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

M. GONDRAN

Des algorithmes linéaires pour les problèmes de partition, de recouvrement et de couplage dans les hypergraphes d'intervalles

Revue française d'automatique, d'informatique et de recherche opérationnelle. Recherche opérationnelle, tome 13, n° 1 (1979), p. 13-21.

<http://www.numdam.org/item?id=RO_1979__13_1_13_0>

R.A.I.R.O. Recherche opérationnelle/Operations Research
(vol. 13, n° 1, février 1979, p. 13 à 21)

DES ALGORITHMES LINÉAIRES POUR LES PROBLÈMES DE PARTITION, DE RECOUVREMENT ET DE COUPLAGE DANS LES HYPERGRAPHES D'INTERVALLES (*)

par M. GONDRAN (1)

Résumé. — *On montre dans cette note que les problèmes de partition, de recouvrement et de couplage dans le dual d'un hypergraphe d'intervalles admettent des algorithmes polynomiaux basés sur la programmation dynamique : ces algorithmes sont respectivement en* $0(n)$, $0(\alpha nm)$ *et* $0(n)$ *où n est le nombre des variables, m le nombre des contraintes et* α *le taux de remplissage de la matrice des contraintes.*

1. INTRODUCTION

Les problèmes de partition, de recouvrement et de couplage d'un hypergraphe se posent dans de nombreux problèmes concrets (cf. [1], chapitre 11 et [7]).

Ces trois problèmes sont dans le cas général NP-complets [2], c'est-à-dire qu'on ne connaît pas pour eux d'algorithmes polynomiaux et qu'il est conjecturé, qu'il n'en existe pas.

Dans le cas où l'hypergraphe est le dual d'un hypergraphe d'intervalles on présente ici pour chacun de ces trois problèmes un algorithme polynomial basé sur la programmation dynamique.

Rappelons la définition d'un hypergraphe d'intervalles ([3], p. 438). C'est un hypergraphe $(S, \mathscr{E})$, où S est un ensemble de points sur une droite D, et où $\mathscr{E}$ est une famille d'intervalles, c'est-à-dire de la forme :

$$[a, b] = \{ s \mid s \in S, a \leq s \leq b \}.$$

La matrice d'incidence sommets-arêtes du dual d'un hypergraphe d'intervalles est donc constituée de lignes dont les éléments 1 sont consécutifs.

(*) Manuscrit reçu Octobre 1977, révisé Mars 1978.

(1) Électricité de France, Direction des Études et Recherches. Service Informatique et Mathématiques Appliquées.

R.A.I.R.O. Recherche opérationnelle/Operations Research, 0399-0842/1979/13/$ 1.00

Exemple :

$$A = \begin{pmatrix} 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \end{pmatrix}$$

On rencontre des hypergraphes d'intervalles par exemple dans certains problèmes d'affectation des équipages [5].

Un tel hypergraphe étant unimodulaire ([3], p. 438) la matrice d'incidence est totalement unimodulaire, et la solution continue des programmes linéaires associés à ces trois problèmes est entière [4]. De tels problèmes peuvent donc être résolus par la méthode du simplexe.

Les algorithmes présentés ici seront plus efficaces que le simplexe puisqu'ils seront linéaires par rapport au nombre de variables pour les problèmes de partition (paragraphe 2) et de couplage (paragraphe 4) et linéaire par rapport au nombre des éléments non nuls de la matrice d'incidence pour le problème de recouvrement (paragraphe 3).

Remarquons que les résultats des paragraphes suivants s'appliquent aux problèmes de partition, de recouvrement et de couplage d'un hypergraphe d'intervalles pour lequel aucune arête n'en contient une autre ; en effet dans ce cas l'hypergraphe et son dual sont tous deux des hypergraphes d'intervalles ([3], p. 394).

Remarquons enfin que ces problèmes peuvent être ramenés à des problèmes de flots comme l'a montré Minoux [6].

2. UN ALGORITHME EN $0(n)$ POUR LE PROBLÈME DE PARTITION

Le problème de partition (PP) s'écrit :

$$\left.\begin{array}{lll} \text{minimiser} & z = \sum_{j=1}^{n} C_j x_j & \\ \text{avec :} & \sum_{j=1}^{n} a_{ij} x_j = 1 & i \text{ de } 1 \text{ à } m \\ & x_j = 0 \text{ ou } 1 & j \text{ de } 1 \text{ à } n \end{array}\right\} \quad (1)$$

avec $C_j \geq 0$ et $A = (a_{ij})$ est la matrice d'incidence du dual d'un hypergraphe d'intervalles.

Notons :

$$J_i = \{ j \mid a_{ij} = 1 \} \tag{2}$$

On a alors :

LEMME 1 : *Si* $J_l \subset J_k$, *alors dans* (*PP*) *on peut supprimer la contrainte* k *et faire* $x_j = 0$ *pour* $j \in J_k - J_l$.

En appliquant en cascade le lemme 1, on obtient une matrice où il n'y a plus d'inclusion entre deux lignes.

Dans l'exemple, on a $J_6 \subset J_1$ et $J_6 \subset J_2$. On supprime donc les lignes 1 et 2 ainsi que les colonnes 2 et 5.

La matrice restante est alors :

$$\bar{A} = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \end{pmatrix}$$

Classons maintenant les lignes de la matrice $\bar{A}$ dans l'ordre de leur premier élément non nul. Puisque nous n'avons pas d'inclusion entre deux lignes dans A cela revient aussi à classer les lignes dans l'ordre inverse de leur dernier éléments non nuls.

Dans l'exemple précédent l'ordre de lignes sera 3, 4, 2 et 1 et on aura :

$$\bar{\bar{A}} = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix}$$

La matrice A mise sous la forme $\bar{\bar{A}}$ sera dite mise sous forme canonique.

On supposera dans toute la suite de ce paragraphe que la matrice A nous est donnée sous forme canonique. Si la matrice A n'est pas irréductible, le problème se décompose. On suppose donc A irréductible (cela entraîne en particulier $n \geq m$).

Pour tout $l \in J_k$ considérons les problèmes $\mathscr{P}_k(l)$ suivants :

$$\left.\begin{array}{lll} \text{minimiser} & z = \sum_{j=1}^{n} C_j x_j & \\ \text{avec :} & \sum_{j \in J_i}^{n} x_j = 1 & i \text{ de 1 à } k \\ & x_j = 0 \text{ ou } 1 & j \text{ de 1 à } n \\ & x_l = 1 & \end{array}\right\} \tag{3}$$

Appelons $f_k(l)$ la valeur de la solution optimale de $\mathscr{P}_k(l)$.

PROPOSITION 1 : *Pour* $p \in J_{k+1}$, *on a :*

$$f_{k+1}(p) = f_k(p) \qquad si \quad p \in J_k \tag{4}$$

$$f_{k+1}(p) = C_p + \min_{l \in J_k - J_{k+1}} (f_k(l)) \qquad si \quad p \notin J_k \tag{5}$$

Démonstration : immédiat par définition des $\mathscr{P}_k(l)$. ■

De la proposition 1, on déduit l'algorithme de programmation dynamique suivant :

Algorithme 1 :

α — *initialisation*

Pour tout $j \in J_1$, *faire* $f(j) = C_j$.

$k = 1$.

β — *étape* k

Si $k = m$, *aller en* γ

Calculer $h = \min_{j \in J_k - J_{k+1}} f(j)$

Pour tout $j \in J_{k+1} - J_k$, *faire* $f(j) = C_j + h$

$k \leftarrow k + 1$

Aller en β.

γ — *Calculer* $z_{\min} = \min_{j \in J_m} f(j)$

($z_{\min}$ *est la valeur de la solution optimale du PP*).

FIN.

Le calcul de h demande à l'étape k, $|J_k - J_{k+1}| - 1$ comparaisons, donc au total $\sum_k |J_k - J_{k+1}| - 1 \simeq n - m$ comparaisons.

Le calcul de $f(j)$ demande à l'étape k, $|J_{k+1} - J_k|$ additions, donc au total $\sum_k |J_{k+1} - J_k| \simeq n$ additions.

On en déduit donc :

THÉORÈME 1 : *L'algorithme 1 est en* $0(n)$.

3. UN ALGORITHME POLYNOMIAL POUR LE PROBLÈME DE RECOUVREMENT

Le problème de recouvrement (*PR*) s'écrit :

$$\left.\begin{array}{lll} \text{minimiser} & z = \sum_{j=1}^{n} c_j x_j & \\ \text{avec :} & \sum_{j=1}^{n} a_{ij} x_j \geq 1 & i \text{ de 1 à } m \\ & x_j = 0 \text{ ou } 1 & j \text{ de 1 à } n \end{array}\right\} \tag{6}$$

LEMME 2 : *Si* $J_l \subset J_k$, *alors dans* (*PR*) *on peut supprimer la contrainte* k.

Après suppression de telles lignes on peut donc mettre la matrice A sous forme canonique.

Pour tout $l \in J_k$ considérons les problèmes $Q_k(l)$ suivants :

$$\left.\begin{array}{lll} \text{minimiser} & z = \sum_{j=1}^{n} C_j x_j & \\ \text{avec :} & \sum_{j=1}^{n} a_{ij} x_j \geq 1 & i \text{ de } 1 \text{ à } k \\ & x_j = 0 \text{ ou } 1 & j \text{ de } 1 \text{ à } l-1 \\ & x_l = 1 & \\ & x_j = 0 & j \text{ de } l+1 \text{ à } n \end{array}\right\} \quad (7)$$

x_l est donc la dernière variable non nulle.

Appelons $f_k(l)$ la valeur de la solution optimale de $Q_k(l)$.

PROPOSITION 2 : *Pour* $p \in J_{k+1}$, *on a :*

$$f_{k+1}(p) = \min\left[f_k(p),\ C_p + \min_{\substack{l \in J_k \\ l<p}} f_k(l)\right] \qquad \text{si} \quad p \in J_k \qquad (8)$$

$$f_{k+1}(p) = C_p + \min_{l \in J_k - J_{k+1}} (f_k(l)) \qquad \text{si} \quad p \notin J_k \qquad (9)$$

Démonstration : Si $p \in J_k \cap J_{k+1}$ les solutions possibles de $Q_{k+1}(p)$ ne peuvent provenir que des solutions des $Q_k(l)$ avec $l \leq p$. Si la solution optimale provient de $Q_k(p)$, on a $f_k(p)$. Si elle provient de $Q_k(l)$, $l < p$ on a $C_p + f_k(l)$. On en déduit (8). (9) est immédiat.

De la proposition 2, on déduit l'algorithme de programmation dynamique suivant :

Algorithme 2

α — *initialisation*

Pour tout $j \in J_1$, *faire* $f(j) = C_j$.

$k = 1$.

β — *étape* k

Si $k = m$, *aller en* γ.

Calculer $h = \min_{j \in J_k - J_{k+1}} f(j)$

Pour tout $j \in J_{k+1} - J_k$, *faire* $f(j) = C_j + h$

Prendre les $j \in J_k \cap J_{k+1}$ *dans l'ordre croissant et faire*

$$h \leftarrow \min(h,\ f(j-1))$$
$$f(j) \leftarrow \min(f(j),\ C_j + h)$$

$k \leftarrow k + 1$
aller en β.

γ — *calculer* $z_{\min} = \min_{j \in J_m} f(j)$
($z_{\min}$ *est la valeur de solution optimale du PR*).

FIN.

Donnons quelques explications de l'étape k et en particulier du calcul de $f(j)$ pour $j \in J_k \cap J_{k+1}$.

La formule (8) donne en posant $h = \min_{\substack{l \in J_k \\ l < p}} f_k(l)$:

$$f_{k+1}(p) = \min [f_k(p), C_p + h] \tag{10}$$

$$f_{k+1}(p+1) = \min [f_k(p+1), C_{p+1} + \min (h, f_k(p))] \tag{11}$$

(10) entraîne :

— ou $f_{k+1}(p) = f_k(p) \leq C_p + h$, et on a donc :

$$\min (h, f_k(p)) = \min (h, f_{k+1}(p)) \tag{12}$$

— ou $f_{k+1}(p) = C_p + h < f_k(p)$, et comme $C_p \geq 0$ on a donc :

$$\min (h, f_k(p)) = h = \min (h, f_{k+1}(p)).$$

On a donc (12) dans tous les cas et donc (11) s'écrit :

$$f_{k+1}(p+1) = \min [f_k(p+1), C_{p+1} + \min (h, f_{k+1}(p))] \tag{13}$$

ce qui explique le calcul de $f(j)$ pour $j \in J_k \cap J_{k+1}$.

En plus des opérations définies pour l'algorithme 1 nous devons recalculer à chaque étape k, $f(j)$ pour $j \in J_k \cap J_{k+1}$. Cela revient à faire une addition et 2 comparaisons pour $J_k \cap J_{k+1}$ à l'étape k. On doit de plus faire une additon pour les éléments de $J_{k+1} - J_k$ à l'étape k. Pour l'étape k, on a donc une complexité en $0(|J_{k+1}|)$.

Appelons α *le taux de remplissage* de la matrice A, αnm correspondant donc au nombre des 1 de la matrice A.

La complexité de l'algorithme 2 est alors :

$$\sum_k 0(|J_k|) = 0(\alpha nm).$$

THÉORÈME 2 : *L'algorithme 2 est en* $0(\alpha nm)$.

4. UN ALGORITHME EN $0(n)$ POUR LE PROBLÈME DE COUPLAGE

Le problème de couplage (PC) s'écrit :

$$\left.\begin{array}{lll} \text{maximiser} & z = \sum_{j=1}^{n} C_j x_j & \\ \text{avec :} & \sum_{j=1}^{n} a_{ij} x_j \leq 1 & i \text{ de } 1 \text{ à } m \\ & x_j = 0 \text{ ou } 1 & j \text{ de } 1 \text{ à } n \end{array}\right\} \tag{14}$$

LEMME 3 : *Si $J_l \subset J_k$, alors dans (PC) on peut supprimer la contrainte l.*

Après suppression de telles lignes on peut donc mettre la matrice A sous forme canonique.

Pour tout $l \in J_k$ considérons les problèmes $\mathscr{R}_k(l)$ suivants :

$$\left.\begin{array}{lll} \text{maximiser} & z = \sum_{j=1}^{n} C_j x_j & \\ \text{avec :} & \sum_{j=1}^{n} a_{ij} x_j \leq 1 & i \text{ de } 1 \text{ à } k \\ & x_j = 0 \text{ ou } 1 & j \text{ de } 1 \text{ à } l-1 \\ & x_l = 1 & \\ & x_j = 0 & j \text{ de } l+1 \text{ à } n \end{array}\right\} \tag{15}$$

et le problème $\mathscr{R}_k(o)$ obtenu de la même façon que les autres mais avec $x_j = 0$ pour tout $j \in J_k$ et pour tous les indices suivants.

Appelons respectivement $f_k(l)$ et $f_k(o)$ les valeurs des solutions optimales de $\mathscr{R}_k(l)$ et $\mathscr{R}_k(o)$.

PROPOSITION 3 : *Pour $p \in J_{k+1}$, on a :*

$$f_{k+1}(o) = \max\left[f_k(o), \max_{l \in J_k - J_{k+1}} f_k(l)\right] \tag{16}$$

$$f_{k+1}(p) = f_k(p) \qquad \text{si } p \in J_k \tag{17}$$

$$f_{k+1}(p) = C_p + f_{k+1}(o) \qquad \text{si } p \notin J_k. \tag{18}$$

Démonstration : Immédiat par définition des $\mathscr{R}_k(l)$ et $\mathscr{R}_k(o)$. ■

De la proposition 3, on déduit l'algorithme de programmation dynamique suivant :

Algorithme 3

α — *initialisation*

Poser $f_0 = 0$

Pour tout $j \in J_1$, *faire* $f(j) = C_j$.
$k = 1$.

β — *étape* k

Si $k = m$, *aller en* γ.
Calculer $h = \max_{j \in J_k - J_{k+1}} f(j)$
$f_0 \leftarrow \max(f_0, h)$
Pour tout $j \in J_{k+1} - J_k$, *faire* $f(j) = C_j + f_0$
$k \leftarrow k + 1$
Aller en β.

γ — *Calculer* $z_{\max} = \max [f_0, \max_{j \in J_m} f(j)]$

($z_{\max}$ *est la valeur de la solution optimale du PC*).

FIN.

La complexité est la même que pour l'algorithme 1. On a donc :

THÉORÈME 3 : *L'algorithme 3 est en* $0(n)$.

Remarquons que l'algorithme 3, qui peut être considéré comme un algorithme de couplage dans un graphe biparti convexe particulier (cf. [9], p. 196), est plus efficace que celui présenté par Glover [8].

Considérons le problème $\max z = \sum_{j=1}^{n} C_j x_j$ avec $\sum_{j=1}^{n} x_j = 1$, $x_j = 0$ ou 1. Il est aisé de voir que la solution de ce problème prend exactement $n - 1$ comparaisons. On peut donc en déduire que la borne inférieure des problèmes 1 et 3 est en $0(n)$, ce qui montre l'optimalité de l'ordre des algorithmes 1 et 3.

Conjecture : Optimalité de l'ordre de l'algorithme 2.

REMERCIEMENTS

Je tiens à remercier le rapporteur de cet article pour ces pertinentes suggestions.

BIBLIOGRAPHIE

1. M. GONDRAN et M. MINOUX, *Graphes et algorithmes*. A paraître chez Eyrolles, 1979.
2. R. KARP, On the computational Complexity of Combinatorial problems. *Network*, 5, 1975, p. 45-68.
3. C. BERGE, *Graphes et hypergraphes*. Dunod, 1970.
4. A. J. HOFFMAN et J. B. KRUSKAL, Integral boundary points of convex polyhedra. *Ann. of Math. Studies*, 38, Princeton, p. 223.

5. J. AGARD, J. P. ARABEYRE et J. VAUTIER, Génération automatique de rotation d'équipages, *R.A.I.R.O.*, n° 6, 1967, p. 107-117.
6. M. MINOUX, *Hypergraphes d'intervalles et problèmes de flots.* Communication orale.
7. M. GONDRAN, Les problèmes de partition et de recouvrement : applications et algorithmes, *Bulletin de la Direction des Études et Recherches E.D.F.*, série C, n° 2, 1976, p. 59-68.
8. F. GLOVER, Maximum Matching in a Convex Bipartite Graph, *Naval Res. Logist. Quart.*, n° 14, 1967, p. 313-316.
9. E. LAWLER, *Combinatorial Optimization: Networds and matroids*, Holt, Rinchart and Winston, 1976.