MATHÉMATIQUES ET SCIENCES HUMAINES

J. P. BORDAT

Calcul pratique du treillis de Galois d'une correspondance

Mathématiques et sciences humaines, tome 96 (1986), p. 31-47

<http://www.numdam.org/item?id=MSH_1986__96__31_0>

Math. Sci. hum. (24^{e} année, n°96, *1986*, p.31-47)

CALCUL PRATIQUE DU TREILLIS DE GALOIS D' UNE CORRESPONDANCE

J.P. BORDAT *

I - INTRODUCTION

Le problème envisagé est une construction algorithmique du treillis de Galois d' une correspondance (cf. [2]). La connaissance de cette structure présente un intérêt certain en Mathématiques Appliquées aux Sciences Sociales. Bien que le nombre d' éléments du treillis construit puisse être exponentiel en fonction de la taille des données, et que ce problème soit clairement NP-complet ([10]), les exemples traités dans la pratique gardent une taille raisonnable, et le calcul du résultat exact mérite dès lors attention.

La section II expose les notions nécessaires à la compréhension de la suite de l' exposé.

Les sections III, IV et V discutent les solutions algorithmiques en les exprimant en termes graphiques, et en évaluant leur complexité. Il est nécessaire pour cela de donner un aperçu d' une structure de données utilisable.

La section V effectue la synthèse des précédentes et cite certains articles se rapportant à ce problème.

II - NOTATIONS ET RAPPELS

II-1 - Graphes et ensembles ordonnés :

Nous supposerons connues les notions de base de Théorie des Graphes, telles que : graphe orienté, sommet, arc, successeur, fonction successeur, chemin, circuit, sous-graphe construit sur un ensemble de sommets. (ouvrage de référence : [3]). Dans la suite de l' exposé seront envisagés des graphes simples, finis, sans circuit.

Soit $G = (X, \gamma)$, X ensemble de sommets, γ fonction successeur.
Une source de G est un sommet qui n' est successeur d' aucun autre sommet.

* Centre de Recherche en Informatique de Montpellier

La fermeture transitive G^t associée à G est le graphe ayant pour ensemble de sommets X et pour fonction successeur γ^t :

$y \in \gamma^t(x) \iff$ il existe un chemin joignant x à y dans G

Un graphe biparti est tel que X soit partitionné en deux classes A et B avec $\gamma(A) \subseteq B$ et $\gamma(B) = \emptyset$. (On pose $\gamma(A) = \bigcup(\gamma(x) \mid x \in A)$)

De même, nous supposerons connues les notions de base de la Théorie des Ensembles Ordonnés telles que : relation d' ordre, relation de couverture, majorant, graphe de Hasse associé à un ensemble ordonné (ouvrage de référence : [2]). Dans la suite de l' exposé ne seront envisagés que des ensembles ordonnés finis.

II-2 - Treillis de Galois d' une correspondance :

On trouvera dans [2] des précisions supplémentaires sur cette section.

Soient E et F deux ensembles finis, et R une relation binaire, ou correspondance entre E et F.

Le treillis de Galois L de R est défini comme suit :

Un élément de L est un produit cartésien A x B avec :

- $A \subseteq E$, $B \subseteq F$, et $\forall\, a \in A$, $\forall\, b \in B$, $R(a,b)$.
- A x B est maximal (au sens de l' inclusion) pour cette propriété.

Les éléments de L sont parfois appelés rectangles maximaux de R.

Relation d' ordre sur L :

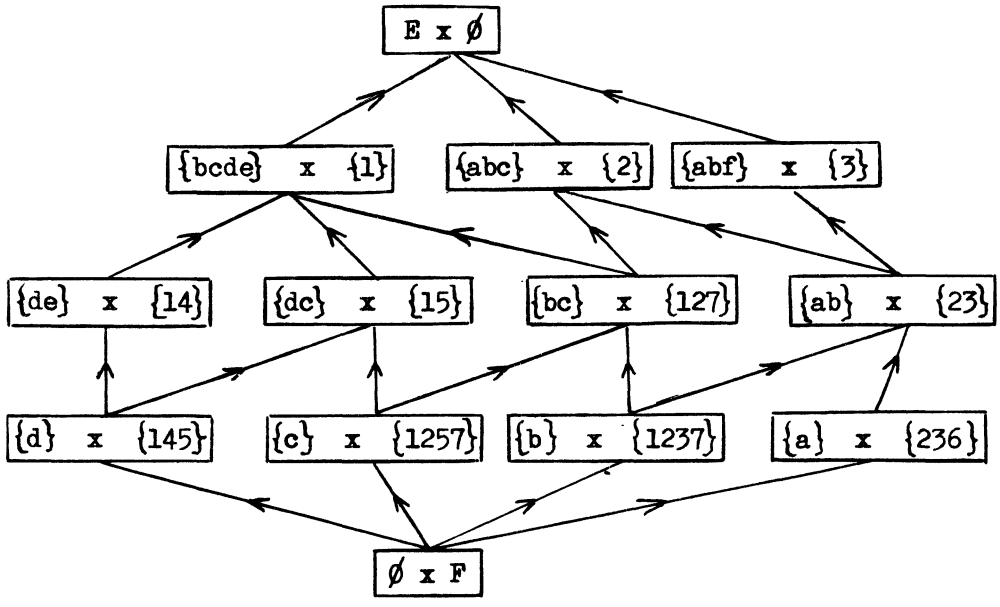
$$A \times B \underset{L}{<} A' \times B' \iff A \subset A' \quad \text{et} \quad B \supset B'$$

Exemple : ([2])

La correspondance R est donnée par la table suivante :

E \ F	1	2	3	4	5	6	7
a		1	1			1	
b	1	1	1				1
c	1	1			1		1
d	1			1	1		
e	1			1			
f			1				

Le graphe de Hasse du treillis L a 13 sommets :

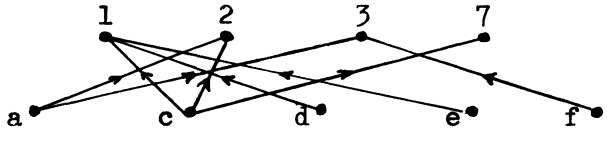


La correspondance R peut être représentée par un graphe biparti G ayant pour ensemble de sommets $E \cup F$, et qui contient l' arc ef si et seulement si R(e,f).

Nous désignerons par G(A,B) le sous-graphe de G construit sur $A \cup B$, $A \subseteq E$, $B \subseteq F$.

On appelle parties maximales d' un graphe biparti les ensembles de successeurs qui sont maximaux au sens de l' inclusion.

Exemple :



Parties maximales : {2,3} (successeurs de a)
{1,2,7} (successeurs de c)

Relation de couverture dans L :

THEOREME : A' x B' couvre A x B dans L si et seulement si B' est une partie maximale de G(E-A,B). Alors A' est l' union disjointe de A et des sommets de G(E-A,B) qui admettent B' comme ensemble de successeurs.

Preuve : Voir Annexe.

En conséquence, nous associerons à chaque élément A x B de L le graphe biparti G(E-A,B). Le graphe biparti de l' exemple précédent correspond à l' élément {b} x {1,2,3,7} de L. Les éléments de L qui couvrent {b} x {1,2,3,7} sont, d' après le théorème : {a,b} x {2,3} et {b,c} x {1,2,7}.

Le nombre de successeurs d' un sommet quelconque de H est borné par |E| , puisqu' il ne peut y avoir plus de |E-A| parties maximales dans G(E-A,B). Dualement, le nombre de prédécesseurs d' un sommet de H est borné par |F| . Le nombre d' arcs de H n' est donc pas trop grand par rapport au nombre de sommets.

La remarque faite à l' alinéa précédent nous conduit à adopter la stratégie algorithmique suivante : Plutôt que de calculer les rectangles maximaux pour les comparer ensuite, L sera engendré directement par sa relation de couverture. Les arcs issus d' un sommet de H seront produits en utilisant le théorème précédent. Cette méthode évite l' apparition de nombreux arcs de transitivité, qui sont toujours difficiles à éliminer par la suite (voir par ex. [8] pour le calcul du graphe de Hasse à partir d'un graphe sans circuit quelconque).

III - L' ALGORITHME DE GENERATION

Le théorème débouche naturellement sur la génération de H par niveaux ("in breadth-first order".Les sommets sont produits par ordre de niveau croissant, le niveau d' un sommet étant la longueur d' un plus court chemin joignant ce sommet à l' unique source.

Nous supposons le lecteur familiarisé avec le parcours par niveaux d' un graphe. L' algorithme classique utilise une structure de File, dont les opérations associées sont :

- INIT : création de la File à partir de son premier élément.
- FRONT : donne l' élément situé en tête de la File.
- SUPPRIMER : supprime l' élément situé en queue de File.
- INSERER : insère un nouvel élément en tête de File.
- FILEVIDE : Booléen testant si la File est vide.

L' algorithme s' écrit alors :

$\{x_o$ source unique$\}$

INIT (File,x_o) ; T := $\{x_o\}$;

Répéter

x := FRONT(File) ; SUPPRIMER (File) ;

{ génération des arcs issus de x }

Pour tout successeur y de x faire

début ENGENDRER (xy) ;

Si $y \notin T$ alors

début T := T $\cup$ y ; INSERER(File,y) ; fin

fin

Jusqu' à FILEVIDE ;

Commentaire : Les sommets ont été désignés par les lettres x_o, x, y. Un ensemble T permet d' effectuer le test d' appartenance (cas d' un sommet déjà produit). L' opération ENGENDRER dépend de la structure de données utilisée pour représenter H et sera décrite par la suite.

La démonstration de l' algorithme et la preuve d' arrêt sont classiques et ne seront pas réexposées ici (voir par ex. [1]). L' algorithme ainsi décrit comporte deux questions essentielles :

a) Description des successeurs d' un sommet x (boucle Pour), c' est-à-dire dans le cas présent calcul des parties maximales d' un graphe biparti, et calcul de ce graphe biparti.

b) Test d' appartenance à un ensemble, et union ensembliste dans le cas où ce test n' est pas vérifié.

Ce problème, ultra-classique dans la littérature, ne peut être résolu de manière efficace sans aborder la question de la structure de données représentative de l' ensemble. Nous proposons dans le paragraphe V un exemple de structure de données agréable.

IV - PARTIES MAXIMALES DANS UN BIPARTI

IV-1 - Une borne supérieure de complexité:

Dans cette section, nous rapprochons deux problèmes :

Problème 1 :

Donnée : un graphe biparti, c' est-à-dire dont l' ensemble des sommets peut être partitionné en 2 classes A et B avec, si γ est la fonction successeur, $\gamma(A) \subseteq B$ et $\gamma(B) = \emptyset$.

Résultat : Pour chaque sommet a_i de A, on obtient les sommets a_j tels que $\gamma(a_i) \subseteq \gamma(a_j)$.

Problème 2 :

Donnée : un graphe triparti, c' est-à-dire dont l' ensemble des sommets peut être partitionné en 3 classes X, Y et Z avec, si μ est la fonction successeur, $\mu(X) \subseteq Y$, $\mu(Y) \subseteq Z$, $\mu(Z) = \emptyset$.

Résultat : La fermeture transitive de ce graphe.

Ce problème, très étudié, ([6], [8], [12]) se résout par un algorithme de complexité $O(n^{\alpha})$, $\alpha \simeq 2{,}5$, si n est le nombre de sommets de la donnée (O notation due à Knuth, cf. [11]). La notation α sera réutilisée dans le paragraphe VI.

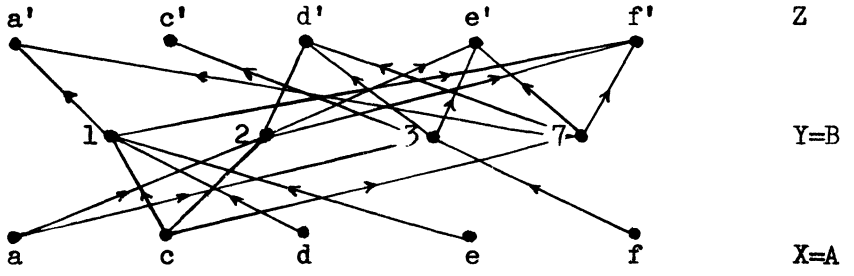
On utilise une technique de réduction entre ces deux problèmes :

Résolution du problème 1 (à partir d' un algorithme résolvant le problème 2).

Soit G biparti. On lui associe G' triparti, tel que

- $X = A = \{a_1, a_2, \ldots, a_p\}$ $\quad Y = B = \{b_1, b_2, \ldots, b_q\}$ $\quad Z = \{c_1, c_2, \ldots, c_p\}$
- La restriction de μ à $X \cup Y = A \cup B$ est égale à γ .
  D' autre part, $\forall\, c_i \in Z$, $\quad \mu^{-1}(c_i) = B - \gamma(a_i)$

Exemple : Si G est le graphe biparti du paragraphe II, le graphe triparti associé G' est le suivant :



G'

Cette construction consiste à rajouter un troisième niveau de sommets en bijection avec le premier niveau et tel que l' ensemble des prédécesseurs d' un sommet soit le complémentaire dans B de l' ensemble des successeurs de son homologue.

On a alors la propriété suivante :

Soient i,j, $1 \leqslant i < j \leqslant p$

$\gamma(a_i) \subseteq \gamma(a_j)$ si et seulement si $a_i c_j$ n' est pas un arc de G'^t.

En effet $\gamma(a_i) \subseteq \gamma(a_j) \iff \mu^{-1}(c_j) \cap \mu(a_i) = \emptyset$

Posons $n = |A \cup B|$, et $n' = |X \cup Y \cup Z|$. Alors $n' < 2n$. On obtient un algorithme résolvant le problème 1 en :

- calculant G' : complexité $O(n^2)$.
- calculant G'^t, ce qui permet de déduire immédiatement, pour tout $a_i \in A$, les indices j, s' il en existe, tels que $\gamma(a_i) \subseteq \gamma(a_j)$.

  La complexité de cette deuxième étape est $O(n'^{\alpha})$, puisqu' il s' agit de la résolution du problème 2.

On obtient ainsi un algorithme de résolution du problème 1, de complexité $O(n^{\alpha})$.

Réciproquement, étant donné un graphe triparti G construit sur $X \cup Y \cup Z$ et de fonction successeur μ, on peut lui associer un biparti G' :

- $A = X \cup Z \qquad B = Y$
- $\forall\, a_i \in A, \quad \gamma(a_i) = \mu(a_i)$ si $a_i \in X$, $\gamma(a_i) = B - \mu^{-1}(a_i)$ si $a_i \in Z$.

On montrerait de la même façon que précédemment la propriété :

Soient $x_i \in X$ et $z_j \in Z$

$x_i z_j$ n' est pas arc de G'^t si et seulement si $\gamma(x_i) \subseteq \gamma(z_j)$

A partir d' un algorithme résolvant le problème 1 en $O(|A \cup B|^{\alpha})$ opérations, $\alpha \geqslant 2$, on obtient là aussi un algorithme de résolution du problème 2, et de complexité $O(n^{\alpha})$, si $n = |X \cup Y \cup Z|$. La démonstration est similaire à la précédente.

D' où la conclusion : Un algorithme de complexité $O(n^{k})$, $2 \leqslant k \leqslant \alpha$, résolvant le problème 1 induit un algorithme de même complexité résolvant le problème 2 et réciproquement.

Le problème 1 n' est pas celui qui nous préoccupe directement, puisque l' on cherche seulement les parties maximales, sans tenir compte des majorations intermédiaires. On obtient donc avec $O(n^{\alpha})$ seulement une borne supérieure de complexité, néanmoins intéressante, comme le montre l' algorithme exposé dans le sous-paragraphe suivant.

IV-2 - Un algorithme simple :

Un algorithme de calcul des parties maximales dans un biparti peut d' après ce qui précède être composé des deux étapes suivantes :

- Résolution du problème 1.
- Pour tout $a_i \in A$, $\gamma(a_i)$ est partie maximale s' il n' existe pas $a_j \in A$ tel que $\gamma(a_i) \subset \gamma(a_j)$.

La complexité globale est clairement imposée par la première étape, soit $O(n^{\alpha})$.

Nous donnons un algorithme simple de parcours graphique, plus avantageux et plus facile à mettre en oeuvre sur des exemples de taille raisonnable. Sa complexité asymptotique est néanmoins supérieure à la borne obtenue précédemment.

Les sommets de A sont supposés classés par degré non décroissant, afin de simplifier la présentation. Ce préalable ne modifie pas la complexité obtenue.

{Calcul des parties maximales dans G biparti}
{construit sur A ∪ B et de fonction successeur γ}
{On suppose $\forall i,j,\ 1 \leq i < j \leq p \Rightarrow |\gamma(a_i)| \geq |\gamma(a_j)|$}

Pour i := 1 à p faire max(a_i) := vrai ;

Pour i := 1 à p faire

début C := $\{a_i\}$;
Si max(a_i) alors {$\gamma(a_i)$ partie maximale}

Pour j := i+1 à p faire
début Si $\gamma(a_j) \subseteq \gamma(a_i)$ alors max(a_j) := faux ;
Si $\gamma(a_j) = \gamma(a_i)$ alors C := C ∪ $\{a_j\}$;
fin

{$\gamma(a_i)$ maximale $\Leftrightarrow$ C = $\gamma^{-1}(\gamma(a_i))$}
fin

Commentaire : A chaque sommet a_i de A est associé le booléen max(a_i). Si $\gamma(a_i)$ est une partie maximale, son image réciproque est calculée dans C.

La preuve d' arrêt est évidente. De plus l' assertion :

max(a_i) $\Leftrightarrow$ ∃ j, $1 \leq j < i$ tel que $\gamma(a_i) \subseteq \gamma(a_j)$

est un invariant vérifié à chaque tour de la boucle extérieure.

Complexité : Dans la boucle Pour intérieure sont parcourus les arcs issus de $a_{i+1}, \ldots, a_p$. Sa complexité est donc $O(|\gamma(A)|)$. Elle est exécutée p fois, ce qui donne $O(|A|\,|\gamma(A)|)$. Remarquons que si $|A \cup B| = n$, on peut obtenir $O(n^3)$ dans le plus mauvais cas. Mais ce calcul peut aussi être très rapide, s' il n' existe qu' une seule partie maximale par exemple.

L' exposant 3 est néanmoins supérieur à la borne α rencontrée précédemment, et qui constitue actuellement le meilleur résultat pour un calcul de fermeture transitive (problème 2).

V - UNE REPRESENTATION EFFICACE DU TREILLIS DE GALOIS

Chaque élément de l' ensemble T (§ III) est en fait un produit cartésien. La première composante de ce produit, par exemple, est une partie de E. Si l' on suppose l' existence d' un ordre total Ω sur E, $\mathcal{P}(E)$ est ordonné totalement par l' ordre lexicographique induit. L' ensemble T est ainsi muni d' un ordre total, puisque chaque élément de T est identifiable grâce à sa première composante.

A partir de cette constatation, on se trouve face à un problème bien répertorié, et qui ne peut être abordé sans faire référence à une structure de données représentant T. Si l' on choisit par exemple des structures d' arbres binaires correctement gérées, on sait que l' opération de recherche-insertion pourra s' effectuer en un nombre de comparaisons proportionnel au logarithme du cardinal de l' ensemble (en moyenne et au sens du plus mauvais cas, voir par ex [1]).

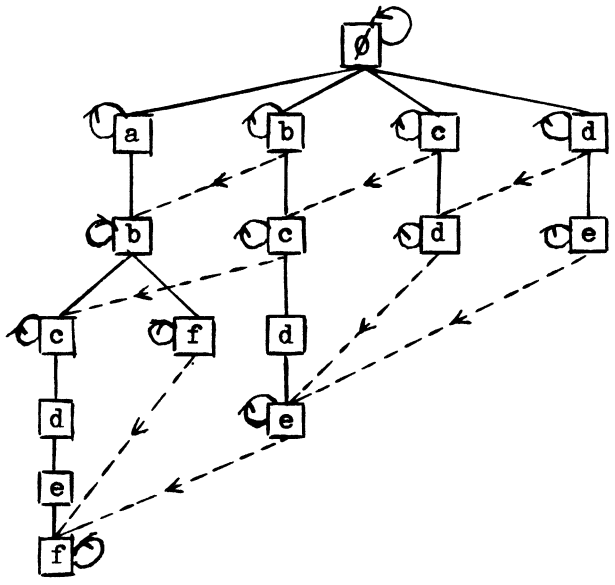
Nous développons ici une structure particulière permettant à la fois la gestion de l' ensemble T et la représentation du résultat cherché H : il s' agit d' une structure d' arbre "cousu".

Dans cette représentation, E joue le rôle d' un alphabet totalement ordonné par Ω . Chaque noeud de l' arbre contient une lettre de cet alphabet, avec les conventions suivantes :

a) La liste des fils d' un noeud quelconque est ordonnée suivant Ω .

b) Les noeuds d' un chemin reliant la racine de l' arbre à un noeud quelconque sont eux-mêmes ordonnés suivant Ω .

On pourra se reporter par exemple à [1] en ce qui concerne cette structure, décrite sous le nom de "trie". Un trie associé à l' exemple du § II sera le suivant :



La première composante d' un élément de T joue ainsi le rôle d' une clé de recherche dans l' arbre. Une boucle a été rajoutée dans la figure sur chaque noeud correspondant à une clé. L' arbre est "cousu" par des arcs traversiers dessinés en pointillés, ce qui lui permet de représenter le graphe de Hasse H du treillis de Galois.

Mécanisme de la recherche-insertion :

La condition 1) permet d' effectuer une recherche-insertion dans la structure sans parcourir tous les fils d' un noeud.

La condition 2) permet de décrire un chemin unique lors d' une recherche-insertion d' une clé. En effet, alors que dans les structures classiques, les mots bcde et bced par exemple sont deux clés distinctes, ici ils représentent le même sous-ensemble $\{bcde\}$ de E et il ne doit pas y avoir d' ambiguité. La condition 2) entraîne comme conséquence qu' une clé doit se présenter déjà classée dans l' ordre lexicographique associé à Ω .

Génération de l' arc $(A \times B, A' \times B')$ dans H :

Deux cas se présentent :

- A'-A ne possède que des caractères supérieurs aux caractères de A suivant Ω . Alors A' ne se trouve pas déjà dans l' arbre. Son insertion peut s' effectuer directement à partir du noeud de clé A que l' on est précisément en train de considérer.

- Dans le cas contraire, on effectue un parcours classique partant de la racine. Un arc pointillé sera alors créé entre le noeud de clé A et le noeud de clé A'.

Notons enfin que :

- la clé d' un noeud s' obtient en remontant le chemin joignant la racine à ce noeud.
- H s' obtient facilement par balayage de l' arbre en partant de la racine.
- La représentation simultanée de H et T permet d' économiser de la place en mémoire.
- La deuxième composante d' un produit cartésien peut être attachée à la clé correspondante de l' arbre, ou bien stockée dans un autre arbre analogue au premier et relié au premier.

VI - L' ALGORITHME DEFINITIF

Compte tenu des paragraphes IV et V, une nouvelle présentation de l' algorithme de génération, utilisant une procédure RECHERCHE de recherche-insertion sera la suivante :

```
INIT (File, ∅ x F) ;

Répéter
A x B := FRONT (File) ; SUPPRIMER (File) ;

Pour  tout  A' x B' successeur de  A x B  faire
        début
        RECHERCHE (A' x B', trouvé) ;
        Si  non trouvé  alors  INSERER (File, A' x B') ;
        fin
Jusqu' à  FILEVIDE ;
```

Commentaire : Le booléen "trouvé" fournit le résultat de la recherche-insertion.

Complexité :

Pour chaque élément A x B de L, on doit pouvoir disposer du biparti G(E-A,B). Ce graphe peut être, pour chaque A x B :

- soit recalculé à partir de G : complexité linéaire suivant la taille de G.
- soit mémorisé dans une file auxiliaire File2. Il faut alors placer en tête de l' algorithme l' instruction

INIT (File2, G)

et dans le cas "non trouvé" ajouter l' instruction

INSERER (File2, G(E-A',B'))

G(E-A',B') est alors calculé à partir de G(E-A,B) en une complexité linéaire suivant la taille de ce graphe. Comme souvent dans ce genre de problème, un gain sur le nombre d' opérations s' obtient au prix d' un sacrifice en place-mémoire.

La complexité du calcul de G(E-A,B) est de toute façon majorée par celle du calcul des parties maximales de ce même graphe, dont on peut donner une borne d' après le paragraphe IV : $O(|E||R|)$, ou $O(|E|^{\alpha})$ suivant la méthode utilisée. Ce calcul étant effectué $|L|$ fois, on obtient comme majorant (grossier) de complexité $O(|L||E||R|)$ ou $O(|L||E|^{\alpha})$.

Rappelons que $|L| \leqslant 2^{\min(|E|,|F|)}$, et $|R| \leqslant |E||F|$.

Un appel de la procédure RECHERCHE demande quant à lui $O(\min(|E|,|F|)|E|)$ opérations : un noeud de l' arbre possède au plus $|E|$ fils, et un chemin partant de la racine contient au plus $\min(|E|,|F|)$ noeuds.

Cet appel est effectué pour chaque arc de H, soit L x $\min(|E|,|F|)$ fois au plus.

Supposons $|E| \leqslant |F|$ (E et F jouent des rôles symétriques dans l' algorithme). La complexité due à cette partie de l' algorithme est donc $O(|L||E|^3)$.

Autres algorithmes :

Les algorithmes plus anciens ([4],[5],[9],[13]) se préoccupent essentiellement du calcul des rectangles maximaux, au moyen d' intersections ensemblistes, ou en incrémentant à chaque pas la taille d' un ensemble E initialisé à $\emptyset$. Aucun ne produit le treillis de Galois associé. De même, aucun n' est évalué ni n' utilise la structure d' ordre naturelle existant sur ces rectangles.

L' utilisation des treillis de Galois pour diverses applications, notamment en Sciences Sociales, a été développée ces dernières années par R. Wille ([14] ,[15]). Dans ce cadre, un algorithme de construction a été écrit et implémenté par B. Ganter ([7]) *

* Indications fournies par un rapporteur.

ANNEXE

Soit $A \times B \in L$, $A \subseteq E$, $B \subseteq F$. Soit γ la fonction successeur dans $G(E-A,B)$. Supposons que ce biparti contienne au moins un arc (Dans le cas contraire, on montre facilement que le seul élément de L qui couvre $A \times B$ est $E \times \emptyset$).

De plus, désignons par $B_1, B_2, \ldots, B_p$ les parties maximales de $G(E-A,B)$ et $A_i = \{x \in E-A \mid \gamma(x) = B_i\}$.

Enfin, soit K l' ensemble de produits $C \times D$, $C \subseteq E$, $D \subseteq F$, tels que $\forall e \in C$, $\forall f \in D$, $R(e,f)$. K contient des rectangles, non nécessairement maximaux, et $L \subseteq K$.

LEMME 1 : <u>$(A \cup A_i) \times B_i$ est un élément de L.</u>

Preuve : $(A \cup A_i) \times B_i \in K$, par définition de A_i.

Supposons qu' il existe $A' \times B' \in K$, avec $A \cup A_i \subseteq A'$ et $B_i \subseteq B'$.

Alors $A \times B' \in K$, et $A \times B \in L$, ce qui implique $B' \subset B$.

$A' \times B' \in K \Rightarrow \exists\, x \in A'-A$, avec $\gamma(x) \supseteq B'$. En conséquence $B_i = B'$, par définition de B_i.

Donc $\forall\, x \in A'-A \;\; \gamma(x) = B_i$ et $A'-A \subseteq A_i$. D' où $A' = A \cup A_i$.

LEMME 2 : <u>Soit $A' \times B'$ couvrant $A \times B$ dans L. Alors il existe i, $1 \leq i \leq p$ avec $B' = B_i$ et $A' = A \cup A_i$.</u>

Preuve : $A' \supset A$ et $B' \subset B$ (relation d' ordre de L).

Soit B_i partie maximale telle que $B' \subseteq B_i \subseteq B$ $(A \times B \neq E \times \emptyset)$.

$\exists\, x \in E-A$ tel que $\gamma(x) = B_i$. Si $x \notin A'$, $(A' \cup x) \times B' \in K$, ce qui est impossible. Donc $A_i = \{x \in E-A \mid \gamma(x) = B_i\} \subseteq A'$, et $A \cup A_i \subseteq A'$. D' autre part, $B_i \neq B$ puisque $(A \cup x) \times B_i \in K$.

$(A \subset A \cup A_i \subseteq A'$ et $B' \subseteq B_i \subset B) \Rightarrow (A \cup A_i = A'$ et $B_i = B')$, puisque d' après le Lemme 1, $(A \cup A_i) \times B_i \in L$.

THEOREME : L' ensemble des éléments qui couvrent A x B dans L est exactement

$$\{(A \cup A_i) \times B_i \qquad i = 1,2,\ldots,p\}$$

Preuve :

- D' après le lemme 2, tout élément couvrant A x B est de la forme $(A \cup A_i) \times B_i$.

- $(A \cup A_i) \times B_i \in L \quad \forall i$ d' après le lemme 1. De plus, ces éléments sont deux à deux incomparables dans L, ce qui achève la preuve.

BIBLIOGRAPHIE

1 - AHO A.V., HOPCROFT J.E., ULLMANN J.D., Data structures and algorithms, Reading, Addison-Wesley, 1983.

2 - BARBUT M., MONJARDET B., Ordre et Classification, Algèbre et Combinatoire, 2 tomes, Paris, Hachette, 1970.

3 - BERGE C., Graphes et hypergraphes, Paris, Dunod, 1970.

4 - CHEIN M., "Algorithme de recherche des sous-matrices premières d' une matrice", Bull. Math. Soc. Sci. Math. R. S. Roumanie, tome 13 (61)(1969), 21-25.

5 - FAY G., "An algorithm for finite Galois connections", CL & CL-Comput. Linguist. Comput. Lang. 10 (1975), 99-123.

6 - FISCHER M.J., MEYER A.R., "Boolean matrix multiplication and transitive closure", Proc. 12th Annual Symposium on Switching and Automata Theory (1971), 129-131.

7 - GANTER B., "Two basic algorithms in concept analysis", Preprint nº 831, Technische Hochschule Darmstadt (1984).

8 - HABIB M., HAMROUN M., JEGOU R.,"Linear equivalences for transitivity in graphs", Rapp. Rech. nº 83-10, E.N.S.M. Saint-Etienne (1983).

9 - HARALICK R.M., "The diclique representation and decomposition of binary relations", J. Assoc. Comput. Mach. 21 (1974), 356-366.

10 - KARP R.M., "Complexity of computer computations", Proc. Sympos., IBM Thomas J. Watson Res. Center, Yorktown heights, N. Y. (1972), 85-103.

11 - KNUTH D.E., Big Omicron and Big Omega and Big Theta, Sigact News (1976), 18-25.

12 - MUNRO I., "Efficient determination of the transitive closure of a directed graph", Information Processing letters, 1 (1971), 56-58.

13 - NORRIS E.M., "An algorithm for computing the maximal rectangles of a binary relation", Rev. Roum. Math. Pures Appl., 23, nº 2 (1978), 243-250.

14 - WILLE R. , "Restructuring lattice theory : an approach based on hierarchies of concepts" in Ordered sets (edit. I. Rival), Reidel, Dordrecht-Boston (1982), 445-470.

15 - WILLE R., "Line diagrams of hierarchical concept systems", Int. Classif. 11, 2 (1984), 77-86.