Un algorithme glouton pour l'affectation des ressources humaines aux lignes d'assemblage

Sana Bouajaja

Labo, Université OASIS – Ecole Nationale d'Ingénieurs de Tunis
Université de Tunis El Manar, BP. 37 le Belvédère, 1002, Tunis, Tunisie

sana_bouajaja@yahoo.fr

RÉSUMÉ. Dans ce travail, nous traitons un problème d'affectation conjointe des ressources humaines et des opérations aux lignes d'assemblage, en industrie automobile. L'objectif est la minimisation du temps de cycle. Pour résoudre ce problème, nous proposons une heuristique gloutonne basée sur des règles de priorité, pour sélectionner l'opération à affecter aux opérateurs en premier parmi une liste candidate et pour sélectionner l'opérateur qui va effectuer l'opération courante. Une étude expérimentale a été menée sur un ensemble d'instances afin de définir les règles donnant les meilleurs résultats.

ABSTRACT. In this paper, we deal with the problem of human resources allocation and task assignment to assembly lines. To solve this problem, we propose a greedy heuristic based on priority rules, used to select from a candidate list, the task to assign and to select a worker for the current task. A numerical study was done to define the priority rules that give the best results.

MOTS-CLÉS : optimisation, affectation de ressources humaines, équilibrage des lignes d'assemblage.

KEYWORDS: optimization, human resources assignment, assembly line balancing.

DOI:10.3166/JESA.49.769-791

1. Introduction

Malgré une automatisation accrue, le facteur humain s'impose comme l'un des éléments clés de la compétitivité des entreprises dans cet environnement en pleine mutation. Cette évolution du monde industriel a impliqué une perpétuelle remise en cause des méthodes de production et en particulier des méthodes de prise en compte des opérateurs. Dès lors, l'utilisation rationnelle des ressources humaines lors de l'organisation des activités devient l'une des préoccupations des entreprises. En effet, l'emploi efficace de ces sources « naturelles » s'avère indispensable à la

satisfaction des objectifs de performance imposée et de la bonne affectation de ces derniers dépend largement l'efficacité des processus.

Plusieurs chercheurs se sont intéressés à la gestion des ressources humaines et en particulier à la problématique d'affectation de ces ressources, laquelle recouvre un ensemble de problèmes qui varient en fonction de l'objectif d'optimisation, des contraintes prises en compte…

Dans ce papier, nous mettons l'accent plus particulièrement sur le problème d'équilibrage des lignes d'assemblage en optimisant l'allocation des ressources humaines disponibles. Notre but est double : nous visons d'une part, à affecter les opérations aux stations de travail et d'autre part, à affecter les opérateurs aux stations, dans l'objectif de réduire le temps de cycle de la ligne et améliorer ainsi sa productivité. Pour résoudre ce problème, nous avons mis en place un algorithme de construction gloutonne basé sur des règles de priorité pour la sélection des opérations à affecter en premier et pour le choix des opérateurs. Une étude expérimentale sur des instances générées aléatoirement et d'autres issues de la littérature nous ont permis d'évaluer la qualité de notre proposition.

L'article est organisé comme suit : dans la deuxième section, nous présentons notre problème (hypothèses, contraintes). Ensuite, nous décrivons brièvement les problèmes qui relèvent du domaine de l'équilibrage des lignes d'assemblage et de l'optimisation de l'allocation des ressources humaines, ainsi que les approches offertes comme support d'aide à la décision. Dans la section 3, nous abordons les deux formulations mathématiques du problème. Nous proposons notre heuristique en section 4. La dernière section présente les résultats d'une étude expérimentale permettant d'évaluer la méthode de résolution proposée. Nous terminons ce travail par une conclusion et quelques perspectives.

2. Présentation du problème

2.1. Problématique

Nous considérons une ligne d'assemblage composée d'un ensemble de stations. Dans chaque station, un seul opérateur est chargé à effectuer ces opérations en série. Notre problème consiste à proposer une solution simultanément à une double affectation : (1) les opérations aux opérateurs disponibles et (2) les opérateurs aux stations. Cette solution doit garantir la réduction du temps de cycle de la ligne tout en respectant les différentes contraintes du problème, ce qui correspond à l'amélioration de la productivité de la ligne. Ce problème est appelé *Assembly Line Worker Assignment and Balancing Problem* 2 (ALWABP-2) par analogie avec le *Simple Assembly Line Balancing Problem* (SALBP-2) dont l'objectif est de réduire le temps de cycle de la ligne d'assemblage pour un nombre de stations fixe.

2.2. Hypothèses et contraintes

Différentes hypothèses et contraintes sont considérées dans la résolution de notre problème :

1. la ligne est conçue pour un seul produit,
2. les stations de la ligne sont en série et sans stock tampon,
3. les temps opératoires sont déterministes,
4. les temps opératoires dépendent de la compétence de l'opérateur qui va les exécuter,
5. les relations de précédence doivent être respectées,
6. toutes les opérations doivent être exécutées,
7. une opération est exécutée sur une seule station
8. un opérateur est assigné à une seule station,
9. une station ne comporte qu'un seul opérateur.

2.3. ***État de l'art***

2.3.1. Problème d'équilibrage des lignes d'assemblage

Le problème de l'équilibrage se pose pour tous les types de lignes de fabrication. Initialement, il a été formulé pour des lignes d'assemblage mono-produit, dans l'industrie automobile. L'objectif du problème est d'affecter les opérations aux stations de travail, de manière à satisfaire les contraintes de précédence, et que chaque opérateur ait le temps de réaliser, durant le temps de cycle, l'ensemble des opérations qui lui sont affectées (Boutevin *et al.*, 2003). Dans la littérature, ce problème est connu sous le nom de *Simple Assembly Line Balancing Problem* (SALBP) i.e. équilibrage des lignes d'assemblage. Plusieurs modèles ont été énumérés par Baybars, (1986) :

– SALBP : où seules les contraintes les plus courantes sont considérées : le temps de cycle et la précédence.

Suite à la variation des objectifs, plusieurs versions du SALBP sont étudiées, nous citons par exemple : SALBP-1 : minimisation du nombre de stations avec un temps de cycle donné, et SALBP-2 : minimisation du temps de cycle avec un nombre de stations donné.

La formulation du SALBP n'exprime pas toute la réalité industrielle à cause des hypothèses simplificatrices qu'elle comprend (Ghosh, 1989). De ce fait, des hypothèses plus générales ont été introduites, dans ce cas on parle de *Generalized Assembly Line Balancing Problem* (GALBP).

– GALBP : Ce modèle considère à la fois des contraintes courantes et d'autres moins usuelles : groupement d'opérations (opérations affectées sur la même station),

incompatibilité (opérations à affecter à des stations différentes), stations parallèles (opérations réalisées par plusieurs stations)…

Dans la littérature, plusieurs études de l'état de l'art ont été menées sur les problèmes d'équilibrage des lignes d'assemblage (Becker et Scholl, 2006 ; Boysen *et al.*, 2007 ; Kumar, 2013 ; Rekiek *et al.*, 2002 ; Sivasankaran et Shahabudeen, 2014). Les premiers travaux mettant l'accent sur ces problèmes d'équilibrage remontent aux années cinquante. Dans ces travaux, différentes classes du problème ont été présentées et plusieurs méthodes de résolution ont été proposées. Nous trouvons des méthodes exactes ou des heuristiques et des métaheuristiques, vu le caractère NP-difficile de ce problème.

2.3.2. Problème d'affectation des ressources humaines

La gestion optimale des ressources humaines joue un rôle important dans la productivité et la compétitivité des entreprises. Pour cette raison, le problème d'allocation des ressources humaines a fait l'objet de nombreuses études, étant donné qu'il touche à plusieurs domaines comme la gestion de production (Corominas *et al.*, 2006 ; Tan *et al.*, 2009), la gestion de la maintenance (Bennour *et al.*, 2005 ; Dakkak *et al.* 2012), la gestion des projets (Selaru, 2012 ; Certa *et al.*, 2009), la gestion des systèmes hospitaliers (Lanzarone et Matta, 2014 ; Schaus *et al.*, 2009), etc.

Bien que plusieurs travaux académiques ont traité les deux problèmes présentés précédemment chacun à part, rares sont les études qui ont mis l'accent sur les deux problèmes simultanément. Nous citons particulièrement (Miralles *et al.*, 2008 ; Chaves *et al.*, 2009 ; Moreira et Costa, 2009 ; Blum et Miralles, 2011 ; Moreira et Costa, 2012 ; Moreira *et al.*, 2012a, 2012b ; Araújo *et al.*, 2012 ; Borba et Ritt, 2012 ; Mutlu *et al.*, 2013 ; Borba et Ritt, 2013 ; Borba et Ritt, 2014 ; Vilà et Pereira, 2014). Ces études couvrent le problème d'affectation des opérateurs et d'équilibrage des lignes d'assemblage dans un contexte particulier, qui est le centre protégé des personnes handicapées (ALWABP in SWD- *Assembly Line Worker Assignment and Balancing Problem in Sheltered Work centres for Disabled*). Nous nous basons sur ces travaux dans la modélisation de notre problème, puisqu'ils optimisent des ressources humaines hétérogènes comme dans notre cas d'étude.

Initialement, le problème a été résolu avec des méthodes exactes dans les travaux de (Miralles *et al*, 2007, 2008). Mais vu le caractère NP-difficile du ALWABP-2 (Chaves *et al.*, 2007), le recours aux méthodes heuristiques et métaheuristiques devient nécessaire pour traiter les grandes instances. Ces méthodes de résolution ne seront pas détaillées ici. Néanmoins, l'histogramme récapitulatif de la figure 1 permet de les visualiser.

Dans la littérature relative à l'équilibrage des lignes d'assemblage (*Simple Assembly Balancing Problem-SALBP*), un ensemble de règles de priorité, pour la sélection des opérations à affecter aux stations, est défini et énuméré dans plusieurs travaux (Scholl A., Voß S., 1996 ; Essafi M., 2010 ; Groussi, 1998). Ces règles ne sont pas toutes valables pour le problème ALWABP, particulièrement celles basées sur le temps d'exécution des opérations, puisque les temps opératoires considérés

pour ce type de problèmes sont déterministes. Par contre, le problème ALWABP, qui tient compte simultanément de l'allocation des ressources humaines et de l'équilibrage des lignes d'assemblage, considère des temps opératoires dépendant de l'opérateur. Pour adapter ces règles à différentes variantes du ALWABP, des modifications leur sont apportées et des nouvelles règles sont proposées (Moreira *et al.*, 2012b ; Araújo *et al*. 2012).

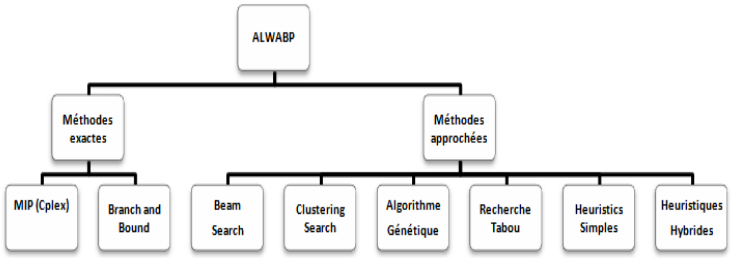

Figure 1. Méthodes de résolution du ALWABP dans la littérature

C'est dans ce contexte d'étude que notre travail a été effectué. Nous ciblons le problème d'équilibrage des lignes d'assemblages et l'optimisation de l'affectation de leurs ressources humaines à la fois, le rendement des opérateurs pouvant être hétérogène.

3. Formulation mathématique

Afin de modéliser notre problème, nous adoptons le modèle mathématique présenté dans (Miralles *et al.*, 2008) pour le cas d'opérateurs handicapés. Ce modèle comporte des contraintes spécifiques tenant compte de certaines incapacités des opérateurs liées à leur handicap. Par exemple, il vaut mieux affecter une personne touchée par une incapacité auditive ou orale à la première ou dernière station qui nécessite moins de coordination et de communication que les stations du milieu de la ligne. Sans ces contraintes, ce modèle devient généralement valable pour les lignes d'assemblage où on souligne une variété dans les temps opératoires, ce qui correspond à notre cas d'étude où on a des opérateurs hétérogènes. La formulation de notre problème est sous la forme d'un programme linéaire en nombres entiers, appelé Modèle 1.

3.1. Notations et paramètres

Notation	Description
i, j	Indice de l'opération

w	Indice de l'opérateur
s	Indice de la station
N	Nombre d'opérations
H	Nombre d'opérateurs disponibles
S	Nombre de stations
C	Temps de cycle
m	Nombre de stations
T_{wi}	Temps opératoire de l'opération i si elle est exécuté par l'opérateur h
D_j	Ensemble des opérations qui précédent directement l'opération j dans le graphe de précédence
X_{swi}	Variable binaire égale à 1 si l'opération i est affectée à l'opérateur w sur la station s
Y_{sw}	Variable binaire égale à 1 si l'opérateur w est affecté à la station s

3.2. Formulations mathématiques

3.2.1. Modèle 1

Dans le tableau 1, la minimisation du temps de cycle est donnée dans (1.1). Les contraintes (1.2) expriment que chaque opération est assignée à une seule station et à un opérateur unique. Les contraintes (1.3) et (1.4) assurent que chaque opérateur est assigné à une seule station et chaque station comporte un seul opérateur. Les contraintes (1.5) permettent de respecter la relation de précédence entre l'opération i et j où i est un prédécesseur de j. Les contraintes (1.6) permettent le calcul du temps de cycle C et les contraintes (1.7) permettent à un opérateur w assigné à une station s d'avoir plus d'une opération.

Afin de résoudre ce programme linéaire nous avons utilisé le solveur CPLEX. Les résultats obtenus seront discutés en détails dans la partie étude expérimentale.

Un nouveau modèle, qu'on a appelé Modèle 2, a été récemment proposé pour résoudre le même problème ALWABP-2 traité par Miralles. Ce deuxième programme linéaire mixte en nombre entiers, formulé par (Borba et Ritt, 2013), est basé sur l'observation qu'il suffit d'affecter les opérations aux opérateurs. Chaque opérateur sera identifié à une station. L'affectation des opérations aux opérateurs doit permettre de respecter les relations de précédence entre les opérations. Un graphe est généré au fur et à mesure qu'une affectation d'opération à une ressource humaine est faite. Ainsi, une relation de précédence entre les opérateurs est établie pour respecter les contraintes de précédence entre les opérations. Dans l'exemple de la figure 2, l'affectation des opérations {2,3} à l'opérateur w_2 et {4} à w_3 implique une relation de précédence entre w_2 et w_3 (w_2 prédécesseur de w_1). De même, l'affectation de {1} à w_1 fait que w_1 doit précéder w_2 et w_3.

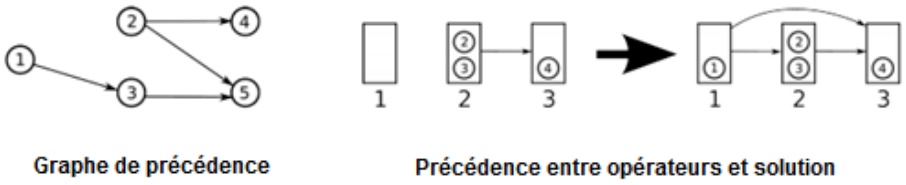

Figure 2. Relation de précédence entre les opérateurs après l'affectation de l'opération 1 (Borba et Ritt, 2012)

3.2.2. Modèle 2

Notation	Description
$A_i \subseteq W$	Ensemble d'opérateurs pouvant exécuter *i*
$A_w \subseteq N$	Ensemble d'opérations faisables par *w*
T_{wi}	Temps opératoire de i si elle est exécutée par l'opérateur *w*
X_{wi}	Variable binaire égale à 1 si l'opération *i* est affectée à l'opérateur *w*
d_{vw}	Variable binaire égale à 1 si l'opérateur *v* doit précéder l'opérateur *w*

Les contraintes (2.2) définissent le temps de cycle de la ligne. Les contraintes (2.3) assurent que chaque opération est exécutée par un seul opérateur. Les contraintes (2.4) reflètent les relations de dépendance entre les opérateurs : si une opération est affectée à l'opérateur *v* et elle est précédée de *j* qui est affectée à *w* donc *w* doit précéder *v*. Les contraintes (2.5) et (2.6) traduisent la transitivité et l'antisymétrie dans les relations de dépendances entre les opérateurs (tableau 1).

Nous adhérons à l'idée de ce modèle, parce qu'elle permet d'avoir une vision plus simple de notre problème en le transformant en un problème à simple affectation (affectation des opérations aux opérateurs), vu que de nouvelles relations de précédence ont été établies entre les opérateurs, pour que chaque opérateur soit assimilé à une station.

4. Méthode de résolution

Nous avons opté dans ce travail pour une méthode approchée parmi les plus usuelles qui sont les méthodes constructives. Nous proposons une heuristique gloutonne, étant donné que c'est une méthode à faible complexité, basée sur des critères de sélection simple et rapide (polynômiale). Avec cette méthode, la construction de la solution se fait progressivement, en se basant sur des listes candidates (d'opérations et d'opérateurs) et en considérant des règles heuristiques de sélection ou règles de priorité. En plus de nous fournir très rapidement une solution réalisable et d'une certaine qualité, elle peut être utilisée dans une méthode exacte

comme borne supérieure, permettant ainsi d'accélérer la convergence, ou encore elle peut permettre d'initialiser une métaheuristique.

Tableau 1. Modèle 1 et Modèle 2

Modèle 1	
Fonction objectif :	
$Min\ z = C$	(1.1)
Sous contraintes :	
$\sum_{w\in H}\sum_{s\in S} X_{swi} = 1\ , \forall i \in N$	(1.2)
$\sum_{s\in S} Y_{sw} \leq 1 \quad , \forall w \in H$	(1.3)
$\sum_{w\in H} Y_{sw} \leq 1 \quad , \forall s \in S$	(1.4)
$\sum_{w\in H}\sum_{s\in S} s.X_{swi} \leq \sum_{w\in H}\sum_{s\in S} s.X_{swj}\ , \forall i,j\ , i \in Dj$	(1.5)
$\sum_{i\in N} T_{wi}.X_{swi} \leq C\ , \forall w \in H, s \in S$	(1.6)
$\sum_{i\in N} X_{swi} \leq N.Y_{sw}\ , \forall w \in H, \forall s \in S$	(1.7)
Avec :	
$Y_{sw} \in \{0.1\}\ \forall s \in S \in, \forall w \in H$	(1.8)
$X_{swi} \in \{0,1\}\ \forall s \in S \in, \forall w \in H, \forall i \in N$	(1.9)
Modèle 2	
Fonction objectif :	
$Min\ z = C$	(2.1)
Sous contraintes :	
$\sum_{i\in N} T_{iw}X_{iw} \leq C\ , \forall\, w \in H$	(2.2)
$\sum_{w\in H} X_{iw} = 1\ , \forall i \in N$	(2.3)
$d_{vw} \geq X_{iv} + X_{jw} - 1\ \ \forall (i,j) \in N, v \in A_i, w \in A_j \backslash \{v\}$	(2.4)
$d_{uw}d_{uv} + d_{vw} - 1\ \ \forall \{v,u,w\} \subseteq H, \lvert\{v,u,w\}\rvert = 3$	(2.5)
$d_{vw} + d_{wv} \leq 1 \quad \forall v \in H, w \in H \backslash \{v\}$	(2.6)
Avec :	
$X_{iw} \in \{0.1\} \qquad \forall w \in H, i \in A_w$	(2.7)
$d_{vw} \in \{0,1\} \qquad \forall v \in H, w \in H \backslash \{v\}$	(2.8)

4.1. Algorithme proposé

4.1.1. Notations

Les différentes opérations sont modélisées par un graphe $G=(S,A)$. L'ensemble des sommets S représente les opérations, et l'ensemble des arcs A représente la relation de précédence : $(i,j)\in A \Leftrightarrow i$ précède j. $N = |S|$ est le nombre d'opérations. G est représenté par sa matrice d'adjacence P de taille $N\times N$ contenant les coefficients $P(i,j)$ =1 si $(i,j)\in A$, et $P(i,j) = 0$ sinon.

De même, une matrice de compétence T de taille $H\times N$, où H est le nombre d'opérateurs à affecter et N le nombre d'opérations à exécuter, est donnée. Cette matrice contient les coefficients $T(i,w)$ qui donne la durée prise par l'opérateur w pour exécuter l'opération i.

Les relations de précédence définies entre les opérateurs au fur à mesure, après chaque affectation, sont traduites par une matrice d'adjacence D. Une matrice binaire X de taille $N\times H$, permet de modéliser les résultats obtenus. Le coefficient $X(i,w) = 1$ si l'opération i a été affectée à l'opérateur w, et $X(i,w) = 0$ sinon.

Nous utilisons aussi les notations suivantes :

- L_a : Liste des opérations affectées,
- L_n : Liste des opérations non encore affectées,
- L_r : Liste courante des opérations à affecter de rang r, dont tous les prédécesseurs sont déjà affectés,
- L_{iw} : Liste candidate des opérateurs w qui peuvent exécuter l'opération i de la liste courante L_r, respectant la matrice de compétence.

4.1.2. Principe et étapes de l'algorithme

La première étape de l'algorithme glouton consiste à initialiser les listes à utiliser ($L_n = N$, $L_a = \emptyset$, $L_r = \emptyset$).

Ensuite, à chaque étape on commence par construire la liste L_r des opérations du graphe de précédence pouvant être affectées.

Une fois qu'une opération de L_r est affectée, elle est ajoutée à la liste L_a et supprimée de L_n. Quand toutes les opérations de L_r sont affectées, on passe aux opérations du rang suivant.

Pour chaque opération courante i de L_r, on définit la liste L_{iw} des opérateurs w qui ont la compétence de l'exécuter, en se référant à la matrice de compétence. Cette liste sera réduite en éliminant tous les opérateurs qui ne permettent pas de vérifier les contraintes de précédence entre les opérations si i leur est affectée.

Le choix de l'opération courante i à affecter parmi ceux de même rang r se fait selon une règle de priorité. De même, une règle de priorité est utilisée pour la sélection de l'opérateur w parmi ceux de la liste Li_w. La condition d'arrêt de cet algorithme est $L_a = N$ et $L_n = \emptyset$, c'est-à-dire toutes les opérations sont affectées.

Afin d'assurer l'équilibrage de la charge horaire entre les différents opérateurs, on doit à chaque itération de l'algorithme et avant d'affecter l'opération courante *i* à l'opérateur adéquat *w*, tenir compte de la charge actuel T_w de cet opérateur.

$$T_w = \sum_{i=1}^{N} X(i,w) * T(i,w) \text{ pour } \forall w = 1 \dots H \qquad (1)$$

Pour ce faire, on a défini une borne supérieure B_{sup}, permettant de limiter la valeur cette charge. Tout d'abord, notons que puisque le temps de cycle de la ligne *C* est égal au temps de l'opérateur le plus chargé, si une borne supérieure a été définie pour le temps de cycle, alors la plus grande charge horaire ne peut pas dépasser cette borne. Ceci nous mène à proposer une borne pour *C*. Dans la littérature relative au problème ALWABP-2, une borne supérieure est généralement définie à partir de la valeur d'une solution heuristique, ce qui n'est pas le cas pour notre travail. En fait, en examinant l'état de l'art du SALBP-2 (Ritt et Costa, 2015 ; Pastor et Ferrer, 2009), nous trouvons que la borne supérieure du temps de cycle $\bar{c}$ est déterminée en fonction sa borne inférieure $\underline{c}$ dont la valeur est :

$$\underline{c} = \max_{i=1\dots N}\left(max(T_i), \left\lceil \frac{\sum_{i=1\dots N} T_i}{M} \right\rceil\right) \qquad (2)$$

Pour le problème SALBP-2, *N* est le nombre d'opérations, *M* est le nombre de stations de travail de la ligne, T_i est le temps opératoire de l'opération i et ⌈x⌉ est le plus petit entier supérieur à *x*.

$$\bar{c} = 2.\underline{c} \qquad (3)$$

En tenant compte du facteur humain qui va influencer le temps opératoire de l'opération à exécuter, cette borne supérieure devient valable pour notre problème. Elle sera notée B_{sup} et dépend de la borne inférieure du temps de cycle ou de la charge des opérateurs B_{inf}.

$$B_{inf} = \max_{i=1\dots N}\left(max(T_i^-), \left\lceil \frac{\sum_{i=1\dots N} T_i^-}{M} \right\rceil\right) \qquad (4)$$

$$B_{sup} = min\left(2.B_{inf}\right) \qquad (5)$$

$$\text{Avec : } T_i^- = min_{w=1\dots H}\left(T(i,w)\right) \qquad (6)$$

T_i^- présente le temps opératoire minimal possible pour l'opération *i*, obtenu au meilleur des cas, lorsqu'elle est exécutée par l'opérateur le plus rapide.

Nous testons ainsi la valeur de T_w, si l'affectation de la nouvelle opération *i* engendre le dépassement de B_{sup}, on passe à l'opérateur suivant qui doit respecter toutes les contraintes du problème et satisfaire la condition suivante : $T_w \leq B_{sup}$. L'objectif est d'assurer une charge de travail équilibrée, le plus proche possible de la moyenne et éviter les surcharges de la main d'œuvre.

4.2. Règles de priorité

Pour les opérations ayant le même rang dans le graphe de précédence, on peut avoir un conflit dans le choix de l'opération à assigner en premier. Dans ce cas, adopter une règle de priorité prédéfinie s'avère nécessaire. En plus, les temps d'exécution des opérations dépendent des ressources humaines qui vont les exécuter. Pour cette raison, nous avons défini également des règles de priorité pour sélectionner l'opérateur qui va exécuter l'opération courante à affecter aux différentes stations de travail. L'allocation des opérations aux ressources humaines sera faite en se basant sur ces règles de priorité, dans le cas où deux ou plusieurs opérateurs sont disponibles pour la même opération.

Pour définir l'ordre d'affectation des tâches aux opérateurs, des contraintes s'imposent :

– Respect des relations de précédence entre les opérations.

– Tenir compte du temps d'exécution de la tâche qui dépend du facteur humain.

La définition des listes L_r groupant les opérations de rang r permet de satisfaire la première contrainte et garantir que toutes les tâches qui précédent l'opération courante soient antérieurement assignées.

Toutefois, au sein d'une même liste on aura des opérations candidates. Un choix quelconque de l'opération suivante qui doit être affectée implique une affectation aléatoire, ne garantissant pas nécessairement une bonne solution. Etant en quête de l'optimalité, des règles de priorités pour l'affectation de l'ensemble des tâches de L_r respectant la deuxième contrainte sont à établir.

Pour notre algorithme glouton, nous considérons la règle de priorité R_0, qui permet de sélectionner l'opération ayant le plus petit temps opératoireT_i^-. Nous avons opté pour cette règle parce qu'elle est basée sur la fonction objectif à optimiser, qui consiste à minimiser le temps de cycle de la ligne d'assemblage, en considérant l'opération ayant le plus petit temps opératoire, au sein de la liste candidate L_r, et ce en tenant compte de l'opérateur le plus rapide à l'exécuter.

En ce basant sur cette règle, les opérations de L_r vont être listées dans l'ordre croissant de leur temps d'exécution T_i^-et affectées aux opérateurs dans ce même ordre. Pour l'opération courante i, la liste L_{iw} groupe tous les opérateurs candidats. Pour sélectionner un des opérateurs, on tient compte du degré de compétence des opérateurs pour l'opération i et de leur charge. Pour cela, deux règles de priorités ont été définies : la première règle R_1consiste à sélectionner l'opérateur le plus rapide à exécuter i, ainsi les opérateurs de la liste L_{iw}, correspondante à l'opération courante i, vont être classés dans l'ordre croissant de leur $T(i,w)$. La deuxième règle R_2 permet de privilégier l'opérateur le moins chargé parmi les autres opérateurs, c'est-à-dire les opérateurs de L_{iw} seront classés dans l'ordre croissant de leur charge T_w.

$$T_w = \sum_{i=1}^{N} T(i,w) * X(i,w) \qquad (7)$$

Algorithme 1. Algorithme glouton proposé

Données : $N, H, P, T,$
Variables de décision : X
Initialisation : $L_n = N$, $L_a = \emptyset$, $r = 1$, $L_r = \emptyset$, D,
Calculer B_{sup} ;

Tant que $L_r \neq \emptyset$ **Faire**
Déterminer les opérations de la liste L_r de rang r ;
Trier les opérations de L_r dans l'ordre croissant de leur T_i^- ; *// Application de la règle R_0*

Pour tout i de L_r **Faire**
Définir la liste L_{iw} des opérateurs pouvant exécuter i ;
Pour tout w de L_{iw} **Faire**
Si précédence entre opérations non respectée **Alors**
Supprimer w de L_{iw} ;
Fin Si

Si ($T_w > B_{sup}$) **Alors**
Supprimer w de L_{iw} ;
Fin Si
Fin Pour

//Sélectionner l'opérateur w qui va exécuter i selon la règle de priorité prédéfinie
Si on va opter pour R_1 **Alors**
Trier les opérateurs de L_{iw} par ordre croissant des $T(i,w)$;
Sélectionner le 1er opérateur w de la liste ;
Fin Si

Sinon Si on va opter pour R_2 **Alors**
Trier les opérateurs de L_{iw} par ordre croissant de la charge de l'opérateur w (T_w) ;
Sélectionner le 1er opérateur w ;
Fin Si

$X(i,w)=1$; *// Affecter i courante à l'opérateur sélectionné w*
$T_w = \sum_{i=1}^{N} X(i,w) * T(i,w)$; *// Mettre à jour la charge de l'opérateur w*
Mettre à jour D; *// Etablir des relations de précédence entre opérateurs si nécessaire*
Retirer i de L_n ;
Ajouter i à L_a ;

Fin Pour

Initialiser L_r ;
$r \rightarrow r+1$; *// passer au rang suivant*

Fin Tant que

Calculer $C = \max_{w=1\ldots H}(T_w)$;
Retourner (C) ;

Dans le cas où on a dans la liste L_r une ou plusieurs opérations candidates à l'affectation ayant le même T_i^-, on choisit une opération aléatoirement. De même, si un ou plusieurs opérateurs de candidats de L_{iw} ont le même $T(i,w)$ (respectivement T_w) lorsqu'on considère la règle R_1 (respectivement R_2), alors on sélectionne un opérateur aléatoirement.

Pour certaines instances du problème, il peut y arriver que notre algorithme ne génère aucune solution réalisable, dans le cas où $L_{iw} = Ø$. Ce cas de figure se présente lorsqu'aucun opérateur w de L_{iw}, pouvant exécuter l'opération courante i, ne peut satisfaire ni les contraintes de précédence entre les opérateurs ni la borne supérieure B_{sup} pour la charge horaire T_w, si on lui affecte i. Pour pallier à cette déficience, nous utilisons une heuristique gloutonne randomisée permettant de faire un choix aléatoire de l'opération courante i parmi celles de L_r, qui sont de même rang r. La mesure à prendre, afin de renforcer la probabilité d'avoir une solution réalisable est d'augmenter le nombre d'exécutions de l'algorithme.

Les modifications à apporter dans ce cas à l'algorithme précédent, pour pouvoir passer à l'heuristique gloutonne randomisée, sont les suivantes. Après avoir testé la satisfaction des différentes contraintes du problème par les opérateurs w de L_{iw}, ajouter après la ligne 17 les instructions suivantes :

Si $L_{iw} = Ø$ **Alors**
Revenir à la ligne 7 ;
Remplacer les instructions de la ligne 7 à 10 par l'instruction :
Choisir l'opération courante i aléatoirement ;
Fin Si

5. Étude expérimentale

5.1. Résolution exacte du modèle 1 et modèle 2

Afin de résoudre notre problème, nous avons codé les modèles 1 et 2 avec le modeleur-solveur Cplex, en utilisant un ordinateur portable équipé d'un processeur Intel™ Core™ i3 CPU M370 @2.40 GHz 2.40 GHz.

Instances testées et résultats

Nous présentons, dans les tableaux 2a, 2b, un ensemble d'instances, généré aléatoirement et de petite taille (N varie de 5 à 18). Ces instances diffèrent par le nombre d'opérations N, le nombre d'opérateurs H et le nombre de stations S ($H = S$).

De même, des instances issues de la littérature (Chaves *et al.*, 2007), de taille moyenne ($N = 25$, $N = 28$), ont été également testées. Nous avons considéré deux groupes d'instances appelés respectivement Roszieg ($N = 25$) et Heskia ($N = 28$) (Chaves *et al.*, 2007). Chaque groupe est formé de 10 instances ayant le même graphe de précédence mais diffèrent par la matrice de compétences. Afin d'évaluer les résultats obtenus, nous avons considéré les critères suivants : le nombre de contraintes (*Ctr*), le nombre de variables (*Var*), le nombre de nœuds (*Nœuds*) et le temps d'exécution du programme.

Tableau 2a. Résultats des instances aléatoires, Roszieg et Heskia (Modèle 1)

Instances		*N*	*H*	*S*	MODÈLE 1			
					Ctr	*Var*	*Nœuds*	*Temps d'exécution*
Aléatoires	**1**	5	3	3	184	105	38	0.14
	2	10	4	4	619	343	0	0.84
	3	10	5	5	901	533	1576	2.74
	4	10	6	6	1245	765	3951	6.49
	5	18	7	7	3059	1823	11023	49.91
Moyenne					**1202**	**714**	**3318**	**12.02**
Roszieg	**1**	25	4	4	1879	823	86	1.32
	2				1879	823	142	1.29
	3				1879	823	115	1.32
	4				1879	823	69	1.27
	5				1879	823	152	1.29
	6				1879	823	119	1.31
	7				1879	823	224	1.49
	8				1879	823	259	1.45
	9				1879	823	157	1.18
	10				1879	823	138	1.54
Moyenne					**1879**	**823**	**147**	**1.34**
Heskia	**1**	28	7	7	2185	919	206	1.76
	2				2185	919	85	0.82
	3				2185	919	153	1.04
	4				2185	919	0	1.03
	5				2185	919	206	1.40
	6				2185	919	60	1.01
	7				2185	919	124	1.21
	8				2185	919	152	1.04
	9				2185	919	97	1.07
	10				2185	919	0	1.27
Moyenne					**2185**	**919**	**108**	**1.16**
Moyenne Générale					**1865**	**839**	**765**	**3,4**

Tableau 2b. Résultats des instances aléatoires, Roszieg et Heskia (Modèle 2)

Instances		*N*	*H*	*S*	MODÈLE 2			
					Ctr	*Var*	*Nœuds*	*Temps d'exécution*
Aléatoires	**1**	5	3	3	753	42	16	0.10
	2	10	4	4	6579	99	88	0.84
	3	10	5	5	12771	128	873	0.95
	4	10	6	6	21995	159	766	01.63
	5	18	7	7	11809	304	3796	7.16
Moyenne					**10781**	**146**	**1108**	**2.13**
Roszieg	**1**	25	4	4	40314	219	2111	1.32
	2				40314	219	2073	1.27
	3				40314	219	2716	1.26
	4				40314	219	564	0.53
	5				40314	219	1383	1.21
	6				40314	219	2290	1.35
	7				40314	219	1636	1.24
	8				40314	219	1924	1.98
	9				40314	219	4077	1.62
	10				40314	219	2453	1.27
Moyenne					**40314**	**219**	**2123**	**1.30**
Heskia	**1**	28	7	7	50517	243	220	0.71
	2				50517	243	23	0.28
	3				50517	243	142	0.40
	4				50517	243	0	0.42
	5				50517	243	150	0.46
	6				50517	243	81	0.42
	7				50517	243	202	0.39
	8				50517	243	46	0.35
	9				50517	243	88	0.37
	10				50517	243	59	0.40
Moyenne					**50517**	**243**	**101**	**0.42**
Moyenne Générale					**38488**	**214**	**1111**	**1,11**

Les résultats présentés dans les tableaux 2a, 2b montrent que Cplex permet d'avoir une solution optimale rapidement en quelques secondes pour les instances de petites et de moyennes tailles. Notons qu'on a testé également les modèles 1 et 2 pour les instances Tonge ($N = 70$, $H = 10$) et Wee-mag ($N = 75$, $H = 11$) de la littérature et qui sont de grandes tailles (Chaves *et al.*, 2007). Pour ces instances, on a dépassé 4 h d'exécution sans résultats.

En comparant les valeurs moyennes des différents paramètres (*Ctr*, *Var*, *Nœuds*, temps d'exécution), on constate que le passage du modèle 1 au modèle 2 a permis de réduire le nombre de variables et par conséquent la taille du problème a été réduite.

On constate aussi que pour le modèle 2, le nombre de contraintes et de nœuds explorés lors de la résolution est plus important mais la résolution du programme se fait dans un temps de calcul plus petit que celui du modèle 1.

5.2. Résolution par l'heuristique gloutonne

L'algorithme glouton a été implémenté en langage C++. Afin d'illustrer les différentes étapes de résolution du problème avec cet algorithme, nous l'appliquons sur un exemple numérique. Notons que la règle de priorité adoptée pour le choix de l'opérateur est R_1.

Il est à souligner également que X(i,w) = (-1) dans la matrice de compétence signifie que l'opérateur w ne peut pas exécuter l'opération i. Et dans la matrice de précédence entre les opérateurs D(u,v)=(-1) lorsque l'opérateur w doit précéder u.

5.2.1. Exemple numérique de résolution du Modèle 2

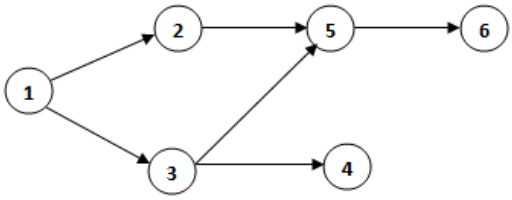

Figure 3. Graphe de précédence

Données du problème : $N = 6$ et $H = 3$

Matrice d'adjacence : $\boldsymbol{P}=$

$$\begin{matrix} 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{matrix}$$

Temps opératoire : ***T(i,w)***=
4 -1 3
4 5 4
3 6 2
1 5 -1
1 2 3
6 4 -1

Affichage des listes initiales : L_n = {1,2,3,4,5,6}, L_r = Ø, L_{iw} = {1,2,3}, L_a = Ø
Détermination des rangs et Affectation :
Rang =1 : L_r = {1}
pour i = 1, L_{iw} = {1,3}
après tri: L_{iw} = {3,1}
X(1,3)=1

Précédence entre opérateurs : ***D***=
0 0 0
0 0 0
0 0 0

L_a = {1}, L_n = {2,3,4,5,6}
Rang = 2 : L_r = {2,3}
pour i=2, L_{iw} = {1,2,3}
après tri: L_{iw} = {1,3,2}
X(2,1) = 1

Précédence entre opérateurs : ***D***=
0 0 -1
0 0 0
1 0 0

pour i=3, L_{iw} = {1,2,3}
après tri: L_{iw} = {3,1,2}
X(3,3) = 1

Précédence entre opérateurs : ***D***=
0 0 -1
0 0 0
1 0 0

L_a = {1,2,3}, L_n = {4,5,6}
Rang = 3 : L_r = {4,5}
pour i = 4, L_{iw} = {1,2}
après tri: L_{iw} = {1,2}
X(4,1) = 1

Précédence entre opérateurs : ***D***=
0 0 -1
0 0 0
1 0 0

pour i = 5, L_{iw} = {1,2,3}

après tri: L_{iw} = {1,2,3}
X(5,1) = 1

Précédence entre opérateurs : ***D***=

$$\begin{matrix} 0 & 0 & -1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{matrix}$$

L_a ={1,2,3,4,5}, L_n ={6}
Rang=4 : L_r ={6}
pour i=6, L_{iw} ={1,2}
après tri: L_{iw} ={2,1}
X(6,2) =1

Précédence entre opérateurs : ***D***=

$$\begin{matrix} 0 & 0 & -1 \\ -1 & 0 & 0 \\ 1 & 1 & 0 \end{matrix}$$

L_a ={1,2,3,4,5,6}, L_n =Ø
Matrice d'affectation : ***X***=

$$\begin{matrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{matrix}$$

Précédence entre opérateurs : ***D***=

$$\begin{matrix} 0 & 1 & -1 \\ -1 & 0 & 0 \\ 1 & 1 & 0 \end{matrix}$$

Charge horaire des opérateurs : T_1 =6, T_2 =4, T_3 =5
Temps de cycle de la ligne : C =6

La solution finale est présentée sous forme d'une matrice d'affectation et qui peut être schématisée par un graphe, comme celui de la figure 4, qui illustre également les nouvelles relations de précédence définies entre les 3 opérateurs.

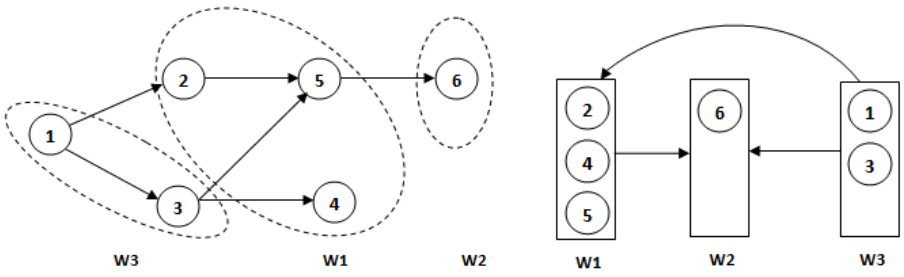

Figure 4. Solution et graphe de précédence entre opérateurs finals

Tableau 3. Résultats de l'heuristique gloutonne (avec R_1 et R_2) instances à taille moyenne

				R_1		R_2		R_1 avec R_0	
	Instance	***N***	***H***	***C***	**Temps d'exécution**	***C***	**Temps d'exécution**	***C***	**Temps d'exécution**
Aléatoire	1	5	3	5	0,103	6	0.59	5	0,119
	2	6	3	9	0,191	16	0.68	9	0,143
	3	10	4	16	0,240	15	1.39	16	0,25
	4	10	5	242	0,263	379	1.53	242	0,309
	5	10	6	172	0,289	606	1.71	194	0,307
	6	18	7	294	0,544	293	1.84	294	0,645
Moyenne				**123**	**0.271**	**219.16**	**1.29**	**126.6**	**0.295**
Roszieg	1	25	4	37	0.365	74	0.26	37	0,53
	2			40	0.302	46	0.273	32	0,56
	3			104	0.255	51	0.317	55	0,366
	4			34	0.312	57	0.324	47	0,368
	5			32	0.322	39	0.296	34	0,63
	6			32	0.308	44	0.309	16	0,409
	7			49	0.234	88	0.303	36	0,63
	8			32	0.321	92	0.287	32	0,64
	9			33	0.309	40	0.292	32	0,61
	10			33	0.336	36	0.222	34	0,59
Moyenne				**42.6**	**0.30**	**56.7**	**0.28**	**35,5**	**0,5333**
Heskia	1	28	7	249	0.315	315	0.321	374	1,571
	2			548	0.452	609	0.392	274	1,266
	3			322	0.325	418	0.344	129	0,351
	4			258	0.356	315	0.324	264	1,844
	5			523	0.382	375	0.348	317	1,304
	6			224	0.323	334	0.355	318	1,021
	7			554	0.301	554	0.332	277	1,538
	8			384	0.308	321	0.308	294	1,333
	9			257	0.347	272	0.338	261	1,351
	10			526	0.325	334	0,305	69	0,504
Moyenne				**384.5**	**0.34**	**384.7**	**0.33**	**257,7**	**1,2083**
Moyenne Générale				**192.65**	**0.271**	**220.34**	**0.538**	**142**	**0,738**

5.2.2. Instances testées et résultats

Les mêmes instances des tableaux 2a, 2b ont été testées avec l'heuristique gloutonne pour résoudre ce problème. Les résultats sont présentés dans le tableau 3, dans le cas où la règle de priorité R_1 a été utilisée pour le choix de l'opérateur qui va exécuter l'opération courante, ensuite pour le cas de R_2 et enfin en tenant compte des règles de priorité R_1 et R_0 en même temps lors de l'exécution de l'algorithme glouton. En fait, en comparant ces résultats avec ceux obtenus par Cplex, pour les mêmes instances, nous constatons que l'algorithme glouton fournit des solutions plus rapidement. Il est également clair que l'utilisation de R_1 mène à des résultats meilleurs que ceux de R_2, en termes de temps de cycle (C) et de temps de calcul du programme. Pour cette raison, nous avons gardé la règle R_1 qui tient compte du degré de compétence de l'opérateur w pour l'opération i et non pas de sa charge horaire actuel, lors de la sélection de l'opérateur disponible qui va exécuter l'opération courante i. Par la suite, nous avons testé notre algorithme en adoptant la règle R_1 et la règle R_0, les deux à la fois. La règle R_0 mène à choisir, de la liste courante L_r, l'opération ayant le plus petit temps opératoire.

La combinaison des deux règles (R0 et R1) a permis d'améliorer davantage la solution finale, puisque le temps de cycle C a été réduit. Toutefois, le temps de d'exécution du programme a légèrement augmenté, ce qui est évident. En effet, des calculs ont été ajoutés à l'algorithme afin de déterminer l'opération la plus petite dans la liste Lr.

Ces résultats nous appellent à adopter les règles R_1 et R_0 dans l'étude de nouvelles méthodes de résolution permettant d'améliorer les résultats obtenus et la résolution des instances de taille plus grande.

6. Conclusion et perspectives

Dans cet article, nous avons présenté des solutions d'affectation des ressources humaines aux lignes d'assemblage satisfaisant l'objectif de minimiser le temps de cycle. Nous avons présenté et testé notre approche de résolution basée sur une heuristique gloutonne. Pour ce faire, des règles de priorités ont été définies afin d'affiner le choix entre plusieurs solutions. Notre étude numérique a montré que la règle R_1 combinée avec la règle R_0 fournit des résultats meilleurs que ceux obtenus avec R_2, ce qui peut justifier le choix de ces deux règles pour une future recherche. Il est à noter aussi que nous avons obtenu grâce à cette méthode des solutions d'affectation en quelques secondes.

Le choix de cette approche constructive se justifie par sa rapidité par rapport aux autres approches telles que les méthodes d'amélioration itératives (ou méthodes de recherche locale) et les méthodes évolutives. En fait, à chaque itération on effectue une affectation d'une opération à un opérateur sans revenir sur les décisions prises à propos des affectations antérieures. Cette heuristique gloutonne s'arrête lorsqu'on obtient une solution complète. Afin de mieux optimiser cette solution, une heuristique d'amélioration peut être hybridée avec l'approche proposée. Cette

solution peut être également considérée comme solution de départ qui sert par la suite comme base d'une recherche plus efficace et plus complexe.

En guise de conclusion, nous affirmons que la résolution de ce problème d'optimisation combinatoire NP-difficile, avec des méthodes exactes, reste limitée à des problèmes de tailles plutôt réduites. Les solutions approchées sont alors l'axe à suivre pour des problèmes de tailles conséquentes. Un effort est à fournir pour trouver un meilleur compromis qualité de la solution/temps de résolution avec l'utilisation des métaheuristiques ou l'hybridation de notre approche avec une approche d'amélioration itérative qui est une perspective ouvrant des voies prometteuses pour améliorer la solution obtenue.

Bibliographie

Araújo F. F., Costa A. M., Miralles C. (2012). Two extensions for the ALWABP: Parallel stations and collaborative approach. *International Journal of Production Economics*, 140(1), 483-495.

Baybars I. (1986). A survey of exact algorithms for the Simple Assembly Line Balancing. *Management Science*, 32, pp. 909-932.

Bennour M., Addouch S., El Mhamedi A. (2005). RCPSP sous contraintes de compétences dans un service de maintenance. *Computers & Operations Research*, 32, 491–507.

Becker, C., Scholl, A. (2006). A survey on problems and methods in generalized assembly line balancing. *European journal of operational research*, 168(3), 694-715.

Blum C., Miralles C. (2011). On solving the assembly line worker assignment and balancing problem via beam search. *Computers & Operations Research*, 38, 328–339.

Boutevin, C., Gourgand, M., Norre, S., (2003). Méthodes d'optimisation pour le problème de l'équilibrage de lignes d'assemblage, *4eConférence Francophone de Modélisation et SIMulatio - MOSIM'03*, Toulouse, France, Avril.

Borba L. M., Ritt M.R.P. (2012). A task-Oriented Branch and Bound method for the Assembly Line Worker Assignment and Balancing Problem. *CLAIO SPBO, Congreso Latino-Liberoamericano de Investigacion Operativa, Simposio Brasileiro de Pesquisa Operacional*, 3192-3201.

Borba L, Ritt M. (2013). Exact and Heuristic Methods for the Assembly Line Worker Assignment and Balancing Problem. *Technical Report Number: 368*, Universidade Federal do Rio Grande do Sul, Instituto de Informatica.

Borba L., Ritt M. (2014). A heuristic and a branch-and-bound algorithm for the Assembly Line Worker Assignment and Balancing Problem. *Computers & Operations Research*, 45, 87-96.

Boysen, N., Fliedner, M., Scholl, A. (2007). A classification of assembly line balancing problems. *European Journal of Operational Research*, 183(2), 674-693.

Certa A., Enea M., Galante G., Manuela La Fata C. (2009). Multi-objective human resources allocation in R&D projects planning. *International Journal of Production Research*, 47(13), 3503-3523. DOI : 10.1080/00207540701824233.

Chaves A. A., Miralles C. Lorena, L. A. N. (2007). Clustering search approach for the assembly line worker assignment and balancing problem. *Proceedings of the 37th international conference on computers and industrial engineering*, Alexandria, Egypt, pp. 1469-1478.

Chaves A. A., Lorena L. A. N., Miralles C. (2009). Hybrid metaheuristic for the assembly line worker assignment and balancing problem. *Hybrid Metaheuristics*, pp.1-14. Springer Berlin Heidelberg.

Corominas A., Pastor R., Rodríguez, E. (2006). Rotational allocation of tasks to multifunctional workers in a service industry. *International Journal of Production Economics*, 103(1), 3-9.

Dakkak B., Chater Y., Talbi A. (2012). Modélisation d'un problème d'allocation des agents de maintenance. *CIGIMS'2012*. pp. 1–14.

Essafi M. (2010). *Conception et optimisation d'allocation de ressources dans les lignes d'usinage reconfigurables*. Thèse de Doctorat, École Nationale Supérieure des Mines de Saint-Étienne.

Ghosh S., Gagnon R. J. (1989). A Comprehensive Literature Review and Analysis of the Design, Balancing and Scheduling of Assembly Systems. *International Journal of Production Research*, 27 (4), pp. 637-670.

Groussi, A. (1998). *Equilibrage des lignes d'assemblage : théorie et application*. Mémoire, Université de Québec à Trois-Rivières, Canada.

Kumar, D. M. (2013). Assembly Line Balancing: A Review of Developments and Trends in Approach to Industrial Application. *Global Journal of Researches In Engineering*, 13(2).

Lanzarone E., Matta A. (2014). Robust nurse-to-patient assignment in home care services to minimize overtimes under continuity of care. *Operations Research for Health Care*, *3*(2), 48-58.

Miralles, C. Garcia-Sabater, J. P., Andres, C. and Cardos, M. (2007). 'Advantages of assembly lines in sheltered work centres for disabled'. A case study. International Journal of Production Economics, 110(1), 187-197.

Miralles C., García-Sabater J. P., Andrés C., Cardós, M. (2008). Branch and bound procedures for solving the assembly line worker assignment and balancing problem: Application to sheltered work centres for disabled. *Discrete Applied Mathematics*, 156(3), 352-367.

Moreira M. C. O., Costa A. M. (2009). A minimalist yet efficient tabu search algorithm for balancing assembly lines with disabled workers. *Pesquisa Operacional na Gestao do Conhecimento*, Porto Seguro, 660–671, XLI SPBO.

Moreira M. C. O., Costa A. M. (2012a). Hybrid heuristics for planning job rotation schedules in assembly lines with heterogeneous workers. *International Journal of Production Economics*, *141*(2), 552-560.

Moreira M. C. O., Ritt M., Costa A. M., Chaves A. A. (2012b). Simple heuristics for the assembly line worker assignment and balancing problem. *Journal of heuristics*, 18: 505–524.

Mutlu Ö., Polat O., Supciller A. A. (2013). An iterative genetic algorithm for the assembly line worker assignment and balancing problem of type-II. *Computers & Operations Research*, 40(1), 418-426.

Pastor, R., Ferrer, L. (2009). An improved mathematical program to solve the simple assembly line balancing problem. *International Journal of Production Research*, 47(11), 2943-2959.

Rekiek, B., Dolgui, A., Delchambre, A., Bratcu, A. (2002). State of art of optimization methods for assembly line design. *Annual Reviews in Control*, 26(2), 163-174.

Ritt, M., Costa, A. M. (2015). Improved integer programming models for simple assembly line balancing and related problems. *International Transactions in Operational Research*, 1-15. DOI :10.1111/itor.12206.

Scholl A., Voß S. (1996). Simple assembly line balancing—Heuristic approaches. *Journal of Heuristics*, 2, 217-244.

Scholl A., Becker C., State-of-the-art exact and heuristic solution procedures for simple assembly line balancing. *European Journal of Operational Research*, vol. 168, p. 666-693, 2006.

Schaus P., Van Hentenryck P., Régin J. C. (2009). Scalable load balancing in nurse to patient assignment problems. Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems. CPAIOR, LNCS 5547, pp. 248-262.

Selaru C. (2012). Resource allocation in project management. *International Journal of Economic Practices and Theories*, 2(4), 274-282.

Sivasankaran, P., Shahabudeen, P. (2014). Literature review of assembly line balancing problems. The International Journal of Advanced Manufacturing Technology, 73(9-12), 1665-1694.

Tan S., Weng W., Fujimura S. (2009). Scheduling of Worker Allocation in the Manual Labor Environment with Genetic Algorithm. *Proceedings of the International MultiConference of Engineers and Computer Scientists*, Vol. 1.

Vilà M., Pereira J., (2014). A branch-and-bound algorithm for assembly line worker assignment and balancing problems. *Computers & Operations Research*, 44, 105-114.

Wee T., Magazine M., Assembly line balancing as generalized bin packing, Operations *Reasearch Letters*, vol. 1, p. 56-58, 1986.

Article reçu le : 17 mai 2015

Article accepté le : 18 décembre 2015

REVUES LAVOISIER TARIFS 2017

TITRES	VOLUME	N°/AN	PRIX € *	ISSN
Annales de chimie	41	4	510	0151-9107
Biofutur		11	255	0294-3506
European Journal of Electrical Engineering	19	6	565	2103-3641
Geographie économie société	19	4	146	1295-926X
Instrumentation - Mesure - Métrologie	16	4	260	1631-4670
Journal européen des systèmes automatisés	50	6	839	1269-6935
Les cahiers du numérique	13	4	250	1622-1494
Politiques et management public	34	4	169	0758-1726
Revue des composites et des matériaux avancés	27	4	368	1169-7954
Revue française de gestion	43	8	295	0338-4551
Revue internationale de géomatique	27	4	420	1260-5875
RSTI – séries DN + ISI + RIA + TSI		**21**	**1300**	
RSTI - Document numérique (DN)	20	3	270	1279-5127
RSTI - Ingénierie des systèmes d'information (ISI)	22	6	525	1633-1311
RSTI - Revue d'intelligence artificielle (RIA)	31	6	409	0992-499X
RSTI - Technique et science informatiques (TSI) l'abonnement TSI inclut 6 TSI + 6 ISI + 3 DN + 6 RIA	36	21	1300	0752-4072
Traitement du signal	34	4	290	0765-0019

* Les prix des abonnements indiqués correspondent au tarif organismes France, pour la version imprimée par voie postale normale (incluant la version on line sans les archives). Les tarifs des autres pays et individuels, ainsi que les tarifs pour la version on line seule sont disponibles sur :

http://www.revuesonline.com ou sur demande (revues.abo@lavoisier.fr).