REVUE FRANÇAISE D'AUTOMATIQUE, D'INFORMATIQUE ET DE RECHERCHE OPÉRATIONNELLE. RECHERCHE OPÉRATIONNELLE

PIERRE NEPOMIASTCHY

Résolution d'un problème d'ordonnancement à ressources variables

Revue française d'automatique, d'informatique et de recherche opérationnelle. Recherche opérationnelle, tome 12, n° 3 (1978), p. 249-261.

<http://www.numdam.org/item?id=RO_1978__12_3_249_0>

R.A.I.R.O. Recherche opérationnelle/Operations Research
(vol. 12, n° 3, août 1978, p. 249 à 261)

RÉSOLUTION D'UN PROBLÈME D'ORDONNANCEMENT A RESSOURCES VARIABLES (*)

par Pierre NEPOMIASTCHY (1)

Résumé. — *Le problème étudié est une extension du « $n/m/G/C_{max}$ job-shop scheduling problem » de Conway. L'extension principale est de considérer que les machines sont réparties en groupes, les machines de chaque groupe étant interchangeables et en nombre variable avec le temps. Une généralisation de la formalisation de Manne nécessiterait l'introduction de variables entières supplémentaires associées à l'affectation des opérations sur les machines de leur groupe respectif; dans les problèmes concrets, le nombre de ces variables entières supplémentaires serait prohibitif.*

La méthode proposée consiste à « oublier » la contrainte cumulative et à résoudre un problème de programmation mathématique ordinaire en minimisant un coût de violation de cette contrainte. Cette méthode, très simple à utiliser, est surtout adaptée à la recherche d'ordonnancements admissibles et, testée sur un exemple concret, a donné des résultats encourageants.

1. INTRODUCTION

Soit le problème suivant. On doit exécuter n tâches indépendantes, chacune d'entre elles étant composée d'opérations élémentaires de durée d'exécution connue à l'avance et strictement ordonnées entre elles (2). Ces opérations élémentaires sont exécutées sur m machines toutes distinctes et toutes constamment disponibles. Il s'agit de trouver l'ordre dans lequel il faut exécuter ces opérations élémentaires de façon à minimiser le temps total d'exécution. Ce problème est appelé « $n/m/G/C_{max}$ job-shop scheduling problem » dans la terminologie de Conway [1].

A notre connaissance et à l'exception de Fisher [2] qui utilise les multiplicateurs de Lagrange et de l'auteur [3] qui utilise une méthode de pénalisation, la plupart des auteurs utilisent une modélisation en PLM (3), le PLM étant résolu par des méthodes combinatoires.

Notons que la modélisation en PLM généralement retenue est celle de Manne [4] et que les méthodes combinatoires les plus souvent utilisées sont l'algorithme connu aux États-Unis sous le nom de « Branch and Bound » (4) (*cf.* [5 à 8], etc.) ou des méthodes dérivées de la théorie des graphes (*cf.* [9]).

(*) Reçu novembre 1976, révisé octobre 1977.
(1) I.R.I.A.-Laboria.
(2) Pour pouvoir commencer l'exécution d'une opération élémentaire, il faut avoir terminé celle de l'opération élémentaire qui la précède dans la même tâche.
(3) Programmation linéaire mixte, c'est-à-dire programmation linéaire dans laquelle une partie des variables ne peut prendre que des valeurs entières.
(4) Procédure arborescente.

Malheureusement, il semble que le problème, tel qu'il est décrit ci-dessus, ne soit pas représentatif des problèmes d'ordonnancement que l'on rencontre dans la pratique. Son plus grave défaut est l'hypothèse essentielle que les machines sont toutes distinctes. Dans cet article, nous étudions une extension du modèle décrit par Conway; cette extension n'a certes pas la prétention d'englober tous les problèmes d'ordonnancement, mais nous pensons qu'elle est suffisamment générale pour que son intérêt dépasse l'étude du problème concret étudié au dernier paragraphe. L'extension la plus importante est, bien sûr, la suppression de l'hypothèse « les machines sont toutes distinctes ». On considère que les machines sont réparties en groupes, les machines de chaque groupe étant interchangeables entre elles et en nombre variable avec le temps. Pour chaque opération élémentaire, on sait sur quel groupe de machines elle est exécutée mais on ignore *a priori* sur quelle machine du groupe. Une généralisation de la formalisation de Manne permettant de ramener ce problème à un PLM nécessiterait l'introduction de variables entières supplémentaires associées à l'affectation des opérations élémentaires sur les machines de leur groupe respectif. Dans les problèmes concrets, le nombre prohibitif de ces variables entières supplémentaires interdit cette approche. C'est la raison pour laquelle, mise à part la méthode proposée par Trémolières [10] qui est spécialement adaptée au cas où toutes les opérations élémentaires ont une durée identique, nous ne connaissons pas de méthode capable de traiter le cas des machines interchangeables.

La méthode que nous proposons ici est basée sur la méthode de pénalisation (*cf.* [3]) mais n'utilise pas celle-ci de manière standard. Elle est très simple à utiliser et paraît être spécialement adaptée au cas où le problème posé est la recherche d'un ordonnancement admissible, ce qui est souvent le cas en pratique.

2. DESCRIPTION DU MODÈLE

On doit exécuter n tâches notées (i), chacune d'entre elles étant composée de a_i opérations élémentaires appelées simplement par la suite opérations. On désigne par s le double indice (i, j), la j-ième opération de la tâche (i) étant notée indifféremment O_{ij} ou O_s; par convention, on note respectivement $s-1$ et $s+1$ les doubles indices $(i, j-1)$ et $(i, j+1)$. On introduit les ensembles d'indices

$$I = \{ s=(i,j)/i=1,\ldots,n, j=1,\ldots,a_i \}, \tag{1a}$$

$$I_1 = \{ s \in I/j=1 \}, \tag{1b}$$

$$I_2 = \{ s \in I/j>1 \}, \tag{1c}$$

$$I_3 = \{ s \in I/j=a_i \}, \tag{1d}$$

$$I_4 = \{ s \in I/j<a_i \}. \tag{1e}$$

On notera que ces ensembles forment les partitions suivantes :

$$I = I_1 + I_2 \qquad \text{et} \qquad I = I_3 + I_4.$$

On appelle $N = \text{card}(I) = \sum_{i=1}^{n} a_i$ le nombre total d'opérations.

Par ailleurs, on dispose pour exécuter ces opérations d'un certain nombre de « machines ». En fait, « machine » désigne ici, en plus des machines proprement dites, de la main-d'œuvre, des biens rares tels que l'électricité, bref toutes les ressources non stockables; nous gardons le mot machine pour être cohérent avec la terminologie standard de l'ordonnancement. Ces machines sont réparties en m groupes, un groupe étant défini par le fait que les machines le composant sont interchangeables. On suppose connu à l'avance le tableau $K(k, t)$ donnant le nombre de machines du groupe k qui sont disponibles à la date t. Le fait de rendre ce tableau dépendant du temps permet de tenir compte à la fois des indisponibilités connues à l'avance de certaines ressources ([5]) et des variations connues des ressources ([6]). Notons que le modèle classique de Conway [1], dans lequel les machines sont distinctes et constamment disponibles, est simplement le cas particulier où le tableau K est identiquement égal à 1.

On dit qu'un ordonnancement est *admissible* si et seulement si son exécution satisfait les six règles suivantes :

RÈGLE 1 : la première opération de chaque tâche ($s \in I_1$) ne peut commencer avant une date connue appelée date de début au plus tôt de la tâche ([7]).

RÈGLE 2 : la dernière opération de chaque tâche ($s \in I_3$) ne peut se terminer après une date connue appelée date de fin au plus tard de la tâche ([8]).

RÈGLE 3 : les interruptions dans l'exécution d'une opération sont interdites; le temps d'exécution p_s de chaque opération est connu et indépendant de l'ordre des opérations.

RÈGLE 4 : *règle de précédence :* pour toute opération O_s qui n'est pas la dernière de sa tâche ($s \in I_4$), il doit s'écouler au moins une durée connue q_s entre le début de l'exécution des opérations O_s et O_{s+1} ([9]).

RÈGLE 5 : *règle d'affectation :* pour tout s, l'opération O_s est exécutée sur une machine *quelconque* du groupe de machines k_s, où k_s est connu.

([5]) Jours de maintenance des machines, congés de personnel, personnel réduit le dimanche, etc.
([6]) Livraison d'une nouvelle machine par exemple.
([7]) Contrainte due, par exemple, aux délais de livraison des matières premières.
([8]) Ces dates correspondent au carnet de commande.
([9]) Cette contrainte, tant technologique qu'organisationnelle, permet de tenir compte de nombreux facteurs, comme nous le verrons dans l'exemple réel traité à la fin de cet article.

RÈGLE 6 : *règle cumulative :* pour tout instant t et tout groupe de machines k, le nombre d'opérations en cours d'exécution sur des machines de ce groupe ne doit pas dépasser un nombre connu $K(k, t)$.

On vérifiera aisément que le modèle classique de Conway est un cas particulier du modèle décrit ci-dessus.

Le problème concret que nous avons été amené à résoudre est la recherche d'un ordonnancement admissible. Notons que dans le cas fréquent d'un atelier fonctionnant avec du matériel qui n'est pas loué à l'heure et du personnel qui est payé au mois, il n'est pas nécessaire de distinguer entre deux ordonnancements admissibles. Le problème, pour ces ateliers, est de respecter les délais de livraison imposés par le carnet de commande.

Par ailleurs, notons que le problème, noté $P(T)$, qui est la recherche d'un ordonnancement admissible dont la durée n'excède pas le paramètre T, est équivalent au problème posé ci-dessus après modification éventuelle des dates de fin au plus tard. Or, on montre (*cf.* [3], p. 71 et suivantes) que la recherche d'un ordonnancement de durée minimale (critère $C_{\max}$ chez Conway) peut se ramener à la résolution d'une suite de problèmes $P(T_k)$.

3. RECHERCHE D'UN ORDONNANCEMENT ADMISSIBLE

La règle 3 implique qu'un ordonnancement admissible est entièrement déterminé par la donnée des dates u_s, $s \in I$, de début d'exécution de ses opérations élémentaires. Nous choisirons donc les N variables u_s comme variables de décision du modèle et nous noterons u le vecteur de composantes u_s.

Avec toujours $s=(i, j)$ on vérifiera que pour que les règles 1, 2, 3, et 4 soient respectées, il faut et il suffit que le vecteur u satisfasse

$$u_s \geqq d_i^-, \qquad \forall s \in I_1, \tag{2a}$$

$$u_s + p_s \leqq d_i^+, \qquad \forall s \in I_3, \tag{2b}$$

$$u_{s+1} \geqq u_s + q_s, \qquad \forall s \in I_4, \tag{2c}$$

où d_i^- et d_i^+ sont (respectivement) les dates de début au plus tôt et de fin au plus tard de la tâche i. Nous poserons

$$u_s^- = d_i^-, \qquad \forall s \in I_1, \tag{3a}$$

$$u_s^- = u_{s-1} + q_{s-1}, \qquad \forall s \in I_2, \tag{3b}$$

$$u_s^+ = d_i^+ - p_s, \qquad \forall s \in I_3, \tag{3c}$$

$$u_s^+ = u_{s+1} - q_s, \qquad \forall s \in I_4. \tag{3d}$$

On a vu que $I = I_1 + I_2 = I_3 + I_4$ donc (3) définit, pour tout $s \in I$, les quantités u_s^- et u_s^+ que nous appellerons respectivement dates de début au plus tôt et au plus tard de l'opération O_s. On vérifiera que, compte tenu des définitions (3), (2) est équivalent à

$$u_s^- \leqq u_s \leqq u_s^+, \qquad \forall s \in I. \tag{4}$$

Il faut noter que (4) n'est pas une contrainte de borne (u_s^- et u_s^+ dépendent de u_{s-1} et u_{s+1}). Enfin, on pose

$$t^- = \min_{1 \leqq i \leqq n} d_i^-; \qquad t^+ = \max_{s \in I_3} (d_i^+ - p_s). \tag{5}$$

Toute solution admissible doit alors être recherchée sur l'intervalle $[t^-, t^+]$. On peut toujours, éventuellement après un changement d'unité de temps, considérer les données du problème comme étant entières. Dans ce cas les quantités t^- et t^+ de (5) sont entières. On pose

$$E = \{ u = \{u_s\} / \forall s \in I, \ u_s \text{ est un nombre entier}, \ t^- \leqq u_s \leqq t^+ \}. \tag{6}$$

On ne se limite pas en cherchant les ordonnancements admissibles dans E. Enfin, pour simplifier les notations ultérieures, on pose

$$T = \{ t / t \text{ entier}, \ t^- < t \leqq t^+ \}, \tag{7a}$$

$$M = \{ k / k \text{ entier}, \ 1 \leqq k \leqq m \}. \tag{7b}$$

Les variables u_s étant entières, l'activité de l'atelier est constante pendant une période $[t-1, t[, \forall t \in T$. On peut alors introduire dans l'étude une *variable d'état* $N(k, t)$ qui est le nombre d'opérations exécutées sur le groupe de machines k pendant la période $[t-1, t[, \forall k \in M, \forall t \in T$. Le graphe des fonctions $N(k, t)$ pour les différentes valeurs de k constitue le *plan de charge* de l'atelier. On vérifiera que si, pour tout $k \in M$, $t \in T$ et $s \in I$, on pose

$$\left.\begin{array}{r} N_s(k, t) = 1, \\ \text{si } u_s < t \leqq u_s + p_s \text{ et si } O_s \text{ est exécutée sur le groupe de machines } k, \end{array}\right\} \tag{8a}$$

$$N_s(k, t) = 0, \text{ dans le cas contraire}, \tag{8b}$$

alors on peut calculer la variable d'état N par la formule

$$N(k, t) = \sum_{s \in I} N_s(k, t), \qquad \forall k \in M, \quad \forall t \in T. \tag{9}$$

Les formules (8) et (9) constituent l'équation d'état du système. Par construction, une solution u vérifie les règles 5 et 6 si et seulement si on a, compte tenu de (8) et (9), les inégalités suivantes :

$$N(k, t) \leqq K(k, t), \qquad \forall k \in M, \quad \forall t \in T. \tag{10}$$

Finalement, une solution u est admissible si et seulement si elle vérifie les contraintes (4) et (10), où u_s^- et u_s^+ sont donnés par (3) et N est donné par (8) et (9).

Posons maintenant

$$g(k, t) = \max(N(k, t) - K(k, t), 0), \qquad \forall k \in M, \quad \forall t \in T, \tag{11}$$

$$G(u) = \sum_{k \in M} \sum_{t \in T} g(k, t), \qquad \forall u \in E. \tag{12}$$

Pour tout $u \in E$, les relations (8), (9) et (11) définissent complètement le tableau g, d'où la notation (12). On vérifiera aisément que G est une fonction non négative qui est nulle si et seulement si la contrainte (10) est satisfaite; G est donc une *fonction de pénalité* de la contrainte (10), dont nous donnerons une interprétation économique au paragraphe suivant.

Définissons maintenant un problème d'optimisation :

PROBLÈME P : minimiser $G(u)$ sous les contraintes : $u \in E$ et vérifie (4).

On démontre aisément le :

THÉORÈME 1 : *S'il existe au moins un ordonnancement admissible, alors le problème de la recherche d'un ordonnancement admissible à composantes entières est équivalent au problème* P. ■

Nous utiliserons pour résoudre le problème P la méthode séquentielle (*cf.*, par exemple, Céa [11], p. 109). Pour cela, on fixe toutes les composantes du vecteur u sauf une, la s-ième, on minimise $G(u)$ par rapport à u_s seul (problème d'optimisation à une seule variable), puis on cycle sur le numéro de la composante. Lors de la minimisation de la somme $G(u)$ définie en (12) par rapport à u_s, seuls interviennent les termes de la somme (12) qui comprennent u_s. Rappelant que k_s est le numéro du groupe de machines sur lequel est exécutée l'opération O_s, on vérifiera que la minimisation de G par rapport à u_s est en général équivalente au problème suivant ([10]) :

PROBLÈME P_s : parmi toutes les valeurs entières u_s qui sont comprises entre u_s^- et u_s^+, trouver celle qui minimise la fonction $G_s(u_s)$:

$$G_s(u_s) = \sum_{t=u_s+1}^{u_s+p_s} g(k_s, t). \tag{13}$$

La technique de résolution du problème P_s est la suivante. On commence par « soustraire » l'opération O_s de l'ordonnancement, ce qui revient à calculer le vecteur $N^0(t)$ qui décrit l'activité du groupe de machines k_s « moins

([10]) Le choix du problème (13) est motivé par le fait que, grâce à (17), le critère de (13) est très rapide à évaluer. Notons que si $G_s(u_s) = 0$ pour tous les s, alors u est bien un ordonnancement admissible.

l'opération O_s ». En appelant $B(s)$ l'ensemble des indices des opérations autres que O_s qui sont exécutées sur le groupe k_s, on a

$$N^0(t) = \sum_{r \in B(s)} N_r(k, t), \qquad \text{pour} \quad t = u_s^- + 1, \ldots, u_s^+ + p_s, \tag{14}$$

où N_r est toujours le tableau calculé à l'aide de (8). Ensuite, on « positionne » l'opération O_s à toutes les positions possibles, c'est-à-dire que l'on fait varier u_s par unité de u_s^- à u_s^+. Chaque positionnement revient à « ajouter » à l'ordonnancement l'opération O_s à la place choisie. On vérifiera qu'en posant

$$Z(t) = \max(N^0(t) - K(k_s, t) + 1, 0), \tag{15}$$

on a

$$G_s(u_s) = \sum_{t=u_s+1}^{u_s+p_s} Z(t). \tag{16}$$

L'intérêt de cette technique de calcul de $G_s(u_s)$ est que, contrairement à la fonction $g(k_s, t)$ intervenant dans (13), le tableau $Z(t)$ intervenant dans (16) ne dépend pas de la position de O_s et, par conséquent, reste constant au cours de la résolution de problème P_s. On commence donc par calculer $Z(t)$ pour $t = u_s^- + 1, \ldots, u_s^+ + p_s$, puis on calcule $G_s(u_s^- + 1)$ à l'aide de (16) et l'on obtient les valeurs suivantes de $G_s(u_s)$ à l'aide de la relation de récurrence

$$G_s(u_s + 1) = G_s(u_s) + Z(u_s + p_s + 1) - Z(u_s + 1), \qquad u_s = u_s^-, \ldots, u_s^+ - 1. \tag{17}$$

On obtient ainsi toutes les valeurs de $G_s(u_s)$ en faisant un minimum de calculs. Enfin, on « garde » pour u_s la valeur qui minimise $G_s(u_s)$. En cas de conflit entre plusieurs valeurs « optimales » de u_s, on garde soit la plus petite (ordonnancement au plus tôt) soit la plus grande (ordonnancement au plus tard) après tirage au sort, ceci pour éviter les blocages possibles de la méthode séquentielle.

Pour initialiser l'algorithme, on tire au sort un ordonnancement initial non admissible mais vérifiant (4). Le problème P n'étant pas convexe, l'algorithme converge généralement vers un optimum local ([11]). Il faut noter qu'il est très facile de distinguer entre optimum local et global : si l'on a convergé vers $\hat{u}$, alors $\hat{u}$ est optimum local si $G(\hat{u}) > 0$ et optimum global si $G(\hat{u}) = 0$. Si $\hat{u}$ est optimum global, alors, d'après le théorème 1, $\hat{u}$ est un ordonnancement admissible et le problème est résolu. Si $\hat{u}$ est optimum local, alors on recommence l'algorithme à partir d'une autre solution de départ.

La méthode ainsi décrite est assez heuristique. Elle reste cependant efficace car le temps de calcul d'un optimum local est généralement très faible. Pour un

([11]) Le test de convergence est ici très simple : on a convergé lorsqu'une itération entière (résolution de P_s pour tous les s) a été effectuée sans modification de la solution.

problème classique d'ordonnancement, nous avons effectué (*cf.* [3], chap. III) une étude comparative de cette méthode avec l'algorithme du Branch-and-Bound appliqué au modèle de Manne d'une part et avec une méthode de gradient appliquée à la résolution du problème P d'autre part. Cette étude a démontré une supériorité incontestable de la méthode séquentielle. Pour le problème d'ordonnancement étudié ici, la généralisation du modèle de Manne introduirait un nombre prohibitif de variables entières supplémentaires. En ce qui concerne les méthodes de gradient, la méthode séquentielle leur est ici supérieure pour la raison suivante. Lors de l'optimisation directionnelle ([12]), toutes les composantes de la solution varient et l'on doit évaluer le critère à l'aide des formules (11) et (12). Par contre, dans la méthode séquentielle, une composante seulement varie et l'on peut utiliser la relation de récurrence (17) pour évaluer le critère. L'expérience montre alors que le temps de calcul moyen de $G_s(u_s)$ est α fois plus petit que le temps de calcul du critère complet $G(u)$, où α est un nombre plus grand que le nombre N de variables du problème.

4. INTERPRÉTATION ÉCONOMIQUE DE LA MÉTHODE

Supposons que l'entreprise utilise en *fonctionnement normal* un matériel qui n'est pas loué à l'heure, un personnel payé mensuellement et que les différents coûts de fabrication soient indépendants des dates d'exécution des opérations et de leur ordre d'exécution. On peut alors admettre que l'exécution de tout ordonnancement admissible entraîne le même coût total de fabrication que nous appellerons par conséquent *coût fixe de fabrication.* Par un changement d'origine, on peut considérer que ce coût est nul.

Supposons maintenant qu'il soit techniquement possible d'exécuter simultanément un nombre plus grand d'opérations que ne l'autorise la contrainte cumulative (10), par exemple en sous-traitant une partie des opérations, en faisant faire des heures supplémentaires au personnel, en embauchant du personnel intérimaire, etc. Dans cette optique, une solution non admissible, donc violant (10), cesse d'être *irréalisable* mais, pour l'exécuter, vient s'ajouter au coût fixe de fabrication un coût supplémentaire. L'interprétation de la méthode est alors claire : résoudre le problème P, c'est cesser de considérer (10) comme une contrainte absolue et chercher à minimiser le coût supplémentaire dû à la violation de (10). En l'absence d'informations économiques sur ce coût supplémentaire, on peut faire simplement comme en (11) : considérer qu'il est proportionnel à la violation et à la durée de cette violation et que toutes les contraintes cumulatives sont d'égale importance.

([12]) La recherche du ρ qui minimise $G(u_n + \rho w_n)$, où w_n est une direction de descente.

Dans le cadre où nous nous sommes placés, c'est-à-dire chercher une solution admissible, le choix de la fonction (11) n'a apparemment pas d'importance : de toute façon, à l'optimum, le coût supplémentaire est nul. En fait, si le problème d'ordonnancement est de grande taille, sa résolution exacte est coûteuse et l'on peut être intéressé par des solutions « approchées », c'est-à-dire violant « peu » la contrainte cumulative; dans ce cas, il est évidemment préférable de remplacer la fonction (11) par un coût *réel* de violation.

Notons enfin qu'il peut arriver qu'il soit impossible de respecter les délais de livraison sans violer la contrainte cumulative. Dans ce cas, bien que le problème posé à l'origine (trouver un ordonnancement admissible) n'ait pas de solution, le problème P garde tout son sens : on cherche l'ordonnancement entraînant le coût supplémentaire le plus faible.

5. APPLICATION A UN PROBLÈME CONCRET

Il s'agit d'un problème d'ordonnancement (pour une usine de fabrication de pièces détachées pour machines textiles) qui nous a été communiqué par l'Université de Manchester par l'intermédiaire de J.-P. Schmidt du C.N.A.M.

Notons ici qu'une autre application de notre méthode à un problème concret (optimisation de la production d'une usine d'armements) a été réalisée avec succès [12].

Pour le problème qui nous occupe, on suppose que la livraison des produits finis s'effectue à la fin de chaque semaine, que la demande est connue assez longtemps à l'avance et que, pour des raisons de stockage et d'organisation, chaque pièce doit être fabriquée durant les deux semaines précédant sa date de livraison. Par ailleurs, pour des raisons organisationnelles et surtout de gain de temps sur le réglage des machines, on suppose que les pièces identiques appartenant à la même commande (c'est-à-dire livrées à la fin de la même semaine) constituent un lot de fabrication indivisible.

Une opération élémentaire O_s est alors le passage (réglage de la machine et exécution) d'un lot de fabrication donné sur un groupe de machines donné et repéré par son numéro k_s, le temps de réglage de la machine étant égal à e_s et le temps d'exécution du lot étant égal à f_s. Le temps total d'exécution de l'opération O_s, noté p_s comme précédemment, est donc égal à $e_s + f_s$. On suppose qu'à l'instant t, on dispose d'un nombre $K(k, t)$ de machines du groupe k.

On suppose d'autre part que toutes les données sont des nombres entiers d'heures (précision jugée suffisante par l'utilisateur) et qu'une semaine de travail représente 80 heures (deux équipes). Reprenant alors les notations de la contrainte (2) du modèle général, on a, pour tout i, $d_i^- = 0$ et $d_i^+ = 160$ pour la première commande dont l'exécution s'étale sur les deux premières semaines,

$d_i^- = 80$ et $d_i^+ = 240$ pour la deuxième commande exécutée pendant les semaines deux et trois et, plus généralement, $d_i^- = (q-1)\ 80$ et $d_i^+ = (q+1)\ 80$ pour la q-ième commande dont l'exécution s'étale sur les semaines q et $q+1$ et qui représente le q-ième problème d'ordonnancement à résoudre.

Les périodes de fabrication se recouvrant dans le temps, ces différents problèmes d'ordonnancement ne sont pas indépendants. Cependant, pour simplifier, on les résoudra en séquence, en tenant compte, bien entendu, lors de la résolution du q-ième problème, qu'entre les dates $(q-1)\ 80$ et $q\,.\,80$ des opérations du $(q-1)$-ième ordonnancement ont déjà été programmées ([13]). En décomposant le problème global en sous-problèmes, on prend un risque ([14]), mais si l'on trouve une solution admissible pour chaque sous-problème, on est assuré d'avoir une solution admissible du problème global qui est de taille trop importante pour être traité directement.

Le recouvrement dans le temps de l'exécution des commandes se justifie par le fait que l'ordre dans lequel un article passe sur les différentes machines dépend certes de l'article mais en dépend peu. Ainsi, pendant la q-ième semaine, on exécutera les opérations de dégrossissage de la q-ième commande et les opérations de finition de la $(q-1)$-ième commande.

Le seul problème pour pouvoir appliquer la méthode décrite ci-dessus est le calcul du tableau q_s intervenant dans la contrainte $(2\,c)$ en application de la règle 4 : q_s est le temps minimal qui doit s'écouler entre le début des opérations O_s et O_{s+1}. Dans le problème étudié ici, trois cas peuvent se présenter :

1[er] *cas* : entre les opérations O_s et O_{s+1} doit être exécutée une opération extérieure ([15]). On suppose qu'il n'y a pas de limitation sur le nombre d'opérations extérieures pouvant être exécutées simultanément. Par conséquent, q_s est égal à la somme de la durée p_s de l'opération O_s, de la durée de cette opération extérieure et du temps de transport.

2[e] *cas* : les opérations O_s et O_{s+1} sont exécutées sur la même machine dont on ne possède qu'un seul exemplaire ([16]). Alors, on a $q_s = p_s$, puisqu'il faut attendre que l'opération O_s soit complètement terminée pour pouvoir commencer l'exécution de l'opération O_{s+1}.

3[e] *cas* (*cas général*) : une opération étant le traitement d'un lot de pièces identiques (plusieurs centaines de pièces), il peut y avoir, sous certaines

---

([13]) Cela revient à modifier le tableau $K(k, t)$ des machines disponibles.

([14]) Le problème décomposé peut ne pas avoir de solution admissible même dans le cas où le problème global en a une.

([15]) Lavage ou séchage de la pièce.

([16]) Ce cas peut se produire lorsque la même machine (mais avec un réglage différent) est utilisée pour traiter de manières différentes la même pièce.

conditions, recouvrement dans l'exécution des opérations O_s et O_{s+1}. Détaillons ces conditions. Premièrement, il doit y avoir un décalage minimal de 1 heure entre le début de l'exécution effective de l'opération O_{s+1} (i. e. la fin de son réglage) et le début de l'exécution effective de l'opération O_s. Ce décalage minimal de 1 heure correspond au temps de traitement de la première pièce du lot augmenté du temps de transport des pièces d'une machine à l'autre. On a donc d'abord :

$$u_{s+1}+e_{s+1} \geqq u_s+e_s+1, \tag{18 a}$$

où e_s est rappelons-le, le temps de réglage pour l'opération O_s.

Deuxièmement, et pour les mêmes raisons, il doit y avoir un décalage minimal de 1 heure entre la fin de l'exécution de l'opération O_s et la fin de l'exécution de l'opération O_{s+1}, c'est-à-dire :

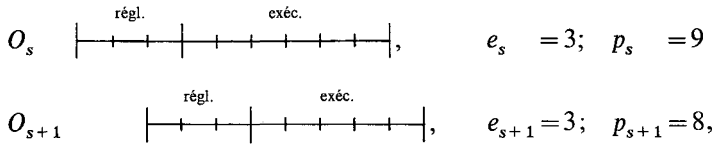
$$u_{s+1}+p_{s+1} \geqq u_s+p_s+1. \tag{18 b}$$

On vérifiera que les contraintes (18) sont équivalentes à la contrainte unique $u_{s+1} \geqq u_s+q_s$ avec :

$$q_s=1+\max(e_s-e_{s+1},\ p_s-p_{s+1}). \tag{19}$$

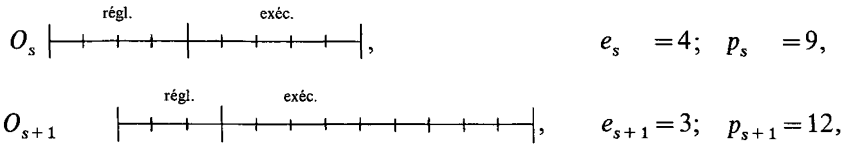
Notons que q_s peut être négatif : dans ce cas, le *réglage* de l'opération O_{s+1} peut commencer avant le réglage de l'opération O_s (*cf.* exemple 3).

Exemple 1 :



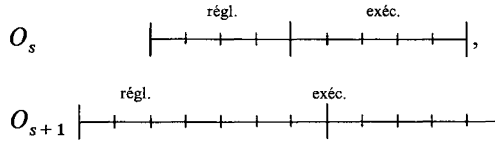
limitation par (18 *b*); $q_s=2$.

Exemple 2 :



limitation par (18 *a*); $q_s=2$.

Exemple 3 :



$e_s = 4$; $p_s = 9$,

$e_{s+1} = 7$; $p_{s+1} = 14$,

limitation par (18 *a*); $q_s = -2$.

Notons que le tableau q_s étant calculé une fois pour toutes avant le début de l'optimisation, son temps de calcul joue un rôle négligeable. Le calcul proposé ici n'étant évidemment qu'un exemple, on peut envisager sans difficultés des règles technologiques et organisationnelles beaucoup plus complexes que celles proposées ci-dessus, pourvu que ces règles puissent, d'une façon ou d'une autre, s'écrire sous la forme $u_{s+1} \geqq u_s + q_s$.

Nous avons programmé notre méthode pour une unité de fabrication comprenant 13 groupes de machines ($m = 13$) et fabricant 8 types d'articles ($n = 8$), l'usinage de ces articles comprenant 48 opérations élémentaires ($N = 48$) sans tenir compte des opérations extérieures qui n'interviennent que pour le calcul des q_s.

Il ne nous est malheureusement pas possible, pour des raisons matérielles, de donner ici les tableaux complets des données du problème. Les personnes intéressées peuvent contacter J.-P. Schmidt au C.N.A.M. En contactant M. Mondain, il est également possible d'obtenir les données d'un autre problème réel qui a été résolu par notre méthode (*cf.* [12]).

Il est clair que la difficulté d'obtention d'un ordonnancement admissible est d'autant plus grande que la semaine est plus chargée : l'importance des commandes influe sur la taille des lots, donc sur la durée d'exécution des opérations. Les deux tableaux ci-dessous donnent le temps de calcul *moyen T* nécessaire pour ordonnancer $k+1$ semaines de travail, c'est-à-dire résoudre successivement k problèmes d'ordonnancement. Le tableau I est donné pour une période moyennement chargée et le tableau II pour une période très chargée.

TABLEAU I

k.....	1	2	3	4	5	6	7	8
T.....	1,90	5,50	7,24	9,33	11,69	13,98	15,31	17,52

TABLEAU II

k.....	1	2	3	4
T.....	3,36	18,58	75,70	80,16

Dans les deux cas, le temps de calcul est exprimé en secondes d'unité centrale d'un ordinateur GE 235 programmé (par l'auteur) en Fortran IV sous le système Mark III.

Les temps de calcul ci-dessus ne sont pas proportionnels à k, entre autres parce que les semaines ne sont pas uniformément chargées.

De nombreuses études statistiques seraient nécessaires pour porter un jugement définitif sur la qualité de la méthode mais, au vu de notre expérience et de celle de Mondain [12], celle-ci semble prometteuse.

BIBLIOGRAPHIE

1. R. W. Conway et W. L. Maxwell, *Theory of Scheduling*, Addison-Wesley, 1967.
2. M. L. Fisher, *Optimal Solution of Scheduling Problems using Lagrange multipliers* (revised), University of Chicago, Report 7210, 1972.
3. P. Nepomiastchy, *Méthode de pénalisation et applications* (t. 1), Institut Européen de Recherches et d'Études Supérieures en Management (Bruxelles), doc. de travail 75-26, 1975.
4. A. S. Manne, *On the Job-Shop Scheduling Problem*, Oper. Res., vol. 8, n° 2, 1960, p. 219-223.
5. S. E. Elmaghraby et A. N. Elshafei, *Branch-and-Bound Revisited: A Survey of Basic Concepts and their Applications in Scheduling*, North Carolina Univ., OR Report n° 95, 1974.
6. M. Florian, P. Bratley et P. Robillard, *On Sequencing with Earliest Starts and Due-Dates with Application to Computing Bounds for the* $n/m/G/F_{max}$ *Problems*, Naval Res. Log. Quat., 20, 1973, p. 57-67.
7. J. K. Lenstra et A. H. G. Rinnoy, *Towards a Better Algorithm for the Job-Shop Scheduling Problem*, Math. Center Amsterdam, Report BN 22-73, 1973.
8. Said Ashour, *A Branch-and-Bound Algorithm for Flow Shop Scheduling Problems*, Ind. Eng. Res. and Devel., vol. 2, n° 6, 1970.
9. B. Roy, *Algèbre moderne et théorie des graphes*, t. 2, Dunod, Paris, 1972.
10. R. Trémolières, *La méthode de la ligne d'enchaînement pour les problèmes d'ordonnancement à contraintes cumulatives*, I.E.R.E.S.M., W. P. 73-13, 73-16, 73-19, 73-26, 73-31, 1973.
11. J. Cea, *Optimisation. Théorie et algorithmes*, Dunod, Paris, 1972.
12. M. Mondain, *Système de calcul des plans de production à long terme avec optimisation assistée*, Thèse de 3e Cycle, Université de Bordeaux, mai 1976.