Un algorithme de partition d'un produit direct d'ordres totaux en un nombre minimum de chaînes

Emmanuel Pichon, Philippe Lenca, Fabrice Guillet, Jian-Wei Wang

MATHÉMATIQUES ET SCIENCES HUMAINES

EMMANUEL PICHON
PHILIPPE LENCA
FABRICE GUILLET
JIAN WEI WANG

Un algorithme de partition d'un produit direct d'ordres totaux en un nombre minimum de chaînes

Mathématiques et sciences humaines, tome 125 (1994), p. 5-15.
<http://www.numdam.org/item?id=MSH_1994__125__5_0>

Math. Inf. Sci. hum. (32e année, n° 125, 1994, p.5-15)

UN ALGORITHME DE PARTITION D'UN PRODUIT DIRECT D'ORDRES TOTAUX EN UN NOMBRE MINIMUM DE CHAINES

Emmanuel PICHON[1,2,3], Philippe LENCA[1,3], Fabrice GUILLET[1,3] et Jian Wei WANG[3,4]

RESUMÉ — *Cette étude s'inscrit dans un prolongement algorithmique d'un travail de Bruno Leclerc, publié dans cette revue, qui discute de la taille maximum d'une antichaîne dans un produit direct P d'ordres totaux. On y présente un algorithme de partitionnement de P en un nombre minimum de chaînes. Enfin, on décrit brièvement une application à l'extraction de connaissance.*

SUMMARY — An algorithm for partitioning a direct product of linear orders into a minimum number of chains.

This paper concerns an algorithmic extension of a work by Bruno Leclerc published in this Journal. He discusses the maximum cardinality of an antichain in the direct product P of linear orders. We present an algorithm for partitioning P into a minimum number of chains. We also briefly describe an application in the field of knowledge extraction.

1. INTRODUCTION

Comment deviner, dans un ensemble ordonné P, une antichaîne A en posant un nombre minimum de questions du type : x est-il au-dessus de A? (réponse Oui ou Non). Une stratégie possible (et pourvue de quelques vertus récursives) consiste à partitionner P en un nombre minimum de chaînes sur lesquelles on applique une procédure de recherche dichotomique. La complexité de cet algorithme est majorée par $\alpha \, \mathrm{Log}_2 \, h$ où α est la largeur de P (taille minimum d'une partition en chaînes) et h la hauteur de P (taille maximum d'une chaîne). Cette borne ne tient compte ni de la diversité des chaînes dans une partition optimale de P, ni des éliminations produites, sur les chaînes suivantes, par l'examen de la chaîne courante. Elle est cependant atteinte lorsque P est la somme disjointe d'une famille de chaînes de même taille (l'algorithme fournit dans ce cas le nombre effectivement maximal de questions à poser ; ainsi $\alpha \, \mathrm{Log}_2 \, h$ est la complexité théorique du problème).

La détermination de la taille minimum d'une partition en chaînes d'un ensemble ordonné P, ou de façon équivalente, de la taille maximum d'une antichaîne de P est, en général, NP-difficile. Elle peut devenir facile lorsque P vérifie la propriété de Sperner (ses éléments peuvent être rangés en niveaux et un de ces niveaux correspond à une antichaîne de taille maximum), puisqu'il suffit d'examiner les différents niveaux de P. Elle devient triviale lorsque les propriétés de P permettent de désigner, sans recherche, le niveau correspondant à une

[1] Laboratoire d'Intelligence Artificielle et Systèmes Cognitifs - Télécom Bretagne
[2] Centre de Recherche public - Centre Universitaire - Luxembourg
[3] GDR 957 Sciences Cognitives de Paris
[4] Département Informatique - Télécom Paris

antichaîne de taille maximum. On est dans ce cas lorsque P est un produit direct d'ordres totaux.

Notons qu'en général - dans le cas d'un ensemble ordonné de Sperner - la connaissance d'une antichaîne de taille maximum ne permet pas de déterminer une partition en chaînes de taille minimum. Elle permet tout au plus de vérifier l'optimalité d'une telle partition. C'est la voie suivie dans cette étude : dans le cas d'un produit direct d'ordres totaux, une partition en chaînes est récursivement construite et on montre que sa taille est minimale.

Nous considérons le cas particulier d'un produit direct d'ordres totaux en raison du domaine d'application où nous travaillons : l'acquisition de connaissances et plus particulièrement l'extraction de règles de décision. Imaginez des objets décrits par p attributs induisant chacun une échelle ordinale. L'univers des possibles est alors un produit direct P de p ordres totaux. Imaginez qu'un sujet doive choisir certains de ces objets et que son comportement soit tel que si $x \leq y$ (dans P) et si x est choisi, alors y est choisi. Déterminer l'ensemble de tous les objets choisis revient à déterminer l'ensemble des objets qui sont minimaux au sens de P, c'est-à-dire une antichaîne A de P. Les éléments de A peuvent être vus comme des règles de sélection dont la détermination permettra d'automatiser le comportement de notre sujet. On cherche alors à les obtenir en lui posant le moins de questions possible.

Après avoir rappelé les définitions et résultats qui sont nécessaires pour notre étude, on présente un algorithme de partition d'un produit P de chaînes. Comme on le verra, le nombre de chaînes créées est minimum. Enfin, un algorithme de recherche d'une antichaîne basé sur cette partition est présenté.

2. DÉFINITIONS

Les lecteurs souhaitant plus d'information peuvent se reporter aux livres et articles suivants : Anderson [1] chapitre 3, Leclerc [6] dont nous avons adopté les notations et Griggs [8]. Des résultats permettant de démontrer que l'algorithme engendre le nombre minimum de chaînes seront présentés dans le paragraphe suivant.

2.1. Chaîne, antichaîne

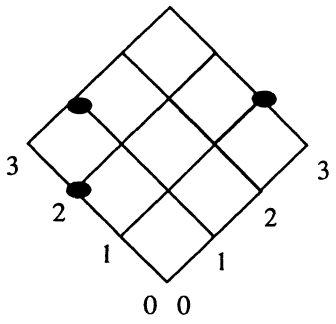
Pour c entier non négatif, on note $(c+1)$ la chaîne (ensemble totalement ordonné) $\{0 < 1 < 2 < ... < c\}$ à $c+1$ éléments. Les éléments de $(c+1)$ sont donc codés par les $c+1$ premiers entiers.

Soit $P = (c_1+1) \times (c_2+1) \times ... \times (c_n+1)$ le produit de n chaînes ayant $N = \prod_{i=1}^{n} (c_i + 1)$ éléments. Pour $x \in P$, on a $x = (x_1, x_2, ..., x_n)$ avec $x_i \in (c_i+1)$.

P est ordonné selon l'ordre usuel : $x, y \in P$, $x \leq y \Leftrightarrow x_i \leq y_i$, pour tout $i = 1, ..., n$.

On appelle *antichaîne* de P tout sous-ensemble A de P dont les éléments sont deux à deux incomparables. Soit $\alpha(P)$ la largeur de P, c'est-à-dire le cardinal maximum d'une antichaîne ; c'est aussi le nombre minimum de chaînes requises pour partitionner P (Théorème de Dilworth). Pour tout entier t, on définit une t-antichaîne comme un sous-ensemble A_t de P dont au plus t éléments sont deux à deux comparables et le nombre de Dilworth $\alpha_t(P)$ comme étant le cardinal maximum d'une t-antichaîne.

EXEMPLE : soit $P = (c_1+1) \times (c_2+1)$ où $c_1 = c_2 = 3$. Une 2-antichaîne de P est {(3,1), (1,3), (0,2)} (Cf. figure 1).



● Points de la 2-antichaîne

Figure 1 : exemple d'une 2-antichaîne

2.2. Fonction de rang, niveaux de P

On dit qu'un ensemble ordonné P est rangé s'il existe une fonction r, appelée fonction de rang, entière non négative sur P telle que $r(x) = 0$ pour au moins un élément (minimal) de P, et que, si x couvre y (x est minimal dans l'ensemble des éléments supérieurs à y) dans P, alors $r(x) = r(y) + 1$.

Pour $x \in P$, $x = (x_1, x_2, ..., x_n)$ on utilise la fonction de rang suivante : $r(x) = \sum_{i=1}^{n} x_i$ La hauteur h de P est le rang maximum d'un élément de P : $h = \sum_{i=1}^{n} c_i$.

Pour tout entier $k \in [0,h]$, le niveau P_k de P est l'ensemble des éléments x de P tels que $r(x) = k$. On note n_k le nombre d'éléments de P_k ($n_k = 0$ pour tout entier $k \{ [0,h]$) et on pose $v = \max_k n_k$.

Un produit de chaînes P est un *ensemble fortement de Sperner*, c'est-à-dire que $\alpha_t(P)$ est égal au cardinal de la réunion des t plus grands niveaux de P ; en particulier $\alpha = v$.

2.3. Produit direct d'un ensemble ordonné et d'une chaîne

Soit $P = P' \times (c+1)$, où P' est un ensemble ordonné et $(c+1)$ une chaîne.

Un élément x de P est noté $x = (x', j)$, avec $x' \in P'$ et $j \in (c+1)$.

Soit r' la fonction de rang de P' et r celle de P. La fonction r est donnée par $r(x) = r'(x') + j$, pour $x = (x', j)$. La hauteur h de P est $h' + c$. On note v' le nombre maximum des n'_k et q' le nombre de niveaux de P' de cardinal v'.

PROPOSITION 1 : (Cf. Leclerc [6])
Les nombres n_k s'obtiennent en fonction des nombres n'_k par l'une ou l'autre des récurrences équivalentes :

$$n_k = \sum_{i=0}^{n} n'_{k-i} \qquad (1)$$

$$n_{k+1} = n_k + n'_{k+1} - n'_{k-c} \qquad (2)$$

DÉFINITION 1 : (Cf. Leclerc [6])
On dit que P est unimodal si la suite finie n_0, n_1, ..., n_h est la concaténation d'une suite non décroissante n_0, ..., n_p et d'une suite non croissante n_{p+1}, ..., n_h.

DÉFINITION 2 : (Cf. Leclerc [6])
P est dit symétrique si on a $n_0 = n_h \leq n_1 = n_{h-1} \leq ... \leq n_k = n_{h-k} \leq ...$

DÉFINITION 3 : (Cf. Anderson [1])
Les éléments x_1, ..., x_k de P forment une chaîne symétrique si :
(i) x_{i+1} couvre x_i pour tout $i < k$
(ii) $r(x_1) + r(x_k) = r(P)$ où $r(P)$ est le rang maximal dans P.

REMARQUES
- Si P est symétrique alors P est unimodal.
- Si P est unimodal alors les niveaux de P de cardinal v sont consécutifs (Cf. figure 2).
- Un produit de chaînes est symétrique.
- Savoir que P est symétrique nous permet de localiser directement l'antichaîne (ou les antichaînes) de taille maximale : $\alpha = v$ est égal à $n_{h/2}$ si h est pair et à $n_{(h+1)/2}$ si h est impair. L'antichaîne de taille α se trouve donc au "milieu" de l'espace.

EXEMPLE : soit $P = (c_1+1) \times (c_2+1)$ où $c_1 = 4$ et $c_2 = 3$.
Sur la figure 2, on peut observer les différents niveaux P_i de P. On peut aisément vérifier qu'un produit de chaînes est unimodal (les niveaux de cardinal maximal étant ici P_3 et P_4) et symétrique : $n_0 = 1$, $n_1 = 2$, $n_2 = 3$, $n_3 = 4$, $n_4 = 4$, $n_5 = 3$, $n_6 = 2$, $n_7 = 1$.

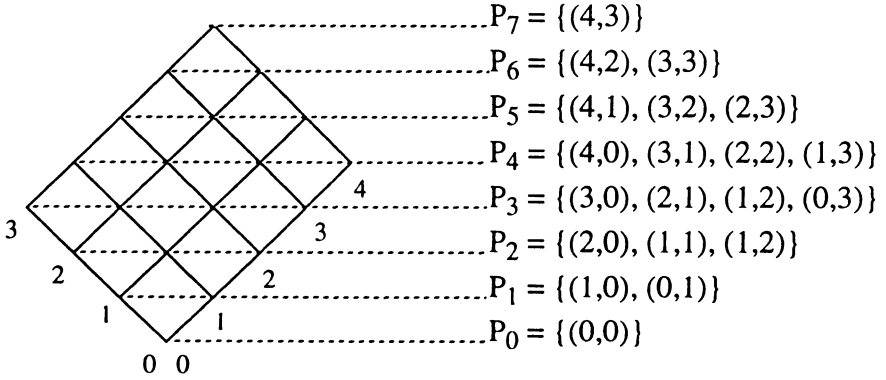


Figure 2 : niveaux d'un produit de chaînes

3. ALGORITHME D'ÉNUMERATION DES CHAINES

3.1. Principe de l'algorithme

On considère l'ensemble ordonné P produit direct de n chaînes représenté, via son codage, dans un espace à n dimensions. On parlera donc de points pour désigner les éléments de P. On veut partitionner P avec un nombre minimum de chaînes.

La création des chaînes est récursive. Les chaînes créées dans l'espace à n-1 dimensions vont servir à créer les chaînes en n dimensions. Les objectifs de cette création sont les suivants : engendrer le nombre minimum de chaînes, parcourir tous les points de l'ensemble que nous appellerons espace, ne parcourir chaque point qu'une fois.

Chaque chaîne possède un élément minimum appelé point de base. Les premiers points des chaînes en n - 1 dimensions sont utilisés comme points de base pour les chaînes en n dimensions. La construction d'une chaîne se décompose en deux phases :

- Les n - 1 premières coordonnées sont celles du point de base (point qui appartient à une chaîne créée en n - 1 dimensions). On fait évoluer la $n^{ième}$ coordonnée de 0 jusqu'au maximum courant pour obtenir la première partie de la chaîne.

- La $n^{ième}$ coordonnée est bloquée sur sa dernière valeur. Les autres coordonnées sont celles des points suivant le point de base sur la chaîne créée en n - 1 dimensions.

Entre deux créations de chaîne, deux mises à jour sont nécessaires :

- La valeur maximale de la $n^{ième}$ coordonnée est réduite de 1,

- On passe au point de base suivant sur la chaîne de base en n - 1 dimensions.

Deux conditions entraînent la création d'une nouvelle chaîne de base au niveau inférieur :

- La chaîne de base courante n'existe pas (elle est vide),

- On ne peut plus faire évoluer la $n^{ième}$ coordonnée (la valeur maximale est nulle).

La chaîne qui sert à créer toutes les autres est la chaîne de l'espace à une dimension : (c_1+1).

3.2. Exemple

Création des chaînes de $P=P_1 \times P_2 \times P_3$ où $P_1=\{0<1<2<3\}$, $P_2=\{a<b<c\}$ et $P_3=\{-<+\}$.

Cet espace contient 24 points et $\alpha(P) = 6$. Pour faciliter la lecture, les chaînes sont présentées niveau par niveau alors que l'algorithme parcourt les niveaux en profondeur d'abord.

Création de la chaîne de base à une dimension à partir de P_1, ce qui donne : $\{0<1<2<3\}$.

Création des chaînes de base à deux dimensions à partir de $P_1 \times P_2$:

- La chaîne de base est $\{0 < 1 < 2 < 3\}$. La seconde dimension est constituée de 3 valeurs, on va utiliser au maximum 3 points de base.

- Première phase : on part du premier point de base (c'est-à-dire 0). La valeur maximale initiale de la seconde dimension est c. On fait évoluer la seconde coordonnée de sa valeur minimale jusqu'à sa valeur maximale (c'est-à-dire de a à c). Ce qui donne la première partie de la chaîne: $\{0a < 0b < 0c\}$.

- Deuxième phase : la valeur maximale est bloquée (ici, sur la valeur c) et on parcourt le reste de la chaîne de base (c'est a dire 1, 2 et 3). Ce qui donne la deuxième partie de la chaîne : $\{1c < 2c < 3c\}$.

- La chaîne ainsi créée est $\{0a < 0b < 0c < 1c < 2c < 3c\}$. Pour générer une nouvelle chaîne, on passe au point de base suivant (c'est à dire le point 1) et la valeur maximale de la seconde coordonnée est diminuée (on passe de c à b).

• Lorsque la valeur maximale atteint son minimum (ici, la valeur *a*), les points de base suivants ne sont pas utilisés (ici, le point 3).

• La suite du texte présente les chaînes créées de la façon suivante :
Point de base {Première partie de la chaîne | Seconde partie de la chaîne}

Les trois chaînes de base sont les suivantes :
0 . {$0a < 0b < 0c \mid 1c < 2c < 3c$}
1 . {$1a < 1b \mid 2b < 3b$}
2 . {$2a \mid 3a$}
3 . n'est pas utilisé

Création des chaînes à trois dimensions à partir de $(P_1 \times P_2) \times P_3$:

• Utilisation de la première chaîne de base {0a < 0b < 0c | 1c < 2c < 3c}
$0a$. {$0a- < 0a+ \mid 0b+ < 0c+ < 1c+ < 2c+ < 3c+$}
$0b$. {$0b- \mid 0c- < 1c- < 2c- < 3c-$}
$0c$, $1c$, $2c$, $3c$. ne sont pas utilisés

• Utilisation de la deuxième chaîne de base {$1a < 1b \mid 2b < 3b$}
$1a$. {$1a- < 1a+ \mid 1b+ < 2b+ < 3b+$}
$1b$. {$1b- \mid 2b- <$ 3b-}
$2b$, $3b$. ne sont pas utilisés

• Utilisation de la troisième chaîne de base {$2a \mid 3a$}
$2a$. {$2a- < 2a+ \mid 3a+$}
$3a$. {$3a-$}

Les six chaînes ainsi créées sont les suivantes :
{$0a- < 0a+ < 0b+ < 0c+ < 1c+ < 2c+ < 3c+$},
{$0b- < 0c- < 1c- < 2c- < 3c-$},
{$1a- < 1a+ < 1b+ < 2b+ < 3b+$},
{$1b- < 2b- < 3b-$},
{$2a- < 2a+ < 3a+$},
{$3a-$}.

3.3. Algorithme d'énumération des chaînes

```
Toutes_les_chaines (n, maximum )                    /* Création des chaînes de P      */
début
faire           chaîne = Chaîner ( n, maximum )
                C = Ajoute_chaîne ( chaîne, C )
tant_que  Existe ( chaîne )
résultat = C
fin

Chaîner ( n, maximum )                              /* Création d'une chaîne dans     */
début                                               /* un espace à n dimensions       */
si non Existe ( chaîne' ) ou max [n] < 0            /* S'il n'existe pas de chaîne    */
                max [n] = maximum [n]               /* au niveau inférieur            */
        chaîne' = Chaîner ( n -1, maximum )         /* Création d'une chaîne          */
fin_si
si Existe ( chaîne' )
        point' = Premier_point ( chaîne' )          /* On part du point de base       */
    pour i de 0 a max [n]                           /* Première phase                 */
                point = Ajoute_coordonnée ( i, point' )   /* Evolution de la n^ième   */
                chaîne = Ajoute_point ( point, chaîne)    /* coordonnée du point courant */
    fin_pour
    point' = Point_suivant ( point', chaîne' )
```

```
    tant_que Existe ( point' )                          /* Deuxième phase              */
            point = Ajoute_coordonnée ( max [n], point' ) /* Blocage de n^ième coordonnée */
            chaîne = Ajoute_point ( point, chaîne )     /* Parcours de la chaîne de base */
            point' = Point_suivant ( point', chaîne' )
    fin_tant_que
    chaîne' = Supprime_premier_point ( chaîne' )        /* Mise à jour de la chaîne de base*/
    max [n] = max [n] - 1                               /* et du maximum courant       */
fin_si
résultat = chaîne
fin
```

Fonctions

Ajoute_chaîne (*chaîne*, *C*)	ajoute *chaîne* dans l'ensemble *C*,
Existe (*chaîne*)	indique si *chaîne* n'est pas une chaîne vide,
Premier_point (*chaîne*)	retourne le premier point de *chaîne*,
Ajoute_coordonnée (*i*, *point*)	ajoute une coordonnée de valeur *i* à *point*,
Ajoute_point (*point*, *chaîne*)	ajoute *point* à la fin de *chaine*,
Point_suivant (*point*, *chaîne*)	retourne le point suivant *point* sur *chaîne*,
Supprime_premier_point (*chaîne*)	retourne *chaîne* sans son premier point.

Lexique

P'	espace à n - 1 dimensions sous forme d'un produit de chaînes,
P	espace à n dimensions sous forme d'un produit de chaînes,
C'	ensemble des chaînes qui recouvrent P',
C	ensemble des chaînes qui recouvrent P,
chaîne'	chaîne appartenant à C',
chaîne	chaîne appartenant à C,
point'	point courant sur chaîne',
point	point courant sur chaîne,
maximum	indice maximum de chaque dimension,
max	indice maximum de chaque dimension évoluant lors du traitement d'une chaîne',
n	nombre de dimensions de P.

3.4. Propriétés

PROPOSITION 2
L'algorithme d'énumération des chaînes construit une partition de P en chaînes.

PREUVE
Tous les points de P sont couverts une fois et une seule par propagation d'une continuité locale :

```
SUIVANT(point1) = point2 avec
si n > 1 alors
    point2[n] = point1[n] + 1 et point2[i] = point1[i] ∀ i < n   /* Première phase  */
ou  point2[n] = point1[n],                                       /* Deuxième phase */
    point_de_base2 = SUIVANT(point_de_base1),
    ∀ i < n     ( point_de_base2[i] = point2[i]
                  et point_de_base1[i] = point1[i] )
sinon
    point2[1] = point1[1] + 1                                    /* Pour la chaîne à une dimension */
```

Les deux points de niveaux extrêmes de l'espace, à savoir le point (0, 0, ...,0) et (c_0, c_1, ..., c_n), sont traités immédiatement dans la première chaîne. Ces deux propriétés (continuité et traitement des extrema) nous garantissent donc que tous les points sont traités. Enfin, l'unicité du traitement est assurée par le fait que le numéro de coordonnée sur laquelle évolue le point est strictement décroissante.

PROPOSITION 3
Le nombre de chaînes créées par l'algorithme est minimal.

PREUVE
Il suffit de montrer que l'on obtient une partition de P en chaînes symétriques (Cf. Définition 3). Comme toute chaîne symétrique a un élément et un seul dans le niveau central, le nombre de chaînes générées est minimal car égal à la largeur de P.

(i) est vérifiée car pour tout $i < k$, on peut écrire x_{i+1} = SUIVANT (x_i)

La première chaîne créée vérifie (ii) car r (point_minimal) = 0, r (point_maximal) = $\sum_{i=1}^{n} c_i$ et r (P) = $\sum_{i=1}^{n} c_i$.

Par construction de l'algorithme, pour chaque nouvelle chaîne :

- Il existe un i unique avec $0 \leq i \leq n$ tel que max[i] diminue de 1 et donc r (x_k),
- r (x_1) augmente de 1 car le nouveau point de base vérifie la relation :
  Nouveau_point_de_base = SUIVANT (Ancien_point_de_base).

La somme r (x_1) + r (x_k) reste donc constante et égale à r (P). La condition (ii) est vérifiée pour toutes les chaînes créées. On peut conclure que la partition de P est constituée de chaînes symétriques et donc qu' elle est de taille minimale.

4. APPLICATION A LA RECHERCHE D'UNE ANTICHAINE

On se place dans le contexte de l'extraction de règles de décision que nous avons évoqué dans l'introduction. On utilise le modèle de processus de décision proposé par Barthélemy et Mullet (Heuristique de la base mobile, Cf. Barthélemy, Mullet [2] et [3]) et fondé sur des recherches en psychologie (Cf. Mullet [7]).

Soit P = (c_1+1) × ... × (c_p+1) le produit direct de p ordres totaux correspondant à l'univers des possibles. Notre sujet (qui fonctionne à seuil : si un objet x est sélectionné et si y est supérieur à x alors y est également sélectionné) doit catégoriser les éléments de P qui lui sont présentés parmi deux catégories possibles (C_1 catégorie des objets sélectionnés et C_2 catégorie des objets rejetés). Déterminer les objets choisis revient à calculer une antichaîne de P dont les éléments sont les règles de sélection des objets. Soit A l'antichaîne séparant C_1 et C_2 (C_1 étant la partie finissante admettant A pour ensemble d'éléments minimaux). On présente ci-dessous le principe et deux méthodes de recherche de A, l'objectif étant de minimiser la taille du questionnaire.

4.1. Principe général de recherche de A

Algorithme simplifié de recherche de A :

début
$i = 0$
$P_0 = P_i$
tant_que $P_i <> \emptyset$
    Choisir (Un_Objet, P_i)
    Proposer_au_Sujet (Un_Objet, Réponse_Sujet)
    Propager_Réponse (Un_Objet, Réponse_Sujet, P_i)
    $i = i + 1$

fin_tant_que
Afficher_Antichaîne (P_i)
Ecrire ("Nombre de questions posées : ", *i*)
fin

Procédures :

- Choisir() : c'est la procédure qui choisit l'objet à présenter au sujet parmi l'ensemble P_i des objets non catégorisés. C'est la procédure la plus importante de l'algorithme : la taille du questionnaire en dépend beaucoup.
- Proposer_au_Sujet() : propose un objet et lit la réponse du sujet.
- Propager_Réponse() : catégorise les éléments de P_i en fonction de l'objet présenté et de la réponse du sujet. Si x élément de P_i est classé dans C_1 (respectivement dans C_2) alors tous les points supérieurs au sens de la relation d'ordre usuelle (respectivement inférieurs) à x sont également dans C_1 (respectivement dans C_2) : on élimine à chaque étape le plus de points possibles.

Ainsi on distingue les méthodes de recherche de A par la façon dont on choisit l'objet courant à présenter au sujet.

4.2. Recherche dichotomique

Une approche simple consiste à effectuer une recherche dichotomique sur les chaînes (les unes après les autres). L'espace P étant partitionné en α chaînes, la taille maximale d'une chaîne étant la hauteur h de P on peut en déduire la complexité de ce type de recherche : $\alpha \text{ Log}_2 h$.

On peut heureusement faire beaucoup mieux. En effet, cette complexité ne tient absolument pas compte de la propagation du classement de x sur l'ensemble P tout entier. De plus, on se place dans le pire des cas où toutes les chaînes sont de taille maximale h.

4.3. Recherche "optimale"

La meilleure stratégie possible est celle qui choisit l'élément x tel que, quel que soit son classement, on puisse classer le plus de points possibles de P par propagation. La recherche de l'élément à proposer à chaque étape se fait en maximisant le critère : $\min(p(x), q(x))$ où $p(x)$ est le nombre d'éléments non classés plus grands que x et $q(x)$ le nombre d'éléments non classés plus petits que x (critère maximisé sur tous les éléments de P_i). Nous n'avons pas d'estimation de la complexité de ce type de recherche. Cette stratégie de recherche est illustrée dans l'exemple ci-dessous.

4.4. Un exemple de recherche d'antichaîne (Cf. figure 3)

On considère un espace P comportant deux dimensions (attributs) pouvant prendre chacune 5 valeurs. Cet espace contient 25 objets possibles. Sa largeur est 5 et sa hauteur 9. Le nombre maximal théorique de questions à poser est donc 16. On simule le comportement du sujet en répondant de telle façon que l'antichaîne A à trouver soit (3,1) + (1,2). C'est a dire que tout élément de P ayant au moins la valeur 3 (ou 1) sur le premier attribut et 1 (ou 2) sur le second attribut appartient à C_1 sinon il est dans C_2. La recherche de A se fait alors en 10 questions.

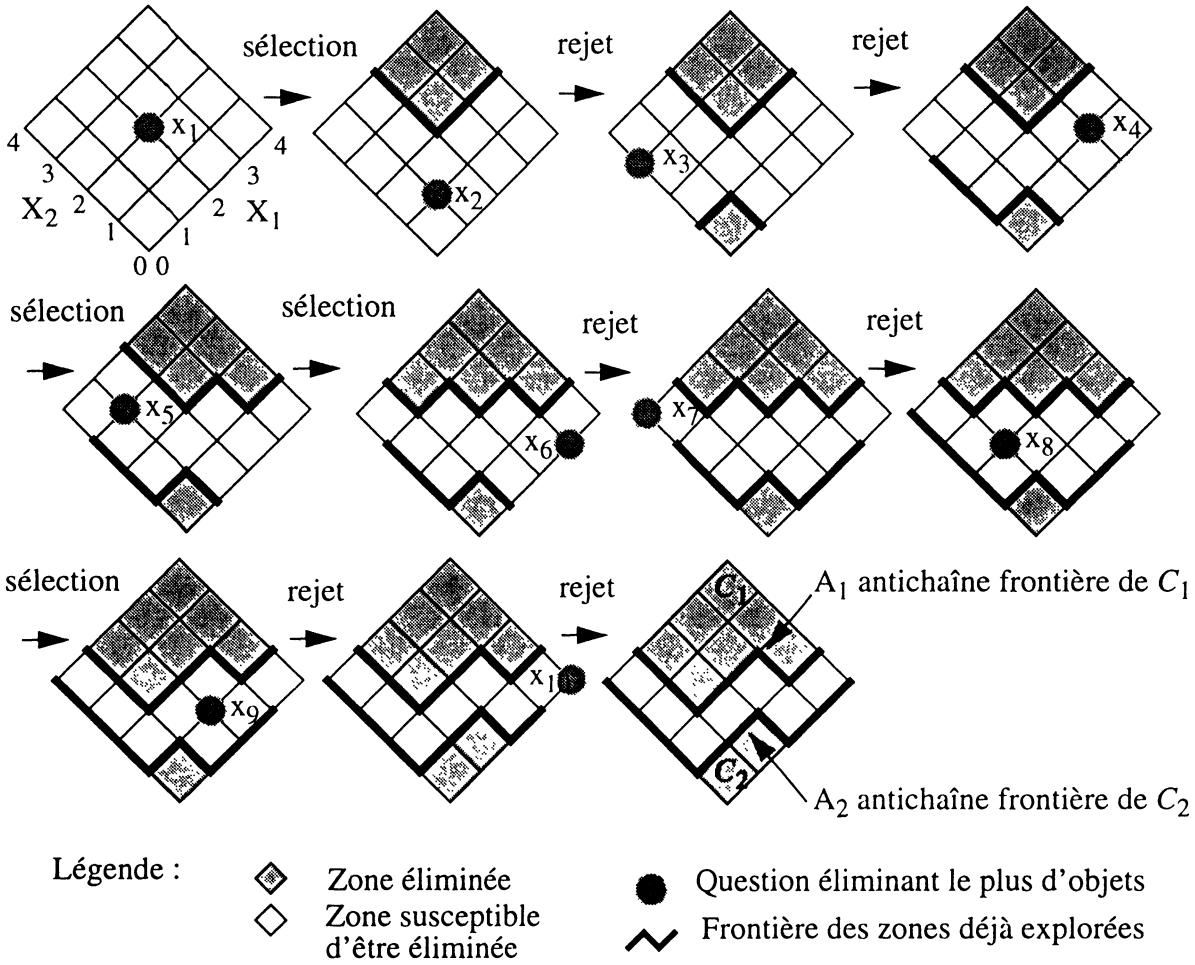


Figure 3 : exemple de recherche d'antichaîne

BIBLIOGRAPHIE

[1] ANDERSON I., *Combinatorics of Finite sets,* Oxford, Clarendon press, 1987.

[2] BARTHELEMY J.P., MULLET E., "Choice basis: A model for multi-attribute preference", *British journal of Mathematical and Statistical Psychology,* 39 (1986), p. 106-124.

[3] BARTHELEMY J.P., MULLET E. (1992), "A model of selection by aspects", *Acta Psychologica*, 79, 1-19.

[4] BEHRENDT G., "The lattice of antichain cutsets of a partially ordered set", *Discrete math.*, 89 (1991), p. 201-202.

[5] De BRUIJN N.G., TENGBERGEN C., KRUYSWIJK D., "On the set of divisors of a number", *Nieuw Arch. Wiskd.*, 23 (1951), p. 191-193.

[6] LECLERC B., "Sur le nombre d'éléments des niveaux des produits de chaînes et des treillis permutoèdres", *Math. Inf. Sci. Hum.*, 112 (1990), p. 37-48.

[7] MULLET E., *L'intégration des informations dans le jugement et la décision*, Thèse de Doctorat d'Etat (1985), Université René Descartes, Sciences Humaines, Sorbonne.

[8] GRIGGS J.R., "Maximum antichains in the product of chains", *Order*, 1 (1984), p. 21-28.

[9] GRIGGS J.R., "Problems on chain partitions", *Discrete Math.*, 72 (1988), p. 157-162.

[10] SANDS B., "Counting antichains in finite partially ordered sets", *Discrete Math.*, 35 (1981), p. 213-228.