RAIRO. RECHERCHE OPÉRATIONNELLE

A. GUÉNOCHE

Représentations arborées des classifications

RAIRO. Recherche opérationnelle, tome 20, n° 4 (1986), p. 341-353

<http://www.numdam.org/item?id=RO_1986__20_4_341_0>

R.A.I.R.O. Recherche opérationnelle/Operations Research
(vol. 20, n° 4, novembre 1986, p. 341 à 354)

REPRÉSENTATIONS ARBORÉES DES CLASSIFICATIONS (*)

par A. GUÉNOCHE (²)

Résumé. — *Dans ce texte nous présentons un algorithme de calcul des longueurs des arêtes d'un arbre de classification correspondant à une hiérarchie de partitions et deux algorithmes de tracé des arbres ainsi valués :*

— Tracé radial, dans lequel on définit un ordre circulaire des sommets pendants et la direction de chaque arête. Puis à partir de la racine on distribue les sommets adjacents, à une distance égale à la longueur de l'arête, et ainsi de suite en partant des sommets placés.

— Tracé arborescent, dans lequel on plante l'arbre en un sommet, ce qui définit une arborescence, et l'on place plus bas les sommets adjacents, à une distance verticale égale à la longueur de l'arête.

Ces algorithmes de tracé sont de complexité linéaire, une fois que l'on a construit un codage approprié de l'arbre.

Mots clés : Analyse de données; Arbres additifs; Tracé d'arbres.

Abstract. — *In this paper, we present an algorithm to approximate lenghs of the edges of a classification tree, and two algorithms of drawing of the resulting valuated tree:*

— Radial drawing, in which we define a circular order of the terminals nodes and the directions of the edges. Then we plot the root and distribute adjacent vertex at a distance equal to the lengh of the edge, and so on from any ploted vertex

— Arborescent drawing, in which we plant the tree, to define an arborescence and we plot lower adjacent vertex at a vertical distance equal to the lengh of the edge.

The drawing algorithms have a linear complexity, when an appropriate coding for the tree is built.

Keywords: Data analysis; Additive tree; Tree embeding.

Cet article est une contribution au problème de la représentation par un arbre valué, d'un tableau de dissimilarités dans lequel sont quantifiés des écarts entre les N items (objets, documents, attributs descriptifs, . . .) d'un ensemble E. Ce problème est abordé de deux façons dans le cadre de l'analyse des données :

(*) Reçu juillet 1985.
(²) G.R.T.C.-C.N.R.S., 31, chemin-J.-Aiguier 13009 Marseille.

— Les méthodes de classification déterminent une hiérarchie indicée de partitions de E, en approximant les dissimilarités par une distance ultramétrique (D.U.).

U est une D.U. ssi

$$\forall i, j, k \in E, \quad U(i, j) \leqq \max(U(i, k), U(j, k)).$$

Ce problème est posé en ces termes par certains auteurs [3], mais toute méthode de classification ascendante hiérarchique permet de construire cette ultramétrique. Elles définissent un arbre hiérarchique que l'on représente par un dendrogramme, c'est-à-dire une arborescence dont la racine est la partition la moins fine constituée d'une seule classe et les sommets pendants sont les éléments de E. Du fait qu'il s'agit de la représentation d'une ultramétrique, les distances de chaque élément de E à la racine sont égales.

— Les méthodes de construction d'un arbre additif [1, 2, 6, 9] réalisent une approximation des dissimilarités par une distance additive d'arbre (D.A.D.A.) encore appelée distance quadrangulaire.

Q est une D.A.D.A. ssi

$$\forall i, j, k, l \in E, \quad Q(i, j) + Q(k, l) \leqq \max(Q(i, k) + Q(j, l), Q(i, l) + Q(j, k)).$$

Il est clair que tout quadruplet de sommets d'un arbre valué vérifie cette inégalité, si l'on définit la distance d'arbre entre sommets comme la somme des longueurs des arêtes du chemin qui les relie. Réciproquement si une dissimilarité sur E vérifie cette inégalité, on sait construire un arbre valué dont les sommets pendants sont les éléments de E et dont la distance d'arbre est égale aux dissimilarités. Si l'on impose à tout sommet intérieur d'être de degré au moins 3, cet arbre est unique.

Ces deux classes de méthodes définissent une structure d'arbre valué. Nous proposons d'améliorer ces longueurs d'arêtes en considérant les chemins dans l'arbre entre sommets terminaux. Leurs longueurs sont égales aux sommes des longueurs des arêtes qu'ils empruntent. Les valeurs des arêtes peuvent alors être calculées comme la meilleure approximation au sens des moindres carrés, des valeurs des dissimilarités initiales, ce qui définit un problème d'optimisation sous contraintes, puisque les longueurs d'arêtes sont nécessairement positives ou nulles [5, 7].

Un tracé de l'arbre ainsi valué, permet de représenter la notion de voisinage, en considérant les longueurs des chemins dans l'arbre comme une approximation des distances entre sommets.

Dans ce texte nous présentons un algorithme de calcul des longueurs des arêtes et deux algorithmes de tracé des arbres valués, dont le premier est original :

– Tracé radial, dans lequel on définit un ordre circulaire des sommets pendants et la direction de chaque arête. Puis à partir de la racine on distribue les sommets adjacents, à une distance égale à la longueur de l'arête, et ainsi de suite en partant des sommets placés.

– Tracé arborescent, dans lequel on plante l'arbre en un sommet, ce qui définit une arborescence, et l'on place plus bas les sommets adjacents, à une distance verticale égale à la longueur de l'arête.

Ces constructions ont recours à des algorithmes de cheminement dans les arbres, dont il est bien connu qu'ils sont de complexité linéaire [4]. Nos algorithmes de tracé respectent cette obligation, ils sont également de complexité linéaire une fois que l'on s'est donné un codage approprié de l'arbre. De plus ils construisent des représentations planaires (sans intersection d'arêtes) des arbres.

1. VALUATION DES ARÊTES PAR APPROXIMATION DES DISSIMILARITÉS

Une hiérarchie de partitions est un ensemble $\mathbf{H}$ de parties de E ordonné par inclusion, tel que : $E \in \mathbf{H}$, $\forall i$ on a $\{i\} \in \mathbf{H}$, $\forall C_i \in \mathbf{H}$ et $C_j \in \mathbf{H}$, on a soit $C_i \supset C_j$, soit $C_i \cap C_j = \varnothing$.

Il lui correspond une structure d'arbre. Les items que l'on classe sont les sommets pendants et les classes sont les sommets intérieurs. La partition la moins fine, constituée d'une seule classe qui contient tous les items, est la racine de l'arbre. Tous les sommets intérieurs, excepté dans certains cas la racine, sont de degré au moins 3, puisqu'ils ont au moins deux successeurs. Si les items sont numérotés de 1 à N, les sommets intérieurs seront numérotés à la suite. Si l'on a NS sommets dans l'arbre, NS est le numéro de la racine; il est au plus égal à $2N-1$.

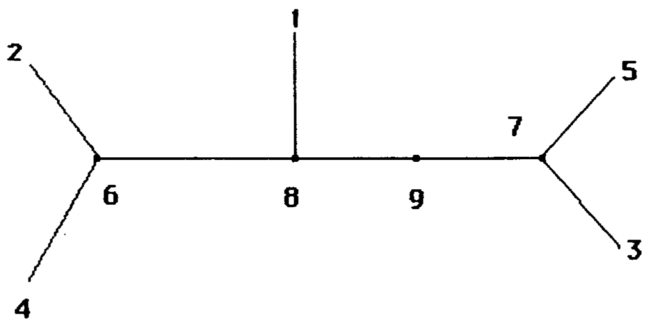
1.1 Codage de l'arbre

Ce paragraphe est plus général que le codage des arbres de classification. On suppose que l'on part d'une liste d'arêtes donnée par leurs sommets début et fin, sans qu'il y ait dans ces termes une notion d'orientation. Si l'on a NS sommets on dispose de deux tableaux à $NS-1$ places, SD et SF et la i-ième arête est $SD(I)-SF(I)$.

Pour ne pas avoir à parcourir plusieurs fois la liste des arêtes, ce qui conduirait à des algorithmes non linéaires, on va se donner un codage de l'arbre à l'aide de trois tableaux de dimension NS.

Le choix d'un sommet dans un arbre permet de considérer toutes les arêtes comme orientées vers cette racine (pour les arbres de classification, la racine est le dernier sommet correspondant à la classe E). On a alors un codage de l'arbre sous forme d'un tableau T des prédécesseurs de chaque sommet. Celui qui est choisi comme racine n'a pas de prédécesseur, on lui donnera la valeur 0, ce qui désigne la racine. On a ainsi un codage de l'arbre, par NS nombres, qui permet d'avoir un algorithme linéaire de cheminement vers la racine.

Exemple :



En choisissant dans l'arbre ci-dessus le sommet 9 comme racine, on obtient le codage suivant de l'arborescence : $T=(8\ 6\ 7\ 6\ 7\ 8\ 9\ 9\ 0)$.

Ce codage est satisfaisant pour ce sens de cheminement, mais insuffisant dans l'autre, car pour trouver les successeurs d'un sommet S on est obligé de balayer le tableau T pour trouver les sommets K tels que $T(K)=S$. On va donc définir une structure de données qui permet de descendre dans l'arborescence. On range dans un tableau SU, à la i-ième place l'un des successeurs du sommet I et l'on crée un tableau de pointeurs FR tel que $FR(I)$ est un sommet qui a même prédécesseur que I. On obtient donc les successeurs d'un sommet S en commençant par $SU(S)$, puis $FR(SU(S))$, $FR(FR(SU(S)))$, etc., jusqu'à tomber sur 0.

Une telle structure de données permet de cheminer dans l'arbre sans balayage de tableaux. Encore faut-il se ramener, de notre liste initiale d'arêtes, à ce codage par trois tableaux T, SU, FR, donc désigner une racine. S'il ne s'agit pas d'un arbre de classification, on prendra comme racine un centre de l'arbre, c'est-à-dire l'un des milieux d'un plus long chemin. Si le diamètre de l'arbre est impair, on choisit arbitrairement l'un des centres. Pour ce faire à partir d'un tableau DG des degrés des sommets, on applique l'algorithme de détermination d'un centre qui consiste à effacer les arêtes conduisant à des sommets pendants de degré 1. Cette partie de l'agorithme est détaillé dans [8].

Algorithme de codage d'un arbre

```
Tant que il existe U tel que DG(U)=1 Faire
    Trouver l'arête non marquée dont l'un des sommets est U
    Marquer cette arête : Soit V son autre sommet
    DG(U):=0 : DG(V):=DG(V)-1
    T(U):=V
    Si SU(V)=0 Alors SU(V):=U:Fin
    Sinon W:=SU(V)
      Tant que FR (W)>0 Faire
      W:=FR(W)
    FR(W):=U:Fin
```

Exemple : Pour l'arbre précédent défini par la liste d'arêtes ($SD(I)$, $SF(I)$ pour $I=1$, $NS-1$), on obtient le codage suivant (les 0 ne sont pas marqués) :

SD	SF		I		T	SU	FR
2	6	:	1	:	8		6
4	6	:	2	:	6		4
3	7	:	3	:	7		5
5	7	:	4	:	6		
1	8	:	5	:	7		
6	8	:	6	:	8	2	
7	9	:	7	:	9	3	
8	9	:	8	:	9	1	
		:	9	:		7	

Cet algorithme se termine lorsque tous les degrés sont nuls. Il n'est pas linéaire puisque l'on parcourt la liste des arêtes tant qu'il existe un sommet de degré 1.

1.2. Approximation quadratique sous contraintes

Le but de ce paragraphe est de calculer les longueurs d'arêtes $L=(l_1, l_2, \ldots, l_{NS-1})$, de façon à représenter dans l'arbre les proximités exprimées dans le tableau initial des dissimilarités. Ces longueurs sont rangées dans un tableau LG dans lequel $l_K=LG(K)$ est la longueur de l'arête $K-T(K)$.

Par définition des arbres, il n'y a qu'un seul chemin qui lie deux sommets, dont la longueur est la somme des longueurs des arêtes empruntées. En écrivant que pour chaque couple de sommets pendants, leur dissimilarité est égale à la longueur du chemin qui les lie, on obtient un système linéaire de $N(N-1)/2$ équations à $NS-1$ inconnues que l'on note :

$$A.L=D.$$

La matrice A est à coefficient 0 ou 1. Pour $N>4$ ce système a plus d'équations que d'inconnues; on le résoud par approximation, au sens des moindres carrés, entre les longueurs des chemins obtenus et les dissimilarités. Ce problème classique d'optimisation [7] a pour solution la solution de $A^T.A.L=A^T.D$ (A^T désigne la matrice transposée de A et $A^T.A$ est une matrice symétrique).

Si la racine est de degré 2, ce qui arrive souvent puisque le dernier sommet peut être la réunion de deux classes, la matrice $A^T.A$ est singulière. En effet les longueurs des arêtes NS-1 $-NS$ et NS-2 $-NS$ sont liées par la relation $l_{NS-1}+l_{NS-2}=\text{Cte}$. On posera $l_{NS-1}=0$, ce qui revient à abaisser la dimension du système d'une unité quitte à placer ultérieurement la racine au milieu de l'arête NS-2 $-NS$-1 si l'on ne veut pas que NS et NS-1 soient confondus, ou si l'on veut conserver exactement la hiérarchie de partitions.

Mais la solution de ce système n'est pas nécessairement à valeurs positives ou nulles, ce que notre modèle de représentation des dissimilarités par un arbre impose. On résoudra donc le système $A^T.A.L=A^T.D$ par la méthode de Gauss Seidel légèrement modifiée [5].

Posons $B=A^T.A$ et $G=A^T.D$. Cette méthode consiste à décomposer B en faisant apparaître ses matrices triangulaires inférieures et supérieures (diagonale non comprise).

$$B=\begin{pmatrix} b_{1,1} & b_{1,2} & & b_{1,NS-1} \\ b_{2,1} & b_{2,2} & -F & b_{2,NS-1} \\ & -E & D & \\ b_{NS-1,1} & & & b_{NS-1,NS-1} \end{pmatrix}=D-E-F$$

On applique alors l'algorithme itératif : $DL^{(k+1)}=EL^{(k+1)}+FL^{(k)}+G$ qui permet de calculer successivement les composantes de $L^{(k+1)}$, longueurs des arêtes à la $k+1$-ième itération, connaissant celles de $L^{(k)}$. La modification apportée consiste simplement à donner à $l_i^{(k+1)}$ la valeur 0 si cette composante est négative. Cette opération revient à une projection dans le cône $L > 0$, ce qui nous assure de trouver une solution satisfaisante. D'autre part on sait que la méthode de Gauss Seidel converge toujours pour une matrice symétrique définie positive, et ce d'autant plus vite que la diagonale est dominante. Ce n'est pas le cas de notre matrice B, mais les éléments diagonaux sont les maximums de leurs lignes et colonnes, condition suffisante pour assurer la convergence de cette méthode de relaxation sous contraintes positives.

Du point de vue de sa mise en œuvre, la principale difficulté est l'écriture de la matrice A. Chaque ligne est la fonction caractéristique des arêtes qui appartiennent au chemin joignant deux sommets pendants. Du fait de notre codage de l'arbre, c'est aussi une fonction caractéristique des sommets autres que la racine, placés sur ce chemin.

1.3. *Exemple* : Soit la matrice de dissimilarités :

	1	2	3	4
2 :	20			
3 :	30	30		
4 :	30	10	50	
5 :	50	40	15	60

La méthode du lien moyen (la distance inter-classes est égale à la moyenne des dissimilarités entre deux éléments de chaque classe) donne la hiérarchie de partitions suivante (les classes sont séparées par /) :

2 , 4 / 1 / 3 / 5
2 , 4 / 3 , 5 / 1
1 , 2 , 4 / 3 , 5
1 , 2 , 3 , 4 , 5

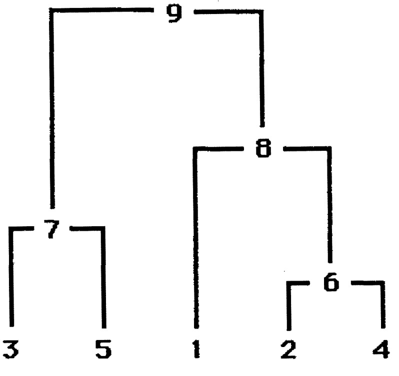


Figure 1. — Dendogramme de la hiérarchie du lien moyen.

Les équations des chemins entre sommets pendants nous donnent le système suivant :

de 1 à 2	1	1	0	0	0	1	0	0		20
1 3	1	0	1	0	0	0	1	1		30
1 4	1	0	0	1	0	1	0	0		30
1 5	1	0	0	0	1	0	1	1		50
2 3	0	1	1	0	0	1	1	1	x L =	30
2 4	0	1	0	1	0	0	0	0		10
2 5	0	1	0	0	1	1	1	1		40
3 4	0	0	1	1	0	1	1	1		50
3 5	0	0	1	0	1	0	0	0		15
4 5	0	0	0	1	1	1	1	1		60

Sa résolution par approximation quadratique donne le système de dimension 7 suivant (la racine 9 étant de degré 2, on pose temporairement $l_8=0$, pour partager ensuite la longueur calculée de l_7 entre l_7 et l_8) :

$$
\begin{pmatrix}
4 & 1 & 1 & 1 & 1 & 2 & 2 \\
1 & 4 & 1 & 1 & 1 & 3 & 2 \\
1 & 1 & 4 & 1 & 1 & 2 & 3 \\
1 & 1 & 1 & 4 & 1 & 3 & 2 \\
1 & 1 & 1 & 1 & 4 & 2 & 3 \\
2 & 3 & 2 & 3 & 2 & 6 & 4 \\
2 & 2 & 3 & 2 & 3 & 4 & 6
\end{pmatrix}
\begin{pmatrix} l_1 \\ l_2 \\ l_3 \\ l_4 \\ l_5 \\ l_6 \\ l_7 \end{pmatrix}
=
\begin{pmatrix} 130 \\ 100 \\ 125 \\ 150 \\ 165 \\ 230 \\ 260 \end{pmatrix}
$$

dont la solution sous contraintes positives (une fois que l'on a rendu égales les longueurs des deux dernières arêtes) est :

Arête		longueur
	1 – 8	9.82
	2 – 6	0
	3 – 7	.91
	4 – 6	13.34
	5 – 7	14.2
	6 – 8	8.78
	7 – 9	11.11
	8 – 9	11.11

2. TRACÉ DES ARBRES VALUÉS

Une fois que l'on a calculé la structure de l'arbre et les longueurs des arêtes, posons-nous le problème du tracé automatique de ces arbres. Plutôt que de se placer dans le système de coordonnées de l'écran graphique utilisé, ce qui est toujours fonction de l'ordinateur auquel on a accès, on se placera dans un plan virtuel sans limite. Les coordonnées des sommets étant fixées, on déterminera les extrema de la figure et on appliquera une translation et une homothétie, de façon à se ramener dans les limites de l'écran.

2.1. Tracé radial

Le principe de ce tracé est de partir de la racine que l'on pose arbitrairement en (0, 0), puis pour tout sommet placé, de calculer les coordonnées de ses successeurs. Pour ce faire, puisque l'on connait la longueur de l'arête, il suffit de déterminer pour chaque sommet S dont le prédécesseur est P, l'angle $AN(S)$ de cette arête $P \rightarrow S$ qui définit la direction du tracé. Afin d'éviter tout croisement d'arêtes, et de calculer l'angle de chaque sommet comme la moyenne des angles de ces successeurs, il faut construire un ordre sur les sommets tels que les sommets pendants soient dans un ordre compatible avec la structure planaire de l'arbre. C'est-à-dire que si un sommet P a deux successeurs S et T, les sommets du sous-arbre de S sont tous placés avant ceux du sous-arbre T.

Il suffit alors de répartir uniformément sur un cercle les sommets pendants de l'arbre, dans cet ordre, puis de calculer les angles de leurs prédécesseurs et ainsi de suite jusqu'à la racine. Tout sommet de l'arbre se trouve alors placé dans un cône défini par la racine et le premier et le dernier sommet pendants de son sous-arbre. En rangeant les sommets dans l'ordre d'un parcours de plus profonde descente [8] qui vérifie la condition requise, on ne peut avoir de croisement d'arêtes.

Algorithme d'énumération de plus profonde descente

Cet algorithme construit dans un tableau CH, d'indice LC, la liste des sommets dans l'ordre de plus profonde descente en partant de la racine R.

```
    LC := 1 : CH(1) := R : I := R
(a) Tant que SU(I)>0 Faire
            LC := LC+1 : CH(LC) := SU(I)
            I := SU(I)
(b) Si FR(I)>0 Faire
            LC := LC+1 : CH(LC) := FR(I)
            I := FR(I) : Aller en (a)
    Sinon I := T(I) : Si I>0 Aller en (b)
```

Sur l'exemple précédent, on obtient l'ordre : 9, 7, 3, 5, 8, 1, 6, 2, 4.

Calcul des angles d'incidence des sommets

On parcourt deux fois le tableau CH. La première fois dans le sens croissant des indices pour donner aux seuls sommets pendants des angles proportionnels à leurs rangs et, s'il y a NP sommets pendants, à $2\pi/NP$. La seconde fois dans le sens décroissant pour donner aux sommets intérieurs la valeur moyenne des angles de leurs successeurs.

```
K := -1 : alpha := 2Π/NP
Pour I variant de 1 à NS Faire
   J := CH(I)
   Si DG(J) = 1 Faire
   K := K+1 : AN(J) := K* alpha
   Sinon AN(J) := 0
Pour I variant de NS à 1 Faire
   J := CH(I)
   Si DG(J) > 1 Alors AN(J) := AN(J)/(DG(J)-1)
   K := T(J) : AN(K) := AN(K) + AN(J)
```

Les directions suivant lesquelles on atteint chaque sommet autre que la racine, sont portées dans la figure ci-dessous.

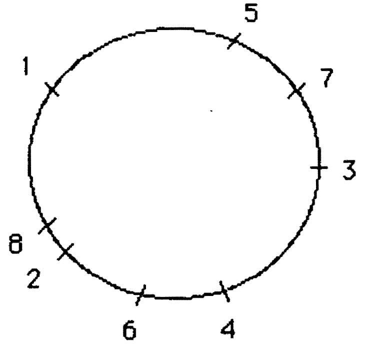


Figure 2. — Angles d'incidence des sommets.

Dès lors il suffit de parcourir la tableau CH dans l'ordre croissant des indices. On est assuré lorsque l'on veut calculer les coordonnées d'un sommet que son précécesseur est déjà placé. En effet pour une arête $U-V=T(U)$, les coordonnées (XV, YV) de V, sa longueur $LG(U)$, et l'angle de cette arête $AN(U)$, permettent de calculer les coordonnées de U.

$$XU = XV + LG(U) * COS(AN(U))$$
$$YU = YV + LG(U) * SIN(AN(U))$$

2.2. Tracé arborescent

Ce principe de tracé est inspiré à la fois par les arborescences et par les dendrogrammes. Comme pour les premières, les sommets ont un niveau égal

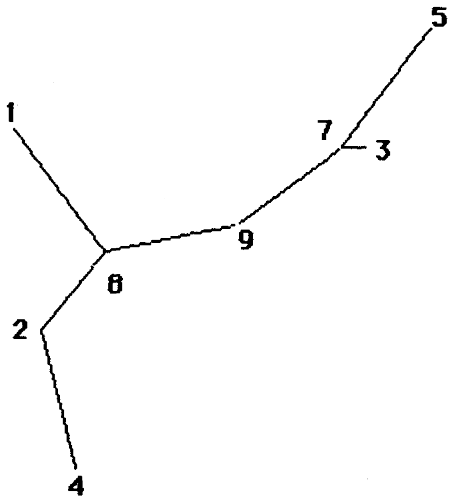


Figure 3. — Tracé radial de l'arbre de classification précédent.

à leur distance à la racine, ici le nombre d'arêtes qui les en séparent. Ceci nous conduit à réaliser une énumération des sommets dans l'ordre des distances croissantes à la racine.

Algorithme d'énumération concentrique des sommets

Cet algorithme construit dans un tableau CH la liste des sommets dans l'ordre croissant de leur distance à la racine R.

$CH(1) := R : I := 1 : J := 1$
Tant que $J \leq I$ Faire
  $P := CH(J) : S := SU(P)$: Si $S > 0$ Faire
    $I := I + 1 : CH(I) := S$
    Tant que $FR(S) > 0$ Faire
      $S := FR(S) : I := I + 1 : CH(I) := S$
  $J := J + 1$

Sur l'exemple précédent, on obtient l'ordre : 9, 7, 8, 3, 5, 1, 6, 2, 4.

On définit récursivement la largeur d'un sous-arbre comme la somme des largeurs de ses successeurs. Les sommets pendants sont de largeur L arbitraire, cette valeur correspond à la place nécessaire au tracé d'un numéro de sommet. En effet, s'il faut k unités de largeur pour tracer un numéro, pour tracer un sous-arbre de largeur L, il faudra $k * L$ unités de largeur.

Algorithme de calcul des largeurs

Les largeurs sont rangées dans le tableau P initialisé à 0. On crée un tableau CH de parcours concentrique à partir de la racine R et on affecte une largeur L à tous les sommets pendants. En parcourant ce tableau de la fin au début, on calcule la largeur de chaque sous-arbre de racine S.

Pour $I := NS$ à 2 Faire
  $K := CH(I) : S := T(K)$
  $P(S) := P(S) + P(K)$.

Les coordonnées d'un sommet sont alors calculées en fonction de celles de son prédécesseur. Soit un sommet S de coordonnées (XS, YS) et de largeur $P(S)$ ayant K successeurs, $SS(1), \ldots SS(K)$.

Algorithme de calcul des coordonnées

XG représente l'abscisse gauche du sous-arbre de sommet U. Avant de traiter les successeurs de S elle est égale à l'abscisse de S moins la demi-largeur du sous-arbre de racine S.

$XG := XS - P(S)/2$
  Pour $I := 1$ à K Faire
  $U := SS(I)$
  $XU := XG + P(U)/2 : XG := XG + P(U)$
  $YU := Y(S) + LG(U)$

Pour ce tracé les longueurs des arêtes sont portées verticalement et l'on trace les arêtes comme pour les dendrogrammes.

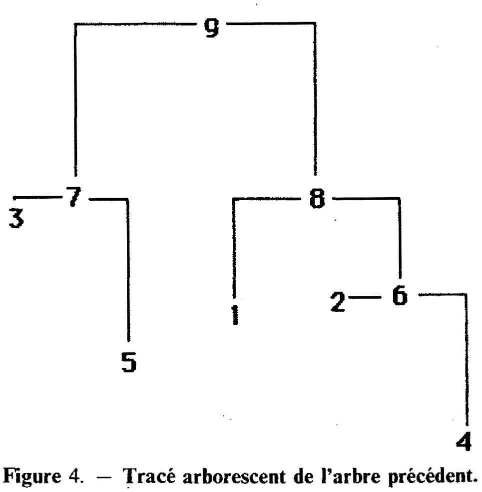


Figure 4. — Tracé arborescent de l'arbre précédent.

Remarque : Cet algorithme de tracé produit le dendrogramme d'une hiérarchie de partitions si pour l'arête $U \to V$ on donne à $LG(U)$ la différence des valeurs de dissimilarités entre la classe V et la classe U, les classes réduites à un seul sommet ayant une valeur de dissimilarité nulle.

3. CONCLUSIONS

Nous avons représenté une dissimilarité par un arbre valué, en utilisant la structure d'arbre définie par une approximation ultramétrique ou quadrangulaire de cette dissimilarité. Ces représentations apportent plus d'information que le dendrogramme. La qualité de cette projection dans des arbres de structure donnée, peut être évaluée au moyen de plusieurs critères métriques, comme l'écart absolu moyen, l'écart absolu maximal, ou l'écart quadratique moyen entre les dissimilarités initiales et les distances d'arbre. On a ainsi la possibilité de comparer des structures d'arbres sur un même ensemble de données et partant les algorithmes de classification ou de construction d'arbres à distances additives [6].

De plus nos méthodes de tracé permettent d'obtenir une représentation des voisinages souvent plus lisible que le niveau des classes d'une hiérarchie de partitions.

BIBLIOGRAPHIE

1. H. Abdi, J. P. Barthélémy et X. Luong, *Tree Representations of Associative Structures in Semantique and Episodic Memory Research*, in *Trends in Mathematical Psychology*, E. Degreef, J. Van Buggenhaut Eds North Holland, 1984, p. 3-31.
2. G. Brossier, *Approximation des dissimilarités par des arbres additifs*, Mathématiques et Sciences Humaines, vol. 91, 1985, p. 5-21.
3. J. L. Chandon, J. Lemaire et J. Pouget, *Construction de l'ultramétrique la plus proche d'une dissimilarité au sens des moindres carrés*, R.A.I.R.O. série R.O., vol. 14, 1980.
4. M. R. Garey et D. S. Johnson, *Computers and intractability*, W. H. Freeman Ed., San Francisco, 1979.
5. R. Glowinski, J. L. Lyons et R. Trémolieres, *Analyse numérique des inéquations variationnelles*, Dunod, Paris, 1976.
6. A. Guénoche, *Cinq algorithmes d'approximation d'une dissimilarités par des arbres à distances additives*, Mathématiques et Sciences Humaines (à paraître, 1987).
7. C. L. Lawson et R. J. Hanson, *Solving Least Squares Problems*, Prentice Hall, Englewood Cliffs, New Jersey, 1974.
8. R. Read, *Graph Theory and Computing*, Academic Press, New York, 1972.
9. S. Sattah et A. Tversky, *Additive Similarity Trees*, Psychometrika, vol. 42, n° 3, 1977.