RAIRO. RECHERCHE OPÉRATIONNELLE

J. C. HENNET

Comparaison de deux méthodes de résolution d'un problème combinatoire quadratique

RAIRO. Recherche opérationnelle, tome 17, n° 3 (1983), p. 285-295

<http://www.numdam.org/item?id=RO_1983__17_3_285_0>

R.A.I.R.O. Recherche opérationnelle/Operations Research
(vol. 17, n° 3, août 1983, p. 285 à 295)

COMPARAISON DE DEUX MÉTHODES DE RÉSOLUTION D'UN PROBLÈME COMBINATOIRE QUADRATIQUE (*)

par J. C. HENNET (1)

Résumé. — *Dans un système composé de N éléments en interaction mutuelle, on désire en sélectionner M (0 < M < N) de façon à maximiser un critère de performance quadratique. Ce problème combinatoire peut être résolu de façon optimale par un algorithme d'énumération implicite, avec des temps de calcul relativement faibles en raison de la simplicité de la technique de relaxation choisie pour l'évaluation des solutions candidates. Cependant, le traitement de problèmes de grande dimension a nécessité l'élaboration d'une méthode sous-optimale, inspirée de la programmation dynamique.*

Mots clés : Combinatoire, quadratique, énumération implicite, relaxation, programmation dynamique.

Abstract. — *In a system which consists of N mutually interacting elements, we want to select M of them (0 < M < N) in order to maximize a quadratic performance criterion. This combinatorial problem can be optimally solved by an implicit enumeration algorithm, with relatively low computing times due to the simplicity of the relaxation technique choosen for evaluating candidate solutions. However, in order to solve large size problems we had to elaborate a sub-optimal method drawn from dynamic programming.*

Keywords: Combinatorial, quadratic, implicit enumeration, relaxation, dynamic programming.

INTRODUCTION

La programmation combinatoire permet de formuler des problèmes qui mettent en jeu un grand nombre de décisions élémentaires liées entre elles. Ces interrelations peuvent apparaître dans l'énoncé du problème soit sous la forme de contraintes, soit par l'intermédiaire de termes mixtes dans le critère d'optimisation.

Nous nous intéressons ici au cas où ces deux types d'interrelations apparaissent sous leur forme la plus simple, à savoir dans une contrainte linéaire de budget et dans un critère quadratique à maximiser.

Le problème d'application sur lequel repose cette formulation est celui du choix de M emplacements parmi N possibles, ces emplacements devant être affectés aux héliostats d'une centrale solaire à tour.

(*) Reçu avril 1982.
(1) Laboratoire d'Automatique et d'Analyse des Systèmes, 7, avenue du Colonel Roche, 31400 Toulouse.

R.A.I.R.O. Recherche opérationnelle/Operations Research, 0399-0559/1983/285/$ 5.00

La plupart des méthodes classiques de résolution d'un problème combinatoire quadratique procèdent par énumération implicite. La mise en œuvre de ce type de méthode est particulièrement aisée dans le cas où la seule contrainte est une contrainte de budget. Cependant, la croissance exponentielle du temps de calcul en fonction de la taille du problème limite considérablement le domaine d'application de cette méthode.

C'est pourquoi nous avons élaboré une autre méthode, inspirée de la programmation dynamique, qui permet de trouver avec rapidité une solution sous-optimale satisfaisante quelle que soit la taille du problème.

I. FORMULATION DU PROBLÈME

I.1. Définition d'une matrice de performances

Considérons un système formé de N éléments. Chacun de ces éléments est caractérisé par un rendement intrinsèque, noté P_{II} pour l'élément indice I. Les interactions entre éléments se traduisent par des baisses de rendement. Le terme P_{IJ} est la baisse de rendement de l'élément I due à la présence de l'élément J, ce terme étant affecté du signe $-$. Le rendement de l'élément I lorsque tous les éléments sont présents dans le système vaut :

$$\rho_I = \sum_{J=1}^{N} P_{IJ}.$$

La matrice $P = ((P_{IJ}))$ est appelée « matrice de performances » du système.

Dans le cas où le système considéré est un champ d'héliostats, on calcule les quantités suivantes :

$P_U(I)$, puissance moyenne concentrée par l'héliostat I supposé ni ombré ni masqué par d'autres héliostats; E_N, flux incident normal moyen; S_{IJ}, portion moyenne de la surface de l'héliostat I ombrée ou masquée par l'héliostat J; S_I, surface de l'héliostat I.

Les termes de la matrice de performances sont définis de la façon suivante :

$$P_{II} = \frac{P_U(I)}{S_I . E_N},$$

$$P_{IJ} = -\frac{S_{IJ}}{S_I} \times P_{II}.$$

I.2. Optimisation des choix d'éléments

Supposons maintenant que parmi les N éléments définis précédemment, on cherche à en sélectionner au plus M ($0 < M < N$). Alors, seuls deviennent actifs les termes de la matrice P correspondant à deux indices I, J conjointement

sélectionnés. On représente la possibilité de sélection ou de rejet de l'élément d'indice I par la variable logique x_I telle que :

— si l'élément I n'est pas sélectionné, $x_I = 0$;

— si l'élément I est sélectionné, $x_I = 1$.

La sélection retenue est caractérisée par le vecteur binaire X tel que :

$$X^T = (x_1, \ldots, x_I, \ldots, x_N).$$

La performance globale de cette sélection d'éléments vaut :

$$Q(X) = X^T P X.$$

Le choix de la meilleure sélection d'au plus M éléments parmi N correspond à la résolution du problème combinatoire quadratique suivant ($\mathscr{P}$) :

$$(\mathscr{P}) \quad \begin{cases} \text{Maximiser } Q(X) = \displaystyle\sum_{I=1}^{N} \sum_{J=1}^{N} P_{IJ}\, x_I\, x_J \\ \text{sous } \displaystyle\sum_{I=1}^{N} x_I \leqq M \\ \text{avec } X^T = (x_1, \ldots, x_I, \ldots, x_N) \\ \text{et } \forall I \in 1, \ldots, N, \quad x_I = 0 \text{ ou } 1. \end{cases}$$

Dans le cadre de la conception des champs d'héliostats, les principales raisons qui ont poussé à adopter cette formulation sont les suivantes :

— on sait calculer les coefficients P_{II} et P_{IJ} lorsque les coordonnées des héliostats sont connues, mais on ne sait pas exprimer analytiquement l'évolution de ces coefficients lorsque les coordonnées varient continûment. On est donc toujours amené à discrétiser le problème;

— cette méthode permet de prendre en compte exactement le relief du terrain. On s'impose par exemple un maillage régulier du terrain disponible, que l'on projette sur la topologie du site. Les nœuds de ce maillage constituent alors les N emplacements possibles.

II. RÉSOLUTION DU PROBLÈME PAR ÉNUMÉRATION IMPLICITE

Les approches les plus courantes conduisant à la résolution des problèmes combinatoires quadratiques sont l'énumération implicite (Hansen, 1972; McBride et Yormark 1980) et la conversion en problème combinatoire linéaire par adjonction de nouvelles variables et de nouvelles contraintes (Glover et

Wolsey, 1974; Granot, Granot et Kallberg, 1979). Le premier type de méthode semble mieux adapté à la résolution de problèmes de taille importante. Nous verrons en outre que la formulation du problème conduit à une évaluation simple des solutions envisagées à chaque branchement de l'énumération.

II. 1. Évaluation par relaxation linéaire

A l'itération courante de la procédure d'énumération implicite, l'ensemble des indices $I=1, \ldots, N$ est partitionné en trois sous-ensembles K_0, K_1 et K_2 respectivement associés aux variables mises à 0, mises à 1 et libres.

On note:

$$N_0 = \text{Card}(K_0),$$
$$N_1 = \text{Card}(K_1),$$
$$N_2 = \text{Card}(K_2).$$

Pour tout indice $I \in K_2$, on définit le coefficient suivant :

$$C_I = P_{II} + \sum_{J \in K_1} (P_{IJ} + P_{JI}).$$

Le problème courant s'écrit :

$$(\mathscr{P}') \quad \begin{cases} \text{Maximiser } q(X) = \sum_{I \in K_2} x_I \left(C_I + \sum_{\substack{J \in K_2 \\ (J \neq I)}} P_{IJ}\, x_J \right) \\ \text{sous la contrainte } \sum_{I \in K_2} x_I \leqq M - N_1 \\ \text{avec } \forall I = 1, \ldots, N; \quad x_I = 0 \text{ ou } 1. \end{cases}$$

Le critère global associé vaut :

$$Q(X) = q(X) + \sum_{I \in K_1} \sum_{J \in K_1} P_{IJ}.$$

On considère la quantité suivante :

$$\bar{q}(X) = \sum_{I \in K_2} C_I\, x_I.$$

Comme tous les coefficients P_{IJ} (avec $J \neq I$) sont négatifs par définition, l'inégalité suivante est vérifiée pour tout vecteur X admissible pour le

problème $(\mathscr{P})$:

$$\bar{q}(X) \geqq q(X).$$

On peut considérer la relaxation linéaire du problème $(\mathscr{P}')$ notée $(\overline{\mathscr{P}'})$:

$$(\overline{\mathscr{P}'}) \quad \begin{cases} \text{Maximiser } \bar{q}(X) = \sum_{I \in K_2} C_I x_I \\ \text{sous les contraintes } \sum_{I \in K_2} x_I \leqq M - N_1 \\ \text{et } \forall I \in K_2; \quad x_I = 0 \text{ ou } 1. \end{cases}$$

Ce problème est extrêmement simple à résoudre. Il est connu sous le nom de Knapsack (Nauss, 1976). Sa résolution consiste à classer les indices $I (I \in K_2)$ par ordre décroissant de C_I. Dans cet ordre les variables sont mises à 1 jusqu'à ce que :

– ou bien $C_I \leqq 0$; aucune variable ne peut plus augmenter le critère $\bar{q}$;

– ou bien $M - N_1$ variables ont été mises à 1; la contrainte a atteint sa borne (cette borne est atteinte exactement puisque $M - N_1$ est un entier).

La solution $\bar{X}$ du problème $(\overline{\mathscr{P}'})$ fournit une borne supérieure à la quantité $q(X^*)$, X^* étant la solution optimale du problème $(\mathscr{P}')$:

$$\bar{q}(\bar{X}) \geqq \bar{q}(X^*) \geqq q(X^*).$$

Pour trouver une borne inférieure de $q(X^*)$, il suffit de calculer $q(\bar{X})$. On obtient alors un bon encadrement de $q(X^*)$:

$$q(\bar{X}) \leqq q(X^*) \leqq \bar{q}(\bar{X}).$$

La comparaison des solutions obtenues pour des branchements différents nécessite le calcul du critère global $Q(\bar{X})$ et de la quantité $\bar{Q}(\bar{X}) = \bar{q}(\bar{X}) + \sum_{I \in K_1} \sum_{J \in K_1} P_{IJ}$. On trouve alors des bornes inférieure et supérieure de $Q(X^*)$:

$$Q(\bar{X}) \leqq Q(X^*) \leqq \bar{Q}(\bar{X}).$$

II. 2. Algorithme proposé

La séparation des solutions candidates peut se faire à partir des coefficients C_I du problème $(\overline{\mathscr{P}'})$.

Si la solution courante de $(\overline{\mathscr{P}'})$ est non admissible pour le problème $(\mathscr{P}')$, le branchement suivant consistera à mettre à 1 la variable x_I correspondant au plus grand coefficient C_I dans le problème $(\overline{\mathscr{P}'})$ courant.

Si l'évaluation montre que la solution courante ne peut pas être optimale, le branchement suivant consistera à mettre à 0 la dernière variable précédemment mise à 1.

Le programme d'optimisation conçu à partir des techniques de séparation, relaxation et évaluation que nous venons de décrire peut se schématiser ainsi :

— 1re étape : mise en forme du problème courant $(\overline{\mathscr{P}'})$;

— 2e étape : calcul de $\overline{Q}\,(\overline{X})$ et de la variable de plus grand C_I;

— 3e étape : si $\overline{Q}\,(\overline{X})$ est supérieur à la plus grande borne inférieure de la solution, continuer à l'étape 4. Sinon aller à l'étape 6;

— 4e étape : calcul de $Q\,(\overline{X})$;

— 5e étape : si $Q\,(\overline{X}) \neq \overline{Q}\,(\overline{X})$ la solution de $(\overline{\mathscr{P}'})$ est non admissible pour le problème $(\mathscr{P}')$. On fixe à 1 la variable de plus grand C_I et on se branche à l'étape 1. Si $Q\,(\overline{X}) = \overline{Q}\,(\overline{X})$, la solution est admissible et non inférieure. Elle est conservée et la procédure continue à l'étape 6;

— 6e étape : on met à 0 la dernière variable mise à 1. Si l'ensemble K_1 n'est pas vide, on se branche à l'étape 1. S'il est vide, l'énumération est terminée. La meilleure solution est la dernière conservée.

Les performances de cet algorithme sont bonnes tant que les valeurs de N et M ne sont pas trop grandes. Or le problème de conception des champs d'héliostats peut atteindre une taille considérable. C'est pourquoi on a élaboré une seconde méthode, sous-optimale mais beaucoup plus performante, inspirée de la programmation dynamique.

III. CONSTRUCTION D'UNE SOLUTION SOUS-OPTIMALE PAR PROGRAMMATION DYNAMIQUE

III. 1. Construction de solutions admissibles

Considérons une solution admissible, notée $\hat{X}$, du problème $(\mathscr{P})$ telle que :

$$\hat{X} = (\hat{x}_1, \ldots, \hat{x}_I, \ldots, \hat{x}_N)$$

et :

$$\sum_{I=1}^{N} \hat{x}_I = j$$

avec $1 \leq j < M$.

La solution $\hat{X}$ est dite d'ordre j. On note $K_1(\hat{X})$ l'ensemble des indices tels que :

$$\forall I \in K_1(\hat{X}), \quad \hat{x}_I = 1,$$
$$\forall I \notin K_1(\hat{X}), \quad \hat{x}_I = 0.$$

A partir de la solution $\hat{X}$ d'ordre j, on peut construire des solutions admissibles $\tilde{X}$ d'ordre $j+1$ de la façon suivante :

$$\tilde{X}^T = (\tilde{x}_1, \ldots, \tilde{x}_I, \ldots, \tilde{x}_N),$$

avec :

$$K \notin K_1(X) \text{ (il existe au moins une valeur de } K \text{ si } j < M \leq N),$$
$$\forall I \in K_1(X); \quad \tilde{x}_I = 1,$$
$$\forall I \notin K_1(X) \quad \text{et} \quad I \neq K; \quad \tilde{x}_I = 0 \quad \text{et} \quad \tilde{x}_K = 1.$$

Le critère associé à la solution $\tilde{X}$ vaut :

$$Q(\tilde{X}) = Q(\hat{X}) + P_{KK} + \sum_{J \in K_1(\hat{X})} (P_{JK} + P_{KJ}).$$

Réciproquement, si $\tilde{X}$ est une solution admissible d'ordre j, avec $1 < j \leqq M$, il est toujours possible de construire une solution admissible $\hat{X}$ telle que :

$$\hat{X}^T = (\hat{x}_1, \ldots, \hat{x}_I, \ldots, \hat{x}_N)$$

avec $K \in K_1(\tilde{X})$:

$$\forall I \notin K_1(\tilde{X}), \quad \hat{x}_I = 0,$$
$$\forall I \in K_1(\tilde{X}) \quad \text{et} \quad I \neq K, \quad \hat{x}_I = 1,$$
$$\hat{x}_K = 0.$$

Pour trouver les meilleures solutions d'ordre M, on considère les ordres $j = 1, \ldots, M$ comme les étapes d'une procédure de programmation dynamique. Chaque solution admissible d'ordre M peut être représentée comme un ensemble ordonné de décisions du type : « $x_I = 1$ à l'étape j ». A chaque étape il y a donc N décisions possibles différentes. A chacune de ces décisions est associé un ensemble optimal de décisions aux ordres $1, \ldots, j-1$.

Pour réduire la taille du domaine de solutions à tester, on suppose qu'il existe une propriété de type markovien entre les étapes. On suppose que la dernière décision prise (à l'ordre $j-1$) résume toutes les décisions antérieures. Alors le nombre d'états possibles à l'ordre j est au plus N, chacun étant défini par la dernière décision : « $x_I = 1$ à l'étape j » (avec $I = 1, \ldots, N$).

III.2. Algorithme de programmation dynamique

La fonction d'évaluation à l'étape j de la décision « $x_I = 1$ à l'étape j » est notée $f_j(I)$. Cette fonction est construite ainsi :

$$\forall I \in (1, \ldots, N), \qquad f_1(I) = P_{II}$$

et pour $2 \leqq j \leqq M$:

$$f_j(I) = P_{II} + \operatorname*{Max}_k \left[f_{j-1}(K) + \sum_{L \in L_{j-1}(K)} (P_{IL} + P_{LI}) \right].$$

L'ensemble d'indice $L_{j-1}(K)$ est l'ensemble noté précédemment $K_1(X)$, X étant le vecteur binaire associé à la décision : « $x_K = 1$ à l'étape $j-1$ ».

Le choix de K est limité aux indices tels que :

$$I \notin L_{j-1}(K).$$

La construction des séquences de décisions de $j=1$ à $j=M$ permet de trouver la meilleure solution du problème, c'est-à-dire le vecteur X^* défini par le couple d'indices (I^*, j^*) tel que :

$$f_{j*}(I^*) = \operatorname{Max} f_j(I),$$

$$I = 1, \ldots, N,$$

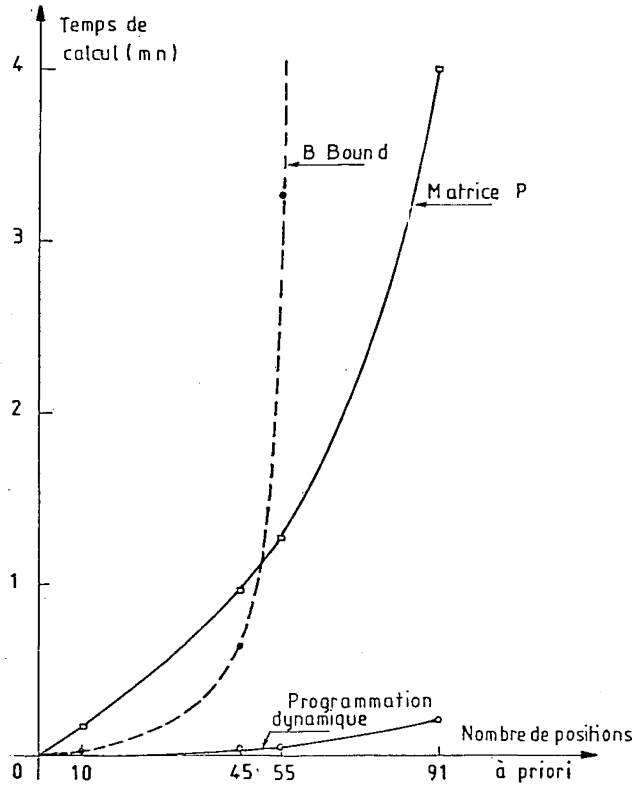
$$j = 1, \ldots, M.$$

La sous-optimalité de cette solution est due à l'approximation markovienne énoncée précédemment.

L'organigramme de programmation utilisé pour tester cette méthode est formé de boucles imbriquées. Cette structure a permis de majorer le temps de calcul par une expression polynomiale dont le terme de plus haut degré est $\zeta M^2 N^2$, ζ étant une constante représentant le temps de calcul nécessaire à quelques opérations de test et d'addition. L'algorithme est donc au plus de complexité polynomiale $O(M^2 N^2)$ en temps de calcul.

IV. ÉVALUATION DES DEUX MÉTHODES

Le calcul de la matrice P et la mise en œuvre de deux méthodes de résolution du problème d'optimisation combinatoire ont été réalisés pour 4 problèmes-type. Les résultats sont représentés sur le tableau. On en a déduit les courbes de la figure. Les calculs ont été effectués sur un ordinateur IBM 370/168.



On peut constater que l'algorithme d'énumération implicite est suffisamment rapide pour des valeurs de N allant jusqu'à environ 55. Au-delà de cette valeur, le seul algorithme utilisable est l'algorithme de programmation

TABLEAU

	Matrice P	Temps B. Bound	Temps P. Dynamique	Solution B. Bound	Solution P. Dynamique	Différences entre les solutions
Problème 1 (5 parmi 10). . . .	10 s	0,96 s	0,88 s	4,033	4,023	0,010
Problème 2 (20 parmi 45). . . .	58 s	38,55 s	2,10 s	16,643	16,643	0
Problème 3 (20 parmi 55). . . .	1 mn 16 s	3 mn 16 s	2,61 s	16,713	16,655	0,057
Problème 4 (25 parmi 91). . .	4 mn	non terminé en 25 mn	12,71 s	20,814 (*)	20,784	0,030

(*) L'optimalité de cette valeur n'est pas garantie puisque le programme a été interrompu après 25 mn de temps de calcul.

dynamique, dont les performances en temps de calcul sont excellentes. La différence entre la solution optimale et la solution sous-optimale est habituellement inférieure à 5 $^0/_{00}$.

Le choix de l'algorithme de programmation dynamique pour résoudre des problèmes de grande taille reporte les limitations en temps de calcul sur la construction de la matrice P dans le cas où cette matrice représente les performances d'un champ d'héliostats fictif.

CONCLUSION

Le problème de choix de M éléments parmi N possibles en interaction entre eux peut être formulé sous la forme d'un programme combinatoire quadratique.

On peut résoudre ce programme de façon optimale par un algorithme d'énumération implicite lorsque le produit $M \times N$ ne dépasse guère la valeur 1 100 (problème 3 du tableau).

Pour des problèmes de taille plus importante, on a recours à une méthode sous-optimale. Il est possible de réduire le temps de calcul en ne sondant qu'une partie de l'ensemble des solutions admissibles, cette partie devant contenir des solutions suffisamment proches de la solution optimale.

La procédure de construction de solutions proposée permet de vérifier cette condition. C'est un algorithme déduit de la programmation dynamique. Nous avons testé cet algorithme et montré qu'il permet de résoudre des problèmes de grande taille sans trop s'écarter de l'optimum.

BIBLIOGRAPHIE

R. BELLMAN, *Dynamic Programming*, Princeton University Press, Princeton, H.J., 1957.

A. M. GEOFFRION et R. E. MARSTEN, *Integer Programming Algorithms: A Framework and a State-of-the-Art Survey*, Management Science, vol. 18, n° 9, mai 1972, p. 465-491.

F. GLOVER et R. E. WOLSEY, *Converting the* 0-1 *Polynomial Programming Problem to a* 0-1 *Linear Program*, Operations Research, vol. 22, 1974, p. 180-182.

D. GRANOT, F. GRANOT et J. KALLBERG, *Covering Relaxation for Positive* 0-1 *Polynomial Programs*, Management Science, vol. 25, n° 3, mars 1979, p. 264-273.

P. HANSEN, *Quadratic* 0-1 *Programming by Implicit Enumeration*, in *Numerical Methods for Non Linear Optimization*, LOOTSMA, éd., Academic Press, 1972, p. 282-296.

J. C. HENNET, *Étude des effets d'ombre entre héliostats d'une centrale solaire*, Note interne LAAS-ASE, novembre 1980.

J. C. HENNET, *Resolution of a Quadratic Combinatorial Problem by Dynamic Programming*, IFIP Conference, New York, 1981.

R. D. McBRIDE et J. S. YORMARK, *An Implicit Enumeration Algorithm for Quadratic Integer Programming*, Management Science, vol. 26, n° 3, mars 1980, p. 282-296.

R. M. NAUSS, *An Efficient Algorithm for the* 0-1 *Knapsack Problem*, Management Science, vol. 23, n° 1, septembre 1976.

G. L. NEMHAUSER et Z. ULLMAN, *Discrete Dynamic Programming and Capital Allocation*, Management Science, vol. 15, 1969, p. 494-505.

J. C. HENNET et J. L. ABATUT, *An Analytical Method for Reflected Flux Density Calculations*, Solar World Forum, Brighton, 1981.

G. GALLA, P. L. HAMMER et B. SIMEONE, *Quadratic Knapsack Problems*, Math. Prog. study, vol. 12, 1980, p. 132-149.