Méthodes exactes pour la résolution d'un problème d'ordonnancement Open-shop avec contraintes de ressources

Omar Souissi, Frédéric Dugardin, Farouk Yalaoui

11ème Conférence Internationale de MOdélisation, Optimisation et SIMulation - MOSIM'16 - 22 au 24 Aout 2016 - Montréal - Canada

Méthodes exactes pour la résolution d'un problème d'ordonnancement Open-shop avec contraintes de ressources

O. Souissi, F. Dugardin, F. Yalaoui

LOSI / Universite de Technologie de Troyes
UMR 6281 12 rue Marie Curie, CS42060,
10004 Troyes Cedex, France
omar.souissi@utt.fr, frederic.dugardin@utt.fr, farouk.yalaoui@utt.fr

RÉSUMÉ : *Dans le présent papier nous considérons le problème d'ordonnancement dans un atelier Open-shop avec contraintes de ressources. La plupart des problèmes d'ordonnancement sont NP-difficile, les chercheurs ont ainsi favorisé en grande majorité le développement d'heuristiques et métaheuristiques au détriment des méthodes exactes. Dans un contexte où les calculateurs haute performance sont en amélioration continue, il redevient intéressant d'explorer des méthodes exactes. Ici, nous nous concentrons sur le développement de méthodes exactes pour la résolution d'un problème d'ordonnancement dans un atelier Open-shop. Nous développons d'abord un Programme Linéaire en Nombre Entiers (PLNE) accéléré en vue de l'optimation de la durée du flux total*. Par la suite, nous exposons une nouvelle borne inférieure obtenu en optimisant la durée du flux total du problème Open-shop relaxé. Les résultats expérimentaux ont permis de valider les performances du PLNE accéléré en comparaison avec le PLNE original. Par ailleurs, nous avons montré que la relaxation lagrangienne du PLNE original produit une borne inférieure de bonne qualité.*

MOTS-CLÉS : *Optimisation combinatoire, Programmation linéaire en nombre entiers, Colonies de fourmis (ACO), Relaxation lagrangienne, Ordonnancement Open-shop*

1 INTRODUCTION

L'ordonnancement représente un des domaines les plus étudiés en recherche opérationnelle. En effet, depuis l'émergence dudit domaine dans les années 50, le nombre de publications scientifiques n'a cessé d'augmenter. L'un des travaux initiateurs dans le domaine est nul doute le modèle de production publié par Johnson en 1954 (Johnson 1954) qui aujourd'hui est communément appelé Flow Shop.

Un problème d'ordonnancement comporte deux éléments fondamentaux: les tâches et les ressources. Une tâche est composée d'opérations nécessitant un certain nombre d'unité de temps et d'unité de ressources en vue d'être ordonnancées. Une ressource peut être technique ou humaine et peut être limitée ou non. Dans le présent travail, nous considérons les ressources humaines en prenant en compte les compétences maitrisées par chaque opérateur en vue de pouvoir réaliser un certain nombre de tâches.

La dernière décennie a connu un nombre très important de travaux dans le domaine d'ordonnancement appliqué notamment à l'optimisation des processus indutriels. Une part non négligeable de ces travaux a traité la problématique de l'ordonnancement des ateliers Open Shop (OSSP).

L'OSSP est un problème industriel complexe qui a émergé afin de faire face à la problématique d'ordonnancement de groupes d'opérations sur différentes machines lorsque celles-ci ne sont pas liées par une contrainte de précédence. Ainsi, dans un OSSP il y a moins de contraintes que dans le cas des problématiques Flow Shop et Job Shop. L'OSSP avec plus de trois machines et en considérant la contrainte de non-préemption est NP difficile(Gonzalez & Sahni 1976). L'OSSP classique possède un espace de solution très large. En effet, avec n "tâches" et m "machines" le nombre maximum de solutions s'élève à n*m!.

Notre papier est structuré comme suit. Dans la Section 2 nous présentons des contributions représentatives dans le domaine. Dans la Section 3 nous decrivons la problématique étudiée. Une méthode exacte accélérée est présentée dans la Section 4. Dans la Section 5, nous introduisons une nouvelle borne inférieure pour l'OSSP avec contraintes de ressources. Les résultats expérimentaux basés sur plusieurs instances sont exposés dans la Section 6. Enfin, dans la

*Flux d'une tache= Date de fin d'exécution - date de disponibilité

Section 7 nous récapitulons les contributions de notre papier et nous présentons les perspectives pour nos futurs travaux.

2 ETAT DE L'ART

L'ordonnancement est l'un des axes clés en vue de l'amélioration d'un processus manufacturier. Parmi les problèmes d'ordonnancement l'OSSP n'est pas celui qui est le plus étudié. Gonzalez et Sahni (Gonzalez & Sahni 1976) ont développé un algorithme polynomial pour la minimisation du makespan dans le cas de deux machines. Dans (Kyparisis & Koulamas 1997), les auteurs ont développé un algorithme polynomial pour le cas d'un OSSP avec deux machines. Brucker al ont proposé dans (Brucker, Hurink, Jurish & Wostmann 1997) un algorithme séparation et évaluation pour la résolution du cas général avec m machines. (Hossein & Doulabi 2010) et (Ng, Cheng & Yuan 2003) ont présenté respectivement une approche pour la minimisation de la somme pondérée des retards de toutes les tâches et la minimisation de la somme pondérée du nombre de tâches en retards. La minimisation du retard total de toutes les tâches a été étudiée dans (Naderi, Ghomi Fatemi, Aminnayeri & Zandieh 2011), en effet, les auteurs ont proposé une résolution par PLNE. Achugbue et Chin (Achugbue & Chin 1982) ont prouvé quant à eux que l'OSSP est NP difficile, même avec juste deux machines, et ce quand la minimisation du flux total est considérée comme fonction objectif.

Tenant compte de la complexité des OSSPs, les chercheurs se sont plus concentrés sur les heuristiques ainsi que les métaheuristiques. Dans (Liaw 1999) et (Liaw 2000) l'auteur propose respectivement un algorithme de recherche Tabou ainsi qu'un algorithme génétique hybrid pour la résolution d'un OSSP. Un algorithme génétique tout aussi performant a été exposé dans (Prins 2000). Dans (Blum 2005) l'auteur propose une approche hybrid en utilisant conjointement les algorithmes colonies de fourmis et recherche en faisceau. Roshanaei et al ont étudié dans (Roshanaei, Esfehani & Zandieh 2010) le cas d'un OSSP non préemptif en prenant en compte les durées de réglage des machines et considérant la minimisation du makespan comme fonction objectif. Dans (Huang & Lin 2011), les auteurs ont développé un algorithme de colonies d'abeilles. Campos et al ont proposé dans (Ciro, Dugardin, Yalaoui & Kelly 2015b) un algorithme de colonies de fourmis et logique floue afin de résoudre un OSSP avec contrainte de ressources. (Zobolas, Tarantilis & Ioannou 2008) et (Anand & Panneerselvam 2015) proposent une revue exhaustive concernant les approches de résolution des problèmes OSSPs.

Notre observation principale à l'issue de notre étude de l'état de l'art est le nombre faible de publications concernant les approches exactes pour la résolution des OSSPs. La particularité de notre travail réside dans notre intérêt spécifique pour les méthodes exactes en vue de la résolution d'un OSSP avec contraintes de ressources.

3 DESCRIPTION DU PROBLEME

Dans le présent travail, nous considérons un OSSP avec contraintes de ressources. Nous visons à minimiser la durée du flux total. L'OSSP étudié est composé de m machines (j = 1 to m) et une liste de tâches (i = 1 to n) où chaque tâche contient au plus "m" opérations qui peuvent être exécutées dans un ordre quelconque. Chaque opération O_{ij} doit être exécutée dans une machine convenable et disponible avec un temps d'exécution p_{ij} et un temps de réglage s_{ij}. De même chaque opération O_{ij} doit être exécutée par un opérateur disponible w qui maîtrise les compétences nécessaires pour effectuer ladite opération.

Dans le cadre de notre étude, nous considérons les hypothèses suivantes:

- la préemption n'est pas permise
- les temps de réglage et d'exécution d'une opération donnée sont considérés de manière séparée.
- le réglage peut être débuté dès que la machine est disponible
- chaque machine peut exécuter uniquement une opération à la fois
- une tâche donnée peut être exécutée sur une seule machine à la fois
- les machines sont disponibles durant toute la période d'ordonnancement
- les tâches peuvent être exécutées en fonction de leur date de disponibilité

Dans la suite du document, nous présentons d'abord une méthode exacte accélérée pour la minimisation de la durée du flux total. Ensuite nous exposons une nouvelle borne inférieur obtenue en optimisant la durée du flux total du problème Open-shop relaxé.

4 ACCELERATION DU PLNE EN UTILISANT ACO

Dans la présente section, nous présentons dans un premier temps un PLNE et un algorithme de colonie de fourmis pour la résolution de l'OSSP avec contraintes de ressources. Par la suite, nous exposons une approche qui intègre conjointement lesdits algorithmes, et ce dans le but d'accélérer le PLNE original lors de la résolution avec l'outil CPLEX.

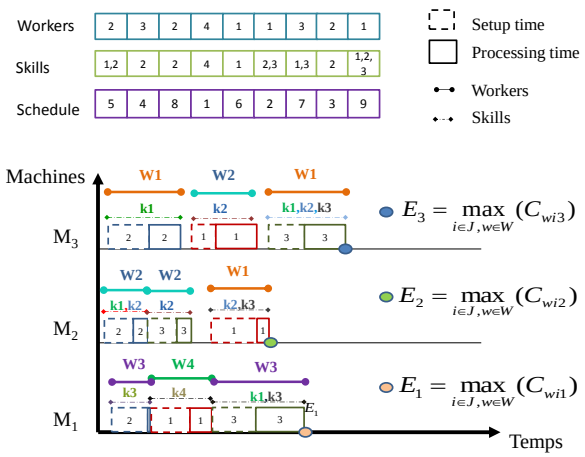

Figure 1: Exemple d'un OSSP avec contraintes de ressources (Ciro et al. 2015b)

4.1 PLNE Original

Ici, nous décrivons le modèle (Ciro, Dugardin, Yalaoui & Kelly 2015a) pour la résolution du problème décrit Section 3. Nous adopterons la notation suivante.

J	Liste des tâches $J = \{1, ..., N\}$
L	Liste des machines $L = \{1, ..., M\}$
W	Liste des opérateurs $W = \{1, ..., N_W\}$
S	Liste des compétences $S = \{1, ..., N_S\}$
i, j	Indice de tâche, $i,j = \{1, ..., N\}$
m, m'	Indice de machine, $m,m' = \{1, ..., M\}$
w	Indice d'opérateur, $w = \{1, 2, 3, ..., N_W\}$
s	Indice de compétence, $s = \{1, 2, 3, ..., N_S\}$
G	Un très grand nombre
C_{wim}	Date de fin d'exécution de la tâche i sur la machine m effectuée par l'opérateur w
C_i	Date de fin d'exécution de la tâche i
SE_{im}	Durée de réglage pour effectuer la tâche i sur la machine m
p_{im}	Durée d'exécution de la tâche i sur la machine m
r_i	Date de disponibilité de la tâche i

Nous utilisons pour la modélisation du problème les variables de décision suivantes:

$$\mathrm{RE}_{ims} = \begin{cases} 1 \ Si\ la\ tâche\ i\ sur\ la\ machine\ m\ requiert \\ \quad la\ compétence\ s \\ 0 \ \ Sinon \end{cases}$$

$$\mathrm{A}_{ws} = \begin{cases} 1 \ Si\ l'opérateur\ w\ a\ la\ cométence\ s \\ 0 \ \ Sinon \end{cases}$$

$$\mathrm{X}_{ijm} = \begin{cases} 1 \ Si\ la\ tâche\ i\ précéde\ la\ tâche\ j\ sur\ la \\ \quad machine\ m \\ 0 \ \ Sinon \end{cases}$$

$$\mathrm{Z}_{imm'} = \begin{cases} 1 \ Si\ la\ tâche\ i\ est\ exécutée\ sur\ la\ machine \\ \quad m\ puis\ sur\ la\ machine\ m' \\ 0 \ \ Sinon \end{cases}$$

$$\mathrm{Y}_{wim} = \begin{cases} 1 \ Si\ l'opérateur\ w\ exécute\ la\ tâche\ i\ sur\ la \\ \quad machine\ m \\ 0 \ \ Sinon \end{cases}$$

$$\mathrm{V}_{imjm'} = \begin{cases} 1 \ Si\ la\ tâche\ i\ sur\ la\ machine\ m \\ \quad précéde\ la\ tâche\ j\ sur\ la\ machine\ m' \\ 0 \ \ Sinon \end{cases}$$

Le modèle mathématique peut être formulé comme suit:

$$Minimiser \sum_{i=1}^{N} F_i = Minimiser \sum_{i=1}^{N} (C_i - r_i) \tag{1}$$

Sous les contraintes suivantes :

$$\forall w \in W; \forall i \in J; \forall m \in L \quad C_i \geq C_{wim} \tag{2}$$

$$\forall i, j \in J, i \neq j; \forall m \in L; \forall w \in W \quad C_{wim} - G(1 - X_{ijm}) \leq C_{wjm} - SE_{jm} - p_{jm} \tag{3}$$

$$\forall i \in J; \forall m, m' \in L, m \neq m'; \forall w \in W \quad C_{wim} - G(1 - Z_{imm'}) \leq C_{wim'} - p_{im'} \tag{4}$$

$$\forall i, j \in J; \forall m, m' \in L; \forall w \in W \quad C_{wim} - G(1 - V_{imjm'}) \leq C_{wjm'} - SE_{jm'} - p_{jm'} \tag{5}$$

$$\forall i, j \in J, i \neq j; \forall m \in L \quad X_{ijm} + X_{jim} = 1 \tag{6}$$

$$\forall i \in J; \forall m, m' \in L, m \neq m' \quad Z_{imm'} + Z_{im'm} = 1 \tag{7}$$

$$\forall i, j \in J; \forall m, m' \in L \quad V_{imjm'} + V_{jm'im} \leq 1 \tag{8}$$

$$\forall i, j \in J, i \neq j; \forall m, m' \in L; \forall w \in W \quad V_{imjm'} + V_{jm'im} \geq Y_{wim} + Y_{wjm'} - 1 \tag{9}$$

$$\forall i \in J; \forall m, m' \in Lm \neq m'; \forall w \in W \quad V_{imim'} + V_{im'im} \geq Y_{wim} + Y_{wim'} - 1 \tag{10}$$

$$\forall i \in J; \forall m \in L; \forall w \in \Gamma; \forall s \in S \quad Y_{wim} \leq RE_{ims} \times A_{ws} + (1 - RE_{ims}) \tag{11}$$

$$\sum_{i=1}^{N} \sum_{m=1}^{M} Y_{wim} \geq 1 \qquad \forall w \in W \tag{12}$$

$$\sum_{w=1}^{W} Y_{wim} = 1 \qquad \forall i \in J; \forall m \in L \tag{13}$$

$$\begin{array}{c} \forall i \in J; \forall m \in L; \forall w \in W \\ C_{wim} - SE_{im} - p_{im} \geq r_i \end{array} \tag{14}$$

$$\begin{array}{c} \forall i, j \in J; \forall m, m' \in L; \forall w \in W \\ X_{ijm}, Z_{imm'}, Y_{wim}, V_{imjm'} \in \{0, 1\} \end{array} \tag{15}$$

4.2 Description de l'algorithme ACO

Les premiers travaux portant sur l'algorithme ACO ont été proposés par Dorigo et al dans (Dorigo, Maniezzo & Colorni 1991) et (Dorigo, Maniezzo & Colorni 1996). L'idée clé de l'ACO fut inspirée par le comportement des fourmis qui retrouvent et empreintent in fine le chemin le plus court entre leur colonie et une source de nourriture. En effet, (Goss, Aron, Deneubourg & Pasteels 1989) a mené des expériences prouvant qu'après une durée de temps donnée toutes les fourmis d'une colonie choisisent (entre deux alternatives) le chemin le plus court pour parcourir la distance qui sépare leur colonie d'une source de nourriture.

Dans le cadre de notre étude nous considérons l'algorithme ACO (Voir (Ciro et al. 2015b)) avec les caractéristiques suivantes:

- Pour les premières 10 % des itérations les fourmis seront générées aléatoirement. Ensuite pour le reste des fourmis les phéromones ainsi que la matrices de visibilité seront utilisés comme critère.
- La valeur initiale de la matrice de visibilité

$$\eta_{0,v} = \frac{1}{r_v + s_v + p_v + s_v} \qquad \forall v \in L_\Phi \tag{16}$$

- La matrice de visibilité permettant d'estimer la désirabilité d'aller de l'opération u vers l'opération v $\forall v \in L_\Phi$; $\forall u \in \Omega$

$$\eta_{u,v} = \begin{cases} \frac{1}{(s_v + p_v)} & if \quad r_v - C_u \leq 0 \\ \frac{1}{(1 + r_v - C_u) + (s_v + p_v)} & Otherwise \end{cases} \tag{17}$$

- La matrice de probabilité de transition qui donne la probabilité pour qu'une fourmis donnée choisisse une opération en se basant sur la trace du phéromone τ et la visibilité η.

$$\eta_{u,v} = \begin{cases} arg\ max_{v \in L_\Phi} \ [\tau_{u,v}]^\alpha * [\eta_{u,v}]^\beta \\ \kappa \quad Otherwise \end{cases} \tag{18}$$

Où κ représente une variable aléatoire définit comme suit

$$P_\Phi(\kappa) = \frac{[\tau_{u,v}]^\alpha * [\eta_{u,v}]^\beta}{\sum_{v \in L_\Phi} [\tau_{u,v}]^\alpha * [\eta_{u,v}]^\beta} \tag{19}$$

4.3 Accélération du PLNE en utilisant ACO

Ici, nous proposons d'utiliser conjointement le PLNE ainsi que l'algorithme ACO et ce dans le but d'accélérér le PLNE. Dans une procédure Branch and cut, l'étape d'énumération des solutions occupe une part importante et affecte largement le temps d'exécution. L'idée de notre approche est d'utiliser l'algorithme ACO en vue de fournir une borne supérieure de bonne qualité permettant de réduire de manière considérable l'espace de solutions et dès lors converger plus rapidement.

Comme décrit dans l'organigramme Figure 2, tout d'abord, l'algorithme ACO est lancé en étape de prétraitement durant 120s. Ensuite, la solution obtenue est utilisée comme contrainte supplémentaire dans le PLNE. En effet, elle servira comme une nouvelle borne supérieur permettant de limiter l'espace de recherche de solutions.

Les résultats expérimentaux (Voir Tab 1) basés sur différents types d'instances ont permis de valider notre approche permettant l'accélération effective du PLNE original.

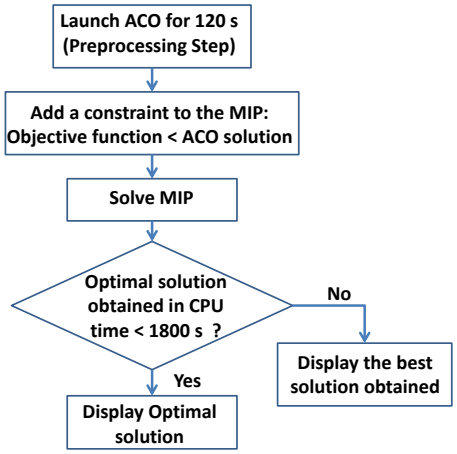

Figure 2: Processus d'accélération du PLNE

5 RELAXATION LAGRANGIENNE

Une des bornes inférieures les plus connues dans le cas de l'ordonnancement dans un atelier est le maximum entre les charges machines et les durées d'exécution des tâches. La relaxation lagrangienne est une approche qui a été proposée pour la première fois par Held et Karp (Held & Karp 1971). Le nom de ladite méthode a été attribué plus tard par Geoffrion (Geoffrion 1974).

Dans plusieurs travaux, des chercheurs ont prouvé que la relaxation lagrangienne améliore de manière significative les méthodes exactes dans plusieurs champs d'application tels que le problème de tournée de véhicule (Bazarra & Goode 1977), le prob-

lème d'affectation (Ross & Soland 1975) ou encore l'ordonnancement (Fisher 1973).

L'idée principale de la relaxation lagrangienne est d'éliminer certaines contraintes du modèle original (celles qui rendent le problème plus complexe) et les injecter par ailleurs dans l'expression de la fonction objectif sous forme d'une combinaison linéaire de pénalités. Les coefficients de cette combinaison sont appelés multiplicateurs de lagrange.

Dans cette section nous introduisons une nouvelle borne inférieure pour l'OSSP avec contraintes de ressources en utilisant une relaxation lagrangienne. Le choix de la contrainte à relaxer est basé sur deux paramètres importants: la qualité de la borne obtenue et la durée d'exécution.

Dans le présent travail, nous avons décidé de relaxer la contrainte ci-dessous mettant en relation deux opérations exécutées par le même opérateur :

$$\begin{array}{c} \forall i,j \in J; \forall m,m' \in L; \forall w \in W \\ C_{wim} - G(1 - V_{imjm'}) \leq C_{wjm'} - SE_{jm'} - p_{jm'} \end{array} \quad (20)$$

Ainsi, la relaxation lagrangienne est obtenue en considérant les mêmes contraintes du PLNE original decrit précédemment dans cette section à l'exception de la contrainte exposée ci-dessus qui est éliminée. Quant à la fonction objectif désormais elle devient comme suit:

$$\begin{array}{l} Minimiser \sum_{i=1}^{N}(C_i - r_i) + \sum_{i,j,m,m',w} \lambda * (C_{wjm'} \\ -SE_{jm'} - p_{jm'} - (C_{wim} - G(1 - V_{imjm'})) \end{array} \quad (21)$$

En effet, en plus de la fonction objectif du PLNE original correspondant à la minimisation de la durée du flux total (*Total flow time*), une pénalité supplémentaire est considérée quand deux opérations exécutées par le même opérateur se chevauchent.

6 RESULTATS EXPERIMENTAUX

Les résultats expérimentaux exposés dans cette section illustrent dans un premier temps l'utilité de l'intégration de la solution obtenue avec l'algorithme ACO autant que borne supérieure pour la la résolution du PLNE. Par la suite, nous nous intéréssons à la nouvelle borne inférieure introduite dans la section précédente. Concernant le dispositif d'expérimentation, nous avons utilisé une machine équipée de processeur 2,5GHz Intel 2520 et 16 GB de mémoire vive. Enfin, et en vue de résoudre les différents PLNE, nous avons utilisé le logiciel d'optimisation CPLEX de IBM ILOG.

Une instance donnée est définit par le nombre de tâches * le nombre de machines * le nombre de d'opérateurs * le nombre de compétences. Les durées d'exécutions sont des entiers uniformément distribués sur [1; w], et les durées de réglage de machines sont des variables aléatoires comprises dans [30; w], où w est un entier égale à 50, 75 ou 100. La date de disponibilité d'une tâche donnée est uniformément distribuée sur $[0; \lambda * \sum_{m=1}^{q} p_{im}]$; où p_{im} représente la durée d'exécution de la tâche i sur la machine m et λ le facteur de date de disponibilité qui peut prendre une des valeurs 0.1, 0.3 ou 0.5.

Dans cette section, tout d'abord nous comparons le PLNE original avec la version accélérée par les solutions de l'algorithme ACO. Ensuite, nous exposons les résultats de la nouvelle borne supérieure obtenue par relaxation lagrangienne du PLNE original.

6.1 PLNE accéléré

L'objectif ici est de lancer une série d'expériences en vue de prouver la validité de notre démarche concernant l'accélération du PLNE original. Pour chaque type de scénario, neuf instances seront testées. Dans le tableau 1, nous exposons dans la colonne "PLNE" la durée moyenne nécessaire pour l'exécution du PLNE original. Dans la colonne "Acc. PLNE + ACO BS" nous présentons la durée d'exécution du PLNE accéléré en incluant $120s$ relative à la durée de pré-traitement pour obtenir la solution de l'algorithme ACO. A noter que nous avons fixé le temps maximal d'exécution du PLNE sur CPLEX à $1800s$.

Nous observons dans la colonne "PLNE Vs Acc. PLNE" que pour les instances de petite taille avec "4" tâches, l'écart est négatif. Ceci est dû à la durée de pré-traitement fixée à $120s$ en vue d'obtenir la solution produite par l'algorithme ACO. Par contre dès que le nombre de tâches devient plus important l'écart augmente de manière significative. Nous constatons alors une amélioration nette des performances en comparant le PLNE accéléré avec le PLNE original. Par ailleurs, le nombre d'instances pour lesquelles le PLNE accéléré permet de trouver une solution en moins de $1800s$ est plus important (Voir les colonnes "Solutions # inst").

6.2 Relaxation lagrangienne

Nous avons précédemment montré que l'utilisation du PLNE en vue de la minimisation de la durée du flux total dans le cas d'un OSSP avec contraintes de ressources demande un temps d'exécution important. Par ailleurs, le nombre d'instances résolues par le PLNE en moins de $1800s$ dimimue significativement quand le nombre de tâches considérées devient important. Ici, nous comparons le MIP original avec la borne inférieure introduite dans la Section 4.1.

Problème	PLNE	Solutions # inst	PLNE + ACO BS	Solutions # inst	PLNE Vs Acc. PLNE
4X4_50-0.1-4-3	47,37	9/9	133	9/9	-180,77%
4X4_50-0.1-4-5	64,43	9/9	133,55	9/9	-107,28%
4X4_50-0.3-4-3	14,55	9/9	127,29	9/9	-774,85%
4X4_75-0.3-4-5	100,4	9/9	129,27	9/9	-28,75%
4X4_75-0.5-4-5	21,5	9/9	126,37	9/9	-487,77%
4X4_100-0.1-4-5	18,59	9/9	127,77	9/9	-587,31%
4X4_100-0.3-4-3	42,23	9/9	133,03	9/9	-215,01%
5X4_50-0.1-4-3	277,76	8/9	169,53	9/9	38,97%
5X4_50-0.1-4-5	294,5	7/9	175,42	9/9	40,43%
5X4_50-0.1-4-7	290,27	6/9	151,14	8/9	47,93%
5X4_75-0.5-4-3	149,71	9/9	142,8	9/9	4,62%
5X4_75-0.5-4-5	381,5	9/9	187,5	9/9	50,85%
5X4_75-0.5-4-7	434,18	7/9	184,95	7/9	57,40%
5X4_100-0.1-4-3	177,93	9/9	145,94	9/9	17,98%
5X4_100-0.1-4-5	428,87	7/9	176,83	8/9	58,77%
5X4_100-0.1-4-7	248,32	8/9	168,68	8/9	32,07%
7X3_50-0.1-3-3	471,57	3/9	191,2	6/9	59,45%
7X3_50-0.1-3-5	747,38	3/9	279,65	6/9	62,58%
7X3_75-0.5-3-3	651,66	5/9	211,49	6/9	67,55%
7X3_75-0.5-3-5	1252,76	2/9	449,87	4/9	64,09%
7X3_75-0.5-3-7	484,04	4/9	183,62	7/9	62,07%
7X3_100-0.1-3-3	704,31	4/9	171,69	5/9	75,62%
7X3_100-0.1-3-5	618,35	6/9	244,18	7/9	60,51%
7X3_100-0.1-3-7	895,61	4/9	586,27	6/9	34,54%

Table 1: PLNE Vs PLNE Accéléré

Comme précédemment nous considérons encore une fois pour chaque type de scénario, neuf instances différentes.

Nous observons dans le tableau 2 (Voir colonne "Temps PLNE Vs Borne inférieuré") que l'écart en terme de temps d'exécution est très important et supérieur à 90% quand le nombre de tâches est supérieur à 3. Par ailleurs, toutes les instances sont résolues dans le cas de la relaxation lagrangienne contrairement au PLNE original où le nombre d'instances résolues descend jusqu'à 2 pour certaines instances. Concernant la qualité des solutions obtenues avec la relaxation lagrangienne, l'écart par rapport à la solution optimale ne dépasse jamais 12%.

7 CONCLUSION ET PERSPECTIVES

Dans le présent papier, nous avons étudié le problème OSSP avec contraintes de ressources. Notre observation principale à l'issue de notre étude de l'état de l'art est le nombre faible de publications concernant les approches exactes pour la résolution des OSSPs. Le problème considéré étant NP difficile(Achugbue & Chin 1982), la majorité des travaux publiés concernent des heuristiques et des métaheuristiques.

Ici, nous avons tout d'abord développé un PLNE accéléré en adoptant les solutions obtenues avec l'algorithme ACO comme borne supérieure pour le PLNE. Les différents tests réalisés nous ont permis de prouver la validité et l'utilité de notre approche. Par la suite nous avons introduis une nouvelle borne inférieure définit comme étant la solution optimale de la minimisation de durée du flux total de l'OSSP relaxé. Les résultats expérimentaux ont permis de montrer que la nouvelle borne inférieure produit dans la quasi-totalité des cas de bons résultats avec moins de 10% d'ecart par rapport à la solution optimale et plus de 90% d'amélioration en termes de temps d'exécution par rapport au PLNE original.

Pour nos futurs travaux, nous souhaitons étendre notre étude au cas bi-objectif en s'intéressant à l'optimisation de l'équilibrage de charge en plus de la minimisation de la durée du flux total. Par ailleurs, nous comptons développer d'autres approches hybrides qui mixent l'utilisation de métaheuristiques ainsi que des méthodes exactes et ce en vue de la résolution du problème OSSP avec contraintes de ressources.

REFERENCES

Achugbue, J. & Chin, F. (1982). Scheduling the open shop to minimize mean flow time, *Society for industrial and applied mathematics journal on computing* **11**: 709–720.

Problème	PLNE	Solutions # inst	Relaxation Lagrangienne	Solutions # inst	Optimalité PLNE Vs Borne inf	Temps PLNE Vs Borne inf
3X3_50-0.1-3-3	0,9	9/9	0,25	9/9	7,60%	72,22%
3X3_50-0.1-3-5	0,58	9/9	0,22	9/9	6,70%	62,07%
3X3_50-0.3-3-3	0,75	9/9	0,2	9/9	6,23%	73,33%
3X3_75-0.1-3-3	0,64	9/9	0,19	9/9	7,62%	70,31%
3X3_75-0.1-3-5	0,57	9/9	0,15	9/9	9,75%	73,68%
4X4_50-0.1-4-3	47,37	9/9	1,38	9/9	12%	97,09%
4X4_50-0.1-4-5	64,44	9/9	0,52	9/9	11,10%	99,19%
4X4_50-0.3-4-3	14,56	9/9	0,69	9/9	3,58%	95,26%
4X4_75-0.3-4-5	100,4	9/9	0,76	9/9	6,39%	99,24%
4X4_75-0.5-4-3	9,38	9/9	0,2	9/9	6,58%	97,87%
4X4_75-0.5-4-5	21,51	9/9	0,49	9/9	6,87%	97,72%
4X4_100-0.1-4-3	7,32	9/9	0,58	9/9	4,18%	92,08%
4X4_100-0.1-4-5	18,6	9/9	0,48	9/9	11,00%	97,42%
4X4_100-0.3-4-3	42,24	9/9	0,45	9/9	6,62%	98,93%
5X4_50-0.1-4-3	447	8/9	3,81	9/9	4,80%	99,15%
5X4_50-0.1-4-5	629,1	7/9	4,6	9/9	11,17%	99,27%
5X4_50-0.1-4-7	794	6/9	3,83	9/9	10,02%	99,52%
5X4_75-0.5-4-3	150	9/9	2,66	9/9	7,25%	98,23%
5X4_75-0.5-4-5	204,21	8/9	3,2	9/9	4,92%	98,43%
5X4_75-0.5-4-7	434,19	7/9	4,9	9/9	8,90%	98,87%
5X4_100-0.1-4-3	178	9/9	2,92	9/9	8,82%	98,36%
5X4_100-0.1-4-5	428,88	7/9	2,85	9/9	8,95%	99,34%
7X3_50-0.1-3-3	472	3/9	16,22	9/9	8,85%	96,56%
7X3_50-0.1-3-5	747,4	3/9	43,7	9/9	10,43%	94,15%
7X3_50-0.1-3-7	861,21	2/9	85,05	9/9	7,70%	90,12%
7X3_75-0.5-3-3	651,66	5/9	12,85	9/9	2,55%	98,03%
7X3_75-0.5-3-5	1252,77	2/9	17,07	9/9	11,7%	98,64%
7X3_75-0.5-3-7	484,05	4/9	13,07	9/9	2,94%	97,30%
7X3_100-0.1-3-3	704,32	4/9	14,29	9/9	1,60%	97,97%
7X3_100-0.1-3-5	618,36	6/9	26,9	9/9	2,50%	95,65%
7X3_100-0.1-3-7	895,62	4/9	19,63	9/9	8,70%	97,81%

Table 2: Modèle avec Relaxation lagrangienne

Anand, E. & Panneerselvam, R. (2015). Literature review of open shop scheduling problems, *Intelligent Information Management* **7**: 33–52.

Bazarra, M. S. & Goode, J. J. (1977). The traveling salesman problem: A duality approach, *Mathematical Programming* **13**: 221–237.

Blum, C. (2005). Beam-aco-hybridizing ant colony optimization with beam search: an application to open shop scheduling, *Computer & operation research* **32**: 1565–1591.

Brucker, P., Hurink, J., Jurish, B. & Wostmann, B. (1997). A branch and bound algorithm for the open-shop problem, *Discrete Appl Math* **76**: 43–59.

Ciro, G. C., Dugardin, F., Yalaoui, F. & Kelly, R. (2015a). A fuzzy ant colony optimization to solve an open shop scheduling problem with multi-skills resource constraints, *INCOM* **48**: 715–720.

Ciro, G. C., Dugardin, F., Yalaoui, F. & Kelly, R. (2015b). Open shop scheduling problem with a multi-skills resource constraint: a genetic algorithm and an ant colony optimisation approach, *International Journal of Production Research* pp. 1–28.

Dorigo, M., Maniezzo, V. & Colorni, A. (1991). Positive feedback as a search strategy, *Technical Report , Dipartimento di Elettronica, Politecnico di Milano* **91-016**.

Dorigo, M., Maniezzo, V. & Colorni, A. (1996). Ant system: Optimization by a colony of cooperating agents, *IEEE Trans Syst Man Cybernet* **21**: 29–41.

Fisher, M. L. (1973). Optimal solution of scheduling problems using lagrange multipliers: Part i, *Operations Research* **21**: 1114–1127.

Geoffrion, A. M. (1974). Lagrangean relaxation for integer programming, *Mathematical Programming Study* **2**: 82–114.

Gonzalez, T. & Sahni, S. (1976). Open shop scheduling to minimize finish time, *Journal of the ACM* **23**: 665–679.

Goss, S., Aron, S., Deneubourg, J. & Pasteels, J. (1989). Selforganized shortcuts in the argentine ant, *Naturwissenschaften* **76**: 579–581.

Held, M. & Karp, R. M. (1971). The traveling-salesman problem and minimum spanning trees: Part ii, *Mathematical Programming* **1**: 6–25.

Hossein, S. & Doulabi, H. (2010). A mixed integer linear formulation for the open shop earliness-tardiness scheduling problem, *Applied Mathematical Sciences* **4**: 1703–1710.

Huang, Y. & Lin, J. (2011). A new bee colony optimization algorithm with idle-time-based filtering scheme for open shop scheduling problems, *Expert Systems with Applications* **38**: 5438–5447.

Johnson, S. (1954). Optimal two- and three-stage production schedules with setup times included, *Nav Res Logist* **1**: 61–68.

Kyparisis, G. J. & Koulamas, C. (1997). Open shop scheduling with maximal machines, *Discrete Applied Mathematics* **78**: 175–187.

Liaw, C. F. (1999). A tabu search algorithm for the open shop scheduling problem, *Computer & operation research* **26**: 109–126.

Liaw, C. F. (2000). A hybrid genetic algorithm for the open shop scheduling problem, *European Journal of Operational Research* **124**: 28–42.

Naderi, B., Ghomi Fatemi, S., Aminnayeri, M. & Zandieh, M. (2011). A study on open shop scheduling to minimize total tardiness, *International Journal of Production Research* **49**: 4657–4678.

Ng, C., Cheng, T. & Yuan, J. (2003). Concurrent open shop scheduling to minimize the weighted number of tardy jobs, *Journal of Scheduling* **6**: 405–412.

Prins, C. (2000). Competitive genetic algorithms for the open-shop scheduling problem, *Mathematical Methods of Operations Research* **52**: 389–411.

Roshanaei, V., Esfehani, M. & Zandieh, M. (2010). Integrating non-preemptive open shop scheduling with sequence-dependent setup times using advanced metaheuristics, *Expert Systems with Applications* **37**: 259–266.

Ross, G. T. & Soland, R. M. (1975). A branch and bound algorithm for the generalized assignment problem, *Mathematical Programming* **8**: 91–103.

Zobolas, G. I., Tarantilis, C. D. & Ioannou, G. (2008). Exact, heuristic and meta-heuristic algorithms for solving shop scheduling problems, *Metaheuristics for Scheduling in Industrial and Manufacturing Applications* **128**: 1–40.