \{0,1\} \quad \forall\ (O_{ij} \neq O_{i'j'})\ s.t.\ mr_{ij} = mr_{i'j'} \quad (18)$$

$$Y_{ij,rk} \in \{0,1\} \quad \forall\ (h_{rk}, O_{ij})\ s.t.\ mr_{ij} = M_r \quad (19)$$

$$Z_{ij,rk} \in \{0,1\} \quad \forall\ (h_{rk}, O_{ij})\ s.t.\ mr_{ij} = M_r \quad (20)$$

où (S_{rk}–t_{ij}) représente la partie de O_{ij} effectuée avant le début de h_{rk}, (p_{ij}–(S_{rk}–t_{ij})) la partie restante à exécuter et (α_{ij}(S_{rk}–t_{ij})) la partie à refaire. Ceci se produit lorsque t_{ij}<S_{rk}<t_{ij}+p_{ij} qui est remplacée par $t_{ij}+\varepsilon \leq S_{rk} \leq t_{ij}+p_{ij}-\varepsilon$ (contraintes (6) et (10)) et β_{ijk} = 1 (préemption autorisée). Le coefficient ε est utilisé car l'introduction d'inégalités strictes dans un modèle le rend non linéaire ; sa valeur doit être la plus petite possible pour garantir l'équivalence des deux expressions. La contrainte (11) signifie que, si l'opération O_{ij} précède la période d'indisponibilité h_{rk}, alors elle précède toutes les périodes d'indisponibilité qui succèdent à h_{rk}. De plus, si l'opération O_{ij} succède à la période d'indisponibilité $h_{r(k+1)}$, alors elle succède à toutes les périodes d'indisponibilités qui précèdent $h_{r(k+1)}$. La contrainte (12) indique que l'ordonnancement ne peut s'achever avant la fin de la dernière opération de chaque job. La contrainte (13) représente la borne des dates de fin des opérations. Elle est pertinente lorsque la machine est disponible en continu ; elle est redondante sinon. Les contraintes (14) à (20) bornent les variables du modèle. Les contraintes (16) et (17) ne sont ajoutées que lorsque les périodes d'indisponibilité varient dans des fenêtres de temps.

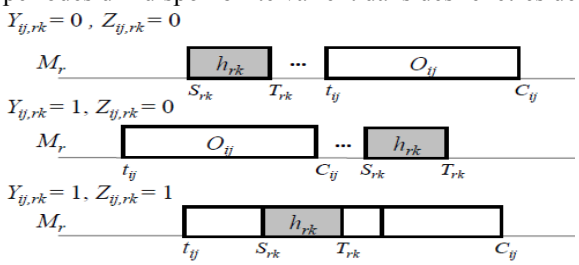

Figure 5. Position de l'opération O_{ij} en fonction des valeurs de $Y_{ij,rk}$ et $Z_{ij,rk}$

Remarques :

- Pour modéliser la minimisation de la somme des dates de fin des jobs, la fonction objectif (1) est remplacée par $Min\ \sum_{i=1}^{n} C_{in_i}$ et la contrainte (12) est supprimée.
- On peut introduire de la flexibilité sur les durées des périodes d'indisponibilité (selon si la priorité est accordée à la production plutôt qu'à la maintenance des machines) en associant un ensemble de valeurs à une durée (voir [Azem, 2010]).

4 HEURISTIQUES DE CONSTRUCTION

Cette section est consacrée aux heuristiques qui visent à trouver rapidement de bonnes solutions aux problèmes d'optimisation combinatoires NP-difficiles. Les heuristiques que nous proposons construisent un ordonnancement sur la base de diverses stratégies de décision. Le choix de ces stratégies est lié à la priorité imposée aux machines et/ou aux opérations, et la manière dont sont gérés les conflits entre les opérations et les périodes d'indisponibilité des machines.

4.1 Positionnement d'une opération dans un intervalle

Pour construire un ordonnancement, chaque opération doit être insérée dans un intervalle qui définit une période de disponibilité sur la machine. L'insertion de l'opération (par exemple, O_{ij}) dans un intervalle est définie en fonction de la date d'achèvement de l'opération précédente dans la gamme opératoire, la longueur de l'intervalle et la présence d'une période d'indisponibilité au début ou à la fin de l'intervalle. Après chaque insertion, les intervalles sont mis à jour.

Nous présentons les quatre cas A, B, C, D qui nous permettent de positionner une opération dans un intervalle. Les cas A et B modélisent la flexibilité d'une période d'indisponibilité ; tandis que le cas C modélise la préemption de l'opération. Nous présentons en premier lieu le cas D car il est le plus simple :

- Dans le cas D (illustré par la Figure 8), l'opération (O_{ij}) peut être insérée dans l'intervalle courant (I_{rl}). Soit elle occupe tout l'intervalle auquel cas ce dernier va disparaitre. Soit elle est placée strictement dans l'intervalle créant un nouveau (I'_{rl}). Soit elle est positionnée au début ou à la fin de l'intervalle ; ce qui réduit ce dernier.
- Dans le cas A (Figures 6 et 7), nous sommes en présence d'une période d'indisponibilité (h_{rk}) au début de l'intervalle courant (I_{rl}) et qui peut bouger dans sa fenêtre de temps. L'opération précédente ($O_{i(j-1)}$) dans la gamme opératoire doit se terminer au plus tard au début de l'intervalle courant (I_{rl}). Dans le cas A.1, l'indisponibilité peut être avancée et l'opération (O_{ij}) est placée après l'indisponibilité dans l'intervalle courant (I_{rl}). Dans le cas A.2, l'indisponibilité peut être reculée et l'opération est placée avant l'indisponibilité dans l'intervalle précédant ($I_{r(l-1)}$).
- Tout comme le cas A, le cas B traite aussi la flexibilité de la date de début de la période d'indisponibilité (h_{rk}). Cependant, contrairement au cas A, la période d'indisponibilité est située à la fin. La reculer permet d'insérer l'opération (O_{ij}) dans l'intervalle courant (I_{rl}). Il existe une similitude entre les cas A.2 et B, la seule différence est l'intervalle d'insertion ; car dans le cas B, l'opération précédente ($O_{i(j-1)}$) dans la gamme opératoire doit se terminer avant la date de fin de l'intervalle courant (I_{rl}). (Figures 6 et 7)
- Le cas C ressemble au cas B (voir Figures 6 et 7). La différence est que dans le cas C, la période d'indisponibilité (h_{rk}) est fixe mais traversable et l'opération (O_{ij}) est préemptive. Ainsi, l'opération peut être interrompue par la période d'indisponibilité et occupe une partie des intervalles courant et suivant (I_{rl}, $I_{r(l+1)}$).

Les traits discontinus verticaux sur les Figures 6 et 7, délimitent, pour chaque cas (B ou C), les zones de

ressemblances définies par les configurations de début de l'opération à insérer (O_{ij}) qui ont une incidence sur l'intervalle courant (I_{rl}) ; car dans l'une des deux configurations, l'intervalle disparait suite à l'insertion de l'opération. Le traitement de la période d'indisponibilité est le même dans chaque cas.

4.2 OIp - procédure d'insertion d'une opération dans un intervalle

On tente d'insérer l'opération au plus tôt dans la première période de disponibilité sur la machine en commençant à partir de l'instant 0 si c'est la première opération du job ou en se basant sur la date de fin de l'opération précédente dans la gamme opératoire. Une procédure générale mettant en œuvre la procédure OIp s'initialise par la décomposition de la disponibilité de chaque machine (M_r) en intervalles (en nombre de m_r+1) : $I_{r1}, I_{r2}, .., I_{rl}, .., I_{r(m_r+1)}$. La procédure OIp procède comme suit : L'étape 1 intègre le cas A, ensuite le cas D pour une insertion de l'opération dans l'intervalle. Le cas où l'opération ne peut être insérée entièrement dans l'intervalle est traité à travers les autres étapes. L'étape 2 représente le cas A tandis que les étapes 3 et 4 intègrent respectivement les cas B et C. Dans la dernière étape, on passe à l'intervalle suivant puisque l'intervalle courant ne convient pas.

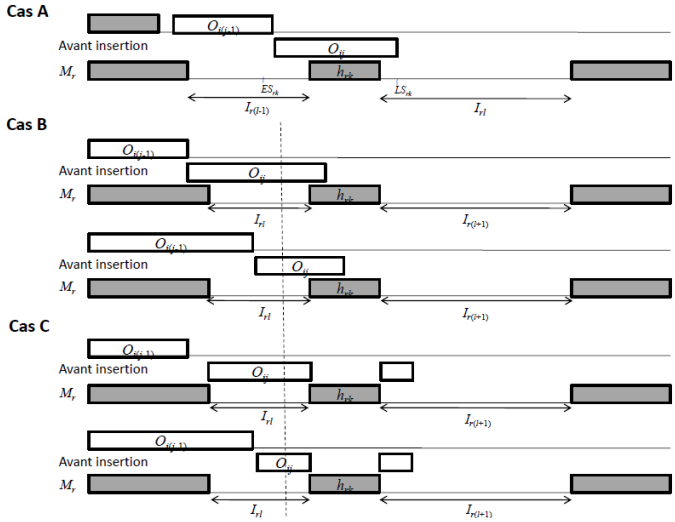

Figure 6. Comparaison entre les cas A, B et C avant insertion de l'opération O_{ij}

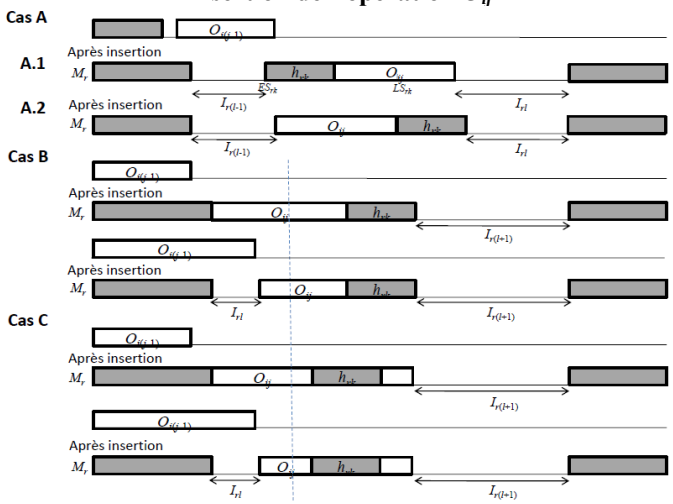

Figure 7. Comparaison entre les cas A, B et C après insertion de l'opération O_{ij}

La situation la plus appropriée est de réduire les longueurs de l'intervalle autant que possible car ils correspondent à des temps morts sur les machines.

Cet enchainement des cas dans OIp favorise la flexibilité des dates de début des périodes d'indisponibilité au détriment de la préemption. Les permutations des cas est possible définissant des stratégies différentes ; ainsi à titre d'exemple, l'ordre C, A, B traite d'abord la préemption ensuite la flexibilité. Contrairement à la modélisation qui peut traiter la flexibilité et la préemption en même temps, cette approche heuristique traite l'un ou l'autre ; ce qui constitue déjà un atout en comparaison aux heuristiques présentes dans la littérature.

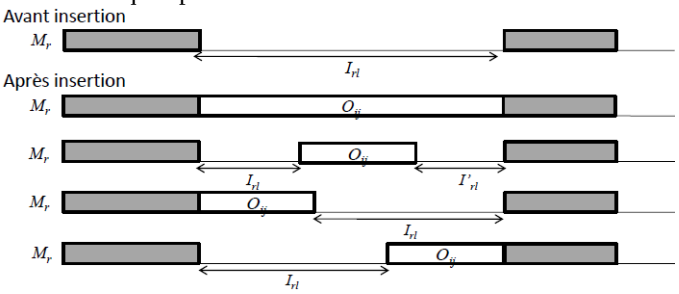

Figure 8. Insertion de l'opération O_{ij} selon le cas D

4.3 Heuristiques basées sur les jobs

Nous présentons trois méthodes JpH, OpH1, OpH2 qui donnent la priorité aux opérations des jobs (ou tâches). Dans JpH et OpH1, nous introduisons l'aspect aléatoire dans les séquences de départ des jobs et des opérations respectivement représentant l'ordre d'insertion des opérations ; on constate que OpH1 définit plus de configurations (les résultats numériques montrent que cela est pertinent pour C_{max} et non pour $\Sigma_i C_i$; puisque les résultats sont meilleurs avec JpH) ; et qu'elles dépendent fortement de ces séquences. Tandis que dans OpH2, ce choix de l'opération à insérer se fait selon une règle donnée. JpH et OpH1 peuvent être utilisées comme des blocs d'évaluation dans des méthodes qui utilisent plusieurs solutions à la fois pour obtenir de meilleures solutions comme cela est le cas dans un algorithme génétique. Quant à OpH2, elle peut être utilisée pour fournir une solution de départ pour des méthodes qui cherchent à améliorer la solution à chaque itération comme une recherche taboue. L'évaluation de la solution à chaque itération se fait avec JpH et OpH1 (selon la structure de la séquence de priorité).

Le principe de JpH et OpH1 est le suivant : on sélectionne en premier lieu une opération (selon l'ordre des jobs pour JpH ou des opérations pour OpH1) ; on insère ensuite l'opération sélectionnée sur la machine associée grâce à OIp et on l'enlève de la liste des opérations restantes. Le processus est réitéré jusqu'à ce que toutes les opérations soient ordonnancées. Comme JpH et OpH1 dépendent fortement de la séquence de départ, chacune d'entre elle est exécutée plusieurs fois, dans la partie expérimentale, pour évaluer sa performance.

Concernant OpH2, elle définit en premier lieu la liste d'opérations prêtes à être ordonnancées pour sélectionner celle qui satisfait la règle choisie. Il faut noter que la règle MWKR (voir remarques) est utilisée pour départager plusieurs opérations. L'opération sélectionnée est ensuite insérée sur la machine associée grâce à OIp et enlevée de la liste des opérations restantes. Le processus est réitéré jusqu'à ce que toutes les opérations soient ordonnancées. La même solution est obtenue pour toute exécution.

Remarques :

- Concernant JpH, nous considérons, dans cet article, le cas où on ne traite le job suivant dans la séquence qu'une fois que les opérations du job courant sont insérées dans leurs

Tableau 1. Résultats des tests du modèle mathématique pour la minimisation de C_{max} dans le cas non-préemptif

Problème	Périodes d'indisponibilité fixées				Périodes d'indisponibilité flexibles			
	Borne inf.	Solu-tion	Ecart (%)	CPU (sec.)	Borne inf.	Solu-tion	Ecart (%)	CPU (sec.)
5m5n1	895	895	0	1.93	825	825	0	0.09
5m5n2	1096	1096	0	0.03	1076	1076	0	0.02
5m5n3	1070	1070	0	0.16	1034	1034	0	0.23
5m5n4	1147	1147	0	0.15	1108	1108	0	0.27
5m5n5	1202	1202	0	0.20	1182	1182	0	0.51
5m10n1	1361	1361	0	65	1300	1300	0	43
5m10n2	1435	1455	1.37	3167	1271	1400	9.21	1519
5m10n3	1527	1607	4.98	2822	1274	1587	19.72	1939
5m10n4	1537	1537	0	609	1478	1492	0.94	1324
5m10n5	1355	1415	4.24	3437	1372	1372	0	174
10m10n1	1887	1948	3.13	2509	1771	1859	4.73	2073
10m10n2	1917	1917	0	64	1839	1839	0	664
10m10n3	1855	1875	1.07	3028	1773	1773	0	466
10m10n4	1772	1772	0	136	1690	1690	0	669
10m10n5	1847	1975	6.48	2858	1725	1833	5.89	1814
10m15n1	1717	2089	17.81	1750	1591	2103	24.35	2708
10m15n2	1880	2316	18.83	2562	1703	2388	28.69	1893
10m15n3	1721	2261	23.88	1844	1637	2221	26.29	2645
10m15n4	1744	2254	22.63	1799	1488	2120	29.81	2679
10m15n5	1746	2282	23.47	1640	1627	2225	26.88	2807

machines respectives. Bien entendu, il peut exister d'autres façon de procéder ; l'une d'entre elles est présentée dans [Azem, 2010].

- Concernant OpH2, le choix de la règle pour sélectionner l'opération à insérer est déterminant. Ce choix se fait parmi six règles r. Ainsi, sélectionner l'opération dont :
  (1) La date de début est la plus petite,
  (2) La date de fin est la plus petite,
  (3) Le temps libre sur la machine est le plus petit et positif,
  (4) Le temps libre de début sur la machine est le plus petit. Si la condition est satisfaite par plus d'une opération, choisir la règle (3),
  (5) La date de début du premier intervalle de disponibilité sur la machine après la fin de l'opération précédente sur la gamme opératoire est la plus petite,
  (6) La somme des durées opératoires des opérations restant à être exécutées sur la machine est la plus grande.

  Les valeurs associées aux règles (1), (2), (3), (4) et (6) sont obtenues en simulant l'exécution de la procédure OIp. Aucune mise à jour des dates de début et de fin des opérations, des périodes d'indisponibilité et des intervalles de disponibilité n'est effectuée. Une fois l'opération sélectionnée, elle est insérée dans l'intervalle associé en utilisant OIp.
- La règle MWKR (Most WorK Remaining) donne la propriété à l'opération du job pour lequel la charge de travail restant à être effectuée est la plus grande ; le but étant d'équilibrer l'exécution des jobs correspondant à ces opérations.

4.4 Heuristiques basées sur les machines

Ces méthodes MOpH1 et MOpH2 sont inspirées de OpH1 et OpH2 respectivement, la première priorité étant donnée aux machines : on dénote par « machine prête » la machine sélectionnée. Elle correspond à la machine dont la date de disponibilité est la plus petite et dont l'ensemble d'opérations prêtes à être ordonnancées est non vide. Nous définissons la date de disponibilité d'une machine comme étant la date de fin de la dernière opération effectuée sur cette machine. En effet, comme il existe des périodes d'indisponibilité sur la machine, cette dernière peut avoir des intervalles de disponibilité situés avant la dernière opération exécutée. Considérer la date de disponibilité comme étant le début du premier intervalle de disponibilité peut conduire à un tout petit intervalle qui ne peut contenir aucune opération. Ainsi, cette machine peut être sélectionnée plusieurs fois avant de passer à une autre machine, par ailleurs, l'écart entre cet intervalle et l'intervalle d'insertion de l'opération peut être très grand.

MOpH1 et MOpH2 procèdent comme suit : le processus suivant est réitéré jusqu'à ce que toutes les opérations soient **o**rdonnancées. Après avoir défini la machine prête, la liste des opérations prêtes à être ordonnancées est établie, dans laquelle est sélectionnée l'opération à insérer (selon le principe de OpH1 ou OpH2). Une fois cette opération insérée sur la machine associée grâce à OIp, la date de disponibilité de la machine est mise à jour et l'opération est enlevée de la liste des opérations restantes.

5 RESULTATS NUMERIQUES

Des expérimentations ont été menées sur quatre classes de cinq instances générées de problèmes (5,5), (5,10), (10,10), (10,15), où les paires représentent (nombre de machines, nombre de jobs). Les résultats sont discutés dans cette section ; cependant, seuls quelques tableaux sont donnés à titre illustratif (pour plus de détails, voir [Azem, 2010]).

Pour ce qui est du modèle mathématique, il a été résolu avec le solveur IBM ILOG CPLEX. La limite de temps de la résolution en nombres entiers pour chaque instance a été fixée à 60 minutes. Le tableau 1 récapitule les résultats des tests pour la minimisation de C_{max} dans le cas non-préemptif lorsque les dates de début des périodes d'indisponibilité sont fixes et non fixes. La première colonne contient le nom de l'instance qui est du type *XmYnZ*, où *X*, *Y* et *Z* sont respectivement le nombre de machines, le nombre de jobs et le numéro de l'instance dans la classe. Les colonnes de 2 à 5 (resp. de 6 à 9) présentent les résultats lorsque les dates de débuts des périodes d'indisponibilité sont fixes (resp. flexibles). Les colonnes 2 et 6 correspondent à la meilleure borne inférieure, et les colonnes 3 et 7 à la fonction objectif de la meilleure solution. Les colonnes 4 et 8 présentent l'écart d'optimalité (en %) entre la meilleure solution et la meilleure borne. Les colonnes 5 et 9 donnent le temps CPU (en secondes).

Aussi bien pour la minimisation de C_{max} que celle de $\Sigma_i C_i$, CPLEX a rapidement résolu de façon optimale la plupart des instances jusqu'à 10 machines et 10 jobs pour le cas non-préemptif et jusqu'à 5 machines et 5 jobs pour les autres cas. Mais, en général, cela prend beaucoup de temps pour résoudre les problèmes à l'optimalité, et seulement une solution réalisable est souvent obtenue.

Les résultats des tests montrent qu'en général, pour les heuristiques JpH, OpH1 et MOpH1, plus le nombre d'itérations est grand, meilleures sont les solutions. Bien entendu, le temps CPU est proportionnel au nombre d'itérations. Cependant, pour les instances de la classe (5,5), les résultats sont stables et peuvent êtres optimaux ; et dans le cas de périodes d'indisponibilité fixes, la plupart sont

optimaux. Ceci suggère la pertinence de les intégrer dans un algorithme qui utilise plusieurs solutions à la fois, comme un algorithme génétique.

En règle générale, les solutions sont optimales pour la plupart des instances de la classe (5,5), l'écart d'optimalité entre les meilleures solutions des heuristiques est soit égal à celui du modèle mathématique, soit inférieur ou égal à 10 %. Les solutions sont parfois meilleures. Les temps de calcul pour toutes les instances n'excèdent pas 25 secondes pour JpH et OpH1 pour 100 000 itérations et 175 secondes pour MOpH1. Ils sont inférieurs à 0,02 secondes pour OpH2 et MOpH2.

Concernant le modèle mathématique, les résultats confirment que cela peut être efficace de permettre le déplacement des dates de début des périodes d'indisponibilité dans des fenêtres de temps car de meilleures solutions sont obtenues. On peut aussi vérifier, qu'à cause de la pénalité induite par la réexécution entière de l'opération lorsqu'elle est interrompue par une période d'indisponibilité et dans le contexte déterministe d'indisponibilités connues à l'avance, il est plus pertinent de ne pas interrompre l'opération. En effet, pour le cas non-sécable, les résultats des tests montrent que l'interruption d'une opération par une période d'indisponibilité n'est pas permise que lorsque le créneau sur la machine, avant la période d'indisponibilité, ne peut pas être utilisé pour effectuer une autre opération. Ainsi, les cas non-préemptif et non-sécable sont équivalents. Les résultats montrent aussi qu'il est plus intéressant de permettre la préemption entre des opérations et des périodes d'indisponibilité car moins de temps est nécessaire pour effectuer tous les jobs. En effet, pour le cas sécable, comme l'interruption est autorisée sans pénalité, toutes les opérations peuvent être exécutées au plus tôt, donc moins de temps libre sur les machines ; pour le cas semi-sécable un gain est réalisé en autorisant une pénalité inférieure à 1 (ceci correspond à 100% de la durée opératoire).

Pour ce qui est des heuristiques, la dominance entre les problèmes non-préemptif, sécable, non-sécable et semi-sécable est conservée pour le cas d'indisponibilités fixes. Dans le cas d'indisponibilités flexibles, les mêmes valeurs de critères sont souvent obtenues entre les problèmes non-préemptif, sécable, non-sécable et semi-sécable ; on remarque aussi qu'il y a moins de solutions optimales. Ceci peut s'expliquer d'une part par le fait que, contrairement au modèle mathématique, certaines configurations ne peuvent être explorées. Cependant, à partir du moment où le problème est NP-difficile, il est aussi difficile de trouver une bonne solution en un temps raisonnable. Ceci peut s'expliquer aussi par le fait que l'ordre des cas A, B, C dans la procédure OIp favorise la flexibilité de la période d'indisponibilité au détriment de la préemption. En prenant à titre d'exemple l'heuristique OpH2, dans le cas d'opérations sécables et des périodes d'indisponibilité traversables, à travers toutes les règles, cet ordre donne de meilleurs résultats que l'ordre C, A, B, dans lequel la préemption est préférée, bien qu'aucune pénalité n'est induite du fait que nous considérons le cas d'opérations sécables.

Concernant l'ordre de dominance des règles pour OpH2 et MOpH2, il est établi par rapport aux nombres des meilleures et des moins bonnes solutions obtenues par chaque heuristique à travers les problèmes non-préemptif, sécable, non-sécable et semi-sécable, dans le cas de périodes d'indisponibilités fixes ou flexibles, pour minimiser C_{max} et $\Sigma_i C_i$. Une agrégation des ordres de dominance entre les cas non-préemptif, sécable, non-sécable et semi-sécable permet d'établir l'ordre global par type de périodes d'indisponibilité. La meilleure règle pour C_{max} est la règle (1) ; tandis que la meilleure pour $\Sigma_i C_i$ est la règle (2) ; la plupart des meilleures solutions sont obtenues grâce à ces règles. Concernant OpH2, la plus mauvaise règle pour C_{max} est la règle (2) ou (3) (resp. (2) or (4)) en fonction du caractère préemptif des opérations et des périodes d'indisponibilité fixes (resp. flexibles) ; tandis que la plus mauvaise règle pour $\Sigma_i C_i$ est la règle (5) (resp. (3)) pour des périodes d'indisponibilité fixes (resp. flexibles). Concernant MOpH2, la plus mauvaise règle pour C_{max} est la règle (2) ou (4) (resp. (2)) en fonction du caractère préemptif des opérations et des périodes d'indisponibilité fixes (resp. flexibles); tandis que la plus mauvaise règle pour $\Sigma_i C_i$ est la règle (6).

Par ailleurs, on remarque que, hormis pour certaines instances, les meilleurs solutions sont obtenues lorsque les périodes d'indisponibilité sont placées au début de leurs fenêtres de temps.

Le tableau 2 présente l'ordre de dominance des heuristiques pour les deux critères objectif C_{max} et $\Sigma_i C_i$ dans le cas d'opérations non-préemptives et des périodes d'indisponibilité fixes (les tableaux 3 et 4 présentent les résultats de la comparaison de ces heuristiques par instance) et le cas d'opérations sécables et des périodes d'indisponibilité flexibles. Ainsi les meilleures heuristiques pour C_{max} sont OpH1 et MOpH1; OpH1 est dominante dans le cas d'opérations non-préemptives et des périodes d'indisponibilité fixes, et MOpH1 est légèrement dominante dans le cas d'opérations sécables et des périodes d'indisponibilité flexibles. La moins bonne heuristique pour C_{max} est MOpH2 dans le cas d'opérations sécables et des périodes d'indisponibilité flexibles. Cependant, dans le cas d'opérations non-préemptives et des périodes d'indisponibilité fixes, c'est JpH qui est la moins bonne heuristique ; alors que cette dernière est largement dominante pour $\Sigma_i C_i$. Pour ce critère, OpH1 et MOpH1 sont presque équivalentes avec une légère dominance de OpH1. Les moins bons résultats sont de loin ceux obtenus avec OpH2 et MOpH2. Cependant, MOpH2 est légèrement moins bonne que OpH2. Ces résultats prouvent la pertinence d'introduire l'aspect aléatoire dans le choix des séquences initiales des jobs et des opérations.

Tableau 2. Ordres de dominances des heuristiques

Ordre	C_{max}		$\Sigma_i C_i$	
	Opérations non-préemptives Indisponibilités fixes	Opérations sécables Indisponibilités flexibles	Opérations non-préemptives Indisponibilités fixes	Opérations sécables Indisponibilités flexibles
1	OpH1	*MOpH1*	**JpH**	**JpH**
2	MOpH1	*OpH1*	*OpH1*	*OpH1*
3	OpH2	JpH ou OpH2	*MOpH1*	*MOpH1*
4	MOpH2	JpH ou OpH2	OpH2	OpH2
5	JpH	MOpH2	MOpH2	MOpH2

Gras : forte dominance
Italique : légère dominance

La résolution du modèle mathématique montre que la relaxation linéaire de la minimisation de $\Sigma_i C_i$ est meilleure que celle de C_{max} car l'écart d'optimalité entre les solutions est plus petit.

Compte tenu de la rapidité d'exécution des heuristiques, il serait peut-être préférable d'intégrer toutes les heuristiques dans une procédure et de choisir la meilleure solution pour chaque instance.

Tableau 3. Comparaison des heuristiques pour la minimisation de C_{max} dans le cas d'opérations non-préemptives et des périodes d'indisponibilité fixes

Problème	JpH	OpH1	OpH2	MOpH1	MOpH2	Model. Math.
5m5n1	*961~*	**895***	**895***	**895***	**895***	895*
5m5n2	*1112~*	**1096***	**1096***	1112~	*1112~*	1096*
5m5n3	*1157~*	**1070***	1122~	**1070***	1122~	1070*
5m5n4	**1147***	**1147***	1237~	1205~	*1255*	1147*
5m5n5	**1202***	**1202***	*1290~*	**1202***	1280~	1202*
5m10n1	1471~	**1420~**	*1560*	**1420~**	*1560*	1361*
5m10n2	1521~	**1455~**	1495~	1485~	*1635*	1455
5m10n3	1722~	**1607~**	1877	1632~	*1884*	1607
5m10n4	1619~	**1557~**	*1767*	1577~	1638~	1537*
5m10n5	1515~	**1425~**	1615	1435~	*1625*	1415
10m10n1	*2226*	**2030~**	2121~	2047~	2141~	1948
10m10n2	2202	**2022~**	2214	**2022~**	*2296*	1917*
10m10n3	*2098*	**1912~**	2062~	**1912~**	2072	1875
10m10n4	1990	**1860~**	1990	1904~	*2110*	1772*
10m10n5	2105~	**1975~**	*2115*	**1975~**	2055~	1975
10m15n1	*2550*	2320	2420	**2274~**	2420	2089
10m15n2	*2742*	2490~	2585	**2480~**	2533~	2316
10m15n3	2513	2423~	*2621*	**2401~**	2551	2261
10m15n4	*2532*	2316~	2452~	**2312~**	2494	2254
10m15n5	*2662*	2462~	2445~	**2422~**	2515~	2282

Gras : meilleure solution *Italique* : moins bonne solution
* : solution optimale
~ : écart d'optimalité entre la meilleure solution et celle du modèle mathématique ≤ 10%

Tableau 4. Comparaison des heuristiques pour la minimisation de $\Sigma_i C_i$ dans le cas d'opérations non-préemptives et des périodes d'indisponibilité fixes

Problème	JpH	OpH1	OpH2	MOpH1	MOpH2	Model. Math.
5m5n1	3748~	**3652***	*4094*	3874~	3947~	3652*
5m5n2	**4669***	**4669***	*4823~*	4706~	4706	4669*
5m5n3	**4008***	**4008***	4735	4130~	*4779*	4008*
5m5n4	**4584***	**4584***	4890~	4630~	*4930*	4584*
5m5n5	**4663***	**4663***	*5229*	**4663***	5064~	4663*
5m10n1	**10304~**	10759~	11312	10574~	*11876*	9850*
5m10n2	**11036~**	11775	12897	11817	*12929*	10637*
5m10n3	**11594~**	12234~	12639	12057~	*12782*	11072
5m10n4	**11353~**	11813	*11839*	11683~	*11839*	10630*
5m10n5	**10179~**	10410~	11473	10898	*11457*	9706*
10m10n1	**17308~**	17935~	18412~	17896~	*19697*	16878
10m10n2	**16539~**	17342~	*19418*	17247~	17087~	15777*
10m10n3	16238~	16641~	*18538*	**16235~**	18165	15352*
10m10n4	**16143~**	16331~	*18004*	16571~	17638	15374*
10m10n5	**16132~**	16795~	*17163*	16559~	16731~	15625*
10m15n1	**28661~**	*30071*	29663~	29703~	29325	27199
10m15n2	**30743~**	32194~	33848	32021~	*34293*	29186
10m15n3	**28814+**	31213+	*31308+*	30698+	31147+	u
10m15n4	**28436~**	30781	30935	30168~	*32854*	27423
10m15n5	**30114~**	*31787*	30911~	31194~	31267~	28609

Gras : meilleure solution *Italique* : moins bonne solution
* : solution optimale u : solution inconnue
~ : écart d'optimalité entre la meilleure solution et celle du modèle mathématique ≤ 10%
+ : solution meilleure que celle du modèle mathématique

6 CONCLUSION

Dans ce papier, un modèle mathématique et des heuristiques ont été présentés pour le problème d'ordonnancement dans un atelier de type job shop avec contraintes de disponibilité des ressources ; la préemption entre les opérations et les périodes d'indisponibilité est modélisée ainsi que la flexibilité des périodes d'indisponibilité. Un modèle mathématique a été proposé. Les résultats numériques montrent que permettre la préemption et introduire de la flexibilité pour les périodes d'indisponibilité sont pertinents, non seulement pour modéliser la réalité industrielle, mais aussi pour obtenir de meilleures solutions aux problèmes. Cependant un solveur standard atteint rapidement ses limites pour la résolution de problèmes de grande taille. Le modèle permet aussi d'évaluer les solutions fournies par nos méthodes heuristiques et d'avoir de bonnes bornes théoriques.

Les heuristiques que nous présentons construisent un ordonnancement sur la base de diverses stratégies de décision. Les difficultés sont liées à la sélection d'une opération à insérer sur la machine associée, à la sélection de la période de disponibilité sur la machine qui peut contenir l'opération, et à l'intégration de la préemption des opérations et de la flexibilité des périodes d'indisponibilité.

Deux types de méthodes sont suggérés : des heuristiques basées sur les jobs et d'autres basées sur les machines. La façon dont ces méthodes pourraient être réutilisées dans des méthodes approchées améliorantes comme un algorithme génétique, pour obtenir de meilleurs résultats, est donnée dans [Azem, 2010].

7 REMERCIEMENTS

Nous remercions le gouvernement du Luxembourg (plus particulièrement le Fonds National de la Recherche ; et le Ministère de la Culture, de l'enseignement Supérieur et de la Recherche) pour le financement de ces travaux ; ainsi que l'Ecole des Mines de Saint-Etienne et l'Université du Luxembourg pour les moyens investis dans cette recherche.

8 REFERENCES

Aggoune, R., (2004) Minimizing the makespan for the flow shop scheduling problem with availability constraints. E*uropean Journal of Operational Research*, 153, pp. 534-543.

Applegate, D., Cook, W., (1991) A Computational Study of the Job-shop Scheduling Problem, *ORSA Journal of Computing*, 3, pp. 149-156.

Azem, S., (2010) Ordonnancement des systèmes flexibles de production sous contraintes de disponibilités des ressources. *Thèse de doctorat*. Ecole Nationale Supérieure des Mines de Saint-Etienne, Gardanne, France.

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

Lee, C.Y., (1997) Minimizing the makespan in two-machine flowshop scheduling problem with an availability constraint. *Operations Research Letters*, 20, pp. 129-139.

Lee, C.Y., (1999) Two-machine flowshop scheduling with availability constraints. *European Journal of Operational Research*, 114, pp. 420-429.

Mauguière, Ph., Billaut, J.C., Bouquard, J.L., (2005) New single machine and job-shop scheduling problems with availability constraints. *Journal of Scheduling*, 8, pp. 211-231.