الجمهورية الجزائرية الديمقراطية الشعبية
REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

2/92

وزارة الجامعات والبحث العلمي
Ministère aux Universités et de la Recherche Scientifique

ECOLE NATIONALE POLYTECHNIQUE

DEPARTEMENT : ELECTRONIQUE

PROJET DE FIN D'ETUDES

SUJET

METHODE DES ARBRES POUR LA RECONNAISSANCE DES FORMES

APPLIQUEE AUXCARACTERES DE L'ALPHABET ARABE

Proposé par :
L. HAMMAMI

Etudié par :
F. AIT BOUDAOUD
&
M. BELHANDOUZ

Dirigé par :
L. HAMMAMI

PROMOTION
Juillet 1992

E.N.P. 10, Avenue Hacen Badi El-Harrach - Alger

الجمهورية الجزائرية الديمقراطية الشعبية
REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

وزارة الجامعات والبحث العلمي
Ministère aux Universités et de la Recherche Scientifique

ECOLE NATIONALE POLYTECHNIQUE

DEPARTEMENT : ELECTRONIQUE

PROJET DE FIN D'ETUDES

SUJET

METHODE DES ARBRES POUR LA RECONNAISSANCE DES FORMES

APPLIQUEE AUX CARACTERES DE L'ALPHABET ARABE

Proposé par :
L. HAMMAMI

Etudié par :
F. AIT BOUDAOUD
&
M. BELHANDOUZ

Dirigé par :
L. HAMMAMI

PROMOTION
Juillet 1992

E.N.P. 10, Avenue Hacen Badi El-Harrach - Alger

DEDICACES

A mon pere.

A ma mere.

A ma soeur et a

mes freres.

A tous mes

amis.

Fouad.

A mes parents,

a mes freres et

soeurs,a mes

deux nieces

Amel et Amina.

A ma

grand-mere,

a ma tante

a tous mes amis

Mehdi

REMERCIEMENTS

Nous voudrions exprimer notre gratitude et nos vifs remerciements :

- à M^{me} L. Hamami qui a bien voulu nous faire confiance en nous chargeant de cette étude,
- à tous les membres du jury qui ont bien voulu examiner ce travail,
- à M. L. AIT BOUDAOUD pour son aide et ses conseils; sans lui ce travail n'aurait jamais vu le jour,
- à tous nos enseignants.

الموضوع : دراسة طريقة الأشجار للتعرف على الاشكال وتطبيقها على الحروف الابجدية العربية .

ملخص : هذه الدراسة تقدم طريقة تركيبية ، وصفية للتعرف على الاشكال تستعمل الاشجار لوصف الصور وتقتضي عملية الهيكلة .
من أجل تطبيقها على الحروف العربية ، تم انجاز برنامج لتحقيق هيكلة الحرف ، تصحيح العيوب الهيكلية الناتجة عن هذه العملية ، نشر الحرف على شكل شجراني وأخيرا التعرف عليه باستعمال مسافة LU .
النتائج المحصل عليها تثبت امكانية تطبيق هذه الطريقة بنجاح على الحروف العربية .

Title : The tree system approach for structural pattern recognition and its application to isolated arabic caracters.

Abstract : This study presents a describing structural method in pattern recognition which uses trees for image description and requires a former thinning operation.

In order to apply it to arabic caracters recognition, a program has been developed to find the caracter skeleton, correct the thinning defects and transform this caracter into its corresponding tree representation and finally recognize it using the LU tree-to-tree distance.

The final results has shown that this approach can be successfully applied to arabic caracters recognition.

Titre : Etude de la méthode des arbres pour la reconnaissance des formes appliquée aux caractères de l'alphabet arabe.

Résumé : Cette étude présente une méthode structurelle, descriptive de la reconnaissance des formes qui utilise les arbres comme descripteurs d'images et nécessite une opération préalable de squelettisation.

En vue de son application aux caractères de l'alphabet arabe, un programme a été élaboré pour réaliser la squelettisation du caractère, corriger les défauts de structure dûs à cette squelettisation et concrétiser le développement en arbre de ce caractère pour enfin le reconnaître en utilisant la distance de LU.

Les résultats obtenus confirment que cette méthode est applicable avec succès aux caractères arabes.

S O M M A I R E

INTRODUCTION

INTRODUCTION

Composant important de l'intelligence artificielle, la reconnaissance des formes donne lieu à un grand nombre d'applications qui arrivent actuellement sur le marché. On y retrouve la reconnaissance de la parole pour la commande de machines, l'identification d'objets par un "oeil" électronique pour faciliter leur manipulation par un robot, le tri de prélèvements de cellules pour le dépistage de différentes maladies, la lecture optique de textes pour éviter une saisie manuelle, ou la reconnaissance de la "signature" d'un sous-marin, c'est à dire son identification à travers les ondes émises.

Il existe deux approches du problème de la reconnaissance des formes. Elles ont respectivement leurs origines l'une en mathématiques, l'autre en informatique. L'analyse et le traitement des signaux ont été longuement étudiés par des mathématiciens appliqués, en particulier depuis l'invention du radar. L'approche algorithmique des informaticiens est assez différente. Elle est souvent basée sur la description syntaxique des objets à reconnaître. Un travail complet nécessite l'exploitation de chaque type d'expertise.Le traitement numérique sert surtout pendant une première phase afin d'identifier les objets. On y retrouve des filtres,comme d'autres techniques statistiques, qui permettent, par exemple, de reconnaître les frontières des objets dans une scène, d'éliminer les effets de la réflection de la lumière (points de brillance), d'identifier des phonèmes dans un discours (avec des possibilités de substitution éventuelle), ou de retrouver les lignes cachées sur une image.

L'approche algorithmique sert davantage à l'analyse des

relations entre les objets. Dans ce domaine on trouve des problèmes comme la reconstitution et l'interprétation d'une phrase formée de phonèmes reconnus, l'interprétation d'une image photographiée à partir d'un satellite pour l'évaluation du rendement de l'agriculture d'un pays, la discrimination de cellules cancéreuses sur une image où les frontières des cellules sont déjà retrouvées,...

Il est illusoire de vouloir résumer des travaux aussi importants dans ces quelques lignes. Pour mieux cerner l'état actuel de cette discipline, on peut consulter les comptes rendus des conférences AFCET. Notons que de telles conférences ont lieu depuis plus de trente ans.

On peut se demander si toutes ces années de travail ont conduit à des résultats concrets et utiles. La réponse est nécessairement mitigée. En effet, la reconnaissance des formes, comme d'ailleurs le reste de l'intelligence artificielle, s'est avérée difficile, et coûteuse en moyens informatiques. Ce dernier facteur est particulièrement sensible en ce qui concerne la reconnaissance des formes, car il s'agit de la modélisation d'une activité humaine qui utilise beaucoup de parallélisme, et ce dans une machine essentiellement séquentielle.

Ainsi, le taux de réussite des systèmes de reconnaissance de formes est assez variable. La lecture est opérationnelle dans le cas d'un texte imprimé, ou frappé à la machine, à la condition d'ajuster soigneusement les paramètres d'entrée en fonction du jeu de caractères utilisé. Les manuscrits sont essentiellement illisibles, mais on peut éventuellement en identifier les auteurs. Dans la reconnaissance de la parole, les systèmes industriels avec un petit vocabulaire de mots détachés fonctionnent bien, en particulier quand ils sont pourvus de mécanismes d'apprentissage leur permettant de s'adapter aux

particularités de différents interlocuteurs. On a l'impression que la reconnaissance de la parole continue, avec de multiples intervenants, est une possibilité qui, pour ne pas être actuelle, est néanmoins envisageable dans un avenir proche. La synthèse de la parole est déjà une réalité industrielle. Le traitement automatisé d'images a reçu un coup de fouet considérable avec la mise à la disposition des chercheurs d'images enregistrées par des caméras embarquées dans des satellites. Il est néanmoins vrai que les travaux dans ce domaine sont loins d'être définitifs. On a mieux réussi des systèmes d'analyse d'images médicales, avec un tri, par exemple, de prélèvements cellulaires entre cellules saines, cellules malades, et cellules indéterminées. Des systèmes de ce type sont déjà opérationnels, et ils ont diminué la fréquence de l'intervention humaine.

CHAPITRE I
GENERALITES

I-1- DIFFERENTES APPROCHES EN RECONNAISSANCE DES FORMES [6]

Il existe deux grandes méthodologies utilisées en reconnaissance des formes : l'approche dite *statistique* et l'approche dite *structurelle*.

I-1-1 APPROCHE STATISTIQUE

Le principe de l'approche statistique consiste à faire un certain nombre de mesures sur un objet à reconnaitre et à étudier la répartition de ces mesures dans un espace metrique afin de trouver la plus forte probabilité d'appartenance de l'objet à une classe de formes.

Il existe deux grands types de méthodes en approche statistique : les méthodes dites *non paramètriques*, où l'on cherche à définir les frontières des classes dans l'espace de représentation, de façon à pouvoir classer le point inconnu par une série de tests simples; les méthodes dites *paramètriques* ou *bayesiennes*, où l'on se donne un modèle de la distibution de chaque classe (en général gaussien), et où l'on cherche la classe à laquelle le point a la probabilité la plus grande d'appartenir. Dans le premier cas, une tactique simple consiste à définir des *hyperplans* qui séparent *au mieux* les classes d'apprentissage. La décision se réduit alors à une série de produits scalaires. Ces méthodes sont dites de *séparation linéaire*. La détermination des équations de ces hyperplans a fait l'objet de nombreux travaux, et les probabilités théoriques d'erreur correspondantes ont été largement étudiées.

Une autre approche non paramètrique très utilisée est celle de la décision par *plus proche voisin* : on attribue au point inconnu la classe de son plus proche voisin de l'ensemble d'apprentissage. Cette méthode simple a de bons résultats statistiques, mais souvent longue en temps de calcul. Des versions accélérées ont été proposées, ainsi que

diverses extensions.

Dans le cas paramètrique, on cherche à minimiser l'erreur moyenne de mauvaise classification grâce aux hypothèses sur la nature statistique des classes. Si l'ensemble d'apprentissage (et l'espace de représentations) se prêtent à ces hypothèses, on est dans une situation optimale pour la décision. La difficulté est bien souvent que la répartition des points dans chaque classe n'est pas assez régulière, ou les points ne sont pas assez nombreux, pour que la précision des calculs soit proche de celle que donne la théorie. Ces réserves faites, ce type de méthode est rapide et donne de bons résultats.

I-1-2 APPROCHE STRUCTURELLE

Contrairement à l'approche statistique qui, elle, ignore complètement le côté topologique et gécmetrique de la forme, l'approche structurelle la conçoit comme un ensemble de formants (caractéristiques) agencés entre eux appelés *primitives*. Ces primitives traduisent les caractéristiques topologiques et géometriques de la forme.

La reconnaissance dans une telle approche implique non seulement la capacité d'attacher une forme à une certaine classe mais aussi la capacité de décrire cette forme de façon à ce qu'elle ne soit pas ratachée à une autre classe.

On peut distinguer deux types de méthodes en approche structurelle:

a) Méthode descriptive : où les relations spatiales entre les primitives sont décrites par des descripteurs d'image (graphes, chaines, arbres), notions que nous introduirons au chapitre 3.

Cette méthode est efficace mais présente l'inconvénient de nécessiter une squelettisation coûteuse et qui, de plus,

ne conserve pas toujours la connexité de la forme engendrant ainsi des erreurs d'interprètation, mais, ceci peut être corrigé comme nous allons le voir au chapitre 3.

b) Methode syntaxique : où chaque classe de formes est décrite par un ensemble de règles dit *grammaire*. Etant donné un échantillon de formes C, la recherche d'une grammaire G telle que C soit inclu dans le langage généré par G, noté L(G), est connu sous le nom d'*inference grammaticale*. Le nombre de solutions d'un problème d'inférence grammaticale croît exponentiellement (la complexité de calcul en fait de même) avec la taille de l'échantillon C, i.e le nombre de formants ou de lettres qu'il comporte. Le choix d'une grammaire parmi ces solutions implique la définition d'un critère de choix (explicite ou non) sur l'ensemble de ces solutions, ce qui exige un temps en plus. Des méthodes visant à réduire l'ensemble des solutions sont l'objet de plusieurs algorithmes, comme celui connu sous le nom UVKW dans le cas des grammaires régulière et celui de SOLMONOFF dans le cas des grammaires algèbriques.

La reconnaissance syntaxique classique d'une forme X consiste en général à chercher une grammaire G telle que $X \in L(G)$, dans ce cas X peut être soit reconnu, soit rejeté. La définition des grammaires stochastiques (i.e : dont les règles de production sont probabilisées) a rendu possible une analyse syntaxique tolérante aux erreurs, autrement dit reconnaitre X avec une probabilité P(X) ou avec une erreur 1-P(X).

I-1-3 COMPARAISON ENTRE LES DEUX APPROCHES

Selon MICLET[6], il ne se dégage aucune supériorité indiscutable de l'une des deux approches sur l'autre. Certains problèmes se résolvent *naturellement* mieux à partir d'une approche statistique (ceux où le nombre de mesures est

important, et la taille de l'ensemble d'apprentissage très significative induisent un traitement de ce type); d'autres sont considérablement simplifiés par l'utilisaion d'outils de type structurel (caractères manuscrits, par exemple). D'un autre côté, forcer à tout prix des données à s'adapter à un formalisme pour pouvoir appliquer les algorithmes correspondants, n'est pas toujours une procédure habile.En regardant la bibliographie des différentes applications, on s'aperçoit que le choix "statistique ou structurel " est avant tout pragmatique : il est très difficile de citer un domaine d'application où les deux types d'approches n'aient pas été utilisés.

Une autre remarque est que, bien sûr, les formalismes mathématiques et les *philosophies* des deux méthodologies sont très opposées; mais comme le remarque K.S.FU, cette distinction n'est significative justement que si l'on se place du point de vue des formalismes : elle est beaucoup moins nette en termes d'application à un problème pratique. La plupart de ces problèmes ont intérêt à s'étudier sous les deux angles, de façon à ce que chaque approche puisse apporter ses avantages. Les deux méthodologies n'ont évidemment rien à gagner à s'ignorer.

Un exemple concret concerne les grammaires probabilisées où l'introduction de mesures de probabilité permet d'accélérer l'analyse syntaxique de la structure de la forme, ou d'aider à l'apprentissage de cette structure. Un autre exemple est l'utilisation de critères de type structurel pour organiser un dictionnaire de représentants de l'apprentissage, afin d'accéder rapidement à un sous-ensemble sur lequel on fera une décision statistique. Dans les deux cas une telle approche mixte combine les avantages des deux types d'approche.

I-2 OUTILS DE LA RECONNAISSANCE [7]

Les systèmes d'acquisition et de traitement sont composés des éléments suivants (voir fig.I-1) :

- Le système d'acquisition et de numérisation d'image :

Ce système permet d'obtenir à partir d'une image analogique, une image numérique pouvant être traitée par un calculateur.

-Le système de visualisation :

Son rôle est de visualiser l'image obtenue à partir de la caméra ou de la mémoire du digitaliseur.

-Le calculateur :

Il pilote le système de traitement d'image et permet ainsi de recueillir les informations issues de celui-ci afin d'engendrer la commande ou la décision appropriée.

-La mémoire de masse :

C'est une mémoire supplémentaire pour stocker les résultats d'un traitement.Une image en noir et blanc de taille 512x512 avec 256 niveaux de gris occupe 256 KO, la même image en couleur en occupe trois fois plus. Il est donc nécessaire de disposer d'une mémoire de masse importante.

Dans tout procédé de reconnaissance, on retrouve deux étapes fondamentales que sont *l'acquisition* de l'image et son *prétraitement*.

I-2-1 ACQUISITION

Le module d'acquisition permet de digitaliser (numériser) un signal analogique (document) donné par un

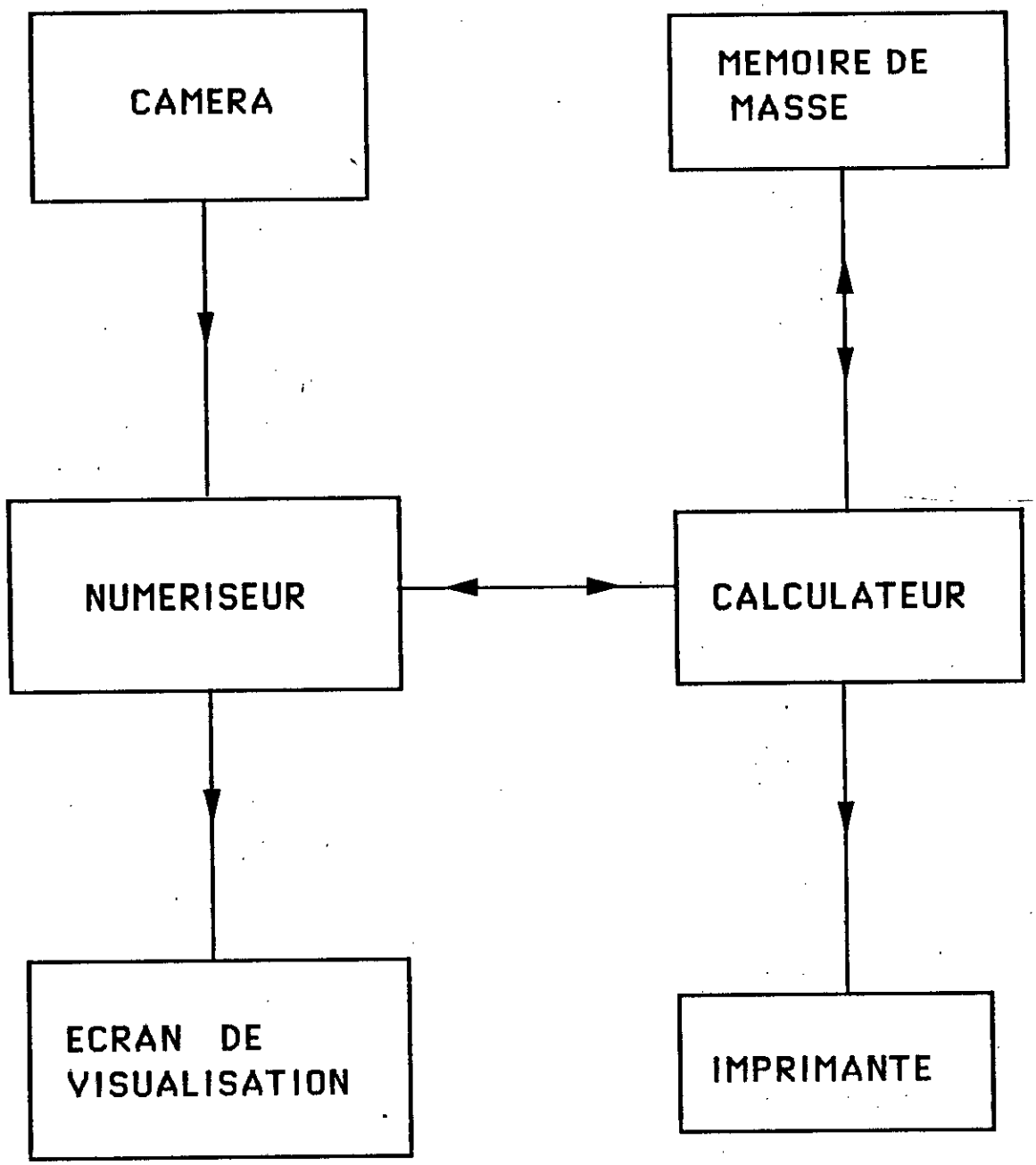


Fig. I-1 Configuration d'un système de traitement d'image

capteur physique. L'image obtenue est différente suivant le type de capteur utilisé.

a) Tablette graphique

L'image obtenue est un ensemble de points séparés par des levers d'un crayon. En effet, à partir du contact avec la surface de la tablette, un crayon spécial émet les coordonnées des points qu'il décrit à une fréquence constante. La rupture du contact interrompt la transmission d'un caractère donné. Ainsi, l'enregistrement sur la tablette produit un ensemble de coordonnées séparées par des signes indiquant les moments où le crayon a cessé de toucher la tablette.

Ce type d'enregistrement offre l'avantage d'une fidèle reproduction de la direction des lignes tracées et l'ordre dans lequel les parties du diagramme ont été dessinées. La tablette graphique est utilisée dans la reconnaissance des caractères manuscrits.

b) La caméra

Il en existe principalement deux types suivant la technologie de construction des capteurs.

-Les caméras à tubes :

Elles sont composées d'ube cible photoconductrice qu'explore un faisceau électonique. L'exploration se fait ligne par ligne. A chaque ligne correspond, en sortie, un signal analogique proportionnel à l'intensité.

-Les caméras C.C.D (Chaged Coupled Device) :

Elles sont constituées par un assemblage de photodiodes. Chacune d'entre elles délivre une intensité proportionnelle àun point de l'image appelé *pixel* (picture element). contrairement aux caméras à tubes qui fournissent un signal continu qu'il faut échantillonner spatialement pour isoler chaque pixel, les caméras CCD fournissent directement l'intensité pour chaque point de l'image puisqu'à chaque

point correspond une photodiode.

c) Le scanner

l'image obtenue à partir du scanner est une image à plusieurs niveaux de gris. On distingue deux types de scanners : les scanners *à main* et les scanners *de table* qu'ils soient *à défilement* ou *à plat*.

Le système comprend toujours une source de lumière constante fournie par des tubes lumineux; plus la zone du document ainsi éclairé est sombre, plus elle réfléchit la lumière. Par le truchement d'un miroir et d'une lentille focalisatrice, des capteurs (phototransistors de très faible diamètre ou cellules C.C.D) receuillent la luminosité réfléchie et émettent une tension proprotionnelle à celle-ci. Grâce à des barrettes qui dépassent les 3000 cellules C.C.D, on obtient une définition atteignant maintenant 400x400 points.

L'image obtenue en sortie de ces différents capteurs est, soit directement,soit à l'aide d'un digitaliseur, numérisée et devient donc une image formée de pixels à plusieurs niveaux de gris.

L'opération de digitalisation nécessite une prise de décision sur la résolution de l'image. Celle-ci se présente sous la forme d'une matrice NxN où N est généralement pris comme une puissance entière de 2.

$$N = 2^{n}$$

et

$$G = 2^{m}$$

où G est le nombre de niveaux de gris.
Le nombre de bits nécesaire pour stocker une telle image est donc

$$b = N \times N \times m$$

Pour une image de taille 128x128 à 64 niveaux de gris, par exemple, b =16384 Octets.

I-2-2 PRETRAITEMENT

Le prétraitement est l'ensemble de toute les techniques qui, directement sur les pixels ou indirectement par le biais des transformées de Fourier, manipulent l'image. Parmi celles-ci on peut citer les plus importantes et les plus utilisées que sont les techniques de filtrage.

Il en existe deux types : le filtrage *fréquentiel* et le filtrage *spatial*.

I-2-2-1 Le filtrage fréquentiel

Ce type de filtrage concerne les atténuations apportées à la gamme de fréquence d'une image, couper les hautes fréquences s'obtient par un filtre passe-bas, tandis que couper les basses fréqauences s'obtient par un filtre passe-haut.

a) Filtre passe-bas :

Il est conçu pour atténuer les composantes hautes fréquences du signal d'image, F(u,v) sans que l'information contenue dans les basses fréquences soit amputée.

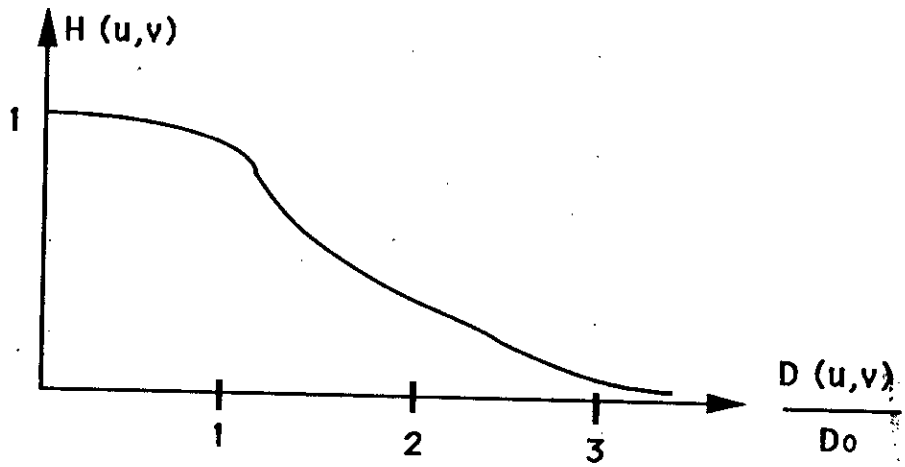
Exemple: Le filtre exponentiel

La fonction de transfert est de la forme :

$$H(u,v) = \exp[-D(u,v)/Do]^{n}$$

D(u,v) : distance du point (u,v) à l'origine du plan U,V.
Do: seuil positif de coupure.
n : paramètre au choix qui donne l'ordre du filtre

cas où n = 2



-Fig. I-2 Le filtre exponentiel(passe-bas)-

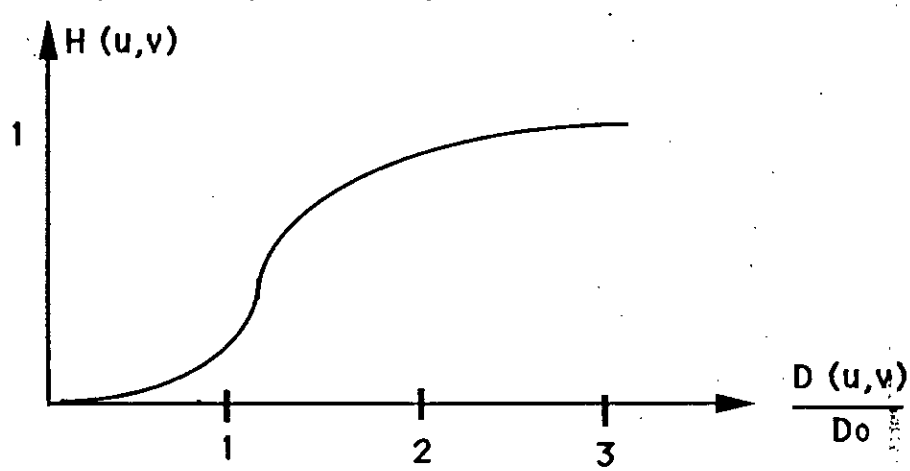
b) Filtre passe-haut:

La fonction de transfert du filtre passe-haut atténue les fréquences basses sans toucher à l'information contenue dans les hautes fréquences. Ce type de filtre permet de renforcer et d'extraire les contours d'une image.

Exemple : Le filtre exponentiel

La fonction de transfert est de la forme :

$$H(u,v) = \exp[-Do/D(u,v)]^{n}$$



-Fig. I-3 Le filtre exponentiel(passe-haut)-

I-2-2-2 LE FILTRAGE SPATIAL

Ce type de filtrage envisage toutes les opérations directes dans le plan de l'image elle-même; le but recherché est d'atténuer le bruit et faire disparaitre les défauts et perturbations contenues dans l'image initiale.

- Transformation d'histogramme

Une image est souvent numérisée dans des conditions non idéales, il est donc utile d'améliorer sa présentation afin de mieux mettre en évidence les informations souhaitées et de permettre des traitements plus efficaces. Ceci peut se faire par des modifications de contraste et de la dynamique à l'aide d'opérations ponctuelles qui modifient l'histogramme de l'image(voir fig.I-4).

- Egalisation d'histogramme :

L'image discrètisée comporte n pixels et chacun d'eux prend une valeur aléatoire de gris rk parmi les k valeurs discrètes rk.

L'histogramme discret s'écrit :

$$Pk = \frac{F(rk) - F(rk-1)}{n} = \frac{nk}{n}$$

nk : nombre de pixels au niveau rk

F(rk) : fonction de répartition

L'histogramme cumulé (fig I-4) discret est :

$$Hk = 1/n \sum_{i=o}^{k} ni$$

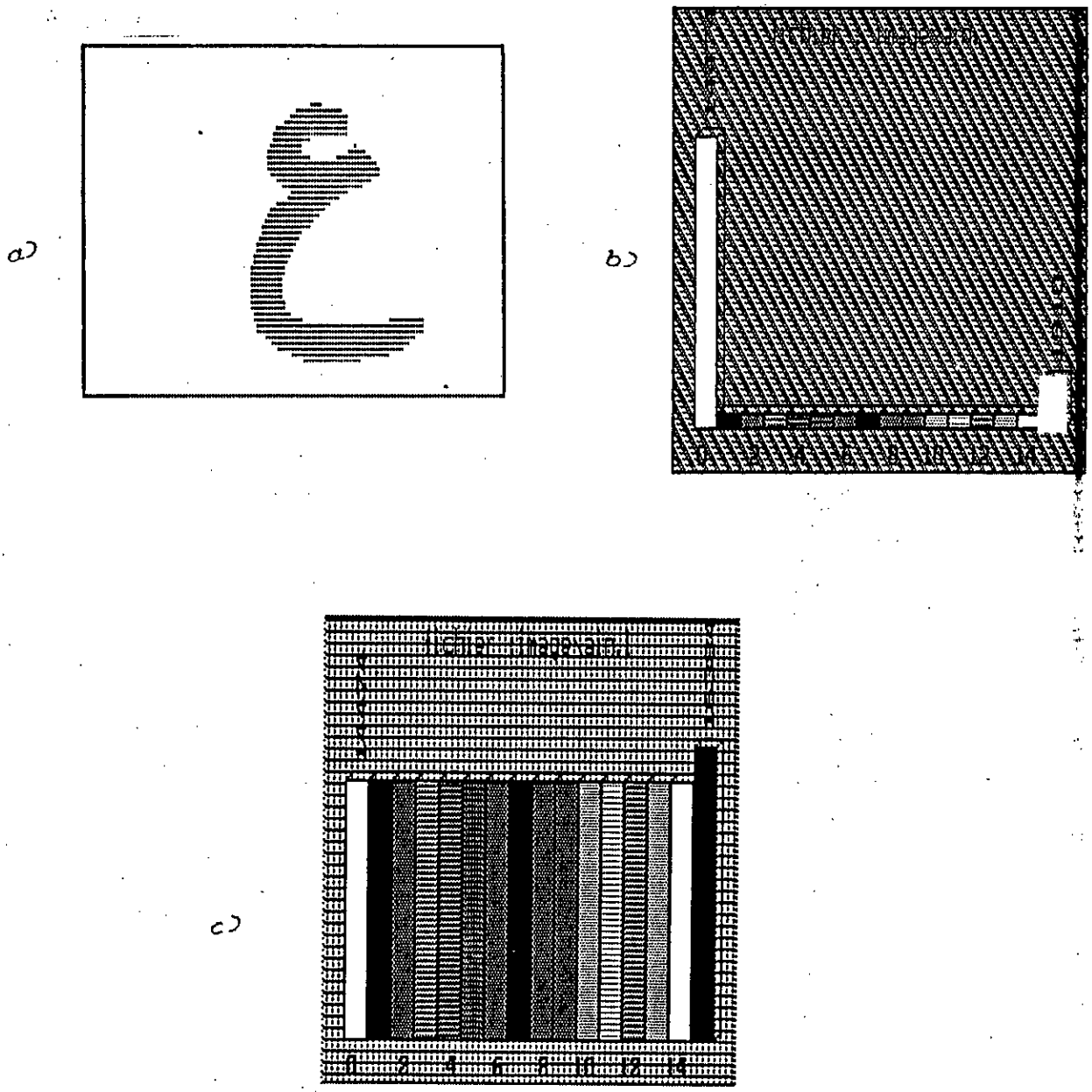


Fig. I-2 Histogramme d'une image a) image initiale b) histogramme simple c) histogramme cumulé source[5]

Si l'image finale est codée sur d niveaux de gris, alors le nombre de pixels pour chaque niveau de l'histogramme égalisé idéal est constant, et est égal à n/d car l'image contient n pixels.

Il faut aussi signaler qu'il existe plusieurs autres méthodes de filtrage spatial parmi lesquelles nous pouvons citer :

- La moyenne des voisins : où le niveau de gris de chaque pixel est remplacé par la moyenne des niveaux de gris de ses voisins.
- Le filtre médian : où la valeur médiane des gris d'une fenêtre 3x3 pixels, centrée sur le point à filtrer est attribuée à ce point.
- La moyenne sur l'image : qui n'agit pas sur les pixels individuellement mais sur l'ensemble des points de l'image en faisant la moyenne de plusieurs images provenant de prises de vues d'une même scène.

I-2-2-3 CONCLUSION SUR LE PRETRAITEMENT

Pour conclure, il faut signaler que cette partie très importante de la reconnaissance à fait l'objet d'une thèse de Magistère[7], d'une part, et d'autre part d'une bonne partie d'un projet de fin d'étude[5].C'est pour cette raison qu'elle n'a pas été détaillée dans ce rapport et que nous invitons le lecteur intéressé à se reporter à ces ouvrages intéressants.

CHAPITRE II
SQUELETTISATION

II-1 INTRODUCTION

La squelettisation joue un rôle très important dans la méthode structurelle .Cette opération permet d'éliminer les points redondants tout en préservant les caractéristiques de la forme. Elle constitue,de ce fait, la première étape d'extraction des primitives.

L'image obtenue, après digitalisation et prétraitement, se présente sous forme matricielle, à plusieurs niveaux de gris. L'algorithme de squelettisation nécessite que l'image soit à 2 niveaux de gris, ce qui impose, en premier lieu, une binarisation de cette image. La binarisation consiste à forcer tout pixel, dont le niveau de gris est différent d'un niveau minimal (niveau 0), à un niveau maximal (niveau 1).

II-2 DEFINITIONS

VOISINAGE D'UN PIXEL

Soit un pixel P situé au point (x,y). On définit 2 ensembles $V_4(P)$ et $V_8(P)$ de voisins de P, appelés respectivement 4-voisins et 8-voisins de P.

	x−1,y	
x,y−1	p	x,y+1
	x+1,y	

Fig.II-1 4-voisins de P

x−1,y−1	x−1,y	x−1,y+1
x,y−1	p	x,y+1
x+1,y−1	x+1,y	x+1,y+1

Fig.II-2 8-voisins de P

CONNECTIVITE

Soit un ensemble E de pixels. On définit deux types de connectivité:

1) 4-Connectivité: Celle de deux pixels P_1 et $P_2 \in E_1$ si P_1 est dans l'ensemble $V_4(P_2)$.

„ 2) 8-Connectivité: Celle de deux pixels P_1 et $P_2 \in E_1$ si P_1 est dans l'ensemble $V_8(P_2)$.

CHEMIN

Un chemin est une séquence de pixels $P_1, P_2, \ldots, P_N$ tels que pour $k > 1$, P_{k-1} est un voisin de P_k et pour $k < 1$, P_{k+1} est un voisin de P_k.

ENSEMBLE CONNECTE

Un ensemble de S pixels est connecté si, pour chaque paire de pixels M et N dans S, il ya un chemin dont le premier et le dernier éléments sont respectivement M et N et si tous les autres pixels appartiennent à S.

PIXEL ALLUME

Dans tous ce qui suit, on appellera pixel "allumé" tout pixel dont le niveau de gris est à '1'. Celui, dont le niveau de gris est à '0' sera dit "éteint".

AMAS DE PIXELS

On appellera *amas*, tout ensemble connexe de pixels allumés.

SQUELETTE D'UNE IMAGE

Définition mathématique

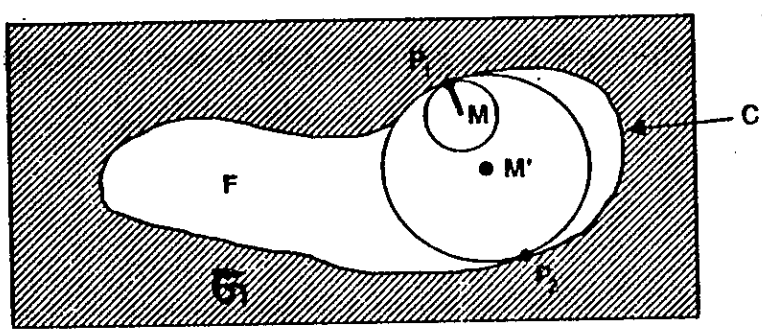


Fig.II-3

F = Figure plane convexe limitée par un contour (c).

G = son complémentaire.

F = ensemble fermé de points; il en résulte : $C \subset F$

Par suite, on introduit : H = G ∪ C ; on remarque que H est à l'extérieur de F.
Soit M un point de F et d sa distance à H, c'est à dire la distance de M à P; et soit P un point de H; d est la valeur inférieure de MP; il s'ensuit que :

$$d = INF \left\{ d(M,P) \right\} \text{ avec } P \in H.$$

On implante un disque Dr centré en M, de rayon r; on constate que d est aussi le plus grand des rayons r, tel que : Dr ⊂ F. On en déduit que d est la valeur supérieure de r, Dr restant dans F et s'écrit :

$$d = SUP \left\{ r \ / \ Dr \subset F \right\}$$

D_d étant le plus grand disque centré en M et inclus dans F, il existe des points M' tels que D_d comporte 2 contacts P_1 et P_2 avec H ; le squelette de F est alors l'ensemble { M' } des points M' ; cet ensemble s'appelle aussi axe médian .

Soit une région R limitée par un contour C; pour n'importe quel point P dans R , on cherche son plus proche voisin dans C. Si p a plus d'un tel voisin, alors p appartient au squelette de R. Il faut noter que le concept de "proximité" dépend directement de la définition de la distance et par conséquent le résultat de la transformation en axe médian sera influencé par le choix de cette distance . Des exemples utilisant la distance Euclidienne sont représentés sur la fig.II-4 .

Fig.II-4 Axes médians de régions simples

REMARQUE

la programmation sur ordinateur de la définition

ci-dessus est pratiquement prohibitive , à cause du nombre d'opérations à éffectuer pour aboutir au squelette de l'image.

AMINCISSEMENT DE CONTOUR

L'opération qui consiste à amincir la région curviligne qui enveloppe un ensemble de pixels, conduit à la squelettisation. La région ou contour, peut subir l'amincissement à condition de ne pas perdre ses propriétés de connexité.

II-3 ALGORITHME DE SQUELETTISATION DE ZHANG ET SUEN

Cet algorithme, que nous avons adopté, est puissant et efficace(voir fig.II-5). Il consiste à appliquer à tous les points de l'image, une série d'itérations où ces points sont traités par 2 étapes de base. Un pixel du contour étant tout pixel "allumé" ayant au moins un 8-voisin "éteint". En se référrant à la définition du 8-voisinage illustrée par la figure II-6, la première étape force à zéro un point du contour P_1 satisfaisant les conditions suivantes :

a) $2 \leq N(P_1) \leq 6$

b) $S(P_1) = 1$

c) $P_2 * P_4 * P_6 = 0$

d) $P_4 * P_6 * P_8 = 0$

où $N(P_1)$ est le nombre de pixels " allumés ", voisins de P_1. Donc $N(P_1) = P_2 + P_3 + \ldots + P_8 + P_9$.

et $S(P_1)$, le nombre de transitions 0-1 dans l'ordre P_2, P_3, ...P_8, P_9.

Par exemple $N(P_1) = 4$ et $S(P_1) = 3$ pour la fig.II-6 .

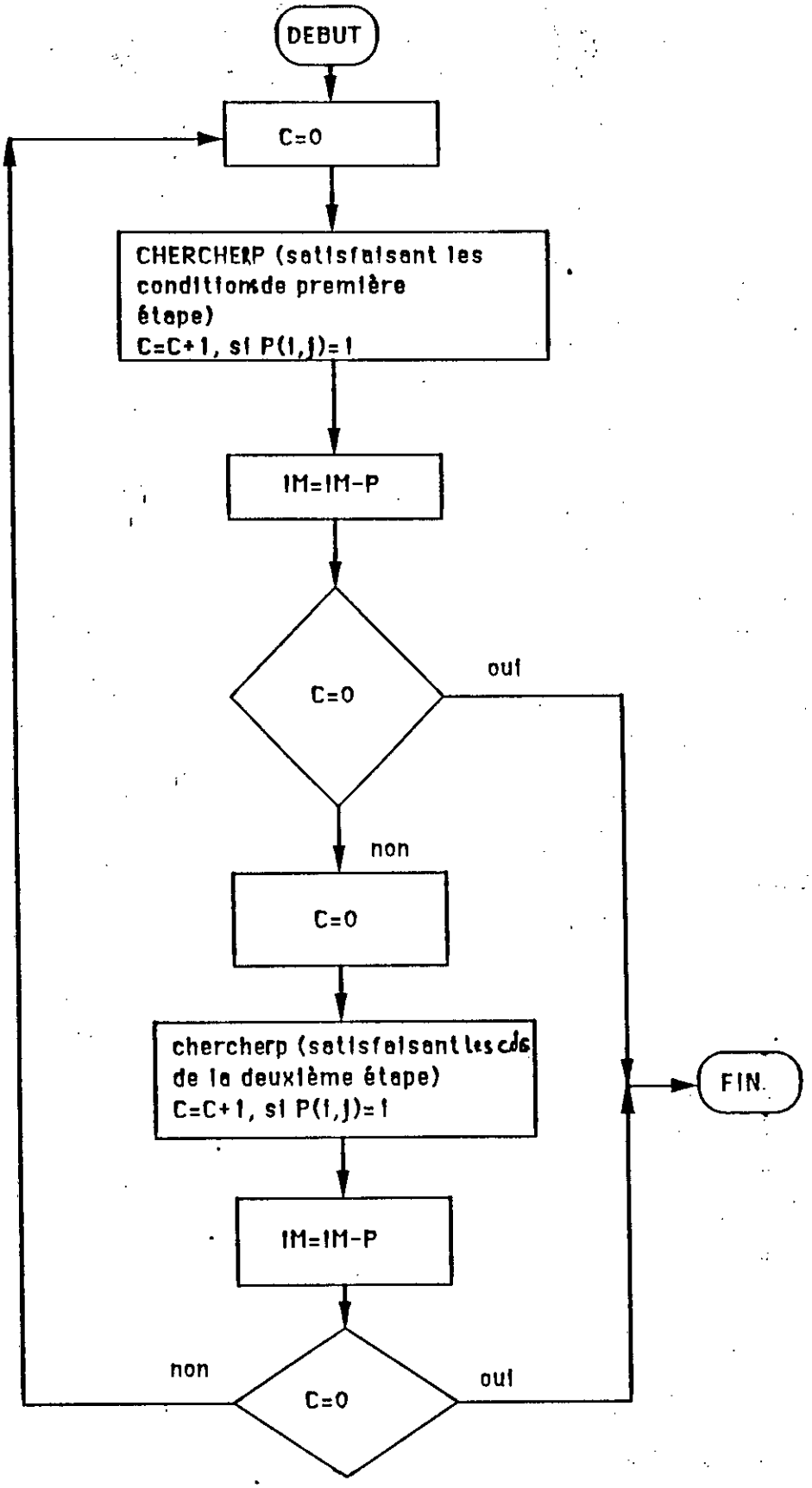


Fig. III-5 Organigramme de l'algorithme de squelettisation.

P_9	P_2	P_3
P_8	P_1	P_4
P_7	P_6	P_5

0	0	1
1	P_1	0
1	0	1

Fig. II-6

Dans la seconde étape, les conditions a) et b) restent inchangées, mais les conditions c) et d) deviennent :

c') $P_2 * P_4 * P_8 = 0$

d') $P_2 * P_6 * P_8 = 0$

L'étape 1 est appliquée à chaque point du contour de la région binaire considérée. Si l'une ou plusieurs des conditions a) à d) sont violées, la valeur du point en question reste inchangée. Si toutes les conditions sont satisfaites, le point est supprimé. Il est cependant important de noter qu'un point n'est supprimé qu'après que tous les points du contour aient été détectés. Ceci évite de changer la structure des données durant l'exécution de l'algorithme.

Après l'étape 1 , l'étape 2 est appliquée de la même manière à tous les points du contour de l'image résultante .

Une itération de l'algorithme consiste donc en :

1- Appliquer l'étape 1 à tous les points du contour
2- Supprimer les points satisfaisant les conditions a) à d)
3- Appliquer l'étape 2 à tous les points du contour
4- Supprimer les points satisfaisant les conditions a) à d')

Cette procédure de base est appliquée itérativement à l'image jusqu' à ce que plus aucun point n'est supprimé, ce

qui termine l'algorithme produisant le squelette de l'image (voir fig.II-7).

- La condition (a) est violée lorsqu'un point du contour P_1 a un ou sept de ses 8-voisins de valeur '1'. Le fait d'avoir uniquement un tel voisin implique que P_1 est l'extrémité d'un trait du squelette et ne doit évidemment pas être supprimé. Si P_1 possède 7 de ses 8-voisins de valeur '1' et est supprimé, celà causerait une érosion dans la région .

- La condition (b) est violée lorsqu'elle est appliquée aux points d'un trait d'épaisseur un pixel. Cette condition évite donc des disconnexions dans les segments du squelette durant l'opération de squelettisation.

- Les conditions (c) et (d) sont satisfaites simultanément par l'ensemble des valeurs suivantes :

$P_4 = 0$, ou $P_6 = 0$ ou ($P_2 = 0$ et $P_8 = 0$)

Donc , en se référant à la figure II-6, les points qui satisfont, aussi bien, à ces conditions qu'aux conditions (a) et (b) sont les points du sud et de l'est du contour ou les points des coins nord-ouest de celui-ci. Dans ces cas-là, P_1 ne fait pas partie du squelette et devrait être supprimé.

- D'une façon similaire , les conditions (c') et (d') sont satisfaites simultanément par l'ensemble des valeurs suivantes:

$P_2 = 0$ ou $P_8 = 0$ ou ($P_4 = 0$ et $P_6 = 0$)

Ceci correspond aux points du nord et de l'ouest du contour , ou des coins sud-est de ce dernier. Notez que les points des coins nord-est satisfont $P_2=0$ et $P_4=0$ et donc les conditions (c) et (d) , de même que (c') et (d'). Ceci est vrai aussi pour les points des coins sud-ouest pour lesquels $P_6 = 0$ et $P_8 = 0$.

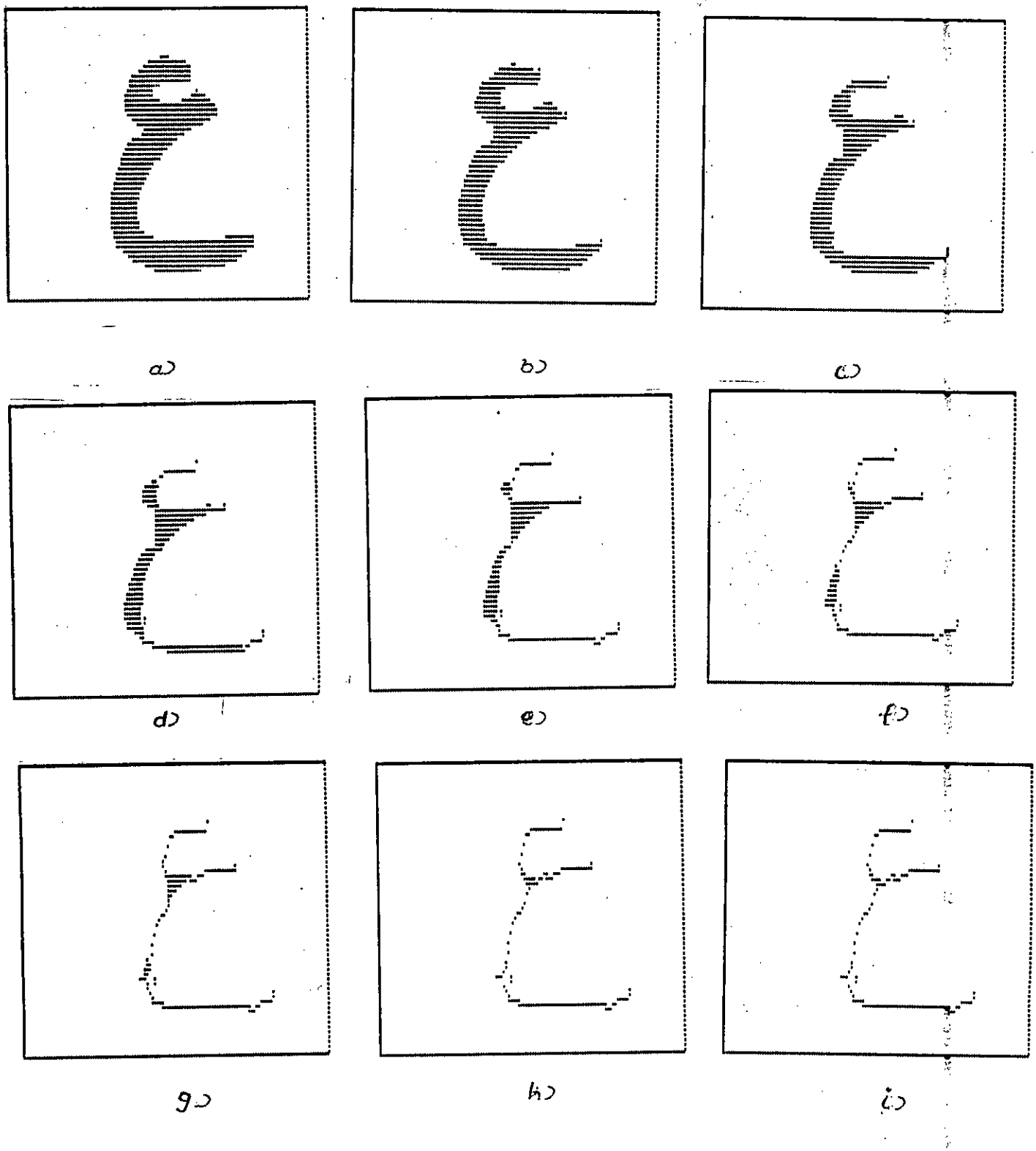


Fig. II-7 Differentes étapes de la squelettisation.
a) image initiale; b), d), f) et h): première étape;
c), e), g) et i) : deuxième étape.

II-4 PERFORMANCES ET DEFAUTS DE L'ALGORITHME

L'algorithme de squelettisation de Zhang et Suen est très efficace et conserve l'essentiel de l'information dans le squelette. De plus, c'est l'un des algorithmes de squelettisation les plus rapides. Néanmoins, cet algorithme présente certains défauts qui se résument dans les points suivants :

1- On perd des pixels qui doivent faire partie du squelette. En général, les traits obliques d'épaisseur deux pixels sont réduits à un ou deux pixels.
2- Certaines formes disparaissent totalement : c'est le cas pour les formes carrées.

Pour remédier au premier défaut, on peut modifier la condition $2 \leq N(P_1) \leq 6$ par $3 \leq N(P_1) \leq 6$. Celà permet de préserver les extrémitées des différents segments du squelette (voir fig.II-8), mais conduit à un squelette dont l'épaisseur est supérieure à un pixel.

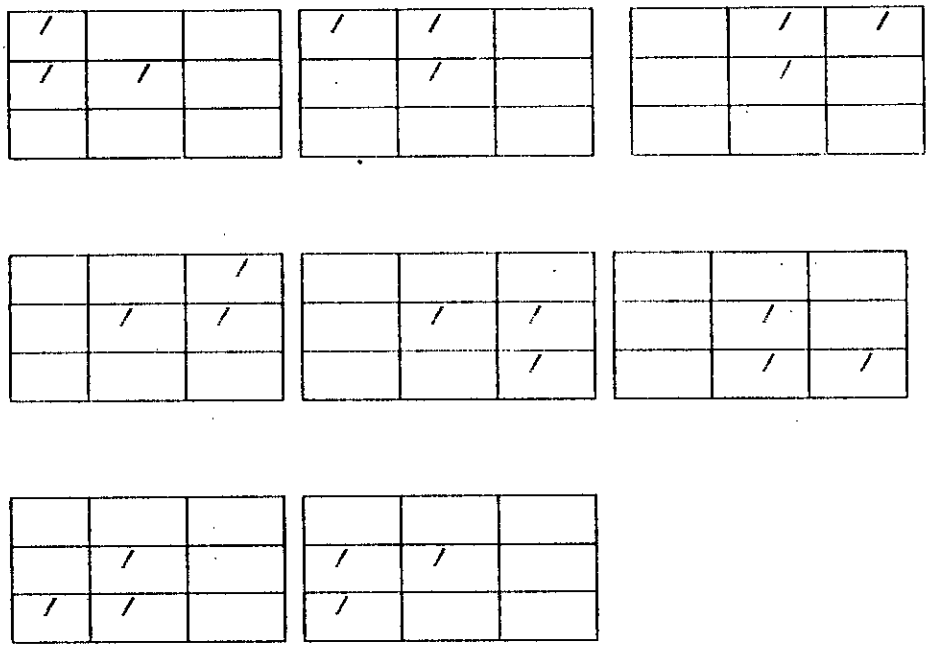


Fig. II-8 Les points des extrémités à préserver

CHAPITRE III
DESCRIPTION D'IMAGE-NOTIONS D'ARBRES

III-1 STRUCTURE DE CHAINES

III-1-1 PRINCIPE

La description structurelle la plus simple consiste à représenter la forme à étudier comme une suite ordonnée de composants élémentaires:la présence ou l'absence de ces composants ainsi que leur position relative, caractériseront la forme globale. La comparaison de deux telles descriptions fournira une mesure de ressemblance entre formes, et donc une technique de reconnaissance.

Exemple :

Une ligne dans une image peut être approchée de façon discrète par une suite de vecteurs de tailles élémentaires et de directions choisies dans un ensemble fini. Un exemple en est le codage de Freeman (fig.III.1).

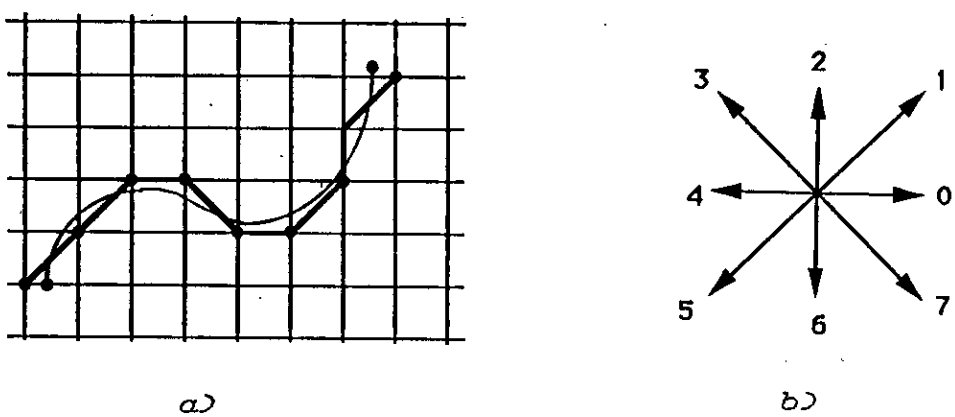


Fig.III-1 a) Description d'une forme
b) Code de Freeman.

La courbe de la figure est représentée sur l'alphabet de Freeman :

$$X = \{ 0, 1, 2, 3, 4, 5, 6, 7 \}$$

par la chaîne numérique :

1 1 0 7 0 1 2 1

III-1-2 COMPARAISON DES CHAINES

Dans le cas où une forme à étudier a fait l'objet d'une description en chaîne, il faut, pour pouvoir la reconnaître, être capable de la comparer à des prototypes décrits dans le même formalisme. Une telle comparaison suppose des algorithmes dont le but est de fournir une mesure de ressemblance entre deux chaînes écrites sur le même alphabet.

Il faut évidemment tenir compte de la signification de cette opération, et éliminer les comparaisons trop directes, comme par exemple la distance de Hamming (celle-ci suppose que la longueur des deux chaînes est identique, ce qui est une contrainte très forte et irréaliste). Nous allons présenter ici un algorithme de distance entre deux chaines qui est l'algorithme de Wagner et Fischer.

ALGORITHME DE WAGNER ET FISCHER

Les trois opérations élémentaires pour transformer une chaîne en une autre sont :

- La substitution :
  a ——> b coût $\nu(a,b)$,
- l'insertion :
  λ ——> a coût $\nu(\lambda,a)$,
- La destruction :
  a ——> λ coût $\nu(a,\lambda)$.

Ces opérations portent sur les lettres de l'alphabet, et sont affectées chacune d'un coût.

L'algorithme calcule la distance $\delta(x,y)$ entre deux chaînes x et y, c'est à dire le coût de la suite de transformations élémentaires la moins coûteuse pour

transformer x en y.Les différentes étapes de l'algorithmes sont les siuvantes :

1) $D(0,0) = 0$
2) Pour i = 1 à n
   Faire $D(i,0) = D(i-1,0) + \nu(a_i,\lambda)$
3) Pour j = 1 à m
   Faire $D(0,j) = D(0,j-1) + \nu(\lambda,b_j)$
4) Pour i= 1 à n
   Pour j =1 à m
   Faire $d_1 = D(i-1,j-1) + \nu(a_i,b_j)$
   $d_2 = D(i-1,j) + \nu(a_i,\lambda)$
   $d_3 = D(i,j-1) + \nu(\lambda,b_j)$
   $D(i,j) = Min(d_1, d_2, d_3)$
5) Distance $\delta(x,y) = D(n,m)$.

III-2 STRUCTURES DE GRAPHES

La recherche de modèles mathématiques structurels généraux a conduit les chercheurs en reconnaissance des formes à s'intéresser aux graphes. Ces outils sont en effet immédiatement utilisables pour décrire les relations dans un ensemble d'objets (par exemple, de formes primitives). De plus, ils ont été très étudiés pour leurs propriétés mathématiques et algorithmiques, car ils peuvent être utilisés dans une grande variété de problèmes.

III-2-1 DISTANCE ENTRE GRAPHES

De même qu'entre chaînes, on peut imaginer de définir une distance entre graphes, fondée sur la façon la moins chère de passer d'un graphe à un autre en combinant des opérations élémentaires.

L'algorithme de KAJITANI utilise cette définition. La complexité de cet algorithme est exponentielle en fonction de la taille des graphes, ce qui interdit son emploi pratique tel quel. Il semble que l'écriture d'un algorithme opérationnel de distances entre graphes soit un problème encore ouvert.

III-3 STRUCTURES D'ARBRES

L'utilisation des arbres comme descripteur d'image peut être très utile en reconnaissance des formes. En effet, leurs propriétés mathématiques autorisent l'utilisation d'algorithmes efficaces pour leur manipulation.

Les arbres permettent aussi de représenter une information structurelle plus abstraite comme les relations hiérarchiques entre des éléments donnés.

III-3-1 DEFINITIONS

UN arbre peut être défini, soit comme un graphe particulier, soit comme une *phrase récursive* c'est à dire une concaténation de sous-arbres.

Definition 1

On appelle graphe le couple (X,U) où :
X est un ensemble de *sommets* ou de *noeuds*.
U est un ensemble de couples ordonnés de noeuds, appelés des arcs, dont les éléments sont nommés *origine* et *extrémité*.

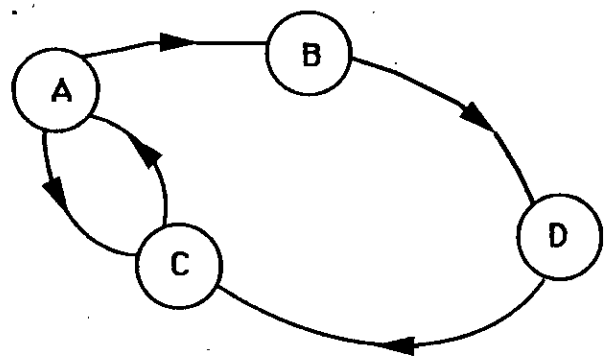
Exemple

Le graphe (X,U) avec :

$$X = \{ A, B, C, D \}$$

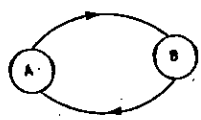
$$U = \{ (A,B), (A,C), (C,A), (B,D), (D,C) \}$$

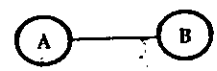
se représente par :



Definition 2

Si U possède toujours l'élément (A,B) quand il possède l'élément (B,A), le graphe est dit *non orienté*.



est representé alors 

Definition 3

On dit que deux noeuds sont *adjacents* s'il existe un arc qui les relie.

Definition 4

Un *chemin* de longueur n est une suite de noeuds $S_0, S_1, \ldots S_n$ tels que :

$\forall k \in [0, n-1]$, S_k et S_{k+1} sont adjacents; S_0 est l'origine du chemin , S_n l'extrémité.

Definition 5

Un arbre est un graphe non orienté tel qu'il existe un chemin et un seul entre tout couple de noeuds différents.

Definition 6 (définition récursive)

Un arbre est un ensemble fini X de noeuds tels que :

i) Il existe dans X un élément distingué noté $ (sommet de l'arbre)

ii) Les autres noeuds, s'il en existe, sont groupés en ensembles disjoints $X_1, X_2, \ldots X_m$ qui sont eux-même des arbres dont l'élément distingué est relié au sommet par un arc.

Definition 7

On dit qu'un noeud.S est le *fils* d'un autre noeud T quand S est la racine d'un sous-arbre de T. Réciproquement, T est dit le *père* de S.

III-3-2 PARCOURS D'ARBRES

Un algorithme de parcours dans un arbre est une procédure qui fournit en résultat la liste des noeuds de l'arbre sans oubli ni répétition. Il existe plusieurs moyens de parcourir les noeuds d'un arbre. Les trois parcours les plus importants sont les parcours préfixé, postfixé et infixé; ces parcours sont définis récursivement de la manière suivante :

. Si un arbre A est vide, la liste vide constitue le parcours préfixé, infixé et postfixé de A.

. Si A consiste en un seul noeud , la liste composée de ce noeud constitue le parcours préfixé, infixé et postfixé de A.

Sinon, supposons que A est un arbre de racine n à k sous arbres A_1, A_2, ...A_k (voir fig. III-2)

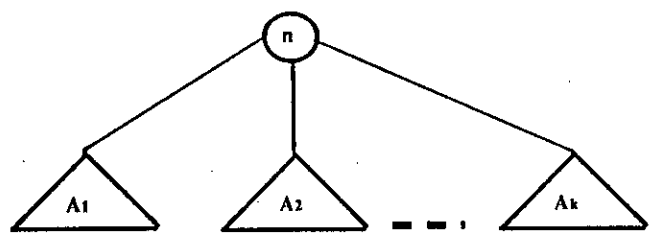


Fig. III-2

1- Pour effectuer le parcours préfixé des noeuds de A, on commence par la racine n puis on entreprend le parcours prefixé des noeuds de A_1, de ceux de A_2 et ainsi de suite jusqu'aux noeuds de A_k.

2- Pour effectuer le parcours infixé des noeuds de A, on commence par le parcours infixé des noeuds de A_1, puis on passe par la racine n, puis on effectue le parcours infixé des noeuds de A_2,..., A_k .

3- Pour effectuer le parcours postfixé des noeuds de A, on effectue le parcours postfixé des noeuds de A_1, puis de A_2 et ainsi de suite jusqu'à Ak et l'on termine par la racine n.

L'algorithme de distance entre arbre adopté, que nous verrons dans le prochain chapitre utilise le parcours postfixé, c'est pour cette raison que nous en donnons ici l'algorithme récursif.

ALGORITHME DU PARCOURS POSTFIXE

```
Procédure postfixe ( n : noeud );
début
    Pour chaque fils f de n, s'il y en a, depuis le plus à
    gauche jusqu'au plus à droite Faire
      Postfixe (f);
    Lister n ;
Fin ;
```

Nous allons à présent introduire certaines définitions et notations utilisées dans la référence [3]

III-3-3 DEFINITIONS ET NOTATIONS

Une façon de numéroter les noeuds d'un arbre permettant de respecter les différents ordres partiels cités est celle du *domaine d'arbre*; elle permet de reconstituer la structure à partir d'une liste (non ordonnée nécessairement) d'étiquettes numériques composées.

Definition 8

Un domaine d'arbre est un langage sur l'alphabet $\mathbb{N}$ des entiers naturels où la concaténation est notée par le symbole « . » dont l'élément neutre est 0.

On définit sur $\mathbb{N}$ la relation « $\leq$ » par :

$a \leq b \iff \exists x \in \mathbb{N} / a.x = b$ (c-à-d a est préfixe de b).

On définit d'autre part la relation :

$a < b \iff a \leq b$ et $a \neq b$.

On dit que a est un *ancêtre* de b et b un *descendant* de a.

Enfin, a et b sont dits *incomparables* s'ils sont tels que ni $a \leq b$, ni $b \leq a$.

Definition 9

$D \subset \mathbb{N}$ est un *domaine d'arbre* si et seulement si :

i) D est fini

ii) $(b \in D$ et $a < b) \Rightarrow a \in D$.

iii) $a.m \in D \Rightarrow a.n \in D$ pour $1 \leq n \leq m$.

avec m et n $\in \mathbb{N}$

Definition 10

Un arbre *étiqueté* défini sur D est une application α : $D \longrightarrow \Sigma$, où Σ est un ensemble fini de symboles. $N(\alpha)$ représente le nombre de noeuds dans α.

L'interprétation par domaine d'arbre permet d'attribuer de façon <u>biunivoque</u> un élément de D à chaque noeud de la structure d'arbre correspondante.

Le domaine d'arbre d'un arbre α est noté $D\alpha$.

<u>Exemple</u>

Soient $D = \{ 0, 1, 2, 1.1, 1.2 \}$, $\Sigma = \{ \$, s , t \}$

Si $\alpha(0) = \$$

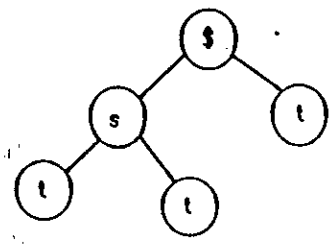
$\alpha(1) = s$

$\alpha(2) = t$

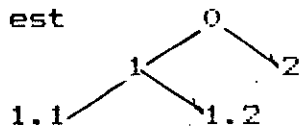
$\alpha(1.1) = t$

$\alpha(1.2) = t$

alors l'arbre α est



Dα est



N(α) = 5

Definition 11

Un *rang* est une fonction r : Dα ———>ℕ telle que r(a)=n si le noeud a de α possède n *fils*. Si a est une *feuille* alors r(a) = 0.

Definition 12

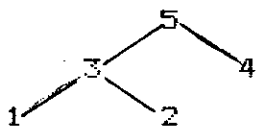
On définit un opérateur hα : Dα ———>ℕ qui ordonne une séquence de noeuds de Dα dans l'ordre postfixé (voir § III-3-2).

Exemple

Soit Dα = { 0, 1, 2, 1.1, 1.2 } alors

hα(1.1) = 1
hα(1.2) = 2
hα(1) = 3
hα(2) = 4
hα(0) = 5

En effet l'ordre postfixé de α est défini par :

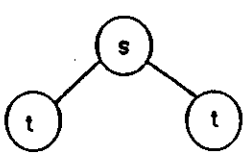


Definition 13

Soit α un arbre étiqueté, et a ∈ Dα. On note α/a le sous-arbre de α dont la racine est a.

Exemple

Soit α l'arbre défini dans les exemples précédents alors α/1 est



III-4 DEVELOPPEMENT EN ARBRES

Nous avons vu que la squelettisation est une étape très importante dans la reconnaissance et que l'efficacité de celle-ci dans la méthode structurelle dépend fortement d'une bonne squelettisation qui doit surtout préserver la connexité de l'image.

Cependant, malgré toutes les précautions prises par les auteurs de l'algorithme, il demeure certaines discontinuités dans le caractère squelettisé comme on le voit sur les figures précédentes.Ces discontinuités, quoique ne dépassent pas 2 ou 3 pixels, sont très néfastes pour notre algorithme de parcours du squelette.

3.4.1 REECHANTILLONNAGE

Pour pallier cette difficulté, nous avons utilisé une technique de *reéchantillonnage* qui a donné de bons résultats comme on le voit sur la figure III-4. Par ailleurs, cette technique permet de simplifier le codage du squelette, que nous verrons plus loin, en diminuant le nombre d'éléments à coder. Cette technique de reéchantillonnage consiste à adopter une grille d'espacement plus large (voir fig.III-3).

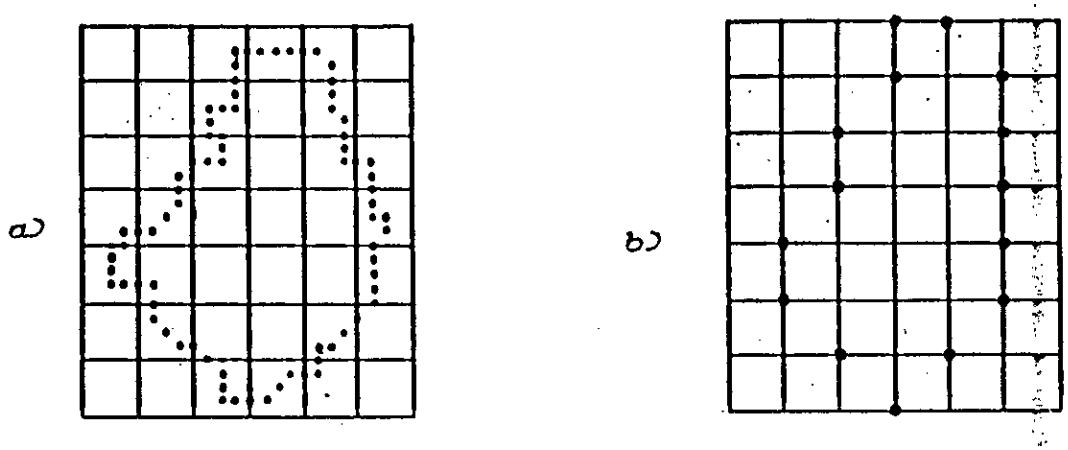


Fig III-3 Réechantillonnage a) Nouvelle grille d'espacement b) forme réechantillonnée.

Un point du nouveau squelette sera assigné à chaque noeud de la grille si ce noeud est proche du squelette original.

Nous avons adopté un pas de reéchantillonnage de trois pixels afin d'éliminer les discontinuités sans pour autant altérer la forme du caractère. L'image originale qui était sous forme d'une matrice 64x128 est donc devenue sous forme d'une matrice 22x43. L'algorithme de reéchantillonnage est le suivant :

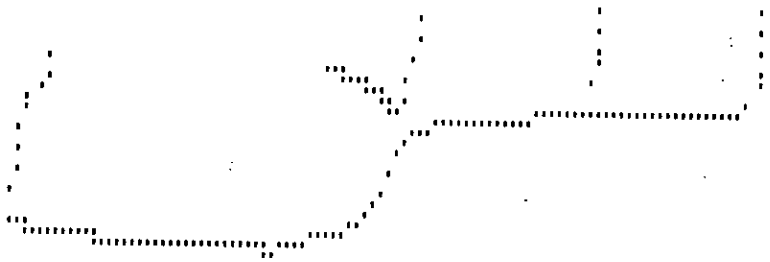
```
Entrée : matrice Dat (64,128)      ( squelette )
Sortie : matrice D (22,43)         ( squelette reéchantillonné )
Do I = 1,64
   Do J = 1,128
      If Dat (i,j) =1 then
      If D [ (i div 3), (j div 3) ] = 0
      then D [ (i div 3), (j div 3) ] = 1
   Return
Return
```

Noter qu'il est nécessaire, en premier lieu, d'initialiser la matrice D à zéro.

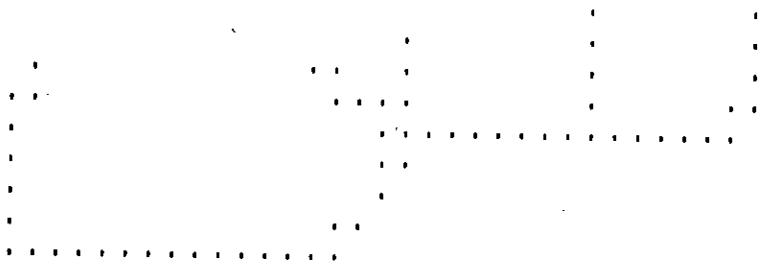
III-4-2 PARCOURS DU SQUELETTE

A ce stade des opérations, nous disposons d'une image formée d'un ou plusieurs amas de pixels allumés qui sont des ensembles connexes qu'il va falloir parcourir afin de coder le squelette du caractère selon le code de Freeman (voir fig.III-1).

- Une première étape consiste à détecter, par un balayage horizontal de l'image, le premier point du caractère qui est le point le plus haut à gauche de l'image. Ce point sera le sommet du caractère et par la même occasion le sommet de l'arbre qui va le coder.



a)



b)

Fig. II-4 Exemple de réechantillonnage a) image initiale
b) image réechantillonnée.

- Le premier point étant détecté, on cherche si un de ses 8-voisins est allumé; ce qui revient à faire une sorte de *balayage circulaire* autour du point considéré. Celui-ci est étiqueté comme *ancien point* et on se déplace vers le nouveau point détécté, on cherche dans son voisinage les points allumés en ignorant les anciens points. On se déplace ainsi, de proche en proche, en parcourant entièrement l'amas de pixels qui est un trait du caractère. Si, au voisinage d'un point, aucun pixel n'est allumé, mis à part les anciens points, celà voudra dire qu'on a atteint l'extrémité du trait.

- Si, au voisinage d'un point, il ya 2 pixels allumés, il s'agit d'un *coin* .

- Si, au voisinage d'un point, il ya 3 pixels allumés, il s'agit d'une *intersection* .

Dans ce dernier cas, les coordonnées de l'intersection sont mémorisées dans une pile; on suit l'un des deux chemins jusqu'à son extrémité, on revient ensuite à l'intersection pour parcourir le deuxième chemin.

L'algorithme de parcours du caractère est décrit par la figure III-5, où

PC : point courant

AC : ancien point

- Lors du parcours du caractère, un trait de moins de 3 pixels est négligé,et les boucles éventuelles sont ouvertes.

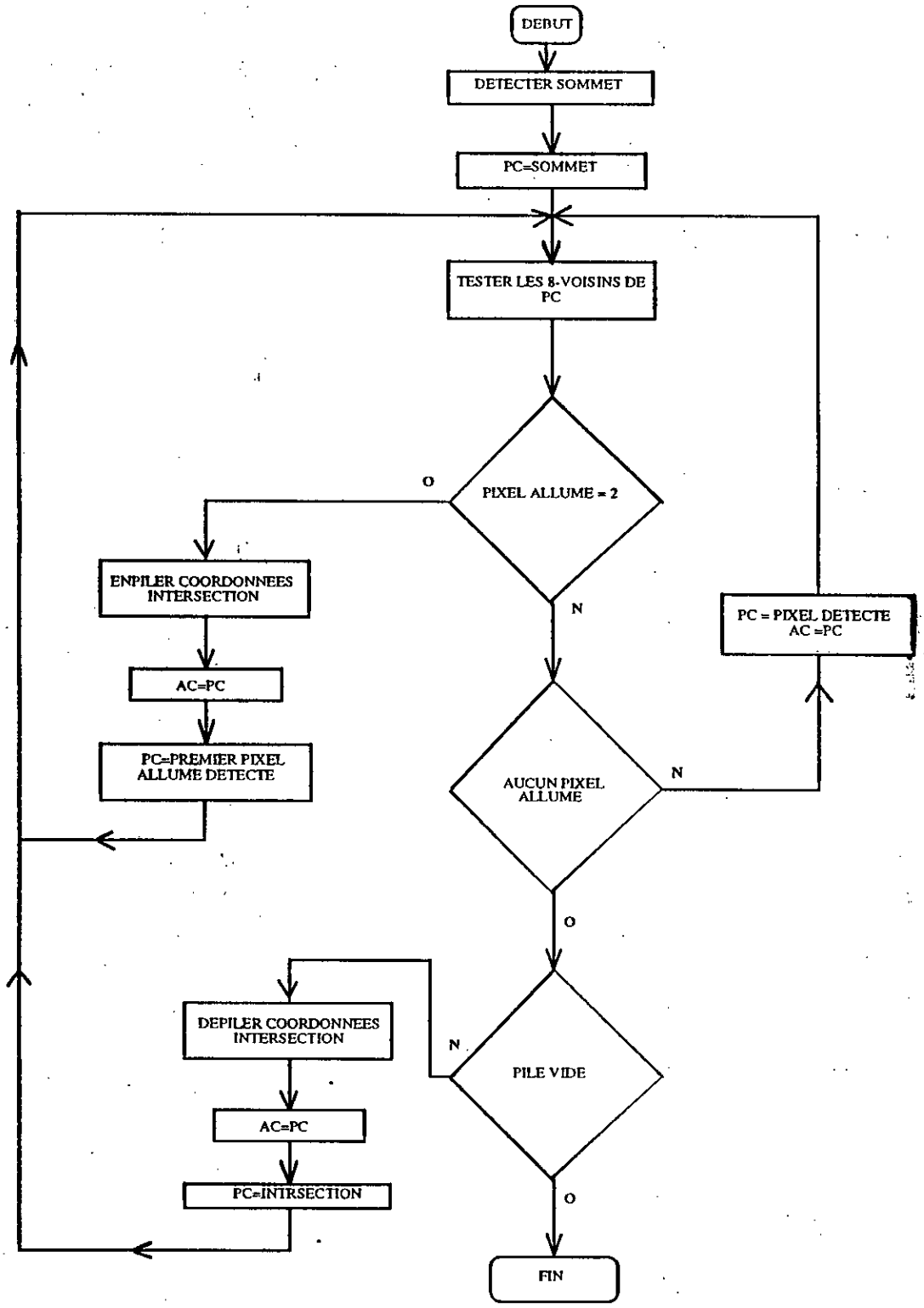


Fig. III-5 Algorithme de parcours du caractère.

III-4-3 CODAGE DU CARACTERE

Lors du parcours du caractère, chaque chemin parcouru est codé selon les primitives du code de Freeman.

P_7	P_0	P_1
P_6	P_c	P_2
P_5	P_4	P_3

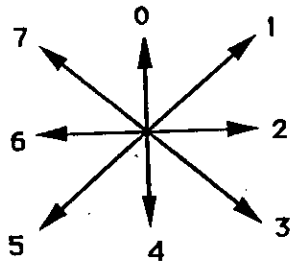


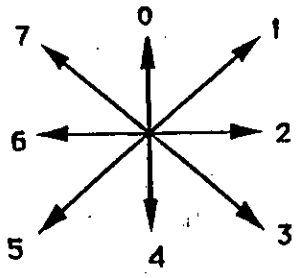
Fig. III-6 Agencement des 8-voisins et code de Freeman.

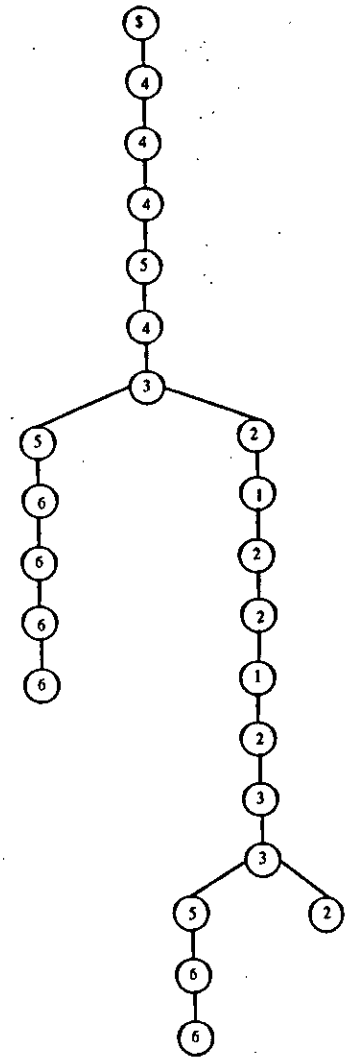
La figure III-6 représente le code de Freeman et les 8-voisins d'un *pixel courant* PC, lors du parcours du caractère, si l'un des 8-voisins, P_i, de PC est allumé alors le chemin PC-P_i est codé par i, $i \in [0,7]$.

Exemple

0	0	0
1	P_c	0
0	0	0

Le pixel allumé du voisinage de P_c est le pixel P_6, ainsi le chemin P_c-P_6 est codé par '6'. Le sommet du caractère va constituer le sommet de l'arbre correspondant. Les différents traits du caractère vont constituer les chemins de l'arbre, et les primitives du code de Freeman (direction prise lors du passage d'un point du caractère à un autre) seront les noeuds de l'arbre (voir fig.III-7).





LA REPRESENTATION EN ARBRE DU CARRACTERE :taa

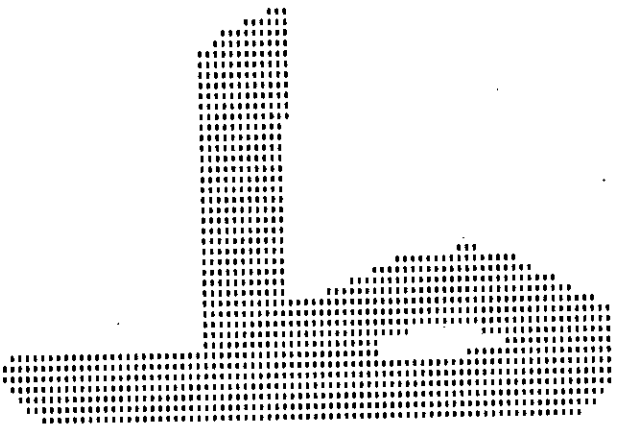


Fig. III-7 Exemple d'arbre.

CHAPITRE IV
RECONNAISSANCE

IV-1 DISTANCES ENTRE ARBRES [6]

L'intérêt de définir une distance entre arbres est de pouvoir se munir d'une metrique qui servira à comparer deux formes codées par un arbre et à décider si une forme appartient à une famille de formes.

Dans le cas qui nous intéresse, tous les caractères de l'alphabet arabe ont été codés sous forme d'arbre et stockés en mémoire dans un dictionnaire. Comme nous l'avons vu au paragraphe précédent, après la première étape qui consiste à affecter le caractère à reconnaitre à l'une des deux classes du dictionnaire, celui-ci va être ensuite comparé à chacun des caractères de la classe. Ceci nous a amené à utiliser une distance entre arbres.

Plusieurs travaux ont été effectués dans ce domaine. On peut en distinguer deux familles :

- Des metriques sur les ensembles ordonnés partiellement ont été définis par des mathématiciens algèbristes ainsi qu'une distance entre *hypergraphes*. Ces techniques peuvent s'appliquer aux arbres, ceux-ci étant en effet des ensembles où l'on peut définir une relation d'ordre partiel d'une part, d'autre part les arbres sont des cas particuliers d'hypergraphes.
- Des distances du type de celle de Wagner ont été généralisés aux arbres étiquetés.

Nous allons parler de deux algorithmes de distance qui appartiennent à la deuxième famille celle-ci étant plus pratique pour la reconnaissance des formes. Ces deux algorithmes sont ceux de Selkow et de Lu.

IV-1-1 ALGORITHME DE SELKOW

Cet algorithme dérive directement de la méthode de wagner, en considérant un arbre comme une *phrase récursive* de sous-arbres.

On se place sur la famille des arbres étiquetés sur un alphabet X, définis de la façon récursive(cf. chapitre 3) :

i) T est un ensemble de noeuds possèdant un élément particulier nommé *racine*,

ii) Les autres noeuds s'il en existe sont partitionnés en $n \geq 0$ ensembles disjoints appelés sous-arbres.

- On note $\lambda(\nu)$ l'etiquette de chaque noeud, $\nu \in T$, et $\lambda(T)$ celle de la racine de T.
- On note $T\langle i \rangle$ l'arbre obtenu en enlevant les sous-arbres $T_{i+1},\ldots,T_m$ (en particulier $T=T\langle m \rangle$).
- On dira que deux arbres A et B sont *égaux*(noté A=B) quand:

  i) $\lambda(A)=\lambda(B)$.

  ii) Si $A_1 \ldots A_n$ sont les sous-arbres de A,

  $B_1 \ldots B_m$ ceux de B,

  alors $m=n$ et $A_i=B_i$ pour $1 \leq i \leq m$

- Sur l'ensemble des arbres étiquetés, on définit maintenant trois opération.

Soit T un arbre dont les étiquettes appartiennent à l'alphabet :

$X=\{s_1,\ldots,s_q\}$

avec $\lambda(T)=s_j$,

$T_1,\ldots,T_m$ sont les sous-arbres de T.

* La première opération notée $L(s_j,s_k)$ a pour effet de transformer T en T^0, défini par :

$\lambda(T^0) = s_k$,

$T_1,\ldots,T_m$ sont les sous-arbres de T^0.

On l'appelle *changement d'étiquette*.

* La seconde, nommée *insertion de sous-arbres*, notée I(A) et appliquée à T à l'indice i ($1 \leq i \leq m$) transforme T en T^0, défini par :

$\lambda(T^0) = s_j$,

$T_1, \ldots, T_i, A, T_{i+1}, \ldots, T_m$ sont les sous-arbres de T^0.

* La troisième nommée *suppression de sous-arbre* S(T_i) transforme T en T^0, défini par :

$\lambda(T^0) = s_j$,

$T_1, \ldots, T_{i-1}, T_{i+1}, \ldots, T_m$ sont les sous-arbres de T^0.

A chaque opération est affecté un coût non négatif.

Pour l'opération de changement d'étiquette, on se donne la matrice des coûts C_L où $C_L(i,j)$ représente le coût de transformer l'étiquette s_i en l'étiquette s_j.

Pour les autres opérations, il faut d'abord se donner le coût de la suppression d'un arbre réduit à un noeud étiqueté par s_i, soit $C_S(s_i)$ et celui de son insertion : $C_I(s_i)$.

Le coût de I(A) sera alors :

$$C_I(A) = \sum_{v \in A} C_I(\lambda(v))$$

de même $C_S(A) = \sum_{v \in A} C_S(\lambda(v))$.

Une transformation d'un arbre en un autre se fera donc par une suite d'opérations élémentaires affectées chacune d'un coût. La *distance* $\delta(A,B)$ entre deux arbres étiquetés A et B sera le coût de la façon la moins coûteuse de transformer l'arbre de départ en un arbre d'arrivée. Ce coût prend en compte à la fois les modifications de structure et celles d'étiquetage.

Selkow démontre (de la même façon que Wagner) le théorème suivant :

Théorème

soient deux arbres A (de sous-arbres $A_1,\ldots,A_m$) et B (de sous-arbres $B_1,\ldots,B_n$).

On a :

i) $\delta(A\langle 0\rangle, B\langle j\rangle) = C_L(\lambda(A),\lambda(B)) + \sum_{k=1}^{j} C_I(B_k)$,

ii) $\delta(A\langle i\rangle, B\langle 0\rangle) = C_L(\lambda(A),\lambda(B)) + \sum_{k=1}^{i} C_S(A_k)$,

iii) $$\delta(A\langle i\rangle, B\langle j\rangle) = \begin{cases} \delta(A\langle i-1\rangle, B\langle j-1\rangle) + \delta(A_i, B_j) \\ \delta(A\langle i\rangle, B\langle j-1\rangle) + C_I(B_j) \\ \delta(A\langle i-1\rangle, B\langle j\rangle) + C_S(A_i). \end{cases}$$

La démonstration s'appuie sur le fait que les opérations ne changent jamais l'ordre des sous-arbres d'où un calcul du type *programmation dynamique* (cf. § IV-3-3).

Un algorithme récursif s'en déduit alors immédiatement. L'auteur évalue sa complexité en définissant la *signature* d'un arbre comme le vecteur $(t_0,\ldots,t_i,\ldots,t_d)$ où t_i est le nombre de noeuds de T au niveau de i. Pour les arbres A et B de signatures :

$(1,a_2,\ldots,a_p)$,

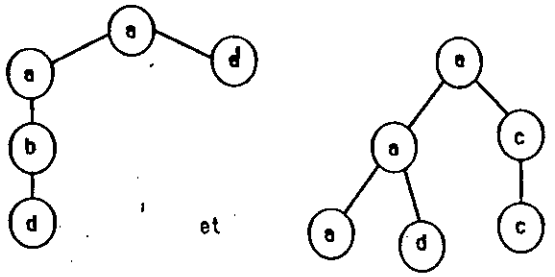
et $(1,b_2,\ldots,b_q)$.

Le nombre de calculs est en :

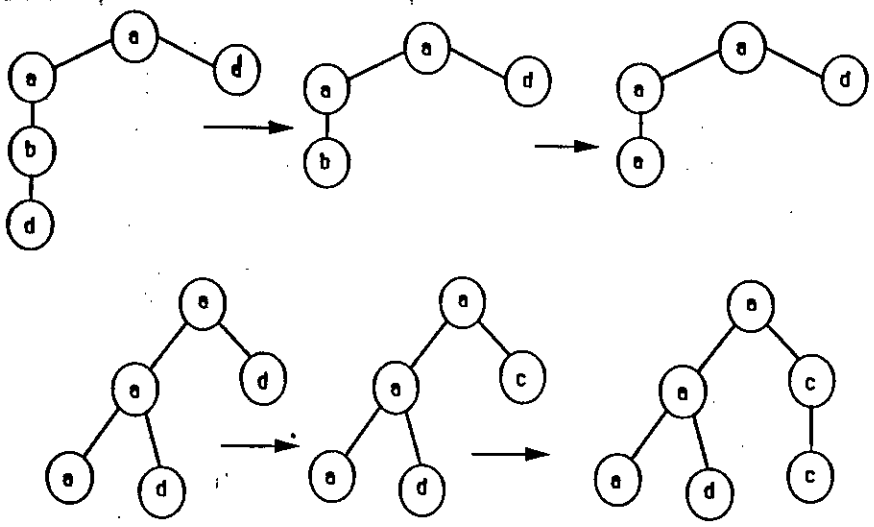
$$O(\sum_{i=1}^{Min(p,q)} a_i b_i).$$

Cette distance est caractérisée par le fait qu'elle procède par insertion et suppression de sous-arbres entiers.

La distance entre les deux arbres suivants :

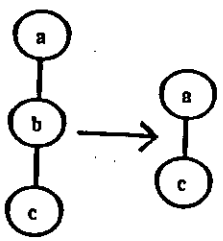


se calcule par la suite d'opérations :



IV-1-2 ALGORITHME DE LU

Cette autre mode de calcul de distance entre arbres est fondée sur le même principe, mais admet des transformations élémentaires plus générale que dans celui de Selkow. En particulier, une opération du type :



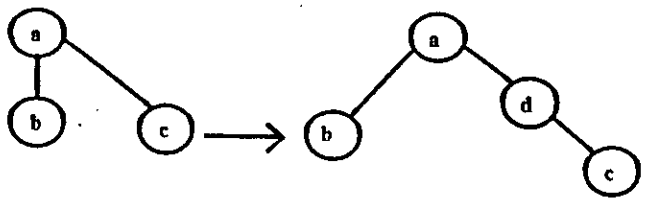
sera considérée ici comme élémentaire (suppression d'un noeud, même si ce n'est pas une feuille).

Les opérations sont définis comme suit :

i) Suppression d'un noeud n'importe où dans l'arbre (cf.exemple précédent).

ii) Insertion d'un noeud.

Par exemple :



iii) Changement de l'étiquette d'un noeud.

A chacune de ces opérations est associé un coût, qui ne dépend que de cette opération (par exemple, l'insertion n'importe où de n'importe quel noeud coûte toujours la même chose).

La distance entre deux arbres est alors définie comme le coût de la suite d'opérations la moins coûteuse transformant un arbre en l'autre, et vérifiant les contraintes suivantes :

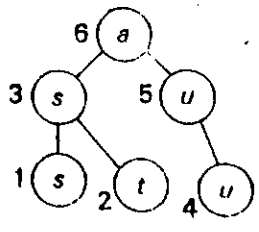
Soient deux noeuds A et B d'étiquettes respectives a et b, transformés par la suite des opérations en noeuds A' et B', d'étiquettes a' et b'.

1) $A = B \Leftrightarrow T(A) = T(B)$.

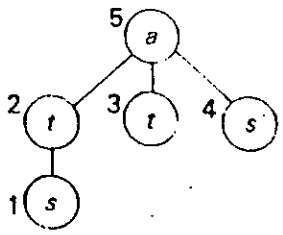
2) La transformation T preserve l'ordre filial.

3) La transformation T preserve l'ordre postfixé.

Exemple :



arbre β



arbre α

Transformation valide entre α et β :

arbre β	arbre postfixé	1	2	3	4	5	6
	étiquette	s	λ	t	s	u	u	a
arbre α	étiquette	s		t	t	λ	λ	s	a
	arbre postfixé	1		2	3		4	5

coût : 4

transformation non valide :

arbre β	arbre postfixé	1	2	3	4	5	6
	étiquette	s	t	s	u	u	a
arbre α	étiquette	s	t	t	s	λ	a
	arbre postfixé	1	2	3	4		5

L'algorithme utilise la programmation dynamique pour construire une matrice D(i,j) où i et j sont les rangs (dans l'ordre postfixé) des noeuds des arbres α et β; D(i,j) est la valeur de la distance entre les sous-arbres de α dont la racine a pour rang i et le sous-arbre de β dont la racine a pour rang j. La complexité est en O(n.m) où n (resp. m) est le nombre de noeuds de l'arbre α (resp. β). nous allons voir à présent l'implémentation de cet algorithme.

IV-2 IMPLEMENTATION DE L'ALGORITHME DE LU [3]

Avant de voir l'implémentation de l'algorithme précédent, nous allons introduire quelques définitions et notations utilisées par l'auteur.

IV-2-1 DEFINITIONS

Definition 1

Soient α et β deux arbres, une transformation $T:D\beta\text{----->}D\alpha$ consiste en l'un des trois types de transformations suivantes :

$-\varepsilon(b)$ -----> λ,p
$-\phi(\lambda)$ -----> a,q
$-\sigma(b)$ -----> a,r

où $b \in D\beta$, $a \in D\alpha$, λ est l'arbre vide (ou symbole nul), et p, q et r les coûts associés respectivement à ε, ϕ et σ.
Une transformation ε ou ϕ provoque respectivement une suppression ou une insertion. Une transformation σ, $\sigma(b)\text{----->}a$ provoque une substitution (ou changement d'étiquette)si les etiquettes des noeuds a et b ne sont pas identiques. Dans ce cas r est non nul, sinon r=0.

Définition 2

La distance entre deux arbres α et β, notée $d(\alpha,\beta)$ est le coût minimal nécessaire pour transformer l'arbre β en l'arbre α en utilisant des séries de transformation T satisfaisant les critères suivants :
1) Si $a<b$ dans $D\beta$, alors $T(a)<T(b)$: si a et b sont incomparables alors T(a) et T(b) sont incomparables.
2) Si $a=b$ dans $D\beta$, alors $T(a)=T(b)$; Si $a\neq b$ alors $T(a)\neq T(b)$.
3) Si $h\beta(a)<h\beta(b)$ alors $h\alpha(T(a))<h\alpha(T(b))$, T(a) et T(b) n'étant pas des symboles nuls.

En d'autres termes, les critères précédents peuvent être

énoncés comme suit :

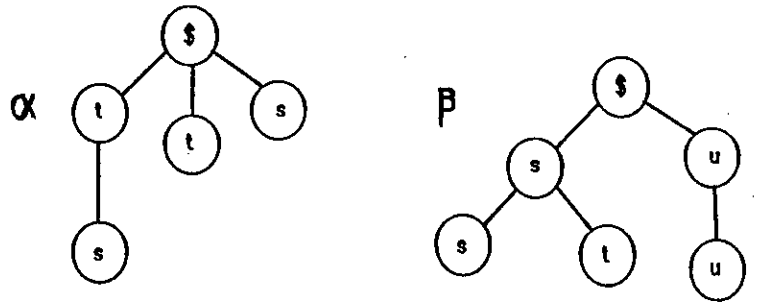
1) La transformation T doit preserver l'ordre filial.

2) Les noeuds de l'arbre β ne doivent pas fissionner ou fusionner.

3) La transformation T doit preserver l'ordre postfixé.

Si on affecte la valeur unité aux coûts associés aux transformations ε et ϕ (suppression et insertion) et à la transformation σ si celle-ci provoque une substitution, alors la distance devient le nombre minimal de suppressions, insertions et substitutions nécessaires pour obtenir un arbre d'un autre.

Exemple : soient deux arbres α et β



considérons l'ensemble de transformation T_1,

T_1 : $D\beta$ -----> $D\alpha$

$\sigma(0)$ -----> 0,0

$\varepsilon(1)$ -----> λ,1 (suppression)

$\sigma(1.1)$ -----> 1.1,0

$\sigma(1.2)$ -----> 2,0

$\phi(\lambda)$ ------> 1,1 (insertion)

$\sigma(2)$ ------> 3,1 (substitution)

$\varepsilon(2.1)$ -----> λ,1 (suppression)

Soit 1 le coût respectif d'une suppression, insertion, et substitution. Alors, T_1 est la séquence de coût minimal qui satisfasse le critère de la définition 2, donc $d(\alpha,\beta)=4$.

Considérons maintenant un autre ensemble de transformations :

T_2 : $D\beta$ -----> $D\alpha$ $\quad\sigma(0)$ -----> 0,0

$\sigma(1)$ -----> 1.1,1

$\sigma(1.1)$ -----> 3,0

$\sigma(1.2)$ -----> 1,0

$\sigma(2)$ ------> 2,1 (substitution)

$\varepsilon(2.1)$ -----> λ,1 (suppression)

Bien que le coût associé à T_2 est 2,T_2 ne satisfait pas la définition 2. Par exemple, on a :

$T_2(1) = 1.1$ et $T_2(1.2) = 1$,

ainsi $T_2(1) > T_2(1.2)$, mais $1 < 1.2$. L'ordre filial n'est pas preservé par T_2.

IV-2-2 IMPLEMENTATION

L'algorithme utilise la programmation dynamique pour calculer les éléments d'une matrice de distances, D(i,j), où i et j sont les indices de α et β, respectivement, dans l'ordre postfixé. Soit $a=h_\alpha^{-1}(j)$, et $b=h_\beta^{-1}(i)$, i.e.,a et b sont des noeuds de α et β, où $h_\alpha(a)=j$, $h_\beta(b)=i$. Alors, D(i,j) est le coût minimal nécessaire pour transformer le sous-arbre β/b en α/a et qui satisfait le critère de la définition 2.

Algorithme : distance entre arbres

Entrée : deux arbres α et β. Les nombres de noeuds de α et β sont $N(\alpha)=m$, $N(\beta)=n$.

Sortie : $d(\alpha,\beta)$, la distance entre α et β.

Méthode : Soit D une matrice $(m+1)\times(n+1)$.

1) D(0,0)=0

2) Do j=1,m

2.1) D(0,j)=N(α/a)×q, où a=hα(j)

return

3) Do i=1,n

3.1) D(i,0)=N(β/b)×p, où b=hβ(i)

return

4) Do i=1,n

4.1) Do j=1,m où a=hα(j) et b=hβ(i)

4.1.1) E(0,0)=0

4.1.2) Do l=1,t où t=r(a)

4.1.2.1 E(0,l)=E(0,l-1) + N(α/a.l)×q

return

4.1.3) E(0,t+1)=E(0,t) + q

4.1.4 Do k=1,s où s=r(b)

4.1.4.1 E(k,0)=E(k-1,0) + N(β/b.k)×p

return

4.1.5) E(s+1,0)=E(s,0) + p

4.1.6) Do k=1,s

4.1.6.1) Do l=1,t

4.1.6.1.1) E(k,l)=

$$\text{Min}\begin{cases} E(k-1,l) + N(\beta/b.k)\times p \\ E(k,l-1) + N(\alpha/a.l)\times q \\ E(k-1,l-1) + D(u,v) \end{cases}$$

où u=hβ(b.k) et

v=hα(a.l)

return

return

4.1.7) Do l=1,t

4.1.7.1) E(s+1,l) =

$$\text{Min}\begin{cases} E(s+1,l-1) + N(\alpha/a.l)\times q \\ E(s,l) + p \\ E(0,l-1) + D(i,v) \end{cases}$$

où v=hα(a.l)

return

4.1.8) Do k=1,s

4.1.8.1) $E(k,t+1) =$

$$\text{Min} \begin{cases} E(k-1,t+1) + N(\beta/b.k) \times p \\ E(k,s) + q \\ E(k-1,0) + D(u,j) \end{cases}$$

où $u=h\beta(b.k)$

return

4.1.9) $D(i,j) =$

$$\text{Min} \begin{cases} E(s,t+1) + p \\ E(s+1,t) + q \\ E(s,t) + r \end{cases}$$

où $r>0$ si $\alpha(a) \neq \beta(b)$, $r=0$ sinon

return

return

5) $d(\alpha,\beta) = D(n,m)$

6) End.

Remarque : Dans cet algorithme, les boucles Do des étapes 4.1.2), 4.1.6) et 4.1.7) doivent être sautées si t=0, ainsi que les boucles Do des étapes 4.1.4), 4.1.6), et 4.1.8) si s=0.

IV-2-3 PROGRAMMATION DYNAMIQUE [1]

Au fil des ans, les informaticiens ont répertorié un certain nombre de techniques générales donnant des solutions efficaces à de grands ensembles de problèmes de même nature. L'une des techniques les plus importantes et les plus répandues d'élaboration d'algorithmes est une stratégie appelée *diviser pour résoudre*; le principe en est de diviser un problème de taille n en un ensemble de sous-problèmes plus petits de manière que la solution de chaque sous-problème facilite la construction du problème entier. Cependant cette subdivision s'avère souvent impossible. Dans de tels cas, on morcelle le problème en autant de sous-problèmes qu'il semble

nécessaire, mais ces sous-problèmes doivent eux-mêmes être subdivisés en sous-problèmes et ainsi de suite. Si l'on n'y prend garde, on obtient une solution du problème original de coût exponentiel.

Malgré tout, il n'existe fréquemment qu'un nombre polynomial de sous-problèmes; on doit en résoudre certains de nombreuses fois. Si, à la place nous gardions trace de la solution de chaque sous-problème résolu et regardions simplement la réponse quand cela devient nécessaire, nous retomberions sur un algorithme de coût polynomial.

Il est parfois plus simple du point de vue de la mise en oeuvre de créer une table des solutions de tous les sous-problèmes susceptibles d'être rencontrés. On remplit cette table sans chercher à savoir si la solution globale est liée à la résolution d'un problème donné ou non. Le remplissage d'une table de sous-problèmes en vue de la solution d'un problème donné s'appelle *programation dynamique*, un terme qui provient de la théorie du contrôle des processus.

La forme d'un algorithme dynamique peut varier du tout au tout, mais tous ont en commun le remplissage d'une table et le respect de l'ordre dans lequel les éléments doivent y être insérés.

Cette technique, comme nous l'avons vu, a été utilisée dans l'algorithme de LU où les distances D(i,j) entre tous les sous-arbres sont calculées dans l'ordre postfixé (et cela même si les sous-arbres en question ne subissent aucune transformation auquel cas D(i,j) = 0) jusquà atteindre D(n,m) = d(α,β) qui est la distance entre les arbres α et β.

IV-3 RECONNAISSANCE ET RESULTATS EXPERIMENTAUX

La méthode de décision que nous avons utilisée est deterministe et est basée sur la distance de LU.

IV-3-1 APPRENTISSAGE ET DECISION

Comme on l'a vu au chapitre précédent, chaque caractère de l'alphabet arabe a été codé sous forme d'arbre, les arbres obtenus sont les arbres *types* des caractères arabes et vont constituer *le dictionnaire*.

La phase de construction du dictionnaire constitue la phase d'apprentissage. Ce dictionnaire est subdivisé en deux classes, l'une contenant les caractères qui possèdent un *groupe de points* (ex: CHIN, NOUN etc...), l'autre contenant les caractères qui n'en possèdent pas (ex: AIN, SIN etc..).

Les caractères appartenant à la première classe diffèrent eux-mêmes selon le nombre et la position dans l'image des points qu'ils possèdent. Une telle image contient deux ou plusieurs amas de pixels allumés selon que le caractère possède un ou plusieurs points.

Le caractère étant parcouru à partir du premier pixel allumé, deux cas peuvent être rencontrés :

- Le pixel rencontré appartient au corps du caractère lui-même qu'on appellera amas du *premier type*.
- Le pixel rencontré appartient au groupe de points du caractère qu'on appellera amas du *second type*.

Pour faire la distinction entre ces deux cas, il suffit de compter le nombre de pixels de l'amas rencontré. En effet, la dimension d'un groupe de points est en général petite par rapport à la dimension du corps principal du caractère. Le rapport de ces deux dimension est au maximum égal à 0.5, ce

qui revient à dire qu'un groupe de points occupe au maximum le tiers du caractère entier.

La règle de décision qui permet de classer une forme dans l'une ou l'autre des deux classes du dictionnaire suit l'algorithme suivant (voir fig.IV-1):

- Le premier amas de pixels rencontré est un amas du second type (ex: CHIN, THA, NOUN).
- Le premier amas de pixels rencontré est un amas du premier type, mais l'image contient encore un nombre non négligeable de pixels allumés n'appartenant pas à cet amas (ex: YAA, BA, DJIM).

Si un caractère répond à l'une de ces deux conditions, il sera alors identifié comme appartenant à la première classe du dictionnaire; sinon il sera affecté à la deuxième classe.

Un caractère inconnu, codé lui-même de la même manière que les caractères du dictionnaire sera ensuite comparé aux caractères de sa classe en utilisant la distance de LU. Il sera associé au caractère du dictionnaire le plus proche c'est à dire celui dont la distance au caractère considéré est la plus petite.

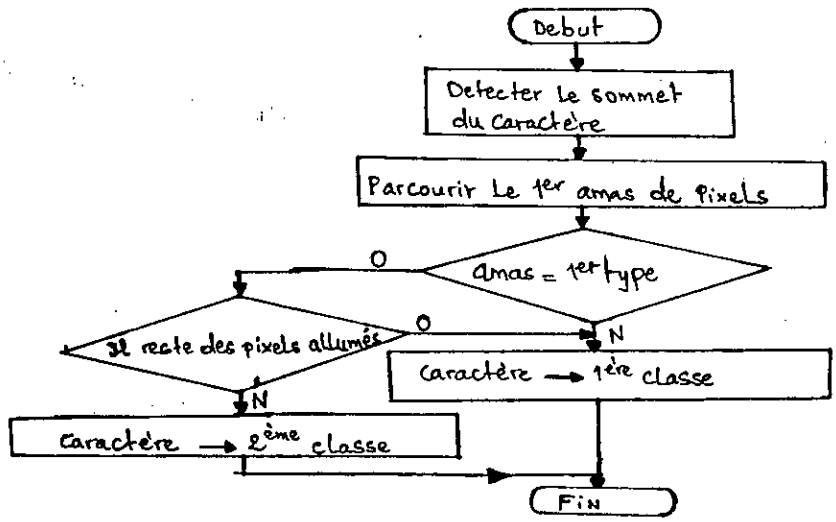


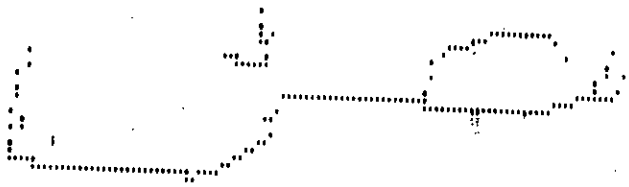
Fig.IV-1 Algorithme de classification du dictionnaire.

IV-3-2 RESULTATS EXPERIMENTAUX

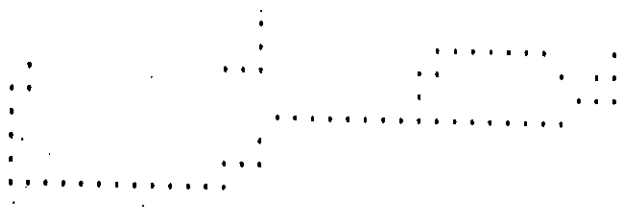
Le programme que nous avons élaboré (voir fig.VI-2) permet d'acquérir un caractère de la mémoire de masse où il est stocké sous la forme d'une matrice de taille 128x128, les éléments de cette matrice représentent les niveaux de gris de chaque pixel de l'image. Celle-ci est ensuite binarisée pour devenir une image à deux niveaux de gris, le niveau haut représentant le caractère et le niveau bas le fond de l'image. Ce caractère est ensuite squelettisé puis réechantillonné comme on l'a vu au chapitre 2. Après le parcours du squelette le caractère est discrètisé en longueur et en direction et codé selon le code de Freeman sous forme de chaines numériques représentant chacune un trait du caractère; ces chaines vont constituer les branches de l'arbre qui va représenter le caractère. La phase de reconnaissance va permettre ensuite d'identifier ce dernier.

Tous les caractères dont nous disposions ont été reconnus mais, faute de moyens, nous n'avons pas pu scanner d'autres fontes de caractères afin de les soumettre à l'expérience. Néanmoins nous avons quand même essayé de *fabriquer* des caractères; nous en avons donc simulé plusieurs qui ont tous été reconnus (voir les exemples de la figure IV-3) mais il a fallu ajuster certains paramètres de la distance de LU que sont, d'un côté, les coûts d'une suppression et d'une insertion, et, d'un autre côté le coût d'une substitution .Nous avons remarqué que le caractère n'était reconnu que lorsque le coût d'une substitution était supérieur à celui d'une insertion ou d'une suppression. Ceci démontre qu'il faut attacher beaucoup plus d'importance à la structure géometrique du caractère (coût d'une substitution) qu'aux dimensions de celui-ci (coût d'une suppression ou d'une insertion), ce qui suit la *philosophie* même de la reconnaissance structurelle.

Acquisition



Squelettisation



Echantillonnage

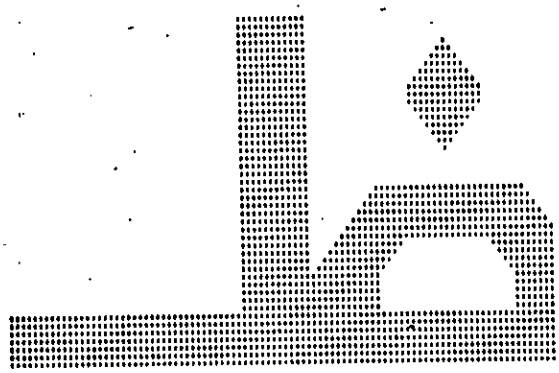
Acquisition **Squelettisation** **Echantillonnage** **Reconnaissance**

```
                    $
              5           2
              4           2
        4           3     2
        6           2     3
        6           2
  7             5   2
  1             6   1
                6
                6
                6
                6
        7           6
        1
        0
```

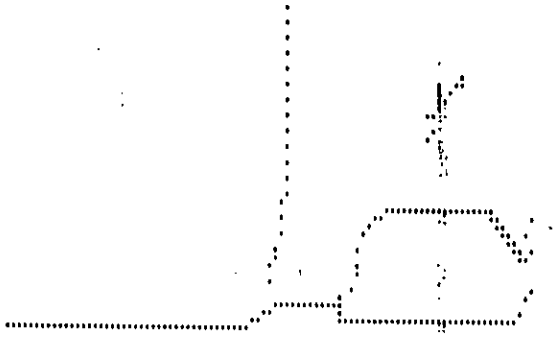
Caractère identifié : sad

Developpement en arbre et reconnaissance

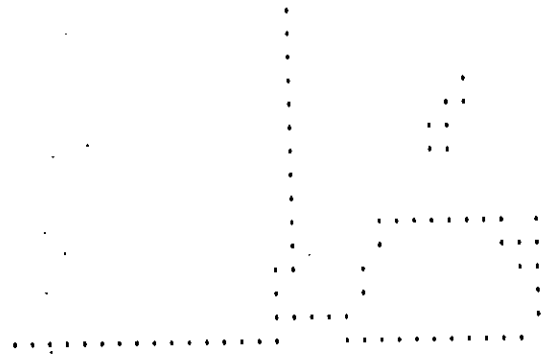
Fig. IV-2 Une exécution du programme.



Acquisition



Squelettisation



Echantillonnage

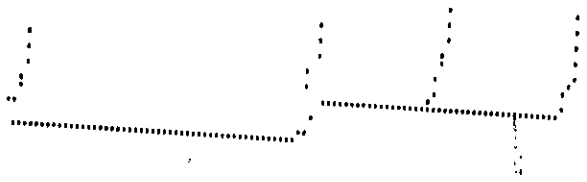
Acquisition	Squelettisation	Echantillonnage	Reconnaissance
	$		
	4		
	4		
	4		
	4		
	5		
	3		
5		2	
6		1	
6		0	
6		0	
6	1		0
6	2		
	2		
	3		
2			
4			
5			
6			
6			
6			

Caractère identifié : thad

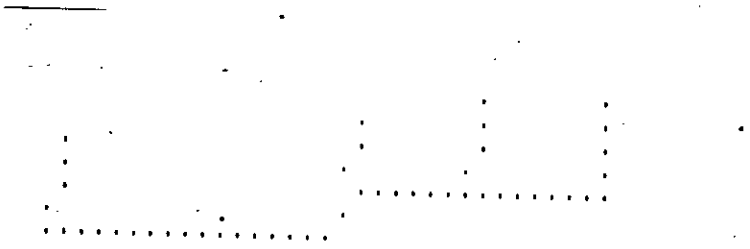
Developpement en arbre et reconnaissance

Fig IV-3 Reconnaissance des caractères simulés.

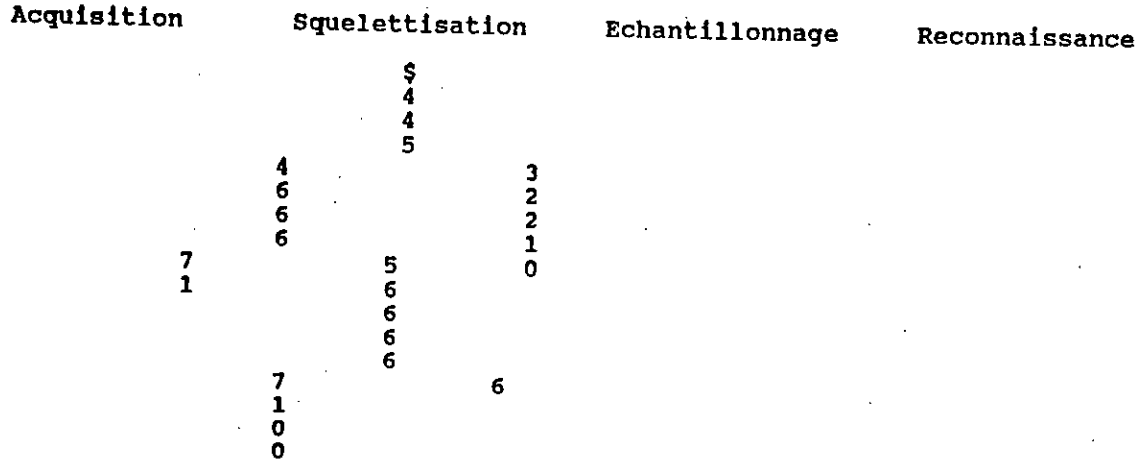
Acquisition



Squelettisation



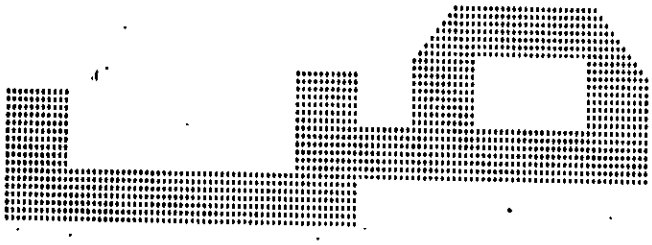
Echantillonnage



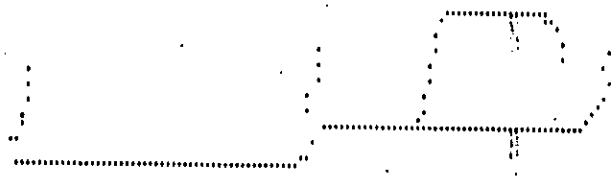
Caractère identifié : sin

Developpement en arbre et reconnaissance

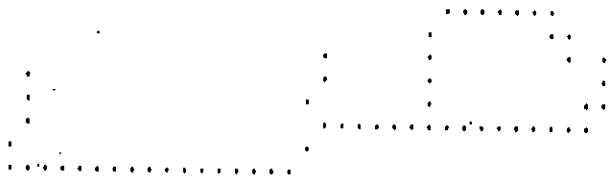
Fig. IV-3 (Suite).



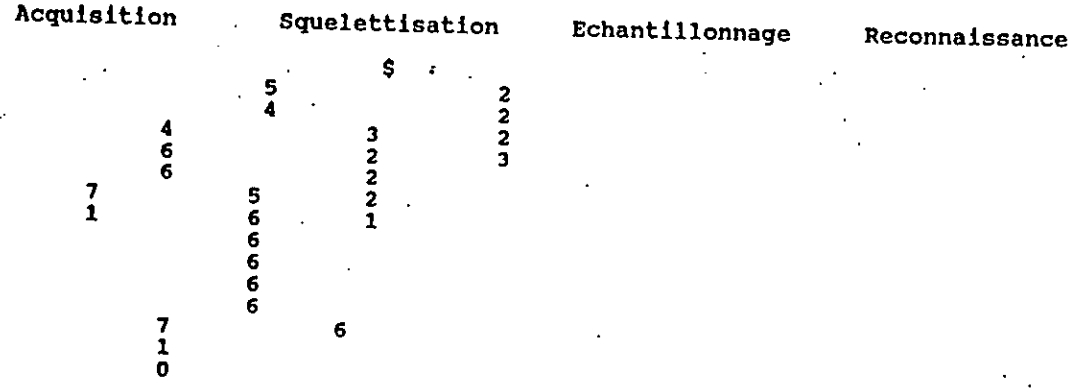
Acquisition



Squelettisation



Echantillonnage



Developpement en arbre et reconnaissance

Fig. IV-3 (Suite).

CONCLUSION

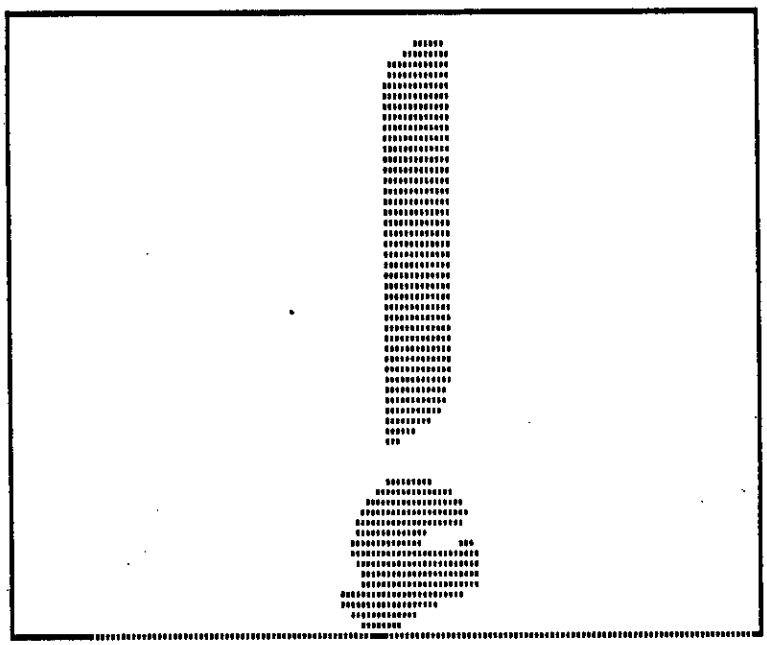
CONCLUSION

Le travail dont nous avons été chargés de faire était l'étude de la méthode des arbres pour la reconnaissance des formes appliquée aux caractères de l'alphabet arabe. C'est une méthode structurelle, descriptive, qui utilise des descripteurs d'images puissants et efficaces que sont les arbres. Ces descripteurs se prêtent d'autant plus aux caractères arabes qui sont caractérisés par une structure qui possède beaucoup de *croisemments*, de *courbures*, et de *changements de direction*. Cependant l'opération de squelettisation que nécessite cette méthode nous a causé beaucoup de problèmes . En effet cette opération est très coûteuse en temps de calcul; d'autre part, elle ne conserve pas toujours la connexité des formes. Nous avons résolu ce dernier problème avec une certaine réussite (voir § III-4) en utilisant une technique simple et rapide qui est le *réechantillonnage* de l'image. La distance entre arbres que nous avons utilisée est efficace et rapide en temps de calcul, en témoigne la reconnaissance à 100 % des caractères du dictionnaire. Les experiences que nous avons tenté sur d'autres caractères ont été concluantes et prouvent que cette méthode est très prometteuse pour la reconnaissance des caractères arabes manuscrits.Un problème important qui pourrait faire l'objet de travaux ultérieurs serait l'étude de la variabilité de cette méthode à la taille et à l'orientation des caractères. Enfin, nous souhaitons que ce travail contribue un tant soit peu à l'évolution de la recherche dans ce domaine.

ANEXE

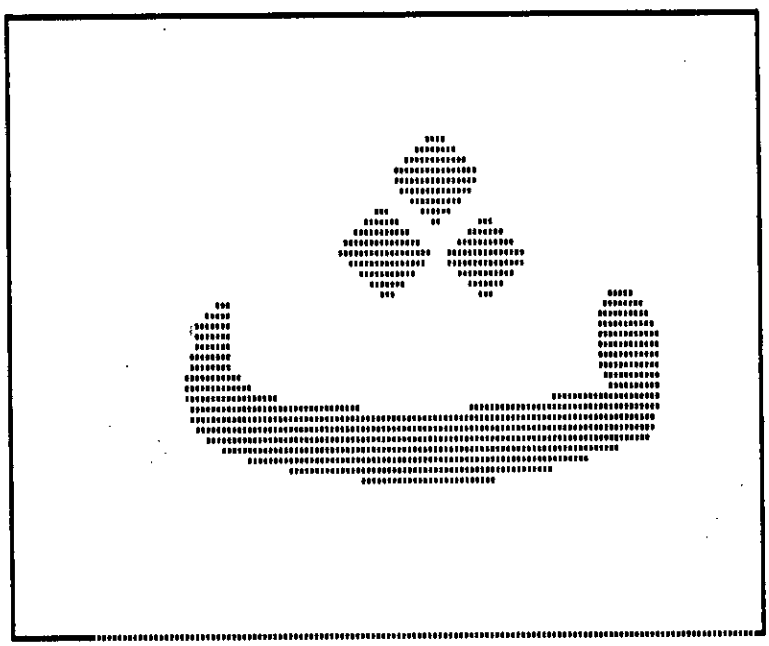
ANNEXE / Caractère et son developpement en arbre.

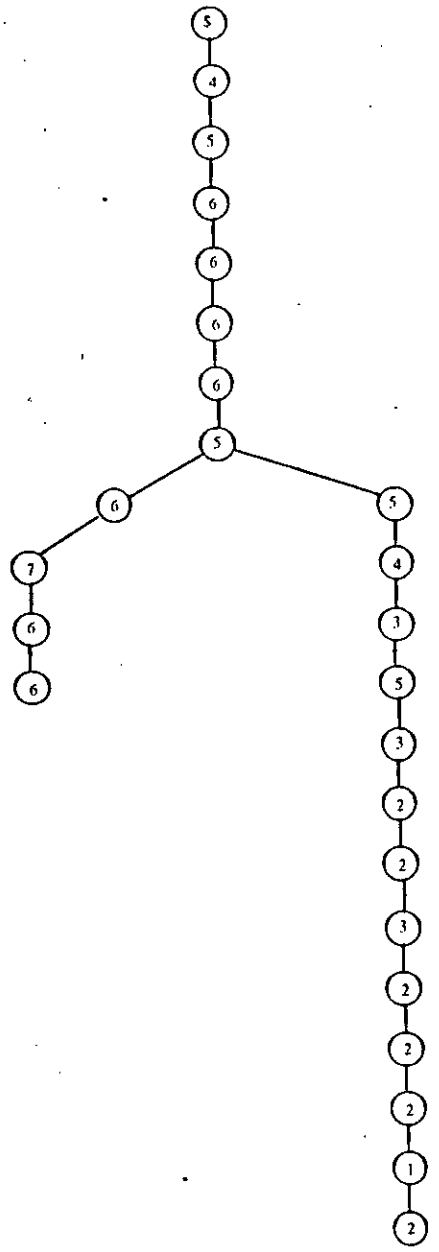
LA REPRESENTATION EN ARBRE DU CARRACTERE : ALIF



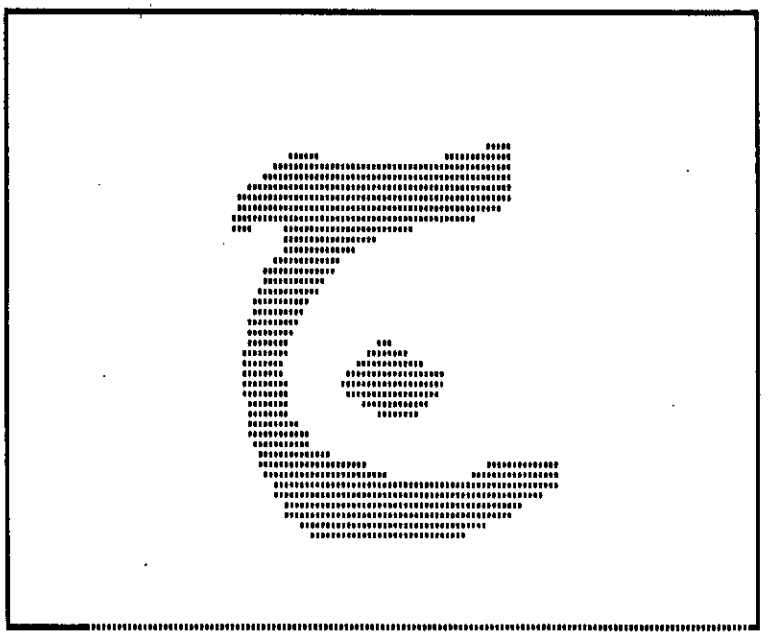
ANNEXE / Caractère et son developpement en arbre.

LA REPRESENTATION EN ARBRE DU CARRACTERE : THA

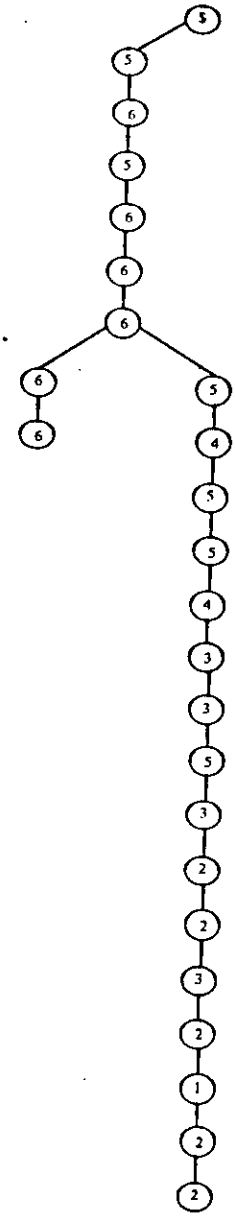




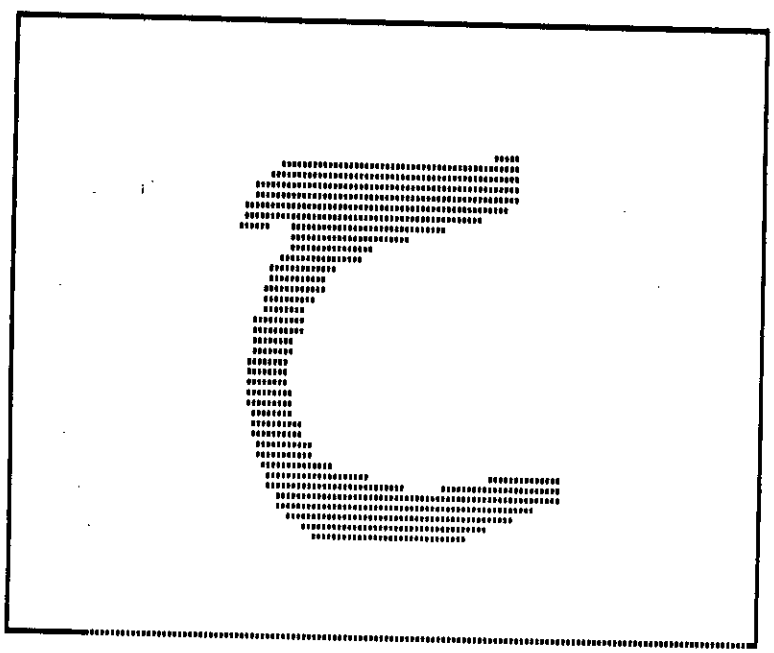
LA REPRESENTATION EN ARBRE DU CARRACTERE : DJIM



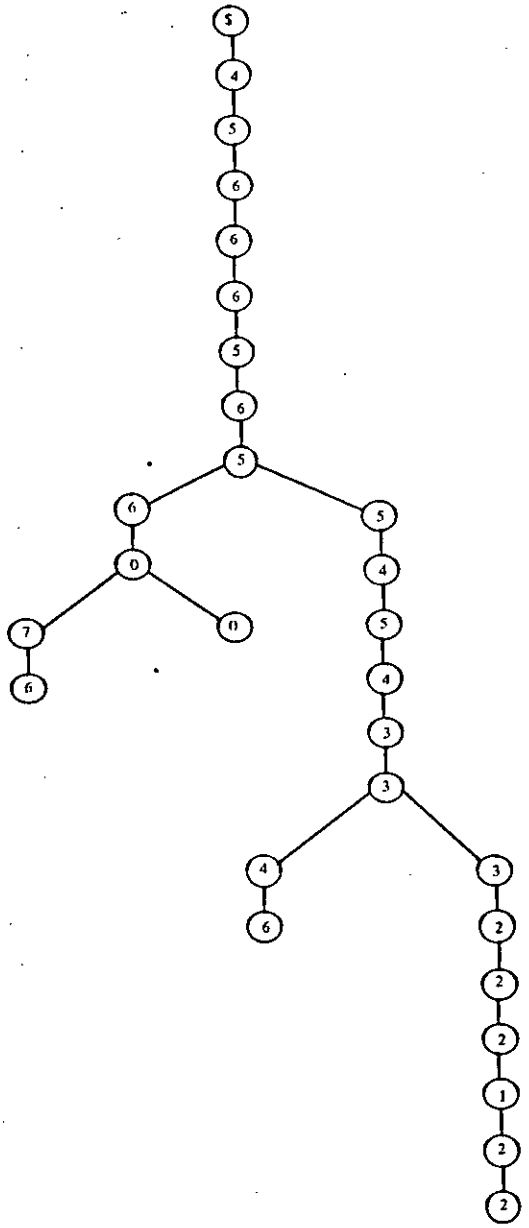
ANNEXE / Caractère et son developpement en arbre.



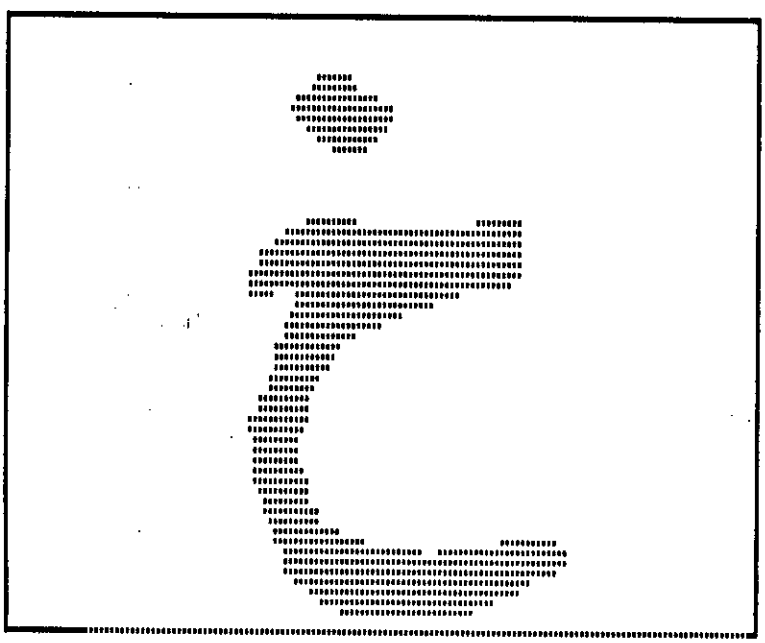
LA REPRESENTATION EN ARBRE DU CARRACTERE : HA



ANNEXE / Caractère et son developpement en arbre.

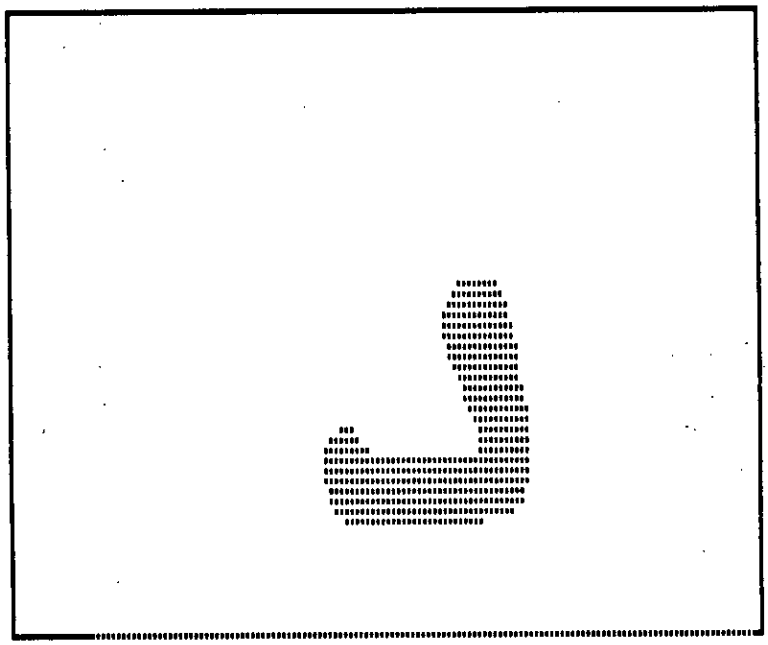


LA REPRESENTATION EN ARBRE DU CARRACTERE : KHA



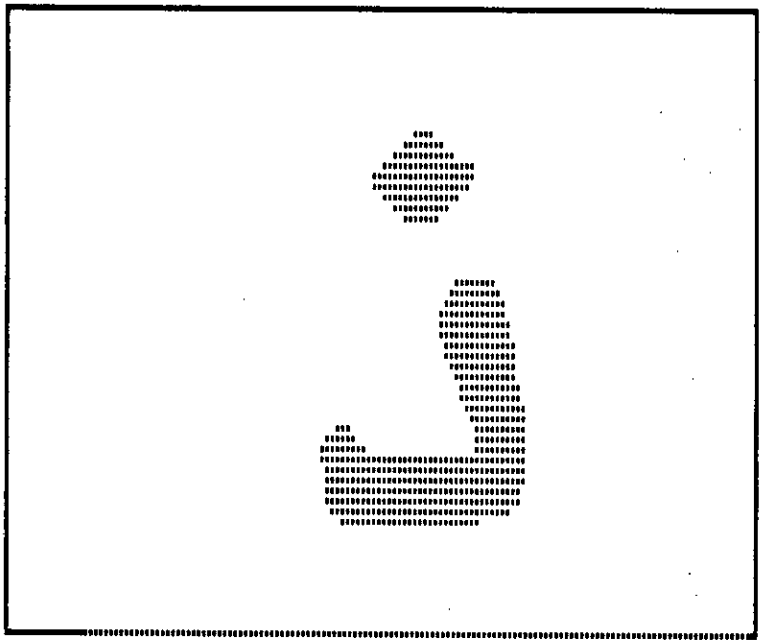
ANNEXE / Caractère et son developpement en arbre.

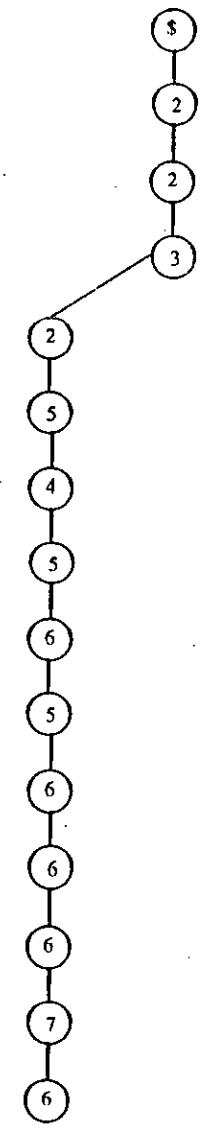
LA REPRESENTATION EN ARBRE DU CARRACTERE :DAL

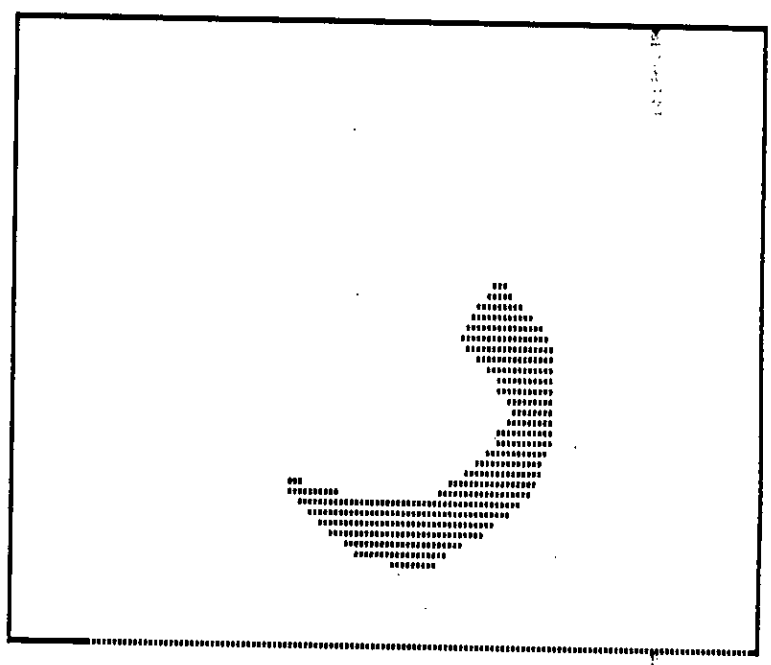


ANNEXE / Caractère et son developpement en arbre.

LA REPRESENTATION EN ARBRE DU CARRACTERE :DHAL

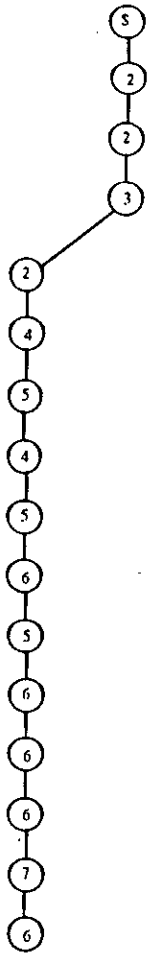




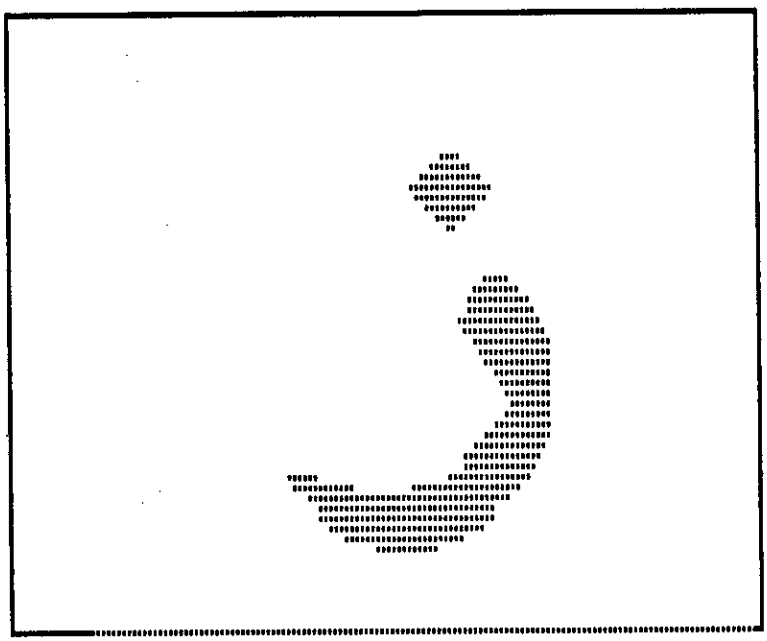


LA REPRESENTATION EN ARBRE DU CARRACTERE :RA

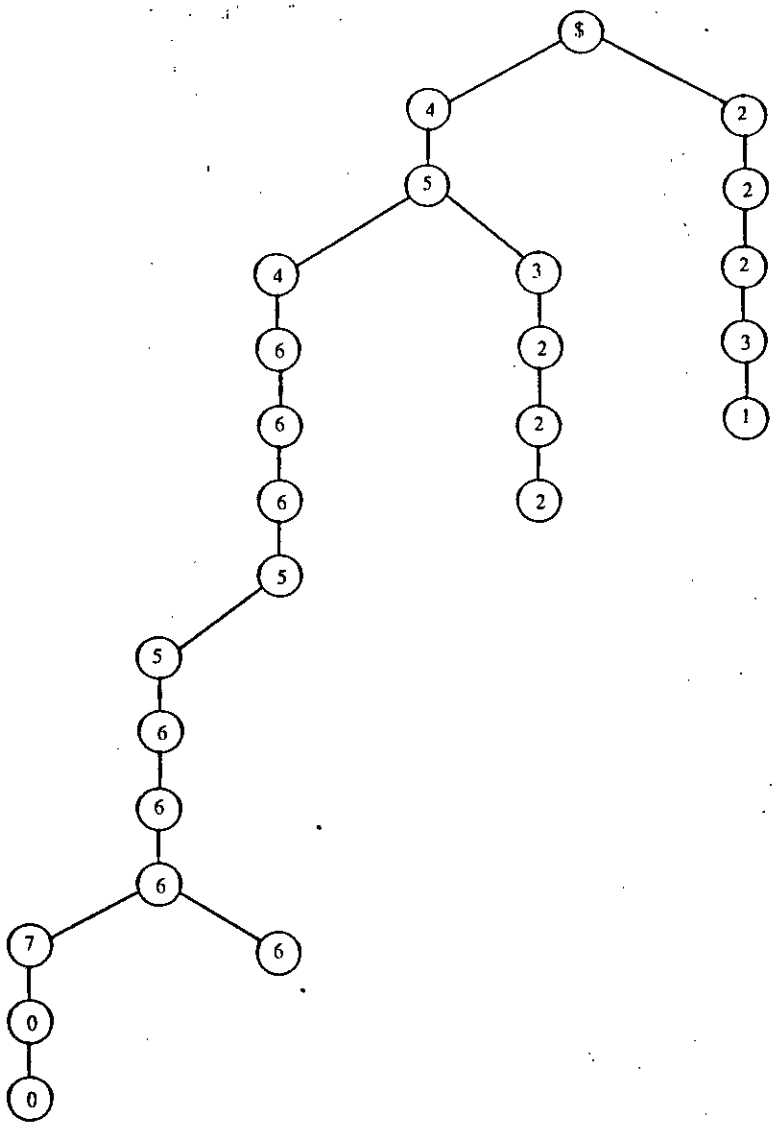
ANNEXE / Caractère et son developpement en arbre.



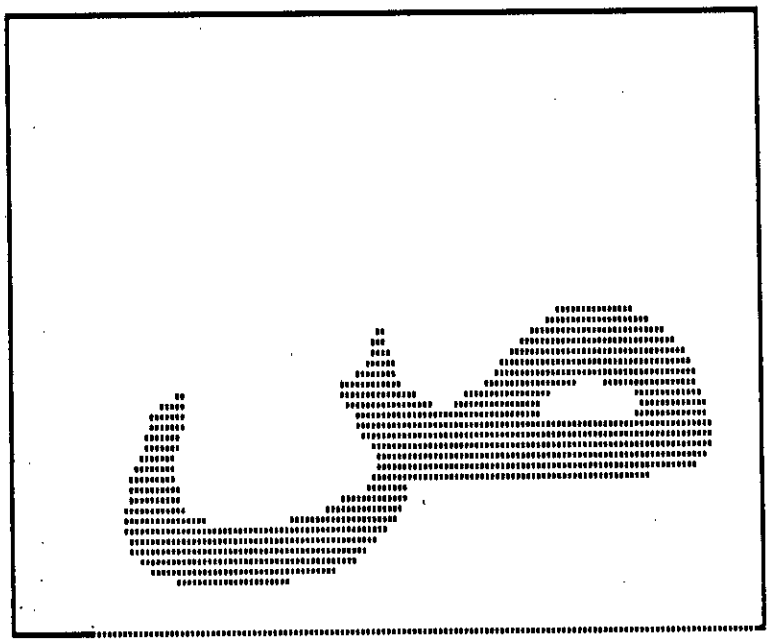
LA REPRESENTATION EN ARBRE DU CARRACTERE : ZIN



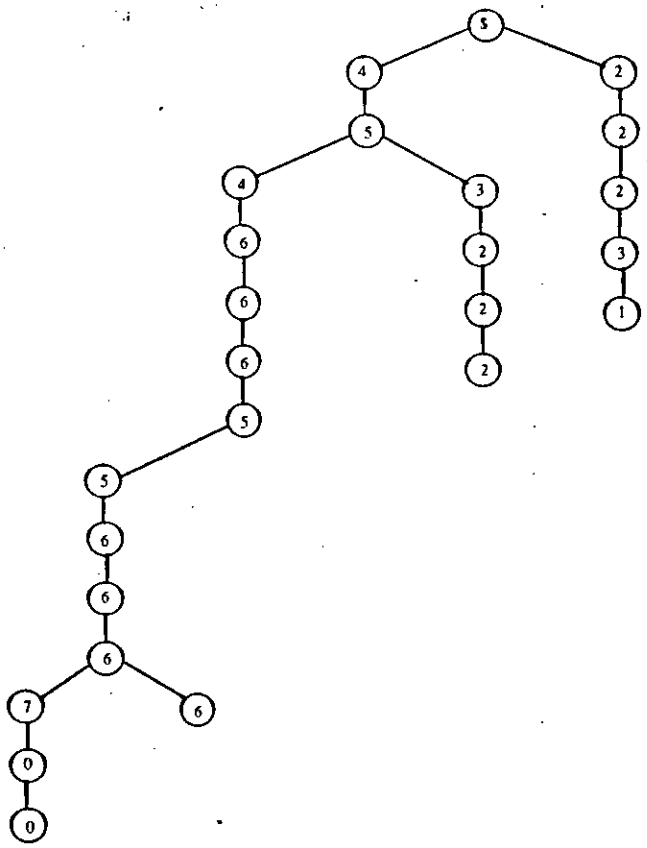
ANNEXE / Caractère et son developpement en arbre.



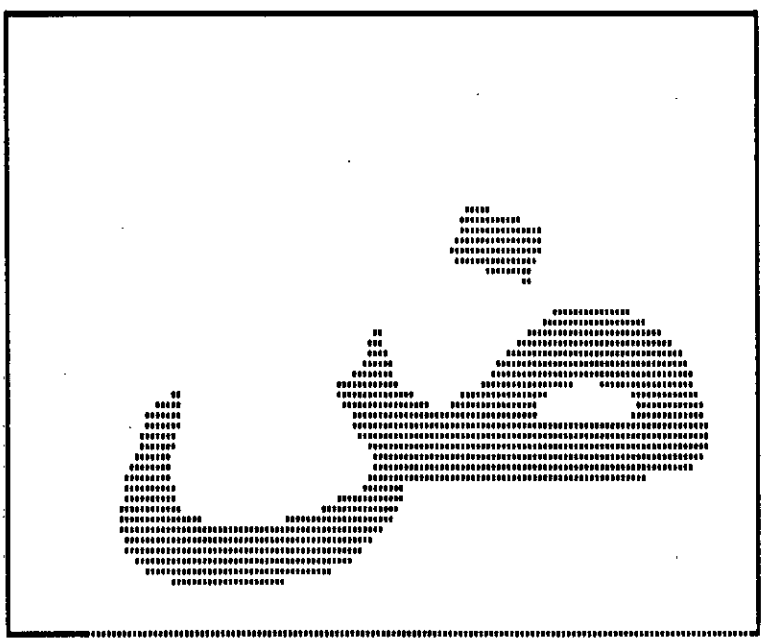
LA REPRESENTATION EN ARBRE DU CARRACTERE :SAD



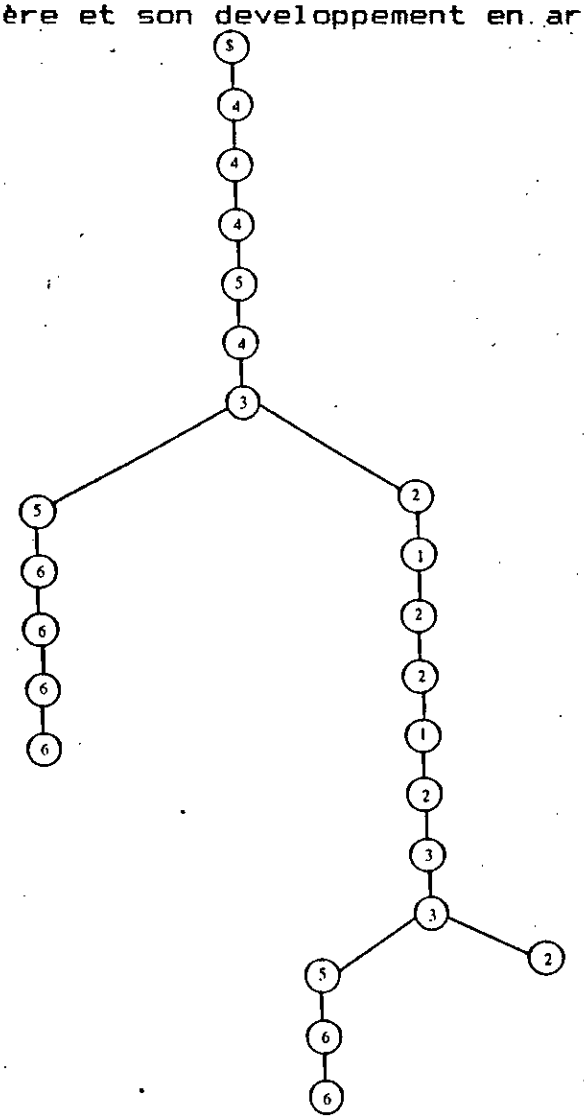
ANNEXE / Caractère et son developpement en arbre.



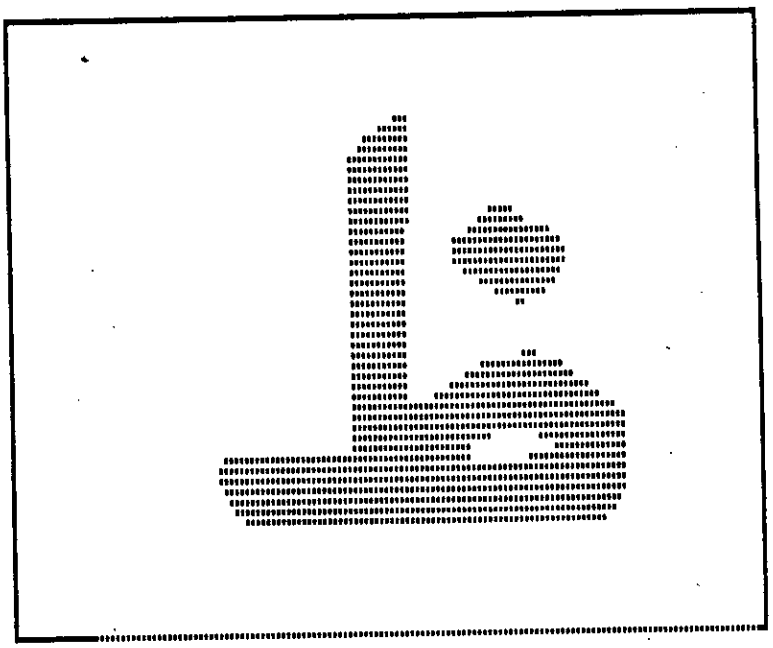
LA REPRESENTATION EN ARBRE DU CARRACTERE :DHAD



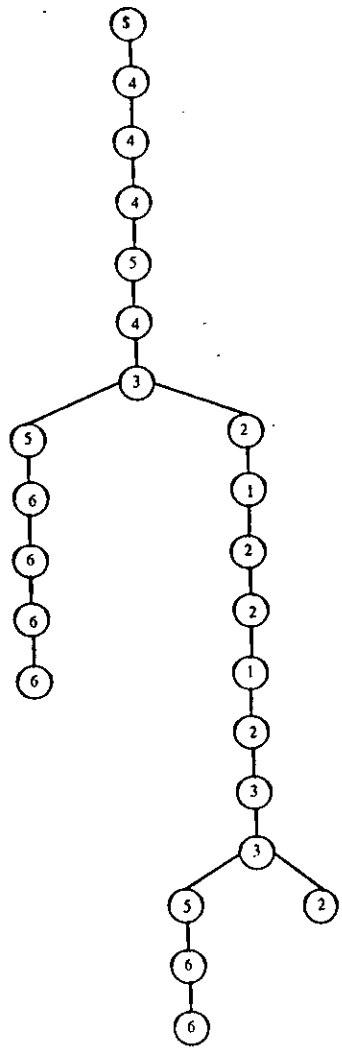
ANNEXE / Caractère et son developpement en arbre.



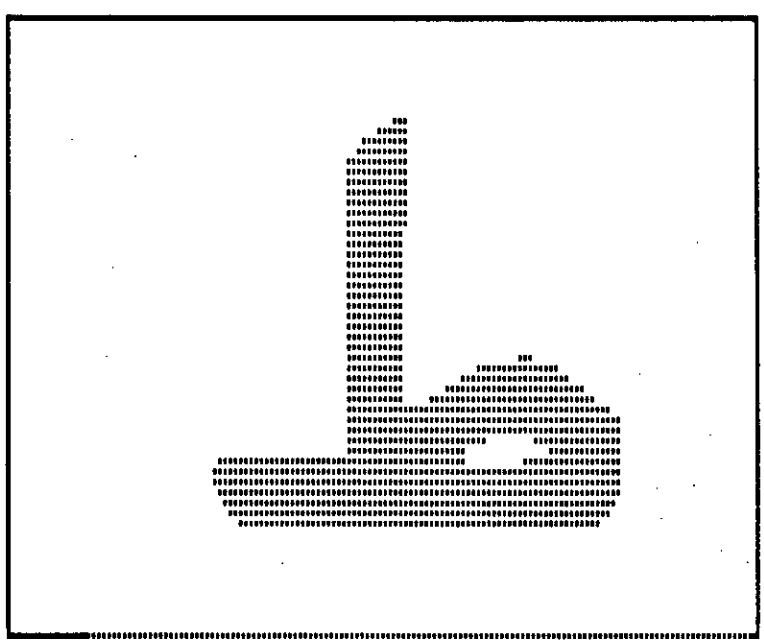
LA REPRESENTATION EN ARBRE DU CARRACTERE : THAD

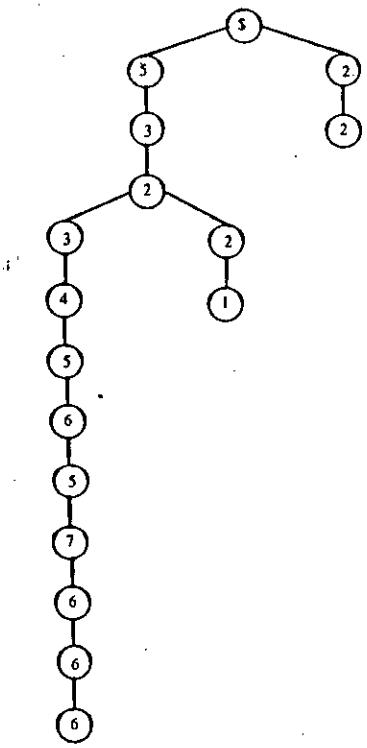


ANNEXE / Caractère et son developpement en arbre.

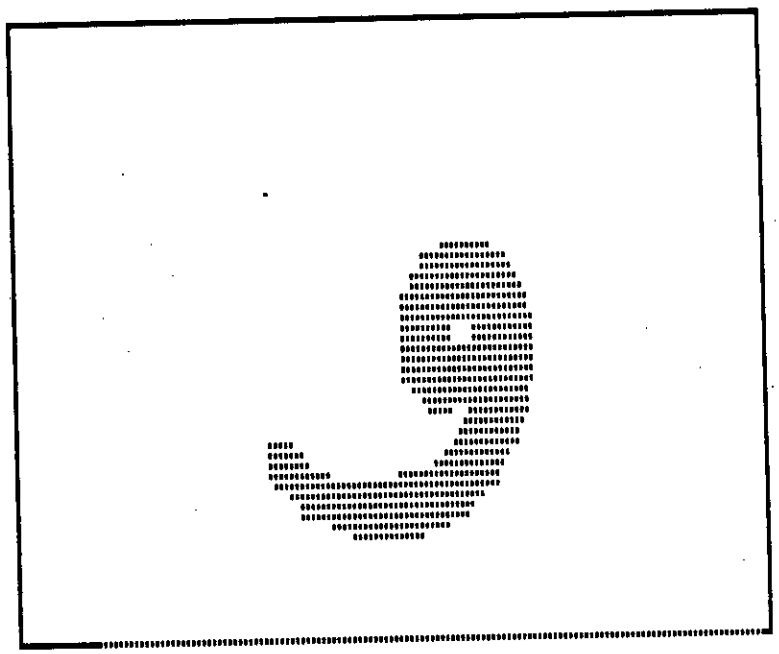


LA REPRESENTATION EN ARBRE DU CARRACTERE :taa

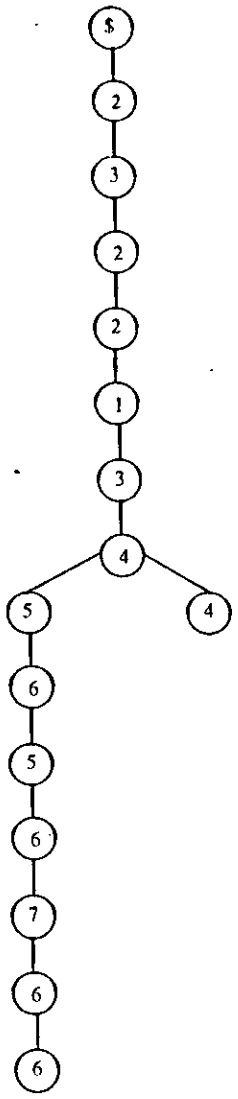




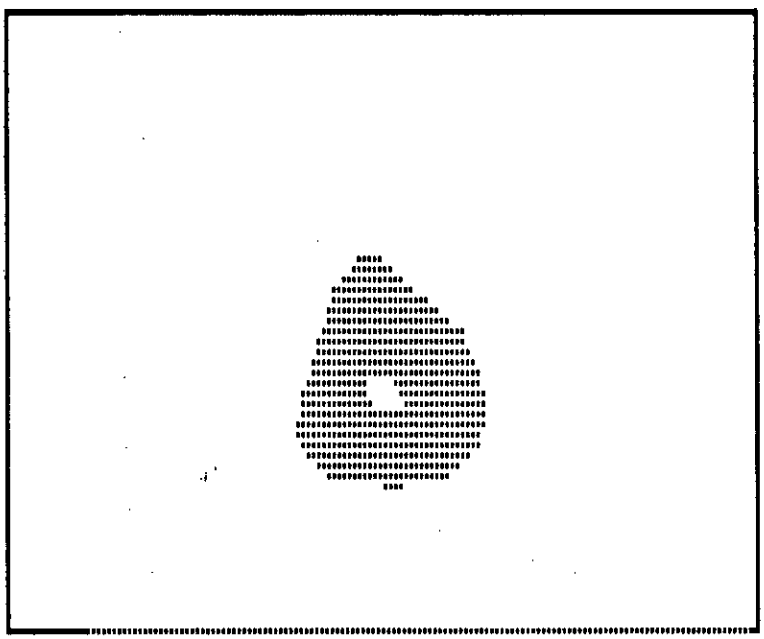
LA REPRESENTATION EN ARBRE DU CARRACTERE :OUA



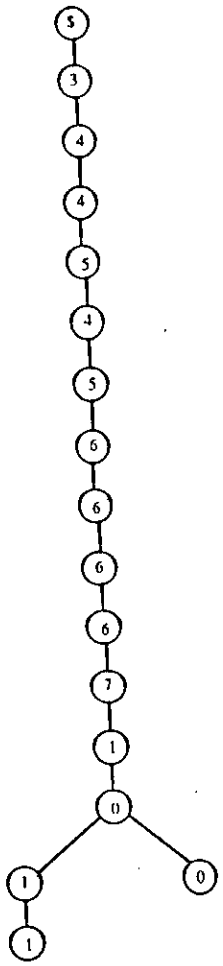
ANNEXE / Caractère et son developpement en arbre.



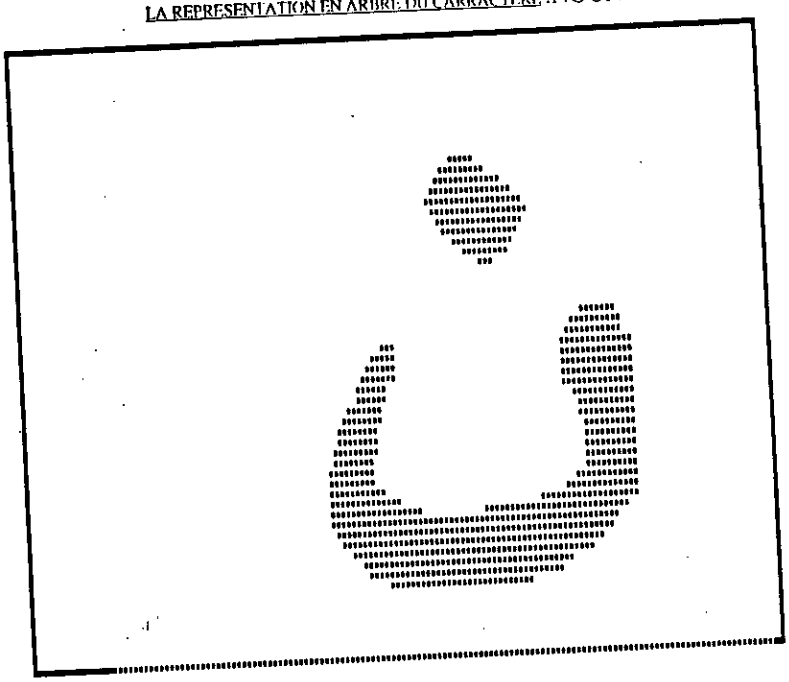
LA REPRESENTATION EN ARBRE DU CARRACTERE :HHA

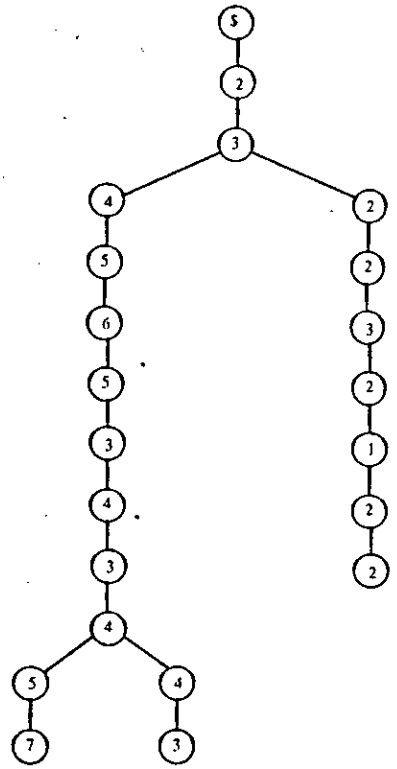


ANNEXE / Caractère et son developpement en arbre.

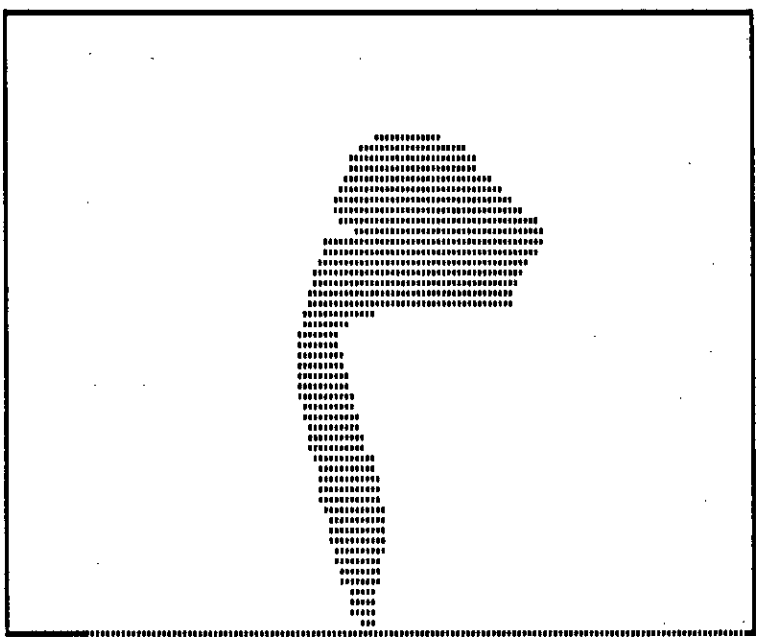


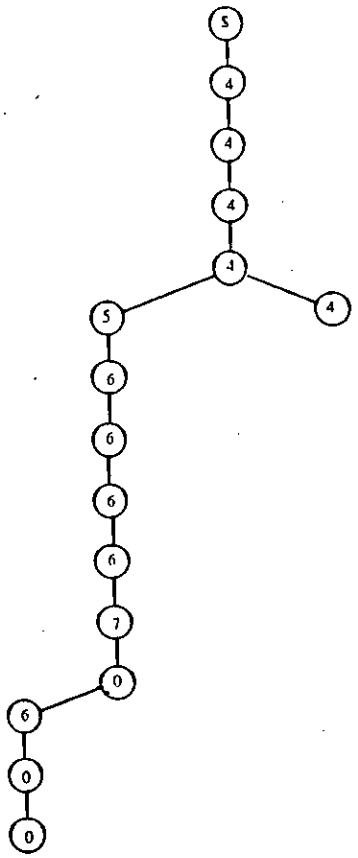
LA REPRESENTATION EN ARBRE DU CARRACTERE :NOUN



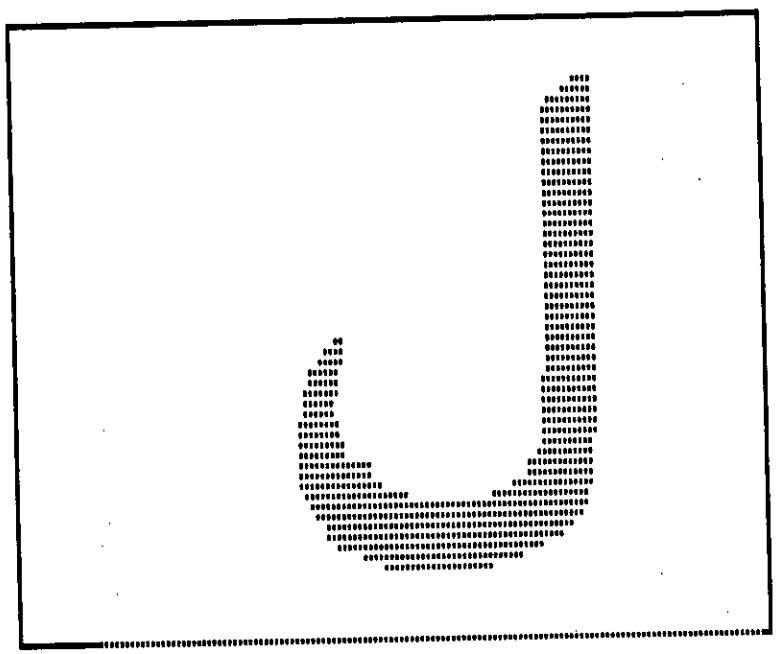


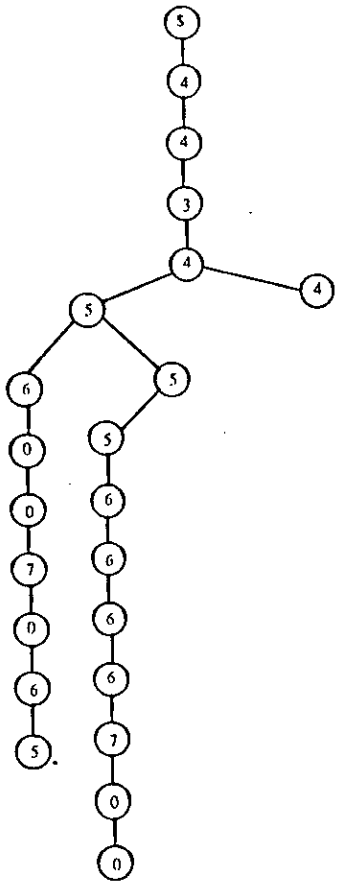
LA REPRESENTATION EN ARBRE DU CARRACTERE :MIM



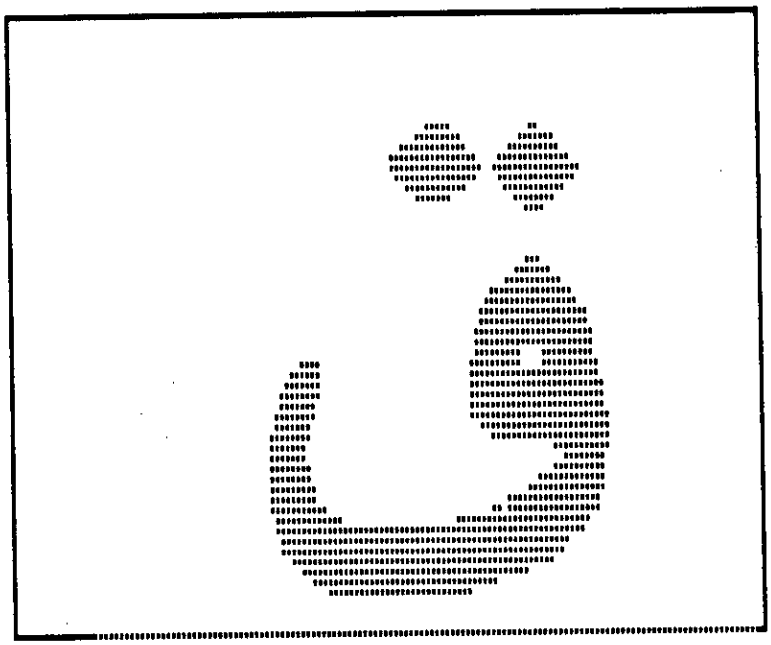


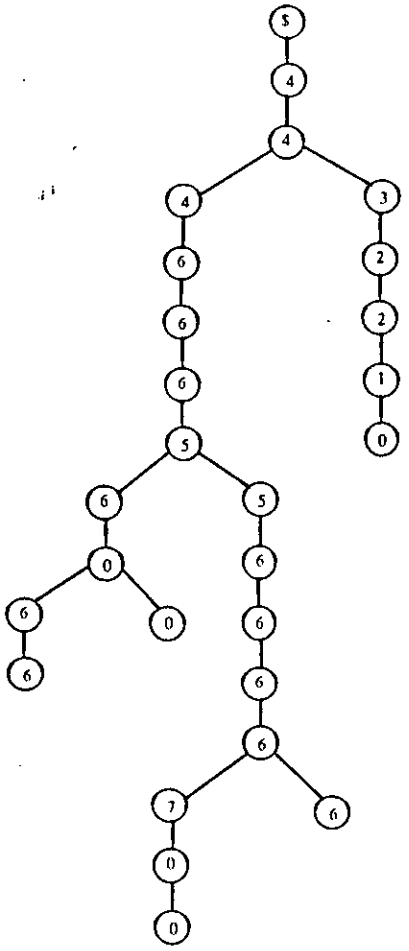
LA REPRESENTATION EN ARBRE DU CARRACTERE :LAM



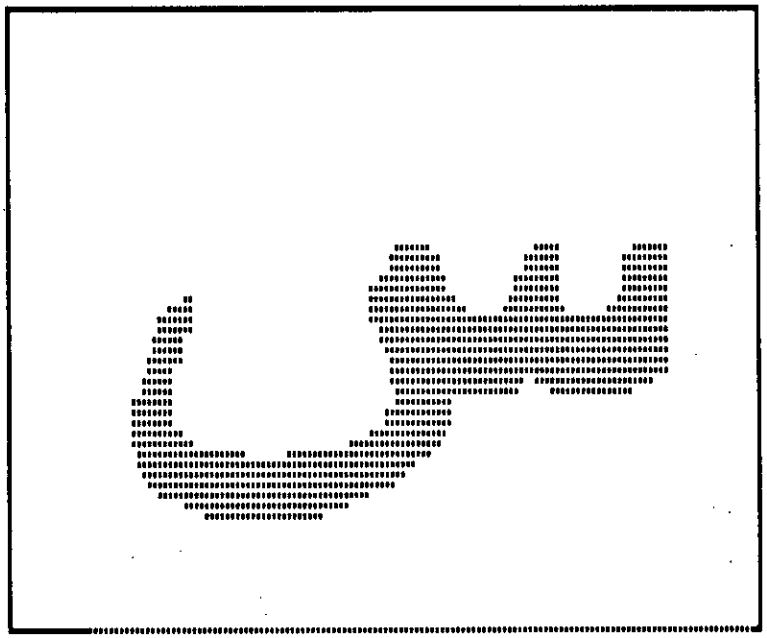


LA REPRESENTATION EN ARBRE DU CARRACTERE :KAF

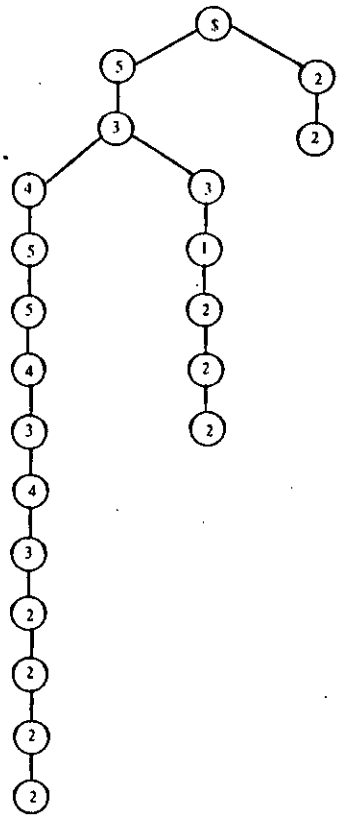




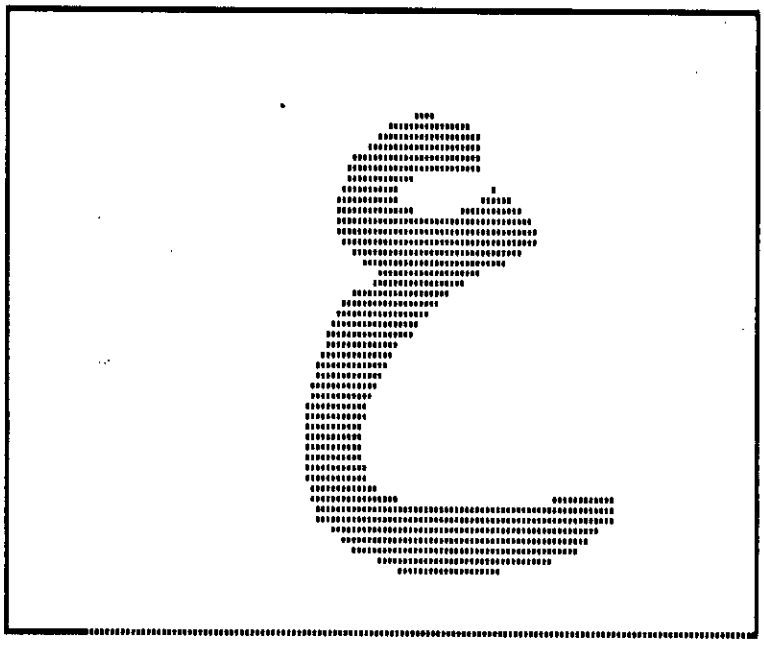
LA REPRESENTATION EN ARBRE DU CARRACTERE :SIN



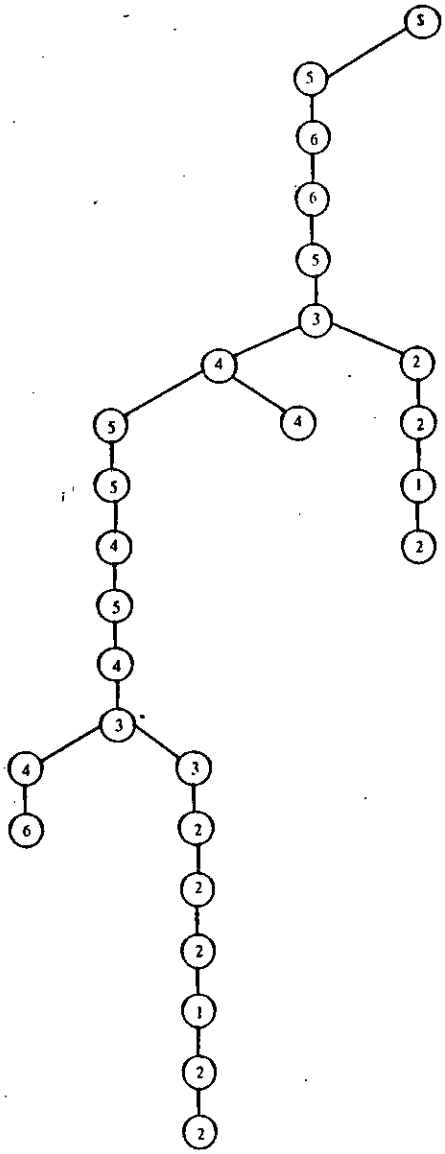
ANNEXE / Caractère et son developpement en arbre.



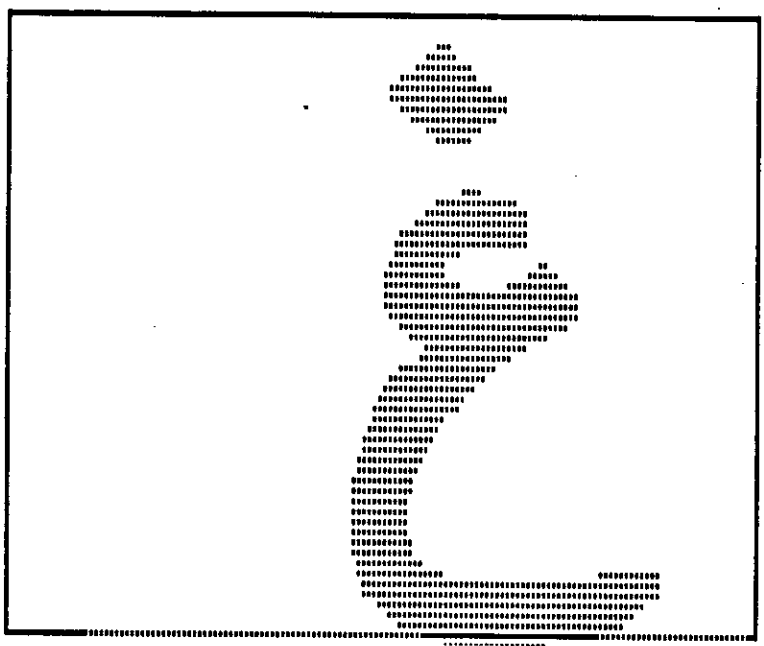
LA REPRESENTATION EN ARBRE DU CARRACTERE :AIN



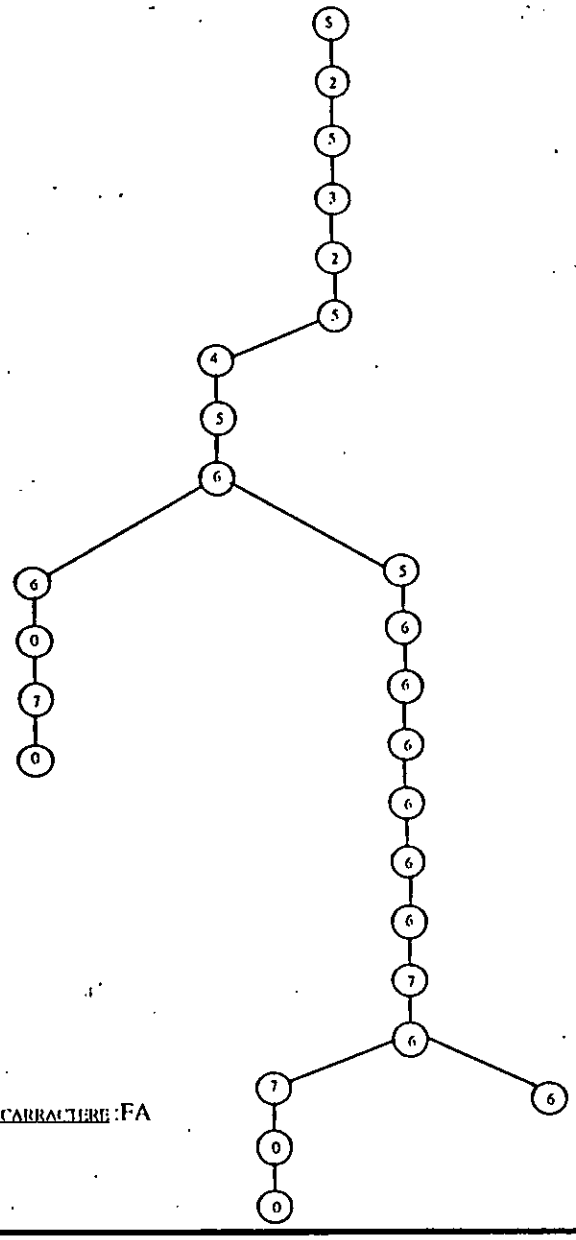
ANNEXE / Caractère et son developpement en arbre.



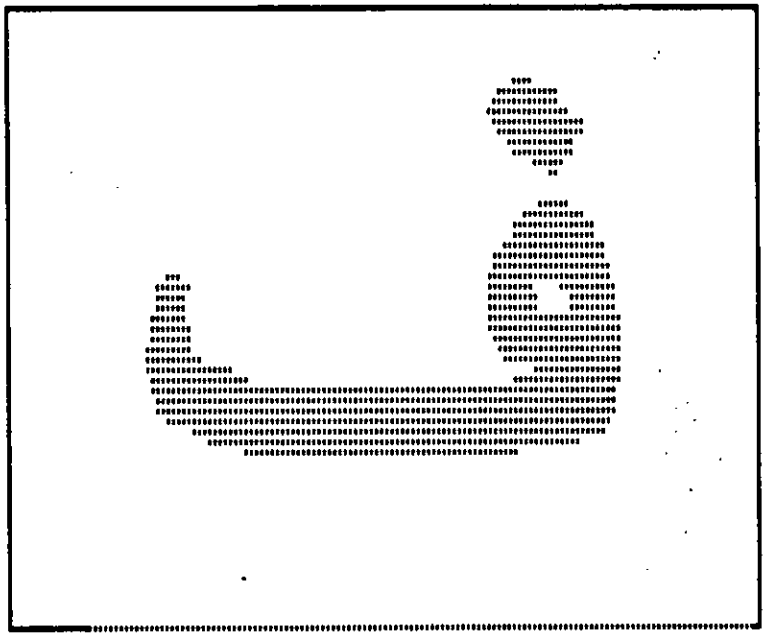
LA REPRESENTATION EN ARBRE DU CARRACTERE : GHA



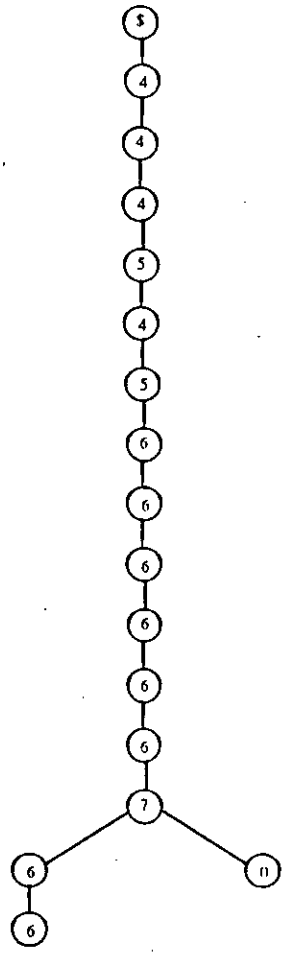
ANNEXE / Caractère et son developpement en arbre.



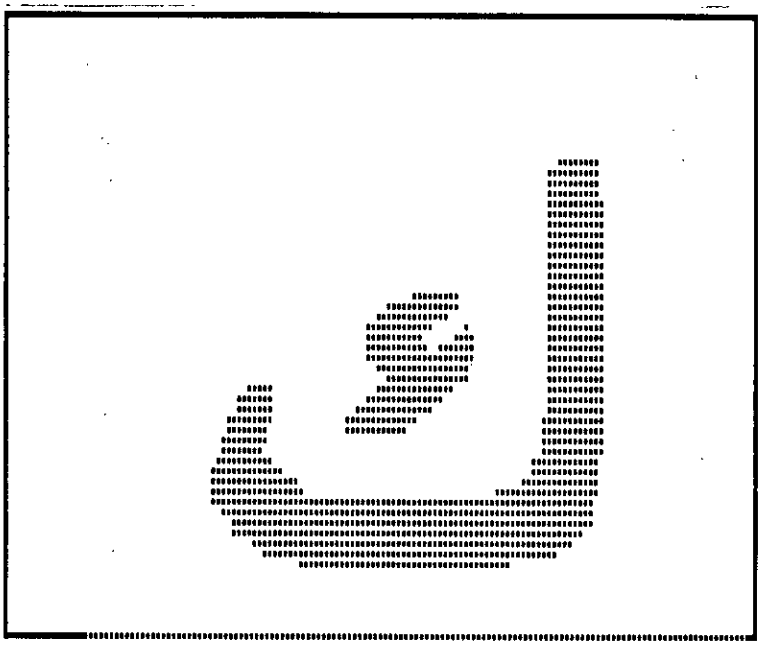
LA REPRESENTATION EN ARBRE DU CARRACTERE :FA

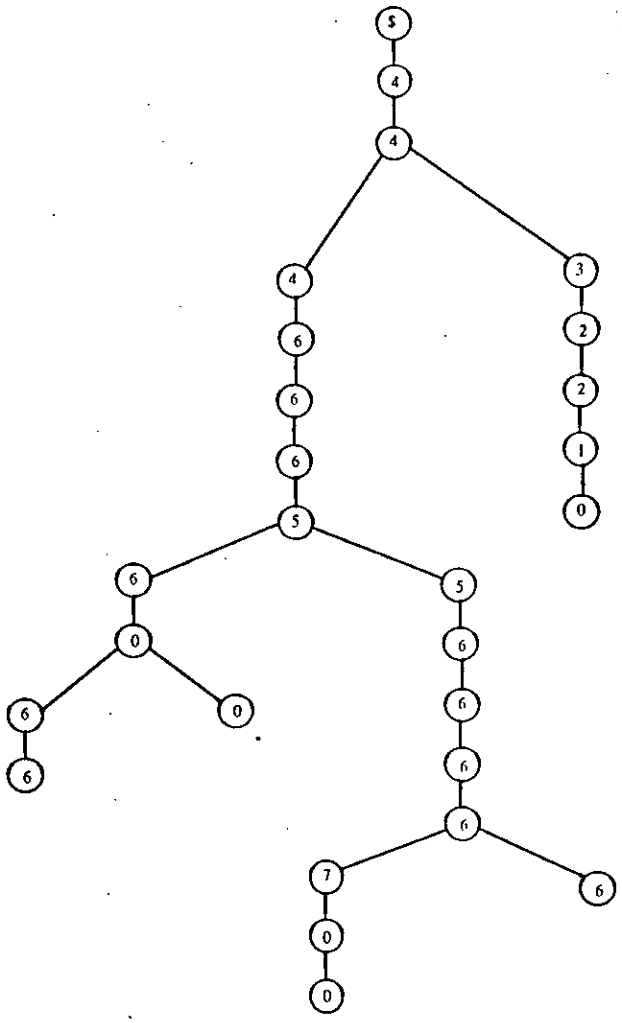


ANNEXE / Caractère et son developpement en arbre.

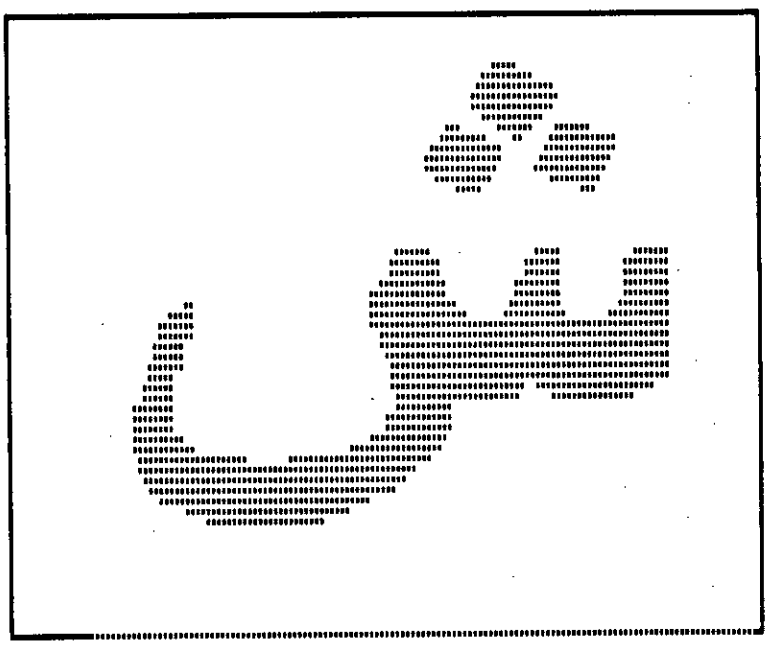


LA REPRESENTATION EN ARBRE DU CARRACTERE : K

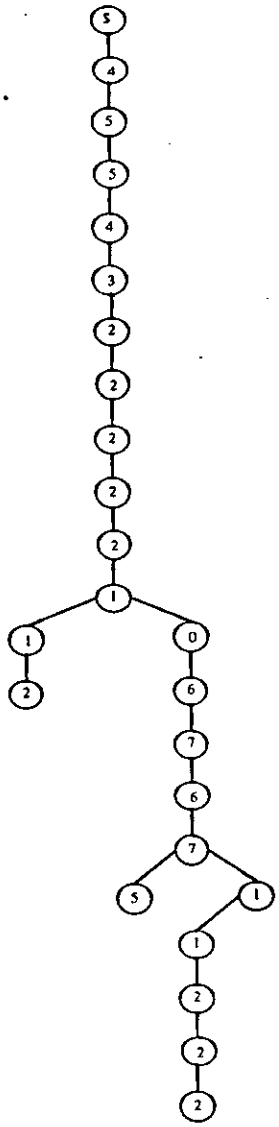




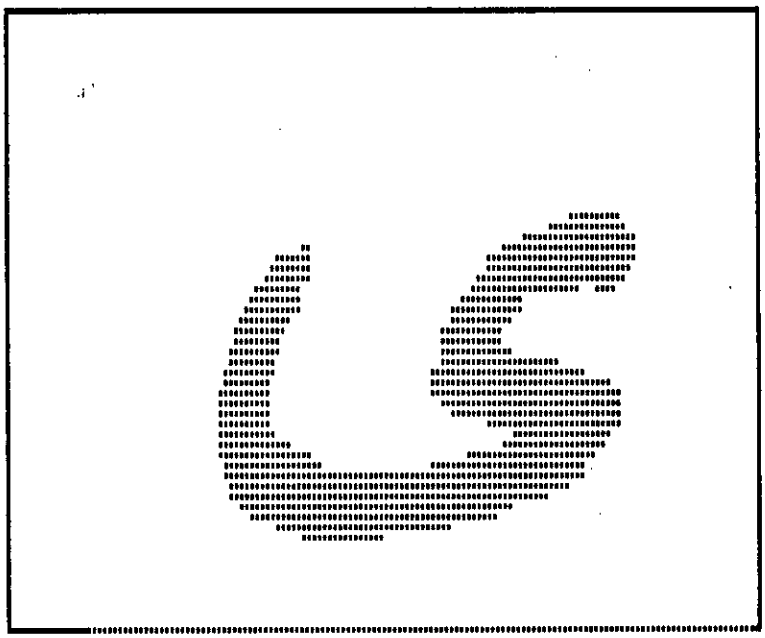
LA REPRÉSENTATION EN ARBRE DU CARRACTERE :CHIN



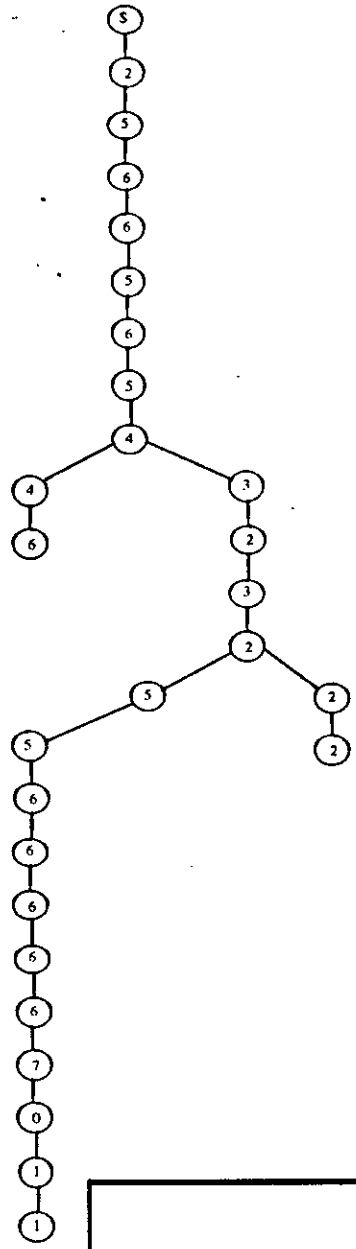
ANNEXE / Caractère et son developpement en arbre.



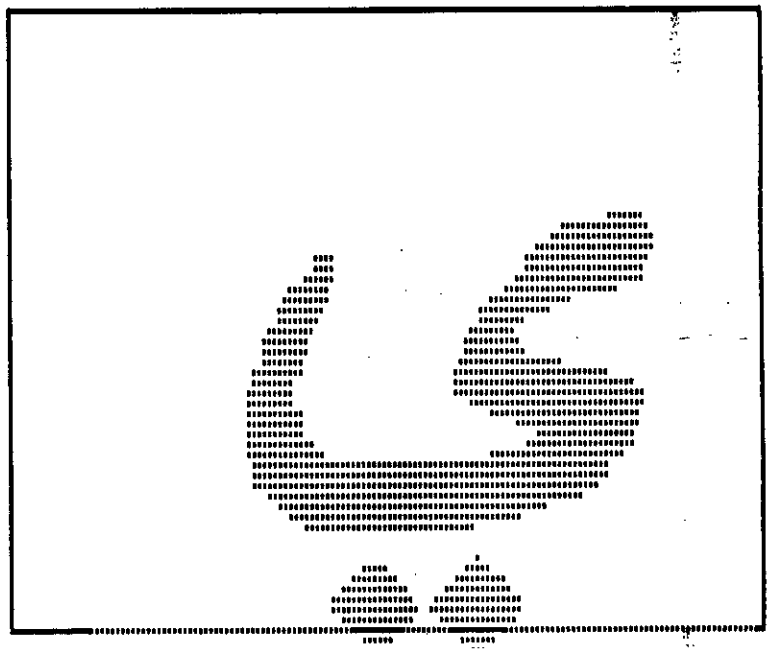
LA REPRESENTATION EN ARBRE DU CARRACTERE : YAA



ANNEXE / Caractère et son developpement en arbre.



LA REPRESENTATION EN ARBRE DU CARRACTERE : YA



BIBLIOGRAPHIE

BIBLIOGRAPHIE

[1] - A. AHO, J. HOPCROFT et J. ULLMAN
" Structures de données et Algorithmes "
I.I.A InterEditions 1987.

[2] - R.C. GONZALEZ et P. WINTZ
" Digital image processing "
Editions Addisson-Wesley; Reading, Mass. ; 1977.

[3] - S. Y. LU
" A tree-to-tree distance and its application to cluster analysis "
I.E.E.E Transactions on PAMI Vol. PAMI.1 N°2
Avril 1979

[4] - S. Y. LU et K. S. FU
" Error-correcting tree automata for syntactic pattern recognition "
I.E.E.E Transactions on comput. Vol.C-27, n°11,
NOV 1978

[5] - MERROUCHE et ABDOU
" Etude de la méthode des moments pour la reconnaissance des formes appliquée aux caractères arabes "
Projet de fin d'études E.N.P
Juin 1991

[6] - L. MICLET
" Methodes structurelles pour la reconnaissance des formes "
Ed. EYROLLES 1984

[7] - L. SAADAOUI
" Techniques de traitement numériques d'images en vue de la reconnaissance des formes "
Thèse de Magister en électronique E.N.P
Avril 1991