LES CAHIERS DE L'ANALYSE DES DONNÉES

J.-P. BENZÉCRI

Construction d'une classification ascendante hiérarchique par la recherche en chaîne des voisins réciproques

Les cahiers de l'analyse des données, tome 7, n° 2 (1982), p. 209-218

<http://www.numdam.org/item?id=CAD_1982__7_2_209_0>

Les Cahiers de l'Analyse des Données
Vol. VII - 1982 - n° 2 - p. 209-218

CONSTRUCTION D'UNE CLASSIFICATION ASCENDANTE HIÉRARCHIQUE PAR LA RECHERCHE EN CHAÎNE DES VOISINS RÉCIPROQUES [C.A.H. CHAÎNE RECIP.]

par J.-P. Benzécri [1]

1 *Principe de la recherche en chaîne*

L'algorithme de base de la classification hiérachique ("base") construit les noeuds un par un en recherchant sur l'ensemble des sommets la paire (sa, sb) réalisant le minimum de la quantité d(sa, sb) prise comme critère : avec cet algorithme, pour un ensemble I de n individus, la classification ne s'achève que par le calcul (ou au moins la consultation) de quelque ($n^3/6$) distances. L'algorithme des voisinages réductibles ("réduc") (dû à M. Bruynooghe et utilisé couramment par M. Jambu) vise à diminuer ce nombre en bornant les comparaisons des distances à une liste définie par un seuil supérieur , et convenablement tenue à jour à chaque création de noeud : " réduc" accélère en effet généralement la construction d'une C.A.H. , mais l'entretien de la liste est fastidieux, et le choix du seuil aléatoire (en fait ce choix se renouvelle, car en bref chaque fois que la liste se trouve vide, on doit élever le seuil), en sorte qu'on ne peut fixer de façon constante une échelle de temps (e. g. en n^2 ou n^2 Log n au lieu de n^3 pour base). L'algorithme des voisins réciproques ("récip") (dont le principe remonte à Mac Quitty(1966);et dont des réalisations furent programmées par Brisse et Granjouan (1977) et de Rham (1980).) introduit,sans compliquer l'algorithme de base,un principe d'accélération très efficace : au lieu de contruire des noeuds un par un, on repère à chaque parcours de la matrice des distances, un ensemble de paires de sommets (sa , sb) , (sa' , sb') (sa" , sb") ... qu'on peut agréger parce que , même s'il ne s'agit pas de la paire (sa , sb) réalisant le minimum du critère d(sa , sb), chaque membre d'une paire est relativement à l'autre celui qui réalise ce minimum (i.e. quel que soit s autre que sa" et sb", d(sa",sb") est inférieur à la fois à d(s,sa") et à d(s , sb"). Utilisée comme un principe d'agrégation simultanée , la notion de voisins réciproques est déjà visiblement efficace, des calculs simples sur un modèle de densité uniforme montrant qu'en moyenne la moitié (voire le tiers) des individus d'un ensemble forment des paires de voisins réciproques, en sorte qu'en un parcours de la matrice des distances on décide simultanément d'un si grand nombre d'agrégations de paires que le nombre CARDSOM des sommets en cours de traitement se trouve réduit à (3/4) CARDSOM (voire (2/3) CARDSOM).

Il existe cependant un modèle d'ensemble I de points en chaîne sur un axe pour lequel à chaque lecture du tableau de distances , il n'apparaît qu'une paire de voisins réciproques : il suffit de poser:

$$I = \{3^{-i} \mid i = 1, \ldots, n\}$$

---

(1) *Professeur de statistique. Université Pierre et Marie Curie.*

On voit en bref que chaque i a pour voisin le plus proche (i + 1) et est le plus proche voisin de (i - 1) en sorte que seule la paire ultime (n, n - 1) satisfait à la condition de réciprocité. (cf. *CHAINE COMP. C.A.H.*).

Mais surtout tant que subsistent le principe de la recherche et de la création simultanée de plusieurs noeuds, il est difficile de se former un modèle assez précis pour calculer les bornes de durée qui vaillent uniformément, quels que soient les aléas des données.

En réalité l'efficacité du principe des voisins réciproques n'est pas liée à la création simultanée de plusieurs noeuds, mais plus exactement, à ce que les paires de voisins réciproques peuvent être agrégées au fur et à mesure qu'on les découvre, sans s'astreindre à créer les noeuds dans l'ordre croissant de leurs niveaux : la seule contrainte étant que (conformément au principe ascendant de la C.A.H.) un noeud n ne peut être créé qu'après ses deux dépendants a(n) et b(n). Dès lors le problème se pose de concevoir au mieux la recherche des paires de voisins réciproques. On propose ici une recherche en chaîne qui d'une part comme l'a suggéré C. de Rham, évite de rechercher le plus proche voisin v(s) d'un sommet s, si un v(s) déterminé antérieurement subsiste, et d'autre part n'opère jamais que sur une seule chaîne (s, v(s), v(v(s)), v(v(v(s))),...) laquelle aboutit nécessairement à une paire de voisins réciproques qu'on agrège ; d'où un raccourcissement de la chaîne, que l'on préserve pourtant (si sa longueur dépassait 2) pour rechercher une paire de voisins réciproques dans le nouvel état de l'ensemble des sommets. Ainsi il apparaît que le coût de la création d'un noeud est essentiellement celui de la création de 3 maillons de chaîne (parfois seulement 2). Or le coût de la création d'un maillon (i.e. de la recherche du n-ième plus proche voisin...) étant de traiter autant de distances qu'il reste de sommets (plus précisément des sommets en dehors de la chaîne...) on voit que ce coût est borné par kn (où n = Card I et k dépend de la complexité des calculs de distance, mais non de Card I) ; en sorte que le coût global de la création de la hiérarchie, admet une borne de l'ordre de n^2 valant, quels que soient les aléas du parcours ; et en particulier pour le modèle en chaîne évoqué plus haut.

Les idées en jeu ici sont toutes simples : la difficulté principale ayant été de les découvrir puis de les conjuguer. Le spécialiste de la classification ascendante hiérarchique n'a sans doute pas de peine à concevoir un algorithme d'après les considérations qui précèdent. Pourtant entraînés de développement en développement assez loin de l'algorithme de base et de ses bases théoriques désormais classiques, il nous paraît préférable de reprendre un exposé général (§ 2) qui ne laisse aucun doute sur la validité des contructions nouvelles ; validité d'ailleurs impliquée par le succès des traitements sur machine. Après l'exposé théorique du § 2 viendra l'algorithme (§ 3).

2 *Construction d'une C.A.H. par une suite croissante d'arbres compatibles avec son critère*

2.1 L'axiome de la médiane : Tout repose sur le caractère local de la relation de voisin réciproque, i.e. sur le fait que cette relation n'est détruite par une agrégation effectuée ailleurs. Le caractère local résulte de l'axiome suivant. Si on suppose définie une notion d de "distance" ou niveau d'agrégation d(a,b) entre parties

finies, l'axiome relatif à 3 parties n, b, s deux à deux d'intersection vide, s'énonce :

$$d(a,b) \leq \inf\{d(a,s), d(b,s)\}$$

$$\Rightarrow \inf\{d(a,s), d(b,s)\} \leq d(a \cup b, s)$$

On peut dire que (a,b,s) sont les trois sommets d'un triangle dont (a,b) est le plus petit côté ; que (a ∪ b) sert de milieu à (a b); et (a ∪ b,s) une médiane. L'axiome s'énonce alors : la médiane opposée au plus petit côté est supérieure en longueur au plus court des autres côtés ; énoncé manifestement faux en géométrie euclidienne ... , mais qui suggère un nom pour l'axiome.

L'axiome (dont l'importance apparaît dans M. Bruynooghe cf. *Cahier* Vol III n°1 pp 9,10, se vérifie pour les principaux critères d'agrégation dont on rappelle ci-dessous les formules :

critère du saut : $d(a,b) = \inf\{d(i,j) \mid i \in a,\ j \in b\}$

c. du diamètre : $d(a,b) = \sup\{d(i,j) \mid i \in a,\ j \in b\}$

distance moyenne : $d(a,b) = \Sigma\{d(i,j)\ mi.mj/ma.mb) \mid i \in I,\ j \in J\}$,

où on a noté mi, mj les masses des éléments et ma, mb les masses totales des parties

moment d'inertie : $d(a,b) = ((ma.mb)/(ma+mb))\ (d(ga,gb))^2$,

où on a noté ga, gb les centres de gravité des parties a et b.

L'axiome vaut aussi pour le critère de D. Domengès (cf. [*C.A.H. FLUX*] ce cahier, pp 169-172.

2.2 *Caractère local de la relation de voisins réciproques* : Soit S un ensemble de parties de l'ensemble initial I ; pouvant être l'ensemble des sommets d'un arbre non connexe A qui est une étape de la construction d'une C.A.H. sur I. Notons v(s) un plus proche voisin dans S d'un sommet s de S : i.e.

$$\forall\ s' \in S-\{s\} : d(s,s') \geq d(s,v(s))$$

alors v(s) reste le plus proche voisin de s si on modifie S par l'agrégation d'une paire (a,b) de voisins réciproques. De façon précise :

soit : a,b,s,v ∈ S quatre sommets distincts ;

$v(a) = b$; $v(b) = a$; $v(s) = v$, i.e.

$\forall\ s' \in S : d(s,v(s)) \leq d(s,s')$;

$S' = S-\{a,b\} \cup \{a \cup b\}$ (on suprime a et b, on crée a ∪ b) ;

alors : $\forall\ s' \in S' : d(s',v(s)) \leq d(s,s')$

La vérification est immédiate : elle porte seulement sur s' = a ∪ b , et s'identifie à l'énoncé de l'axiome de la médiane pour le triangle s, a, b.

Si, en particulier s et v sont voisins réciproques (i.e. v = v(s) et s = v(v)) , ils le restent après agrégation de a et b.

2.3 *Arbre compatible avec d* : Par l'algorithme de base, (et aussi avec "réduc", et "récip" sous sa forme la plus simple) on construit des états successifs de l'arbre, qui relativement à la hiérarchie binaire complète (au sommet de laquelle est l'ensemble I lui-même, considéré

comme le noeud supérieur) peuvent être définis par la suppression des noeuds dont le niveau dépasse un seuil x. Présentement on désire bénéficier d'une liste plus grande dans l'ordre de création des noeuds. Nous définissons donc la notion d'arbre binaire sur I compatible avec le critère d ; la construction s'effectuera par une suite croissante d'arbres compatibles aboutissant à un arbre convexe (à un seul sommet) qui réalise la C.A.H. cherchée.

Définition : Soit A un arbre binaire sur I admettant comme ensemble de terminaux les parties à un élément i.e. $\{\{i\} | i \in I\}$; on définit par récurrence la notion d'arbre compatible avec d par les deux conditions suivantes :

1°) $A = \{\{i\} | i \in I\}$ est un arbre compatible (cet arbre constitue l'état initial de la C.A.H.).

2°) A est compatible s'il existe un sommet s de A ($s \in SOM(A)$) dont les deux successeurs a(s) et b(s) sont voisins réciproques relativement à $SOM(A-\{s\})$; et $A-\{s\}$ est lui-même un arbre compatible (i.e. on peut donc considérer A comme provenant d'un état antérieur $A-\{s\}$, par agrégation de deux sommets qui sont v. réciproques).

L'intérêt de cette définition est que l'arbre compatible A peut avoir été obtenu par adjonction de l'un quelconque de ses sommets à un état antérieur également compatible.

Proposition : Si A est compatible, il en est de même de $A-\{s\}$, quel que soit le sommet s de A ; et a(s) et b(s) sont voisins réciproques au sein de $SOM(A-\{s\})$.

On démontre la proposition par récurrence sur le nombre N de noeuds de A. Si N = 1, la p. résulte immédiatement de la définition même : l'unique sommet s est réunion de deux individus a(s) et b(s) qui sont voisins réciproques dans I. Soit N quelconque supposons la proposition démontrée jusqu'au rang N-1 ; et soit A un arbre compatible ayant N noeuds. Il existe par définition un noeud s' tel que A-s' (nous omettons désormais les accolades autour de s, s',...,) soit compatible et a(s') et b(s') sont v.r. au sein de SOM(A-s') ; d'autre part, quel que soit $s \in SOM(A)$, a(s) et b(s) sont par hypothèse de récurrence des voisins réciproques au sein de SOM(A-s'-s) (car A-s' a N-1 noeuds). Dès lors de par l'axiome a(s) et b(s) restent voisins réciproques si on agrège a(s') et b(s') : c'est-à-dire qu'ils sont v.r. au sein de SOM(A-s) ; ce qu'il fallait démontrer.

2.4 Recherche des sommets par une chaîne de plus proches voisins :

La proposition du § 2.3 permet de montrer qu'un arbre obtenu à partir de I par Card I-1 agrégations successives (de deux plus proches voisins d'entre les sommets de l'état antérieur de l'arbre) aboutit au résultat désiré . Reste à guider la recherche des paires de voisins réciproques. On utilisera pour cela les chaînes de plus proches voisins.

Définition : Soit A un arbre binaire sur I compatible avec d , une suite ordonnée s(1),..., s(p) de sommets distincts de A est appelée une chaîne de plus proche voisins de longueur p si on a :

$$\forall\, q \in \{2,\ldots, p\} : \quad s(q) = v(s(q-1)) \ ,$$

i.e. si au sein de SOM(A) chaque sommet de la chaîne est un plus proche voisin de son prédécesseur ($\forall\, s \in A - s(q-1) : d(s(q-1), s(q)) \leq d(s(q-1), s)$). Nous dirons "un" plus proche voisin et non "le" plus proche voisin, car on peut ici faire abstraction de l'unicité du plus proche voisin, quitte à admettre plusieurs solutions pour la C.A.H.

elle-même que l'on construit.

Proposition : Toute chaîne de plus proches voisins, convenablement prolongée, aboutit à une paire de voisins réciproques. Pour prolonger la chaîne s(1),..., s(p) il faut déterminer s(p + 1) = v(s(p)), le plus proche voisin de s(p) au sein de SOM(A) : il est clair d'abord que v(s(p)) n'est pas à chercher parmi les (p - 2) premiers s(q): car on a

$$d(s(p), s(p-1)) \leq d(s(q), s(q+1)) \leq d(s(q), s(p))$$

la deuxième de ces inégalités résulte de ce que s(q + 1) est un plus proche voisin pour s(q) ; et la première, de ce que dans une chaîne de plus proche voisins la longueur des maillons successifs ne peut que décroître, i.e. :

$$d(s(q+1), s(q+2)) \leq d(s(q+1), s(q))$$

parce que s(q + 2) = v(s(q + 1)). Ceci posé, on voit qu'éventuellement en épuisant SOM(A) par une chaîne de longueur p= Card(SOM(A)), on aboutit à une paire (s(p), s(p - 1)) de voisins réciproques.

2.5 *Schéma et coût de l'algorithme de recherche en chaîne* : Initialement A0 = I (plus exactement A est l'ensemble des parties de I formées d'un seul élément) ; et SOM(A) = A0 et on part d'un élément quelconque il = s(1) et on construit une chaîne qui aboutit à une paire de v.r. (s(p),s(p - 1)) qu'on agrège d'où un arbre A1 : tel que si on note n1 = s(p) ∪ s(p - 1)

$$A1 = A0 \cup \{n1\} \; ; \; SOM(A1) = A0 - \{s(p), s(p-1)\} \cup \{n1\}.$$

Si p = 2 (i.e. si l'élément il dont on est parti admet un voisin réciproque), la chaîne est détruite par l'agrégation de ses deux éléments : on doit donc choisir dans A1 un sommet quelconque s(1) à partir duquel on construit une nouvelle chaîne aboutissant nécessairement à une paire de v.r. dont l'agrégation fournira le noeud n2.

Si p > 2 l'agrégation de s(p) et s(p - 1) laisse subsister une chaîne de longueur (p - 2)

$$s(1), s(2), s(3), \ldots, s(p-3), s(p-2),$$

il s'agit bien d'une chaîne au sens défini au § 4, car d'après le § 2.2, l'agrégation de s(p - 1) et s(p) ne détruit pas la relation s(q) = v(s(q - 1)) pour q = 2,..., p - 2. On reprendra donc la recherche d'une paire de p.p.v. en prolongeant cette chaîne : éventuellement s(p - 3) et s(p - 2) peuvent être devenus v. r. du fait de l'absorption de s(p - 1) au sein du nouveau sommet s(p - 1) ∪ s(p). C'est en particulier ce qui se produit nécessairement si la chaîne créée a épuisé l'ensemble des sommets, comme c'est le cas pour l'exemple de progression géométrique considéré au § 1.

Après création du second noeud n2, l'algorithme se poursuit en n'opérant jamais que sur une seule chaîne, éventuellement recréée à partir d'un sommet quelconque, si elle a été épuisée.

On voit que le coût de la création d'un noeud s(p) ∪ s(p - 1) pour p > 2, équivaut à 3 recherches de p.p.v. : celles de v(s(p-2))=s(p-1) v(s(p - 1)) = s(p) et v(s(p)), qui nous ramène à s(p - 1). Si p = 2 on a seulement deux recherches v(s(1)) = s(2) et v(s(2))=s(1). Quant à la recherche de v(s(p)) son coût est proportionnel à Card SOM(A) - p (nombre des sommets à envisager). Pour la création du r-ème noeud (si Card I = n) le coût est donc

$C(r) \leq 3(n-r)k$ (k dépendant non de n mais de la formule de distance).

En tout état de cause le coût total admet une majoration en $K n^2$ (K ne dépendant pas de n).

3 *L'algorithme de C.A.H. par recherche en chaîne des paires de voisins réciproques*

Nous écrivons cet algorithme en apportant le minimum de modifications à l'algorithme de C.A.H. du traité TI B n° 4 § 1.2. Cet algorithme opère à partir d'un tableau de distances. Il est particulièrement intéressant d'appliquer la recherche en chaîne dans le cas où l'on part d'un tableau de description I × J de l'ensemble I : soit tableau de correspondances, soit tableau de coordonnées sur les axes factoriels etc. . La structure de l'algorithme est la même dans les deux cas : seules varient les procédures de permutation des sommets, et d'agrégations : celles-ci étant d'ailleurs plus simples dans le cas d'un tableau de description que dans celui d'un tableau de distance.

```
entier CARDI, CARDSOM, S, SP, P1, P2, KI;
réel INF, L1, L2, D1, D2;
entier tableau A,B[CARDI +1 : 2*CARDI -1],P[1:2*CARDI-1],
    [SOM 1:CARDI];
réel tableau L,D[1:2*CARDI-1],DIS[1:(CARDI*(CARDI-1)/2)];
```

Commentaire : CARDI est le cardinal de l'ensemble I sur lequel on construit une arborescence binaire. L'arbre achevé doit comprendre 2*CARDI-1 éléments : d'une part les éléments terminaux (ou parties à un élément de I), numérotés de 1 à CARDI ; d'autre part les CARDI-1 noeuds numérotés de CARDI+1 à 2*CARDI-1, au fur et à mesure de leur création. L'entier CARDSOM donne le nombre des sommets de l'arbre en cours de construction. CARDSOM varie de CARDI à 1, la différence CARDI -CARDSOM est le nombre de noeuds déjà créés. Les entiers S,SP,P1,P2,KI et les réels INF, L1,L2,D1,D2 sont des grandeurs intermédiaires dont le rôle apparaîtra au cours du programme. Pour déduire l'organisation hiérarchique de l'arbre, on note pour chaque noeud N (N est donc un entier variant de CARDI +1 à 2*CARDI -1), les numéros de ses deux successeurs immédiats A[N]et B[N], qu'on peut appeler fils aîné et benjamin (si on lit l'arbre comme une généalogie...) : le choix de qui est aîné qui est benjamin est fait de manière contingente par l'algorithme, tout ce qui nous intéresse est la paire {A[N], B[N]} de deux entiers positifs et inférieurs strictement à N. Le tableau P reçoit les poids de l'éléments de l'arbre, i.e. le cardinal de la partie correspondante de I : e.g. on a P[2CARDI -1] = CARDI ; et, si $T \leq$ CARDI, P[T] = 1 ; cette dernière égalité dispenserait de tabuler P de 1 à CARDI, mais on a préféré noter une suite de 1 pour que P[N] soit toujours connu par consultation d'une table, quel que soit N. Le tableau D reçoit l'indice de diamètre des éléments de l'arbre : l'indice D[T] est nul pour T = 1,..., CARDI : mais on note ces zéros, comme les P[T]=1 ;l'indice D calculé peut présenter des paliers ou même des inversions (D[N] $<$ D[A[N]]) : ces anomalies ne disparaîtront que lors du tracé de l'arbre, par suppression de certains noeuds : mais on le sait (cf. [D. M. Cl.]) il est avantageux de noter toujours un arbre binaire, quitte à déterminer quels noeuds sont à supprimer, plutôt que de noter la description hétérogène d'un arbre non binaire. Les tableaux A, B, P, D (et aussi L, cf. *infra*) qui certes ne s'emplissent qu'au cours du calcul, reçoivent du moins toujours des informations de caractère définitif, qui sont justement les résultats obtenus par le programme ; aussi ces informations sont-elles indicées par les numéros définitifs attribués aux noeuds. Au contraire, les tableaux DIS, SOM contiennent des informations constamment modifiées, relatives aux sommets de l'arbre , qui sont à tout

instant numérotés de 1 à CARDSOM, et dont l'ensemble varie sans cesse. On lit sur le tableau SOM le numéro SOM[S] de l'élément de l'arbre qu'est le sommet numéroté (présentement) S : si SOM[S] est compris entre 1 et CARDI, le S-ième sommet est un terminal (i.e. une partie à un élément de l'ensemble I) ; au-delà on a un noeud (de poids P[SOM[S]] supérieur à 1) ; au début du calcul, tous les sommets sont aussi terminaux : on a SOM[S] = S, quel que soit S de 1 à CARDI = CARDSOM. Les distances mutuelles des sommets sont contenues dans le tableau DIS : ce tableau est fait à partir du triangle inférieur (i.e. { (V,U)| {U $<_s$ V}) du tableau carré symétrique CARDSOM*CARDSOM, des distances d(V,U), tableau dont on recopie les lignes les unes après les autres: ainsi DIS[1] = d(2,1), DIS[2] = d(3,1), DIS[3] = d(3,2), DIS[4] = d(4,1), DIS[5] = d(4,2), DIS[6] = d(4,3), etc. ; et, plus généralement d(U,V) (pour U $<_s$ V), se trouve dans la case (U + ((V-1)(V-2)/2)) du tableau DIS (cf. *infra*, procédure K). Les distances contenues dans le tableau DIS ne reflétant jamais qu'imparfaitement les proximités qui existent réellement dans la nature, on peut, si on le juge commode, imposer que la distance ne prenne qu'un nombre fini (e.g. de 10 à 20) de valeurs distinctes : toutefois nous ne le ferons pas ici. Initialement le tableau DIS contient les distances mutuelles des points de l'ensemble I. Par la suite, dans le cours du calcul, CARDSOM décroît : seule une partie du tableau DIS (de 1 à CARDSOM(CARDSOM-1)/2) , contient des distances utiles : ainsi la fin du tableau se trouve-t-elle disponible pour qu'on y range les états successifs du tableau SOM: ces états, qui sont des coupes horizontales successives de l'arbre A (sommets du sous arbre non connexe A_x...) peuvent être utiles, e.g., pour comparer deux arbres A construits sur un même ensemble de base I. Enfin le tableau L contient sur la structure des sommets des informations internes latentes complémentaires qui peuvent être utiles pour mettre à jour le tableau DIS, i.e. pour calculer les distances d'un nouveau noeud (créé par réunion de deux anciens) aux noeuds subsistants. Nous verrons plus loin que, dans certains cas , la mise à jour ne fait pas usage d'une dimension interne L ; on pourrait au contraire utiliser plusieurs paramètres analogues à L : nous ne le ferons pas ici. Dans les applications, on peut soit poser que L = 1 pour une classe réduite à un point, soit lire les valeurs de L pour une telle classe, etc. (cf. *infra* procédure LI).

```
entier procédure K(U,V);entier U,V;début entier K;
K=inf(U,V)+((sup(U,V)-1)*(sup(U,V)-2)/2);K(U,V):=K fin;
```

Commentaire : La procédure K donne le rang de la paire non ordonnée {U,V} (où U ≠ V), dans la suite, ordonnée lexicographiquement des paires V U pour lesquelles V $<_s$ U ; on a donc (cf. *supra*): K(2,1) = K(1,2) = 1 ; K(3,1) = K(1,3) = 2 ; K(3,2) = K(2,3) = 3 ; K(4,1) = K(1,4) = 4 ; K(4,2) = K(2,4) = 5 ; K(4,3) = K(3,4) = 6, etc. . Les fonctions sup et inf sont utilisées ici, comme de coutume, pour désigner respectivement le plus grand et le plus petit des deux nombres :

```
réel procédure LI(I); entier I;début LI(I) := ...fin;
```

Commentaire : Cette procédure est soit un ordre de lecture de la dimension interne des éléments de l'ensemble I étudié, soit un ordre de mise à la valeur initiale 1 pour le tableau L : L(I):= 1, etc. . Nous ne donnons pas le corps de cette procédure :

```
réel procédure LINT(P1,L1,D1,P2,L2,D2,DIS12);
entier P1,P2;réel L1,L2,D1,D2,DIS12;début LINT:=...fin;
```

Commentaire : Cette procédure calcule en fonction des poids, dimensions internes et indices des classes S1 et S2, ainsi que leur distance D12, la dimension interne LINT du sommet créé par réunion de S1 et S2.

```
réel procédure;
DISTANCE(P1,L1,D1,P2,L2,D2,DIS12,PS,LS,DS,DIS1S,DIS2S);
entier P1,P2,PS;réel L1,L2,LS,D1,D2,DS,DIS12,DIS1S,DIS2S;
début DISTANCE:=...fin;
```

Commentaire:Supposons que l'on crée un sommet par réunion de deux sommets S1 et S2 dont les poids sont P1, P2, les dimensions internes (latentes) L1, L2, les indices D1, D2, DIS12 étant la distance entre les deux sommets réunis. Soit S un troisème sommet de poids PS, dimension interne LS, indice (ou diamètre) DS. La procédure calcule la distance du sommet S au nouveau sommet créé par réunion de S1 et S2.

Nous complétons maintenant les déclarations et les procédures du programme de C.A.H. en vue de bénéficier de la recherche en chaîne des voisins réciproques :

```
entier SX ; entier tableau AA,BB[CARDI+1:2*CARDI-1];
réel tableau DM[1 : CARDI - 1]; DD[CARDI+1:2*CARDI-1];
étiquettes ETIQ1;ETIQ2
```

Commentaire : Les tableaux AA et BB conservent les numéros des aînés et benjamins lors du tri final. DD sert aussi au moment du tri. DM(I) est la "distance" minimale de I à son plus proche voisin et ETIQ1, ETIQ2 repèrent respectivement les instructions de recherche d'un voisin et celles concernant l'agrégation de deux individus.

```
Procédure PERMU(A,B);début...fin;
```

Commentaire Cette procédure a pour effet de permuter le contenu des deux cases A et B (e.g. en utilisant une case auxiliaire X, avec : X:=A, B:=A ; A:=X). Nous ne la préciserons pas, et ne distinguerons même pas les cas A, B réels et A, B entiers.

```
procédure TRI(RA,RN,D,NA,NB);
     entier NA,NB; entier tableau RA,RN[NA:NB];
     réel tableau D[NA:NB];début...fin;
```

Commentaire : Afin de ranger dans l'ordre croissant les valeurs de D, on construit deux permutations RN et RA de [NA:NB], inverses l'une de l'autre et caractérisées de façon précise par les propriétés suivantes :

$$\forall\ N \in [NA:NB] : RN[RA[N]] = N\ ;$$
$$D[RA[NA]] \leq D[RA[NA+1]] \leq \ldots \leq D[RA[NB]]\ .$$

Nous ne préciserons pas le corps de cette procédure.

```
lire CARDI ; lire tableau DIS;
pour I:=1 pas 1 jusqu'à CARDI faire début
     SOM[I]:=I;P[I]:=1 ; L[I]:=LI(I)fin;
N:=CARDI;SP:=0;
ETIQ1;
SP:=SP+1
```

Commentaire : On passe à ETIQ1 chaque fois qu'on doit allonger (ou recréer) la chaîne des P.P.V. (plus proches voisins) par laquelle on cherche les paires de voisins réciproques. SP est la longueur de cette chaîne ; N le numéro du dernier noeud créé.

```
Si CARDSOM=SP+1 alors aller à ETIQ2
Sinon début INF:=∞;S:=SP-1
     Si SP > 1 alors INF:=DM(S)
     Pour SX:=SP+1 pas 1 jusqu'à CARDSOM faire :
          Si DIS(K(SP,SX)) < INF alors début:
             S:=SX ; INF:=DIS(K(SP,SX)) fin ;
     fin ;
```

Commentaire : On cherche un P.P.V. de SP parmi les élément numérotés SP+1, SP+2,..., CARDSOM sauf si SP+1=CARDSOM auquel cas SP et CARDSOM sont nécessairement voisins réciproques et on doit les agréger en allant à ETIQ2. La chaîne est à reconstruire si SP:=1. La valeur de INF est alors initialisée à ∞ (sinon cette valeur vaut DM(SP-1)). En fin d'étape, S est, un P.P.V. de SP et INF la distance les séparant.

```
Si S =SP-1 alors aller à ETIQ2
Sinon début : DM(SP):=INF
     Si S=SP+1 alors aller à ETIQ1
     Sinon début : PERMU(SOM(S),SOM(SP+1))
          Pour SX:=1 pas 1 jusqu'à CARDSOM faire
          Si SX≠S et SX≠SP+1 alors PERMU(DIS(K(SX,SP+1)),DIS(K(SX,S)))
          fin;
     aller à ETIQ1 fin;
```

Commentaire : Le P.P.V. de SP n'est pas SP-1. (En particulier si SP =1 ce sera toujours le cas) mais S. On doit donc placer S en SP+1 pour former la chaîne de P.P.V. . Pour cela on permute les numéros S et SP+1 sauf si S vaut SP+1. On a noté DM(SP) la distance de SP à son P.P.V. on revient ensuite à ETIQ1 pour allonger la chaîne.

```
ETIQ2;
N:=N+1;
A[N]:=SOM[S]; B[N]:=SOM[SP] ; D[N]:=INF;
P1:=P[A[N]] ; P2:=P[B[N]];    P[N]:=P1+P2;
D1:=D[A[N]] ; D2:=D[B[N]];
L1:=L[A[N]] ; L2:=L[B[N]];
L[N]:=LINT(P1,L1,D1,P2,L2,D2,INF);
SOM[S]:=N ; SOM[SP]:=SOM[CARDSOM];
```

Commentaire : On note les successeurs A et B, et le diamètre (niveau d'agrégation) D du noeud nouvellement créé ; on calcule son poids P comme somme des poids de ses successeurs immédiats. Les paramètres P, L, D des successeurs A[N], B[N] sont notés en P1, L1, D1, P2,L2,D2, pour le calcul de L[N] et (cf. *infra*) la mise à jour du tableau DIS. Le noeud N nouvellement créé reçoit en tant que sommet le n° S qui était celui de A[N] ; le n° SP, du noeud B[N] qui n'est plus sommet, se trouve libre ; on l'attribuera au dernier sommet dont le numéro était CARDSOM (toutefois, si SP=CARDSOM cette opération équivaut à ne rien faire...).

```
Pour X:=1 pas 1 jusqu'à S-1, S+1 pas 1 jusqu'à SP-1,
     SP+1 pas 1 jusqu'à CARDSOM faire
        DIS[K(S,X)]:=DISTANCE(P1,L1,D1,
           P2,L2,D2,DIS12,P[SOM[X]],L[SOM[X]],
           D[SOM[X]],DIS[K(S,X)],DIS[(SP,X)]);
```

Commentaire : Pour tous les sommets dont le rang X était distinct de S et de SP, on calcule la distance au niveau du sommet créé, qui est le noeud N, réunion des sommets antérieurement numérotés S et SP. Puisque le noeud N est pris comme sommet de rang S, les distances nouvellement calculées sont inscrites dans le tableau DIS aux places K(S,X)

```
pour X:=1 pas 1 jusqu'à SP-1,
     SP+1 jusqu'à CARDSOM-1 faire
        DIS[K(X,SP)]:=DIS[K(X,CARDSOM)];
```

Commentaire : Puisque le sommet antérieurement numéroté CARDSOM reçoit le numéro SP laissé libre par la fusion de A[N] et B[N] en N, il faut transférer aux adresses K(X,SP) les distances antérieurement inscrites, en K(X,CARDSOM). (Si SP=CARDSOM, cf. *supra* , l'opération est inutile).

```
CARDSOM:=CARDSOM-1;
SP:=SUP(0,SP-3)
Si CARDSOM ≥ 2 aller en ETIQ1;
```

Commentaire : Le noeud a été créé par agrégation de SP et SP-1. On diminue de 1 le nombre des sommets. Le numéro de l'élément en haut de chaîne est alors soit 0 (si SP=2) soit SP-3 (si SP < 2) car dans le cas où il reste plus de 1 sommet on revient à ETIQ1 et donc à l'instruction SP:=SP+1.

Si CARDSOM=1, la construction de la hiérarchie est achevée ; à ceci près que les noeuds ayant été créés dans un ordre qui n'est pas celui des niveaux croissants, il faut les renuméroter ; ce qu'on fera, par exemple, comme suit :

```
TRI(RA,RN,D,CARDI+1,2 CARDI-1);
pour N:=CARDI+1 pas 1 jusqu'à 2 CARDI-1 faire début
     AA[N]:=A[N];BB[N]:=B[N];DD[N]:=D[N]fin;
```

Commentaire : On recopie les tableaux A, B, D avant de les recoder suivant le nouveau numérotage des noeuds.

```
pour N:=CARDI+1 pas 1 jusqu'à 2*CARDI-1 faire début
     D[N]:=DD[RA[N]];
     A[N]:=AA[RA[N]];B[N]:=BB[RA[N]];
     si CARDI ≤ A[N] alors A[N]:=RN[A[N]];
     si CARDI ≤ B[N] alors B[N]:=RN[B[N]]fin ;
```

Commentaire : Pour suivre ce calcul il suffit de se souvenir que RA exprime le rang ancien du noeud en fonction de son rang nouveau ; tandis que RN fait l'inverse ; de plus les individus gardent leur numérotage de 1 à CARDI.