

J.-J. FREY

A. YEHA-ALCOUHLABI

Comparaisons par paires : une interprétation et une généralisation de la méthode des scores

RAIRO. Recherche opérationnelle, tome 20, n° 3 (1986),
p. 213-227

http://www.numdam.org/item?id=RO_1986__20_3_213_0

© AFCET, 1986, tous droits réservés.

L'accès aux archives de la revue « RAIRO. Recherche opérationnelle » implique l'accord avec les conditions générales d'utilisation (<http://www.numdam.org/conditions>). Toute utilisation commerciale ou impression systématique est constitutive d'une infraction pénale. Toute copie ou impression de ce fichier doit contenir la présente mention de copyright.

NUMDAM

Article numérisé dans le cadre du programme
Numérisation de documents anciens mathématiques
<http://www.numdam.org/>

COMPARAISONS PAR PAIRES : UNE INTERPRÉTATION ET UNE GÉNÉRALISATION DE LA MÉTHODE DES SCORES (*)

par J.-J. FREY (1) et A. YEHIA-ALCOULABI (2)

Résumé. — *Cet article propose un critère pour classer des objets à partir des résultats d'une expérience de comparaisons par paires quelconque. L'application de ce critère nécessite la résolution d'un problème d'affectation quadratique à structure particulière, pour lequel nous avons défini, et mis en œuvre sur ordinateur, un algorithme de programmation dynamique. Puis nous établissons une formule qui, dans le cas particulier des tournois, prend une forme très simple et permet de montrer que la méthode que nous proposons est équivalente à la méthode des scores. Celle-ci se trouve ainsi interprétée et généralisée.*

Mots clés : Affectation quadratique, comparaisons par paires, méthode des scores, programmation dynamique, tournois.

Abstract. — *This paper proposes a criterion for ranking objects from results of any paired comparisons experiment. Applying this criterion implies solving a quadratic assignment problem with special structure, for which we designed and implemented a dynamic programming algorithm. Then we obtain a formula which, in special case of tournaments, becomes very simple and allows to prove that the method we propose is equivalent to score method. So the latter is interpreted and generalized.*

Keywords : Dynamic programming; paired comparisons; quadratic assignment; score method; tournaments.

1. POSITION DU PROBLÈME

Soit $X = \{x_1, x_2, \dots, x_n\}$ un ensemble d'objets.

Soit une expérience de comparaisons par paires, où x_i a été comparé m_{ij} fois à x_j ($j \neq i$), et préféré a_{ij} fois.

Les résultats de l'expérience sont résumés par la matrice $A(a_{ij})$ carrée d'ordre n , dite « matrice de préférence », dans laquelle :

$$a_{ij} + a_{ji} = m_{ij} \quad \forall i, j/i \neq j, \quad \text{et} \quad a_{ii} = 0 \quad \forall i.$$

(*) Reçu en juin 1985.

(1) Centre d'Informatique de Toulouse, U.E.R. M.I.G. de l'Université Paul-Sabatier, Toulouse.

(2) Laboratoire Modèles et Logiciels d'Analyse des Données, U.E.R. M.I.G. de l'Université Paul-Sabatier, Toulouse.

On peut associer à cette expérience le graphe orienté $G = (X, U)$ obtenu en traçant un arc de x_i vers x_j chaque fois que x_i est préféré à x_j dans une comparaison. Ce graphe est un tournoi si tous les m_{ij} sont égaux à 1. On peut parler de « tournoi généralisé » lorsque tous les m_{ij} sont égaux au même entier m positif.

Le problème est de définir un classement ou ordre des n objets « aussi compatible que possible » avec les résultats de l'expérience.

Une première méthode, dite « méthode des scores », consiste à classer les objets suivant leurs scores décroissants, en appelant score d'un objet x_i la quantité : $\sum_{j=1}^n a_{ij}$. Cette méthode, très facile à mettre en œuvre, n'a de sens que si tous les couples d'objets ont été comparés un nombre égal de fois. On ne peut donc l'appliquer que pour des tournois ou pour des tournois généralisés.

Une seconde méthode, due à Slater [9], consiste à déterminer les classements qui minimisent le nombre de résultats (ou réponses) incompatibles. Cette méthode a conduit à de nombreux algorithmes, dont certains sont valables dans le cas de tournois, et d'autres dans le cas général.

Le problème de classement que nous venons d'évoquer a été étudié par de nombreux auteurs; le lecteur pourra, en particulier, se reporter à [1], [3], [5] et [6].

2. LA MÉTHODE PROPOSÉE

Dans la méthode de Slater, on détermine une bijection « $p : X \rightarrow \{1, 2, \dots, n\}$ » qui minimise la quantité : $\sum_{\substack{(i,j) \\ p(x_j) < p(x_i)}} a_{ij}$, où la sommation porte sur l'ensemble des couples (i, j) tels que $p(x_j) < p(x_i)$.

Cette méthode minimise donc le nombre d'arcs qu'il faut inverser dans le graphe G associé à l'expérience, pour que celui-ci soit entièrement compatible avec le classement fourni par p . En d'autres termes, elle minimise le nombre de résultats de l'expérience contraires au classement.

Toutefois, si x_i a été préféré à x_j dans une comparaison, et si p place x_i après x_j , il est naturel de considérer que cette incompatibilité est d'autant plus grave que l'écart entre $p(x_i)$ et $p(x_j)$ est grand.

C'est pourquoi nous proposons de pondérer chaque arc (x_i, x_j) inversé, c.à.d. chaque résultat contraire au classement, par « $p(x_i) - p(x_j)$ ».

La fonction à minimiser s'écrit alors :

$$F(p) = \sum_{p(x_j) < p(x_i)} a_{ij}(p(x_i) - p(x_j)). \quad (1)$$

Nous allons, dans un premier temps, donner un algorithme qui permet de déterminer une bijection p minimisant la fonction F . Nous calculerons la fonction de complexité de cet algorithme, et préciserons comment il a été programmé pour réduire la place occupée en mémoire.

Puis nous établirons une formule donnant « $F(p') - F(p)$ » lorsque p' est obtenu à partir de p par le simple échange de deux éléments.

Enfin, nous utiliserons la formule précédente pour montrer que, dans le cas particulier des tournois ou des tournois généralisés, la méthode que nous proposons est équivalente à la méthode des scores. Ce résultat permet une interprétation théorique de la méthode des scores.

3. DESCRIPTION DE L'ALGORITHME

Le problème que nous nous proposons de résoudre s'apparente aux problèmes d'affectation quadratique, qui sont NP -complets. Si nous posons : $X = \{1, 2, \dots, n\}$, la fonction à minimiser est de la forme :

$$\sum_i \sum_j a_{ij} d_{p(i)p(j)},$$

où p est une permutation de l'ensemble des n premiers nombres entiers, et $d_{kl} = k - l$ si $k > l$, $= 0$ sinon.

Les deux matrices (a_{ij}) et (d_{kl}) qui définissent cette fonction sont donc asymétriques. En outre la seconde est triangulaire.

Nous allons donner un algorithme de résolution de type « programmation dynamique ».

Nous noterons désormais :

$$F(p) = \sum_{p(j) < p(i)} a_{ij}(p(i) - p(j)).$$

Soit S l'ensemble des objets placés par p aux $|S|$ premières places. On peut écrire :

$$F(p) = \sum_{\substack{p(j) < p(i) \\ i, j \in S}} a_{ij}(p(i) - p(j)) + \sum_{\substack{p(j) < p(i) \\ i, j \in X-S}} a_{ij}(p(i) - p(j)) \\ + \sum_{\substack{i \in X-S \\ j \in S}} a_{ij}(p(i) - |S| - 1) + \sum_{\substack{i \in X-S \\ j \in S}} a_{ij}(|S| + 1 - p(j)).$$

Posons :

$$f_S = \sum_{\substack{p(j) < p(i) \\ i, j \in S}} a_{ij}(p(i) - p(j)) + \sum_{\substack{i \in X-S \\ j \in S}} a_{ij}(|S| + 1 - p(j)).$$

Cette quantité représente la partie $F(p)$ qui peut être déterminée dès que l'on connaît les éléments de X qui occupent les $|S|$ premières places. Par ailleurs : $f_X = F(p)$.

Nous nous trouvons donc dans les conditions d'application du principe d'optimalité de Bellman.

Si k est l'élément placé à la $|S|^\text{e}$ place par p , on a :

$$f_{S-(k)} = \sum_{\substack{p(j) < p(i) \\ i, j \in S-(k)}} a_{ij}(p(i) - p(j)) + \sum_{\substack{i \in X-S+(k) \\ j \in S-(k)}} a_{ij}(|S| - p(j)),$$

et l'on peut écrire :

$$f_S = \sum_{\substack{p(j) < p(i) \\ i, j \in S-(k)}} a_{ij}(p(i) - p(j)) + \sum_{\substack{i=k \\ j \in S-(k)}} a_{ij}(p(i) - p(j)) \\ + \sum_{\substack{i \in X-S \\ j \in S-(k)}} a_{ij}(|S| + 1 - p(j)) + \sum_{\substack{i \in X-S \\ j=k}} a_{ij}(|S| + 1 - p(j)),$$

avec $p(k) = |S|$.

D'où :

$$f_S - f_{S-(k)} = - \sum_{\substack{i \in X-S \\ j \in S-(k)}} a_{ij}(|S| - p(j)) + \sum_{\substack{i \in X-S \\ j \in S-(k)}} a_{ij}(|S| + 1 - p(j)) \\ + \sum_{\substack{i \in X-S \\ j=k}} a_{ij} = \sum_{\substack{i \in X-S \\ j \in S-(k)}} a_{ij} + \sum_{\substack{i \in X-S \\ j=k}} a_{ij} = \sum_{\substack{i \in X-S \\ j \in S}} a_{ij}.$$

Finalement, on peut calculer $\min_p F(p)$ par la formule de récurrence :

$$f_S = \min_{k \in S} \{ f_{S-(k)} + \sum_{\substack{i \in X-S \\ j \in S}} a_{ij} \} = \min_{k \in S} \{ f_{S-(k)} \} + \sum_{\substack{i \in X-S \\ j \in S}} a_{ij}, \tag{2}$$

en partant soit de $f_\emptyset = 0$, soit de :

$$f_{(h)} = \sum_{\substack{i \in X-(h) \\ j=h}} a_{ij}(|\{h\}| + 1 - p(j)) = \sum_{i \in X-(h)} a_{ih} = \sum_{i \in X} a_{ih}$$

somme des termes de la h^e colonne de $A(\forall h \in X)$.

Le programme devra être tel que, en fin d'algorithme, non seulement on connaisse $\min F$, mais encore on puisse retrouver la bijection p (ou l'une d'entre elles s'il en existe plusieurs) qui minimise F . Nous y reviendrons dans la section 6.

4. FONCTION DE COMPLEXITÉ

Soit $|S| = q$. La détermination de $\min_k \{ f_{S-(k)} \}$ nécessite $(q - 1)$ comparaisons; le calcul de $\left(\sum_{\substack{j \in S \\ i \in X-S}} a_{ij} \right)$ nécessite $(q(n - q) - 1)$ additions; d'où, pour le

calcul de f_S , un nombre d'opérations élémentaires égal à :

$$(q - 1) + (q(n - q) - 1) + 1 = -q^2 + q(n + 1) - 1.$$

Pour calculer f_X , il faut donc un nombre total d'opérations élémentaires égal à :

$$g(n) = \sum_{q=1}^n C_n^q (-q^2 + q(n + 1) - 1) = \sum_{q=0}^n C_n^q (-q^2 + q(n + 1) - 1) + 1.$$

D'où :

$$\begin{aligned} g(n) - 1 &= - \sum_{q=0}^n C_n^q (q^2 - q(n + 1) + 1) \\ &= - \sum_{q=0}^n q^2 C_n^q + (n + 1) \sum_{q=0}^n q C_n^q - \sum_{q=0}^n C_n^q \\ &= -n(n + 1)2^{n-2} + (n + 1)n2^{n-1} - 2^n = 2^{n-2}(n^2 + n - 4). \end{aligned}$$

Finalement : $g(n) = 2^{n-2}(n^2 + n - 4) + 1$, de l'ordre de : $n^2 2^{n-2}$.

REMARQUE : On peut également établir la formule de récurrence suivante :

$$f_S = \min_{k \in S} \left\{ f_{S-(k)} + \sum_{i \in X-S+(k)} \sum_{j \in S-(k)} a_{ij} \right\}, \quad \text{avec} \quad f_S = 0 \quad \text{si} \quad |S| = 1.$$

Mais alors, la fonction de complexité a pour ordre :

$$n^3 2^{n-3} = \frac{n}{2} g(n).$$

5. EXEMPLE

$$\text{Soit } (a_{ij}) = \begin{pmatrix} 0 & 2 & 1 & 3 \\ 1 & 0 & 3 & 2 \\ 4 & 2 & 0 & 1 \\ 1 & 3 & 3 & 0 \end{pmatrix}.$$

Phase I :

$$f_{\{1\}} = 6 \quad f_{\{2\}} = 7, \quad f_{\{3\}} = 7, \quad f_{\{4\}} = 6.$$

Phase II :

$$f_{\{1,2\}} = \min \{ f_{\{1\}}, f_{\{2\}} \} + a_{31} + a_{32} + a_{41} + a_{42} = 6 + 10 = 16,$$

et $OR(1, 2) = (1, 2)$; de même $f_{\{1,3\}} = 14$ et $OR(1, 3) = (1, 3)$, $f_{\{1,4\}} = 14$ et $OR(1, 4) = (1, 4)$ ou $(4, 1)$, $f_{\{2,3\}} = 16$ et $OR(2, 3) = (2, 3)$ ou $(3, 2)$, $f_{\{2,4\}} = 14$ et $OR(2, 4) = (4, 2)$, $f_{\{3,4\}} = 15$ et $OR(3, 4) = (4, 3)$.

Phase III :

$$f_{\{1,2,3\}} = \min \{ f_{\{1,2\}}, f_{\{1,3\}}, f_{\{2,3\}} \} + a_{41} + a_{42} + a_{43} = 21$$

et $OR(1, 2, 3) = OR((1, 3), 2) = (1, 3, 2)$; de même $f_{\{1,2,4\}} = 21$ et $OR(1, 2, 4) = OR((1, 4), 2)$ ou $OR(2, 4), 1) = (1, 4, 2)$ ou $(4, 1, 2)$ ou $(4, 2, 1)$, $f_{\{1,3,4\}} = 20$ et $OR(1, 3, 4) = OR((1, 3), 4)$ ou $OR((1, 4), 3) = (1, 3, 4)$ ou $(1, 4, 3)$ ou $(4, 1, 3)$, $f_{\{2,3,4\}} = 20$ et $OR(2, 3, 4) = OR((2, 4), 3) = (4, 2, 3)$.

Phase IV :

$$f_{\{1,2,3,4\}} = \min \{ f_{\{1,2,3\}}, f_{\{1,2,4\}}, f_{\{1,3,4\}}, f_{\{2,3,4\}} \} + \sum_{\substack{i \in \emptyset \\ j=1,2,3,4}} a_{ij} = 20$$

et $OR(1, 2, 3, 4) = OR((1, 3, 4), 2)$ ou $OR((2, 3, 4), 1) = (1, 3, 4, 2)$ ou $(1, 4, 3, 2)$ ou $(4, 1, 3, 2)$ ou $(4, 2, 3, 1)$.

Il existe donc, dans cet exemple, quatre classements équivalents.

6. MISE EN ŒUVRE SUR ORDINATEUR

Pour représenter un sous-ensemble S de X , nous utilisons la méthode suivante : à chaque élément de X , on associe un chiffre binaire, égal à 1 si l'élément appartient à S , et à zéro sinon. Par exemple, pour l'application numérique précédente, $\{1, 2\}$ est représenté par 0011, et $\{1, 2, 4\}$ par 1011. Chaque sous-ensemble S de X est ainsi associé à un nombre compris entre 0 (pour $S = \emptyset$ et n^{n-1} (pour $S = X$). Soit J ce nombre, et $BIN(J)$ son codage binaire.

Pour l'application de la formule de récurrence donnant f_S , on traite les sous-ensembles S dans l'ordre correspondant aux nombres qui leur sont associés. Alors, $\forall S' \subset S$, $f_{S'}$, est calculé avant f_S .

Soit donc à calculer pour S (associé à J) :

$$f_S = \min_{k \in S} \{ f_{S - \{k\}} \} + \sum_{i \in X - S}^{j \in S} a_{ij}.$$

Notons $f_S = f(J)$.

Dans $BIN(J)$, on remplace l'un des chiffres 1 par un 0. Soit J' le nombre obtenu, et S' le sous-ensemble associé. On a $J' < J$, et $S' \subset S$. Si la position binaire utilisée ci-dessus correspond à l'élément k , on a : $S' = S - \{k\}$, et l'on peut écrire : $f_{S'} = f_{S - \{k\}} = f(J')$. Cette quantité a déjà été calculée, et mémorisée dans la ligne correspondant à J' . Il suffit donc de remplacer, dans $BIN(J)$, chacun des 1 par 0 (mais un seul à la fois), pour obtenir tous les $S - \{k\}$, d'où toutes les valeurs $f_{S - \{k\}}$; on retient la plus petite et l'on y ajoute : $\sum_{i \in X - S}^{j \in S} a_{ij}$; la valeur f_S ainsi obtenue est mémorisée dans la ligne du tableau correspondant à J .

Nous nous sommes inspirés, pour ce qui précède, d'une méthode due à Baker [2], dont on trouvera également une application dans [4].

En fin d'algorithme, on obtient f_x , égal à $\min_p F(p)$. Mais nous voulons aussi connaître la bijection p qui a minimisé F ; ou l'une d'entre elles s'il en existe plusieurs.

A cette fin, lorsque nous venons de calculer f_s , nous mémorisons, outre cette quantité, la valeur (ou l'une d'entre elles s'il y en a plusieurs) k_0 de k pour laquelle $f_{S - \{k\}}$ est minimale, et le nombre J'_0 associé au sous-ensemble $S - \{k_0\}$. Ainsi, en fin d'algorithme, il suffira de remonter la chaîne ainsi définie pour reconstituer l'une des bijections p qui minimise F . Soit p_0 cette bijection.

Nous illustrerons ce qui précède en définissant le tableau correspondant à l'exemple présenté dans la section 5. Lorsque k_0 n'était pas unique, nous avons retenu le plus grand.

J	BIN(J)	S	S	f_s	k_0	J'_0
0	0000	\emptyset	0	0	0	0
1	0001	{1}	1	6	1	0
2	0010	{2}	1	7	2	0
3	0011	{1, 2}	2	16	2	1
4	0100	{3}	1	7	3	0
5	0101	{1, 3}	2	14	3	1
6	0110	{2, 3}	2	16	3	2
7	0111	{1, 2, 3}	3	21	2	5
8	1000	{4}	1	6	4	0
9	1001	{1, 4}	2	14	4	1
10	1010	{2, 4}	2	14	2	8
11	1011	{1, 2, 4}	3	21	2	9
12	1100	{3, 4}	2	15	3	8
13	1101	{1, 3, 4}	3	20	4	5
14	1110	{2, 3, 4}	3	20	3	10
15	1111	{1, 2, 3, 4}	4	20	2	13

Nous voyons sur ce tableau que $f_{\{1,2,3,4\}} = f_x = \min F = 20$. Par ailleurs, $p_0 = (1, 3, 4, 2)$. En effet, la ligne 15 nous indique que dans p_0 , le dernier élément est 2 et qu'on trouvera l'élément précédent à la ligne 13; c'est 4 et l'élément précédent se trouve à la ligne 5; c'est 3, et donc le premier élément de p_0 est 1 (qu'on peut trouver à la ligne 1).

7. RÉSULTATS OBTENUS

La mémorisation de f_s , k_0 et J'_0 nécessite un vecteur de taille 3×2^n .

La méthode utilisée pour déterminer p_0 — reconstitution à partir d'une chaîne indicée — a permis de réduire sensiblement l'occupation mémoire, par rapport à

la méthode plus immédiate consistant à mémoriser, pour chaque sous-ensemble S , l'ordre optimal des éléments (taille du vecteur correspondant : $(n + 1)2^n$).

Par ailleurs, l'obtention d'une formule de récurrence dans laquelle l'opération « min » ne porte que sur les $f_{S-\{k\}}$ — le terme $\sum_{\substack{j \in S \\ i \in X-S}} a_{ij}$ étant indépendant de k — a permis de réduire sensiblement le temps de calcul (cf. remarque, section 4).

Nous avons programmé notre algorithme en FORTRAN 77 (NEW FORTRAN), sur DPS 8 sous MULTICS.

Avec une mémoire disponible de 2^{20} octets, on peut résoudre des problèmes de taille $n \leq 18$.

Nous donnons en annexe trois exemples de problèmes qui ont été traités par ce programme.

Les temps d'exécution ont été les suivants : 3 secondes pour $n = 10$, 16 secondes pour $n = 12$, et 183 secondes pour $n = 15$.

L'utilisation de la fonction de complexité étudiée à la section 4 permet d'obtenir un ordre de grandeur pour le temps d'exécution nécessaire à la résolution de problèmes de taille supérieure : 6 à 7 minutes pour $n = 16$, une quinzaine de minutes pour $n = 17$, et environ une demi-heure pour $n = 18$.

8. VARIANTE DANS L'IMPLEMENTATION(*)

Dans la formule de récurrence, le terme le plus long à calculer est $\sum_{\substack{j \in S \\ i \in X-S}} a_{ij}$. Au lieu de le calculer entièrement à chaque itération, on peut penser à mettre à jour la valeur obtenue pour un ensemble « $S - \{k\}$ » comportant un élément de moins.

$$\text{Soit : } f_S = \min_{k \in S} \{ f_{S-\{k\}} \} + \sum_{\substack{i \in X-S \\ j \in S}} a_{ij} = f_{S-\{k_0\}} + \sum_{\substack{i \in X-S \\ j \in S}} a_{ij}.$$

On peut écrire :

$$f_S = f_{S-\{k_0\}} + \sum_{\substack{i \in X-S+(k_0) \\ j \in S-\{k_0\}}} a_{ij} + \sum_{i \in X-S} a_{ik_0} - \sum_{j \in S-\{k_0\}} a_{k_0j}.$$

L'application de cette formule nécessite l'adjonction d'une colonne dans le tableau décrit à la section 6 : pour chaque valeur de J , donc pour chaque S , on mémorisera le terme « $\sum_{\substack{i \in X-S \\ j \in S}} a_{ij}$ » correspondant.

(*) Cette variante a été définie à partir d'une suggestion du rapporteur; nous l'en remercions.

Calculons la fonction de complexité $v(n)$ correspondant à cette variante. Comme $f_{S-(k_0)}$ est obtenue à partir de $(q - 1)$ comparaisons, le nombre d'opérations élémentaires nécessaires au calcul de f_S est :

$$(q - 1) + 1 + (n - q) + (q - 1) = n + q - 1.$$

D'où :

$$\begin{aligned} v(n) &= \sum_{q=1}^n C_n^q(n + q - 1) = \sum_{q=0}^n C_n^q(n + q - 1) - n + 1 \\ &= (n - 1) \sum_{q=0}^n C_n^q + \sum_{q=0}^n q C_n^q - n + 1 = (n - 1)2^n + n2^{n-1} - n + 1 \\ &= 2^{n-1}(3n - 2) - n + 1 \sim 3n2^{n-1}. \end{aligned}$$

Si l'on compare à la fonction de complexité $g(n)$ obtenue à la section 4, et en notant que :

$$\frac{v(n)}{g(n)} = \frac{2^{n-1}(3n - 2) - n + 1}{2^{n-2}(n^2 + n - 4) + 1} \sim \frac{3n2^{n-1}}{n^22^{n-2}} = \frac{6}{n},$$

on constate que la variante que nous venons de décrire est plus performante à partir de $n = 6$ (pour $n = 5$, $v(n)$ et $g(n)$ sont presque égales).

Pour les trois problèmes décrits en annexe, l'application de cette variante a entraîné les temps d'exécution suivants :

2,5 secondes pour $n = 10$, 13 secondes pour $n = 12$ et 142 secondes pour $n = 15$.

En comparant aux résultats obtenus à la section 7, on vérifie que l'intérêt de cette variante croît avec la valeur de n , du point de vue du temps d'exécution.

Toutefois, la dimension du vecteur à mémoriser est augmentée d'un tiers, puisqu'il faut rajouter « $\sum_{i \in X-S}^{j \in S} a_{ij}$ » aux trois résultats conservés (f_S, k_0, J'_0).

En fonction de l'ordre des problèmes à résoudre, et de la mémoire disponible, on pourra dans certains cas être amené à devoir arbitrer entre une compression sur le temps d'exécution et une diminution de l'espace-mémoire nécessaire.

9. ÉCHANGE DE DEUX ÉLÉMENTS DANS p

Soit $(h, k) \in X^2$, avec $p(h) < p(k)$.

Soit p_h^k définie par :

$$\begin{aligned} p_h^k(i) &= p(i) & \forall i/i \neq h, \quad i \neq k, \\ p_h^k(h) &= p(k), & p_h^k(k) &= p(h). \end{aligned} \tag{3}$$

Nous allons montrer :

PROPRIÉTÉ 1 : Dans une expérience de comparaisons par paires quelconques, on a la relation :

$$F(p_h^k) - F(p) = \left[d^+(h) - d^+(k) + \sum_{p(j) > p(k)} (m_{kj} - m_{hj}) \right] [p(k) - p(h)] + \sum_{p(h) < p(j) < p(k)} (m_{kj} - m_{hj}) [p(j) - p(h)],$$

où $m_{ij} = a_{ij} + a_{ji}$, et $d^+(i) = \sum_{j=1}^n a_{ij}$. (4)

DÉMONSTRATION :

On peut décomposer $F(p)$ de la façon suivante :

$$F(p) = \sum_{\substack{p(j) < p(i) \\ i \neq h, k; j \neq h, k}} a_{ij}(p(i) - p(j)) + \sum_{p(j) < p(h)} a_{hj}(p(h) - p(j)) + \sum_{p(j) < p(h)} a_{kj}(p(k) - p(j)) + \sum_{p(h) < p(j) < p(k)} a_{jh}(p(j) - p(h)) + \sum_{p(h) < p(j) < p(k)} a_{kj}(p(k) - p(j)) + \sum_{p(k) < p(j)} a_{jh}(p(j) - p(h)) + \sum_{p(k) < p(j)} a_{jk}(p(j) - p(k)) + a_{kh}(p(k) - p(h)).$$

En notant que $p_h^k(k) < p_h^k(h)$, on décompose $F(p_h^k)$ de façon analogue, et en utilisant (3) on obtient :

$$F(p_h^k) = \sum_{\substack{p(j) < p(i) \\ i \neq h, k; j \neq h, k}} a_{ij}(p(i) - p(j)) + \sum_{p(j) < p(h)} a_{kj}(p(h) - p(j)) + \sum_{p(j) < p(h)} a_{hj}(p(k) - p(j)) + \sum_{p(h) < p(j) < p(k)} a_{jk}(p(j) - p(h)) + \sum_{p(h) < p(j) < p(k)} a_{hj}(p(k) - p(j)) + \sum_{p(k) < p(j)} a_{jk}(p(j) - p(h)) + \sum_{p(k) < p(j)} a_{jh}(p(j) - p(k)) + a_{hk}(p(k) - p(h)).$$

D'où :

$$F(p_h^k) - F(p) = \sum_{p(j) < p(h)} a_{kj}(p(h) - p(k)) + \sum_{p(j) < p(h)} a_{hj}(p(k) - p(h)) + \sum_{p(h) < p(j) < p(k)} (a_{jk} - a_{jh})(p(j) - p(h)) + \sum_{p(h) < p(j) < p(k)} (a_{hj} - a_{kj})(p(k) - p(h) + p(h) - p(j)) + \sum_{p(k) < p(j)} a_{jk}(p(k) - p(h)) + \sum_{p(k) < p(j)} a_{jh}(p(h) - p(k)) + (a_{hk} - a_{kh})(p(k) - p(h)) = [p(k) - p(h)] \left[- \sum_{p(j) < p(h)} a_{kj} + \sum_{p(j) < p(h)} a_{hj} + \sum_{p(h) < p(j) < p(k)} a_{hj} \right]$$

$$\begin{aligned}
 & - \sum_{p(h) < p(j) < p(k)} a_{kj} + \sum_{p(k) < p(j)} (a_{jk} + a_{kj} - a_{kj}) \\
 & - \left. \sum_{p(k) < p(j)} (a_{jh} + a_{jh} - a_{hj}) + a_{hk} - a_{kh} \right] \\
 & + \sum_{p(h) < p(j) < p(k)} (a_{kj} + a_{jk} - a_{hj} - a_{jh}) [p(j) - p(h)].
 \end{aligned}$$

On obtient alors (4) en remarquant que :

$$\sum_{p(j) < p(h)} a_{hj} + \sum_{p(h) < p(j) < p(k)} a_{hj} + a_{hk} + \sum_{p(k) < p(j)} a_{hj} = \sum_{j=1}^n a_{hj} = d^+(h);$$

de même pour $d^+(k)$; et en notant :

$$a_{hj} + a_{jh} = m_{hj}, \quad a_{kj} + a_{jk} = m_{kj}.$$

10. CAS PARTICULIER DES TOURNOIS

Si l'expérience de comparaisons par paires considérée consiste en un tournoi ou en un tournoi généralisé, tous les m_{ij} sont égaux entre eux. La formule (4) se simplifie et l'on peut énoncer :

PROPRIÉTÉ 2 : Dans le cas de tournois, on a la relation :

$$F(p_h^k) - F(p) = (p(k) - p(h))(d^+(h) - d^+(k)). \tag{5}$$

Ici, $d^+(h)$ et $d^+(k)$ représentent les scores respectifs de h et k .

* On peut alors démontrer :

THÉORÈME : Dans le cas de tournois, les classements qui minimisent la fonction F sont les mêmes que ceux fournis par la méthode des scores.

La méthode des scores apparaît ainsi comme un cas particulier — correspondant aux tournois — de la méthode que nous proposons ici, et qui s'applique dans le cas général d'une expérience de comparaisons par paires quelconque.

Démontrons le théorème précédent :

• Soit p un classement de X tel que la fonction F soit minimale. Supposons que le vecteur des scores associé à p n'est pas décroissant, c.à.d. qu'il existe h et k tels que :

$$d^+(k) > d^+(h) \quad \text{et} \quad p(k) > p(h).$$

La formule (5) montre que : $F(p_h^k) - F(p) < 0$.

Donc $F(p)$ n'est pas maximal. Contradiction.

Le vecteur des scores associé à p est donc décroissant.

• Inversement, soit p un classement dont le vecteur des scores associé est décroissant.

Soit p' un classement minimisant F . D'après ce qui précède, le vecteur des scores associé à p' est décroissant.

Si $p = p'$, $F(p) = F(p') = \min F$.

Sinon, comme p et p' sont associés au même vecteur des scores (décroissant), on peut définir une suite de classements équivalents du point de vue des scores, $\{p_{b_1}, p_{b_2}, \dots, p_{b_t}\}$, où $p_{b_1} = p'$ et $p_{b_t} = p$, et où $p_{b_{i+1}}$ s'obtient à partir de p_{b_i} par l'échange de deux éléments h_i et k_i tels que : $d^+(h_i) = d^+(k_i)$.

D'après la formule (7) :

$$F(p_{b_{i+1}}) = F(p_{b_i}) \quad \forall i \in \{1, 2, \dots, t-1\}, \quad \text{d'où} \quad F(p) = F(p') = \min F.$$

11. REMARQUE

Il est à noter que M. Kano et A. Sakamoto, dans des travaux très récents portant sur le classement des sommets d'un graphe pondéré [7] — travaux réalisés en même temps que les nôtres et indépendamment — ont étudié la minimisation d'une fonction qui, avec des notations analogues aux nôtres, pourrait s'écrire :

$$G(p) = \sum_{p(j) < p(i)} b_{ij}(p(i) - p(j)), \quad \text{avec} \quad b_{ij} + b_{ji} = 1. \quad (8)$$

Si l'on considère une expérience de comparaisons par paires, on peut poser : $\frac{a_{ij}}{m_{ij}} = b_{ij}$. Cette quantité représente alors le pourcentage de comparaisons entre i et j qui ont donné un résultat favorable à i .

Il ne s'agit donc pas du même critère de classement que le nôtre. En particulier, dans la mesure où l'on ne retient que des pourcentages, on ne tient pas compte du nombre de comparaisons effectuées entre i et j (m_{ij}).

Par contre, dans le cas particulier des tournois ou des tournois généralisés, où tous les m_{ij} sont égaux à m , on a $F(p) = mG(p)$, et les deux critères deviennent équivalents. On obtient alors pour la méthode des scores une interprétation analogue à celle que nous avons présentée à la section 10.

Signalons enfin que, dans les travaux de Kano et Sakamoto, et de quelques autres auteurs [p.e. 8], l'expression « generalized tournament matrix » désigne une matrice (b_{ij}) telle que $b_{ij} + b_{ji} = 1$, c.à.d. en fait une matrice de pourcentages, ou de probabilités.

12. CONCLUSION

Pour une expérience de comparaisons par paires quelconque, cet article propose un critère de classement qui a priori paraît « raisonnable » :

$$\text{minimiser } \sum_{p(j) < p(i)} a_{ij}(p(i) - p(j)),$$

c.à.d. minimiser le nombre de résultats contraires au classement, pondérés par l'écart de classement.

Nous avons défini un algorithme de type « programmation dynamique », permettant de déterminer les classements optimaux d'après ce critère. Cet algorithme a été adapté et complété de façon à pouvoir être traduit par un programme n'entraînant qu'un temps de calcul et une occupation mémoire modérés.

Puis nous avons étudié la variation de la fonction $F(p)$ lorsqu'on échange deux éléments de p . La formule obtenue, valable dans le cas général, prend une forme très simple dans le cas particulier des tournois ou des tournois généralisés. En l'utilisant, on démontre que, dans ce cas particulier, la méthode des scores et la méthode que nous proposons sont équivalentes. Cette dernière peut alors apparaître comme une généralisation de la méthode des scores.

ANNEXE

Données et résultats de problèmes résolus par l'algorithme présenté dans les sections 3 à 6 :

	1	2	3	4	5	6	7	8	9	10
1	0	1	1	1	2	0	0	1	0	1
2	0	0	1	0	1	2	2	1	1	1
3	0	0	0	1	0	1	0	1	1	1
4	1	1	1	0	0	0	1	1	1	1
5	1	1	1	1	0	1	1	0	0	0
6	1	1	1	2	0	0	0	1	1	1
7	1	1	0	0	0	0	0	1	0	1
8	0	1	1	1	0	0	0	0	0	1
9	1	1	1	1	1	1	1	0	0	0
10	1	1	0	1	1	1	1	0	0	0

matrice (a_{ij})
avec $n = 10$

$\min F(p) = 79$

$p_0 = (9, 5, 1, 6, 2, 10, 4, 3, 8, 7)$

	1	2	3	4	5	6	7	8	9	10	11	12
1	0	1	1	1	2	0	0	1	0	1	1	0
2	0	0	1	0	1	2	2	1	1	1	1	1
3	0	0	0	1	0	1	0	1	1	1	1	1
4	1	1	1	0	0	0	1	1	1	1	1	0
5	1	1	1	1	0	1	1	0	0	0	1	1
6	1	1	1	2	0	0	0	1	1	1	1	0
7	1	1	0	0	0	0	0	1	0	1	1	1
8	0	1	1	1	0	0	0	0	0	1	1	0
9	1	1	1	1	1	1	1	0	0	0	3	1
10	1	1	0	1	1	1	1	0	0	0	1	0
11	1	1	1	1	1	0	0	0	0	1	0	0
12	1	0	0	0	0	1	1	1	1	0	0	0

matrice (a_{ij})
avec $n = 12$

$\min F(p) = 131$

$p_0 = (12, 9, 7, 2, 6, 5,$
 $1, 10, 4, 11, 3, 8)$

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	0	1	0	1	1	1	1	1	1	0	1	1	1	1	1
2	0	0	1	0	0	0	0	2	1	2	0	0	1	1	1
3	0	0	0	0	1	0	1	2	3	1	1	0	1	1	0
4	1	1	1	0	1	1	0	0	0	0	0	1	0	1	0
5	1	1	1	2	0	3	1	1	2	1	1	0	0	1	1
6	1	1	1	0	0	0	0	0	1	2	1	0	0	1	1
7	1	1	1	1	1	0	0	0	1	2	2	3	0	1	1
8	1	1	0	0	0	0	0	0	0	0	1	1	2	1	0
9	0	0	0	1	1	1	0	0	0	0	0	0	1	1	1
10	0	1	2	2	2	2	2	2	0	0	0	0	2	1	1
11	1	1	1	1	4	2	1	0	0	0	0	2	1	0	1
12	1	1	1	1	1	2	1	2	1	0	0	0	0	1	1
13	1	1	0	1	1	1	1	1	1	0	1	0	0	1	0
14	1	1	2	1	1	1	1	1	1	2	0	0	0	0	0
15	1	2	1	0	1	0	1	0	1	0	1	1	1	2	0

matrice (a_{ij}) avec $n = 15$; $\min F(p) = 283$

$p_0 = (12, 7, 11, 15, 1, 5, 10, 6, 4, 14, 3, 13, 2, 8, 9)$.

BIBLIOGRAPHIE

1. A. ASTIE-VIDAL, *Comparaisons par paires : recherche d'un classement ou d'un ensemble de classements des objets comparés*, rev. stat. appl., Vol. 21, n° 3, pp. 51-63, 1973.
2. K. R. BAKER, *Introduction to sequencing and scheduling*, New York, Wiley, 1974.
3. J. P. BARTHELEMY et B. MONJARDET, *The median procedure in cluster analysis and social choice theory*, Mathematical Social Sciences 1, pp. 235-267, 1981, N. H. Pub. Company.
4. M. BEGHIN-PICAVET, P. HANSEN, *Deux problèmes d'affectation non linéaires*. R.A.I.R.O. Recherche Opérationnelle n° 3, pp. 263-276, août 1982.
5. R. R. DAVISON et P. H. FARQUHAR, *A bibliography on the method of paired comparisons*, Biometrics n° 32, pp. 241-252, juin 1976.
6. M. GRÖTSCHEL, M. JÜNGER et G. REINELT, *A cutting plane algorithm for the linear ordering problem*, Operations Research, Vol. 32, n° 6, pp. 1195-1220, 1984.
7. M. KANO et A. SAKAMOTO, *Ranking the vertices of a weighted digraph using the length of forward arcs*, Networks Vol. 13 n° 1, pp. 143-151, 1983.
8. J. W. MOON et N. J. PULLMAN, *On generalized tournament matrices*, S.I.A.M. Review Vol. 12 n° 3, pp. 384-399, juillet 1970.
9. P. SLATER, *Inconsistencies in schedule of paired comparisons*, Biometrika Vol. 48, pp. 303-312, déc. 1961.
10. A. YEHA-ALCOUTLABI, *Sur le classement des sommets d'un graphe de comparaisons par paires*, Thèse de 3^e Cycle, U.P.S., Toulouse, mars 1985.