VII LE TRAITEMENT DES DESCRIPTIONS BIOLOGIQUES : KATE ET CASEWORK

Dans notre méthode d'acquisition des connaissances expliquée au chapitre 2, le traitement des descriptions biologiques est la troisième phase importante du processus qui permet de répondre aux deux objectifs de classification et de détermination en biologie. Initialement en apprentissage, la logique inductive est le raisonnement qui a été utilisé pour répondre à ces deux objectifs afin de trouver une alternative aux systèmes experts traditionnels à base de règles. Aujourd'hui, nous souhaitons aller plus loin dans le choix de différentes méthodes qui seront mieux adaptées chacune à un objectif particulier. Le but de ce chapitre est de montrer en quoi le raisonnement inductif est plus approprié à la problématique de la classification telle que nous l'avons définie au § 2.3.2, et pourquoi une forme de raisonnement analogique, le raisonnement par cas, nous permet de mieux résoudre le problème de la détermination en biologie. Les mérites et limites respectives de ces méthodes sont analysées à la lumière de notre application sur les *Hyalonema*.

En effet, une fois formalisée la base d'exemples (chapitre 5), nous pouvons appliquer différentes méthodes de traitement issues à la fois des recherches en analyse des données et en apprentissage. Il s'agit ici de formaliser le processus de génération automatique de critères de décision pour :

1) fabriquer un arbre de classification (par induction avec KATE),
2) utiliser cet arbre pour la détermination (par déduction avec KATE),
3) comparer directement les exemples pour la détermination (avec un raisonnement de type analogique : CaseWork[1]).

7.1 Formalisation de l'approche inductive

Le programme KATE [Manago, 1991] est une extension des techniques d'induction utilisées dans ID3 [Quinlan, 1983]. Il autorise le traitement d'un ensemble d'exemples $\Omega = \{w_1,...,w_n\}$ composé de descriptions complexes d(w) sous forme d'objets de synthèse [Diday, 1987], et comportant des relations entre objets de la description. A chaque d(w) est associée une classe d'identification c_i

[1] Ce programme informatique constitue notre véritable contribution lors de cette thèse où il s'agissait de répondre à la question sur la robustesse de la consultation.

(voir la classe soulignée de l'exemple du § 5.6). Comme nous avons émis l'hypothèse que les descriptions sont celles de spécimens (§ 5.1.1), nous assimilerons d(w) à w, c'est-à-dire qu'un cas représente la description d'un individu (qu'il soit prototypique ou unitaire).

7.1.1 Rappel des Notations

$C = \{c_1,\ldots,c_k\}$ est l'ensemble des k Classes ou chaque c_i représente la décision de l'expert pour le cas w.

$\Omega = \{w_1,\ldots,w_n\}$ est l'ensemble des cas observés à traiter par induction, $k < n$.
$Y = Y_1 \times Y_2 \times \ldots \times Y_p$ est l'ensemble des variables observées du domaine,
$Y_i = \{y^i_1,..,y^i_j,..,y^i_q\}$ est l'ensemble des variables de Ω_i représentant un objet ou une partie p_i de la description d'un cas,
$Q_i = \{q^i_1,..,q^i_j,..,q^i_q\}$ est l'ensemble des **qualités** ou caractères observés de Ω_i.
Les qualités d'un objet appelées aussi attributs descriptifs en intelligence artificielle sont à distinguer du **statut** S_i de l'objet, c'est-à-dire de la propriété de présense - absence qui conditionne la description de cet objet.

$N = N_1 \times N_2 \times \ldots \times N_p$ est l'ensemble des noms d'objets ou parties observées d'individus.

Par exemple, $N_i = \{n_i\}$ est l'ensemble singleton comportant le nom de la partie p_i de Ω_i. Si n_i se spécialise en n'_i, on a $N_i = \{n_i, n'_i\}$ avec $n'_i < n_i$.

$U_i = (u^i_1,..,u^i_k,..,u^i_n) \quad [\Omega_i]^{ni}$ est l'ensemble des instances de l'objet i,

$M_i = N_i \circ U_i$ est l'ensemble des objets i multi-instanciés,

on a $Y_i = Q_i \circ M_i$ et $S_i = \{exist\} \circ M_i$.

Exemples : $Q_1 = \{taille\}$, $N_1 = \{amphidisques\}$, $U_1 = \{1, 2\}$

$M_1 =$ [amphidisques (1)], $M_2 =$ [amphidisques (2)]
$S_1 = \{$ exist [amphidisques (1)] $\}$, $Y_1 = \{$ taille [amphidisques (1)] $\}$
$S_2 = \{$ exist [amphidisques (2)] $\}$, $Y_2 = \{$ taille [amphidisques (2)] $\}$

$Q_2 = \{forme\}$, $N_2 = \{corps\}$, $U_2 = \{1\}$
$M_2 = \{corps\}$, $S_2 = \{$ exist (corps)$\}$, $Y_2 = \{$ forme (corps)$\}$

V_k^{ij} est l'ensemble des valeurs observées de y_j^i lorsqu'il existe k instances de l'objet i pour le cas w. Si k = 1, on a $V_k^{ij} = V_j^i$ et $M_i = N_i$.

7.1.2 Principe de la classification par arbre de décision

Le but de la méthode de création d'un arbre de décision est d'obtenir une caractérisation des classes décrites dans les exemples en construisant une fonction caractéristique de reconnaissance suffisante des classes entre elles (ce qui correspond à une diagnose, voir figure 2.5).

L'idée centrale des algorithmes d'apprentissage par arbre de décision consiste à diviser récursivement les exemples de l'ensemble Ω d'apprentissage à l'aide des attributs jusqu'à obtenir des sous-ensembles d'exemples qui soient suffisamment purs, c'est-à-dire ne contenant (presque) que des exemples appartenant tous à la même classe.

Ces sous-ensembles sont alors regroupés au niveau des feuilles ou nœuds terminaux de l'arbre de décision.

Une division d'un nœud intermédiaire est déterminée par l'un des attributs qui décrivent les exemples. Cette division est fonction du nombre de valeurs possibles associées à l'attribut. Par exemple, dans le cas d'un attribut booléen, numérique ou testant l'existence d'un objet, la division est binaire. Elle est n_aire en considérant un ensemble fini de valeurs qualitatives nominales ou classifiées.

La division peut aussi être vue comme une question à poser à l'utilisateur pour permettre la séparation des exemples en autant de groupes qu'il y a de valeurs possibles attachées à l'attribut.

L'autre idée est que cette division soit la plus efficace possible de manière à ce que l'effort de recherche pour trouver la solution soit minimal : on désire poser le minimum de questions à l'utilisateur. Cette idée est néanmoins subordonnée à l'utilisation de l'arbre de décision pour faire de la détermination.

Soit T un arbre de décision n_aire construit à partir de Ω et *d* un nœud intermédiaire de T correspondant à un sous-ensemble E Ω, et défini par la division *s* (figure 7.1). Le nœud d correspond au choix d'un attribut A parmi *s*, s étant la liste des attributs ordonnés en fonction de leur pouvoir de discrimination. E est l'ensemble des exemples au nœud *d*, c'est-à-dire l'ensemble qui vérifie la liste des valeurs indexées sur le chemin conduisant de la racine d_0 à d (voir § 7.1.4.2.5).

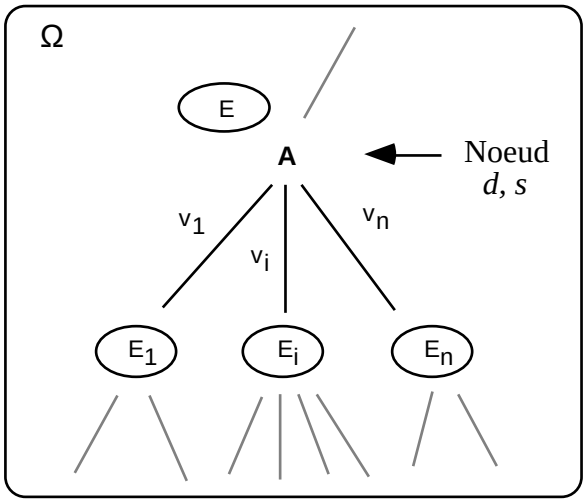
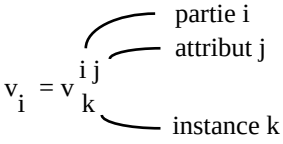


Fig. 7.1 : Schéma d'un nœud de l'arbre T Soit $A = y_j^i$, un attribut d'une partie de la description et n descriptions ou valeurs de cet attribut $\{v_1,...,v_i,...,v_n\}$,



Une fonction de partitionnement R induisant une partition sur E est définie de la manière suivante :

$R : \quad w \quad E , A(w) = v_i \quad \Leftrightarrow \quad w \quad E_i$

$R(E) = \{E_1,...,E_n\}$ est alors une partition de E avec les propriétés suivantes :

$E = \quad E_i$

$i = 1,...,n \quad$ on a $E_i \cap E_j =$

7.1.3 Algorithme

On peut décrire notre algorithme par une procédure générale de construction d'arbre de décision [Vignes, 1992].

Cela consiste à sélectionner un attribut selon un certain critère pour former le premier nœud de l'arbre, puis à créer les différentes branches qui partent de ce nœud, une branche étant étiquetée par l'une des valeurs possibles de l'attribut sélectionné. Ensuite, on répartit la liste des exemples restants en fonction de leur compatibilité avec chaque branche au nœud courant. Enfin, on réitère le processus jusqu'à n'obtenir que des exemples de la même classe qui forment alors une feuille de l'arbre de décision.

L'algorithme demande donc trois opérations principales :

1) Tester si un nœud est terminal : si chaque élément de E appartient à la même classe, on construit une feuille de l'arbre libellée par le nom de la classe,

2) Sélectionner la meilleure division pour placer son représentant (A) dans un nœud intermédiaire,

3) Partitionner les exemples restants à un nœud intermédiaire en sous-ensembles disjoints.

Une quatrième opération spécifique à KATE vient s'ajouter entre 1) et 2) :

1') Construire l'espace des attributs possibles au nœud d.

Soit l'ensemble des exemples restants E et l'ensemble Y_d des attributs restants attachés au nœud *d* :

Algorithme :

Début :
E = Ω,

ConstruireArbre (E, Y_d)
si Critèred'Arrêt (E) alors
ConstruireFeuille (E)
sinon
Y_d = ConstruireEspace (E)
s = OrdonnerCritères (E, Y_d)
A = Meilleure_division (E, s)
d_i = CréerNœud (A)
partition = R (E)

Pour tout E_i partition
CréerBranche (v_i)
ConstruireArbre (E_i , Y_d)
Fin Pour tout
Fin si

Fin.

ConstruireEspace (E)

Y_d =
Pour tout m_i M_i
si w E, m_i w alors $Y_d = Y_i$ Y_d
sinon si w E , m_i w alors Y_d = Y_d
sinon $Y_d = S_i$ Y_d
Fin si

Pour tout A Y_d
si V_k^{ij} R ou si V_k^{ij} N alors T = CalculerSeuil (A, E)
Fin Pour tout
Fin Pour tout

retourner Y_d

7.1.4 Description des fonctions principales de l'algorithme

7.1.4.1 OrdonnerCritères (E, Y_d)

Soit $Y_d = \{A_1,...,A_p\}$, l'ensemble des attributs applicables au nœud courant. L'ordonnancement des critères est fondée sur l'application des principes de la théorie de l'information et de l'entropie [Shannon, 1949].

L'entropie de Shannon est une mesure de probabilité sur la difficulté de prévoir laquelle des valeurs possibles d'un attribut est applicable à un nouvel individu choisi au hasard parmi l'ensemble des exemples décrits [Estabrook, 1967]. Elle possède une signification *statistique* indépendante du contenu du message véhiculé : l'entropie est basée sur la mesure du degré d'incertitude de la réalisation d'évènements aléatoires par rapport à des expériences possédant un nombre k d'issues ayant chacune une certaine probabilité d'apparition [Yaglom A.M. & Yaglom I.M., 1957]. Par exemple, si notre expérience consiste à déterminer la couleur de la première corneille rencontrée, nous pourrons considérer la couleur noire avec une certitude presque entière (bien qu'il existe quelques rares corneilles blanches dans la nature selon les zoologues). Notre incertitude sera plus grande si l'expérience est de déterminer si le premier homme rencontré sera gaucher : le résultat de l'expérience peut être prédit presque sans hésitation, mais les craintes quant à l'exactitude de cette prédiction seront plus fondées que dans le premier cas. Il est sensiblement plus difficile de prédire d'avance le sexe de la première personne que nous rencontrerons. Mais cette expérience possède encore un degré d'indétermination relativement petit en comparaison de la tentative de désigner d'avance le vainqueur d'un tournoi dont

les vingts participants nous sont entièrement inconnus ou de donner le numéro du billet de loterie gagnant du gros lot au prochain tirage...
Le degré d'incertitude de chaque expérience est déterminé par le nombre k d'issues possibles ainsi que de leurs probabilités d'apparition : elle est notée f(k). Cette fonction est déterminée par les trois propriétés suivantes:

1) Si k = 1, il n'y a aucune incertitude dans ce cas et la mesure doit s'annuler.
2) Lorsque k augmente, la mesure f(k) est une fonction croissante des k issues.
3) Si l'on considère deux expériences indépendantes α et β (avec k issues pour α et l issues pour β), le degré d'incertitude de l'expérience composée αβ est égal à la somme des incertitudes qui caractérisent les expériences α et β: f(kl) = f(k) + f(l).

On peut montrer [Guiasu & Theodorescu, 1971] que la seule fonction de la variable k qui vérifie ces trois conditions est la fonction logarithmique : f(k) = log k. Dans les applications, les logarithmes de base 2 sont le plus souvent utilisés, ce qui signifie que l'on prend comme unité de mesure du degré d'incertitude, l'incertitude d'une expérience possédant deux issues également probables. Ce choix n'est pas essentiel : un facteur constant existe entre les différentes bases.

Cette mesure est aussi une information sur la capacité d'un attribut à séparer efficacement les exemples. Supposons en effet un attribut ayant 2 valeurs possibles, p_1 et p_2 sont les proportions d'exemples prenant respectivement les valeurs 1 et 2. Si tous les exemples d'apprentissage prennent la première valeur ($p_1 = 1$ et $p_2 = 0$), alors le fait d'observer l'état 1 n'apporte aucune information supplémentaire pour séparer les exemples. Inversement, si la répartition des exemples selon les valeurs de l'attribut est homogène ($p_1 = 1/2$ et $p_2 = 1/2$ pour l'équi-répartition), l'efficacité de discrimination est maximale.

On peut donc associer à chaque attribut A Y_d une entropie Ent(E) qui est la somme des distributions de probabilités des n valeurs de son domaine de définition. Ces probabilités sont calculées en fonction des états que prennent les exemples pour l'attribut A.

Pour chaque valeur discrète de A, on définit la fréquence d'occurrence P_i de E_i qui est la probabilité associée à chaque valeur d'attribut pour qu'un exemple w appartenant à E appartienne à E_i :

$P_i = \frac{Card(E_i)}{Card(E)}$ est donc la probabilité de choisir un exemple ayant l'état i de A.

L'entropie est alors calculée selon la formule : $Ent(E) = -\sum_{i=1}^{n} P_i \times \log_2 P_i$

ou n est le nombre de valeurs possibles de A.

Par exemple, pour l'attribut A = C (la classe) au nœud courant, on peut calculer la proportion d'exemples de E qui sont de la classe c_i et l'entropie de C sera alors la quantité d'information nécessaire pour déterminer les classes dans le sous-ensemble E.

7.1.4.2 Meilleure_division (E, s)

Par principe, l'entropie est mesurée sur un attribut que l'on désire apprendre (C par exemple), en fabriquant une caractérisation de cet attribut à l'aide d'autres attributs. C'est alors que se justifie la mesure du gain d'information :

Le Gain d'information est la mesure de l'accroissement d'ordre sur C qu'introduit le choix d'un autre attribut A : plus ce gain est élevé, plus la répartition des exemples pour chaque classe est homogène (le meilleur gain est celui qui représente l'équi-répartition des exemples). C'est le gain calculé le plus élevé qui permet de choisir le meilleur attribut permettant d'apprendre C, qui permet donc la meilleure division au nœud courant.

La formule du gain d'information est la suivante :

$$Gain(A,E) = Ent(E) - Ent(A,E)$$

avec $Ent(A,E) = \sum_{i=1}^{n} P_i \times Ent(E_i / C)$ étant l'entropie moyenne pondérée des informations conditionnelles des n valeurs possibles de A.

En effet, $Ent(E_i / C) = -\sum_{i=1}^{n} p_i \times \log_2 p_i$ est l'entropie conditionnelle calculée pour chaque valeur de A avec $p_i = \frac{Card(E_i \cap C_j)}{Card(E)}$ est la probabilité conditionnelle associée à A (probabilité de choisir un objet ayant l'état i de A et l'état j de C).

Remarque : On a pu constater dans différentes applications médicales [Kononenko *et al*, 1984] que la mesure du gain d'information favorise les attributs ayant un domaine de définition avec beaucoup de valeurs. Quinlan (1986) a introduit la notion de **gain d'information relatif** pour compenser ce biais en divisant le gain d'information précédent par l'information contenue dans le choix de l'attribut A : IV(A)

$$IV(A) = -\sum_{i=1}^{n} P_i \times \log_2 P_i \qquad Gain\ (A,E) = \frac{Ent(E) - Ent(A,E)}{IV(A)}$$

Dans notre algorithme, A = meilleure_division (E, s) correspond au choix de l'attribut de s possédant le gain d'information relatif le plus élevé pour séparer au mieux les exemples en fonction du but à atteindre qui est de faire de la discrimination sur l'attribut C.

7.1.4.3 Critèred'Arrêt (E)

Il existe plusieurs moyens d'arrêter la construction d'un arbre de décision :

1) w E , Classe(w) = c_i.

C'est la condition d'arrêt la plus naturelle, c'est-à-dire lorsque tous les cas d'un nœud ont la même modalité c_i pour la variable décision.

2) Card (E) > seuil donné.

Un inconvénient du premier critère d'arrêt est qu'il conduit à une séparation totale des classes, ce qui fait que certaines branches terminales ne possèdent que très peu d'exemples. Donc, séparer les exemples lorsqu'il n'en reste que 2 ou 3 n'est pas significatif d'un point de vue statistique : cela relève le plus souvent du hasard et ne contribue pas à une véritable connaissance du domaine [Crémilleux, 1991].

C'est pourquoi certains algorithmes imposent un nombre minimal d'exemples pour continuer à construire le sous-arbre (segmenter le nœud courant) comme le fait le système CART [Breiman *et al.*, 1984] en attribuant *a priori* la valeur 5 à ce seuil.

3) Card (E) / Card (Ω) > seuil donné.

Au lieu d'appliquer le critère absolu du 2), on peut fixer un seuil relatif dépendant du nombre total de cas [Cestnik, 1987].

4) Card ({ w E / Classe(w) = c_i }) > seuil donné.

Au lieu de comptabiliser les cas indépendamment de la classe auxquels ils ont été attribués, on peut décider d'arrêter la construction de l'arbre lorsque le nombre de cas d'une même classe dépasse un certain seuil.

5) La profondeur de l'arbre est limitée à un seuil donné.

Soit D = {d_i}, l'ensemble des nœuds de l'arbre T, soit d_0 un nœud particulier appelé la racine de l'arbre.

Tout nœud d_i autre que d_0 est relié par un arc à un autre nœud $d_{i'}$ appelé le *fils* de d_i. Si $d_{i'}$ est *fils* de d_i alors d_i est appelé *père* de $d_{i'}$. Cet arc est une branche avec un sommet $d_{i'}$ et une extrémité d_i. Elle contient la valeur v_i à observer pour déterminer l'individu (cf. figure 7.1).

$D_t = \{d_t\}$ est l'ensemble des nœuds terminaux ou feuilles de l'arbre T, une *feuille* est un nœud $d_k = d_t$ qui n'a pas de fils.

Soit la relation ">" ("père de").

Supposons que $d_1, d_2, \ldots, d_k$ soit une séquence de nœuds de T telle que $d_1 > d_2 > \ldots > d_{k-1} > d_k$. Cette séquence est appelée un *chemin* depuis d_1 jusqu'à d_k dans T. La longueur du chemin est k - 1.

La *profondeur de l'arbre T* est la longueur du chemin maximal menant de d_0 à d_t.

6) Tester si toutes les variables candidates à un nœud de l'arbre sont jugées "indépendantes" de la variable décision. Pour ce faire, on calcule le test du χ^2 pour chaque variable à partir du tableau de contingence défini par celle-ci et la variable décision. Puis on compare ce calcul avec le gain d'information. Ce dernier tend vers un χ^2 lorsque le nombre de cas au nœud courant est élevé.

Remarque : ce dernier point n'est souvent pas vérifié dans nos application pour la significativité du test, ce qui est un inconvénient pour arrêter la construction de l'arbre de manière fiable. Ce test est à considérer pour les nœuds terminaux dont le nombre d'exemples est élevé ainsi que le nombre de modalités de la variable décision [Crémilleux, 1991].

7) Il ne reste plus aucune variable candidate pour segmenter le nœud. En effet, à chaque fois qu'une variable est choisie comme test pour l'arbre de décision, elle est éliminée de la liste des variables candidates pour les nœuds suivants. Cette règle ne s'applique pas pour les variables numériques qui peuvent être réutilisées plusieurs fois (voir § 7.1.4.4). De même, les variables classifiées présentent des valeurs différentes si elles ont déjà été utilisées une fois pour la segmentation : il faut pour cela exploiter l'ordre introduit par les nœuds intermédiaires de la taxonomie des valeurs possibles : la variable est examinée paliers par paliers jusqu'aux feuilles terminales avant d'être éliminée de la liste des variables candidates.

7.1.4.4 ConstruireFeuille (E)

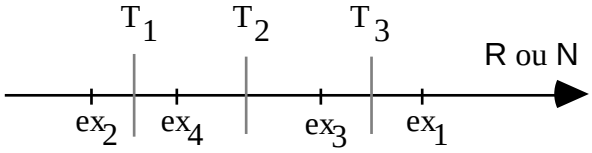
Chaque feuille construite est libellée par le nom de la classe correspondante.
Si tous les exemples à un nœud d_t (feuille ou nœud terminal) n'ont pas la même valeur de classe, on calcule la "probabilité" P_d associée à chaque classe c_k présente en d_t :

$$P_d = \frac{Card(E_d)}{Card(E)}$$

A chaque libellé de classe étiquetant le nœud terminal est associé la probabilité calculée P_d. Cette configuration correspond à une ambiguïté ou un "clash" (voir § 1.6.1.3).

7.1.4.5 CalculerSeuil (A,E)

Cette fonction ne s'applique qu'aux attributs numériques (à valeurs ordonnées). Tout d'abord, l'ensemble des exemples E est trié selon les valeurs croissantes prises pour l'attribut A considéré :



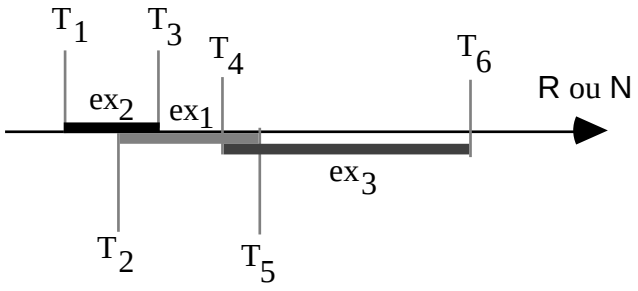
L'ensemble des valeurs de A prises par E est fini et noté $\{v_1,...,v_n\}$. Chaque point entre deux paires d'exemples dans la liste triée est alors calculé pour former un seuil potentiel de discrimination. Etant données n valeurs distinctes de A prises par E, il y a (n - 1) évaluations possibles, ce qui donne un ensemble de T_{n-1} seuils potentiels avec

$$T_i = \frac{v_i + v_{(i+1)}}{2}.$$

A chaque évaluation de T_i , les exemples E sont séparés en deux parties E_1 et E_2 (binarisation de l'attribut) et on calcule comme avant le gain d'information de chaque seuil potentiel T_i, le test étant alors booléen : $A(E_1) \leq T_i$ et $A(E_2) > T_i$.

Après les (n - 1) évaluations, on choisit le seuil T qui possède le meilleur gain d'information.

S'il existe des intervalles dans l'ensemble des valeurs prises par A, on applique le même principe d'ordonnancement des exemples selon l'axe des entiers ou des réels. Les seuils potentiels sont les bornes des intervalles de chaque exemple. S'il y a n valeurs (intervalles) pour tous les exemples, cela donne 2n - 2 seuils potentiels à calculer (en ôtant les bornes les plus extrèmes) :



Néanmoins, pour le calcul du gain d'information de chaque seuil, le problème est ici plus délicat du fait du recouvrement des intervalles entre les différents exemples : le même exemple peut être comptabilisé deux fois pour un seuil donné : $A(Ex) \leq T_i$ et $A(Ex) \geq T_i$. Mais cela ne gêne pas le calcul du gain d'information lorsque les modalités ne sont pas *disjointes* (un individu peut prendre plus d'une modalité pour la variable A), de même que lorsqu'elles ne sont pas *exhaustives* (un individu peut ne pas prendre une modalité de la variable, ce qui est le cas de la réponse «inconnu»).

En effet, que ce soit pour une variable numérique où le test est binarisé (n = 2 branches ou valeurs possibles) ou pour une variable nominale n_aire, si la valeur de A est inconnue pour un exemple, alors toutes les valeurs sont possibles : l'exemple est propagé sur les n branches. Par contre, si l'exemple possède plusieurs valeurs résultant de l'imprécision des descriptions observées de l'observateur (voir § 3.6.10.2), il est propagé sur ces branches uniquement. Afin que la mesure du gain d'information reste consistante, la taille du sous-ensemble E au nœud d est artificiellement modifiée :

Card (E) = Card (E) + (p - 1), ou p est le nombre de branches ou l'exemple a été propagé.

Remarques : Dans [Fayyad & Irani, 1992], il est montré qu'il est inutile de calculer le gain d'information des (n - 1) points possibles de la partition engendrée par les n valeurs d'un attribut numérique : il suffit de ne considérer que les seuils qui séparent deux classes différentes après avoir trié les exemples par ordre croissant. Cette fonctionnalité n'est pas encore implantée dans l'algorithme.

Par contre, un attribut de type entier ou réel, s'il est choisi à un nœud de l'arbre, peut être réutilisé dans la liste des tests possibles pour engendrer le sous-arbre du nœud (contrairement aux autres tests non numériques qui sont éliminés de la liste). Les valeurs possibles de ce nouveau test ont alors un sous espace d'observation O' restreint et déterminé par le calcul du seuil du test initial.

Pour une étude de la complexité globale de l'algorithme, on peut se reporter à [Manago, 1988], [Crémilleux, 1991].

7.2 De l'induction au raisonnement par cas

Depuis une dizaine d'années, la technologie de l'induction a été utilisée pour faciliter la mise en œuvre d'un projet de système expert. Au lieu de demander à un expert humain de fournir lui-même des règles logiques qui modélisent son savoir-faire, on lui demande de constituer une *base d'exemples* représentative de son domaine. Les connaissances nécessaires au système expert sont ensuite extraites par induction à partir des exemples. Cette *base de connaissances* se présente sous la forme d'un *arbre de décision* ou de *règles.*

Considérons une base d'exemples pour notre application de détermination d'éponges marines (figure 7.2) :

cas	**Classe**	**Forme(corps)**	**Extrémité(dents)**	…
Ex1	Paradisconema	Ellipsoïde	Elargies	…
Ex2	Coscinonema	Conique	En-lancette …	
Ex3	Corynonema	Ellipsoïde	En-lancette …	
…	...	...	… …	

Fig. 7.2 : Exemples sous la forme d'un tableau de données

Le but est de reconnaître efficacement une classe à partir de ses caractéristiques. L'algorithme d'induction KATE permet de construire automatiquement un arbre de classification tel celui de la figure 7.3. Cet arbre de décision peut dans un deuxième temps être exploité pour déterminer un nouveau cas : les nœuds intermédiaires de l'arbre correspondent à des questions posées à l'utilisateur, les feuilles ou nœuds terminaux correspondent à la conclusion donnée par le système expert.

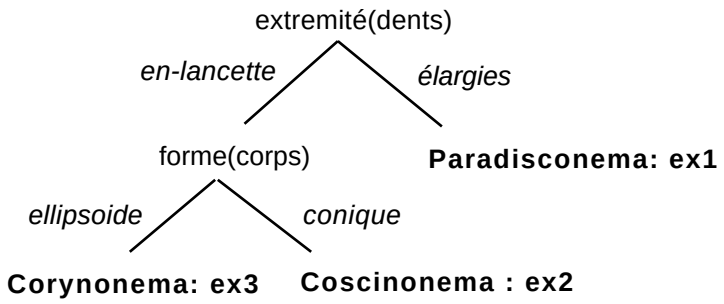


Fig. 7.3 : Un arbre de classification (ou de décision)

7.2.1 Formalisation de la procédure de détermination

Une détermination à partir d'un arbre de décision se fait en débutant à la racine de l'arbre T, que nous appellerons d_0. A chaque nœud de T, se trouve un critère de détermination $d_i = (A_i, V_i, E)$:

- A_i est le nom d'un attribut (A_i [Y_i S_i),
- V_i est l'ensemble des valeurs observables de A_i, $V_i = \{v_1,...,v_i,...,v_n\}$,
- E est l'ensemble des exemples w restant au nœud d.

Le cas w à déterminer est apparié à $d_i = d_0$, puis en fonction de la (ou des) valeur(s) qu'il prend pour l'attribut A_i, le (ou les) nœud(s) fils d_i' devien(nen)t candidat(s) pour l'(les) appariement(s) suivant(s). La procédure s'arrête lorsque l'on atteint une (les) feuille(s) libellée(s) par un nom de concept. Ce nom devient le résultat de la détermination. Lorsque plusieurs feuilles sont atteintes, le résultat est une combinaison de concepts avec des coefficients de vraisemblance associés à chacun d'eux, et calculés en fonction du nombre d'exemples indexés à chacune des feuilles.

Soit $D = D_n$ $D_t = \{d_i\}$, l'ensemble des nœuds de T.
$D_n = \{d_n\}$ est l'ensemble des nœuds intermédiaires,
$D_t = \{d_t\}$ est l'ensemble des nœuds terminaux.

Le parcours de l'arbre pour la détermination est exprimé par l'algorithme récursif suivant :

Algorithme :

Début :
$d_i = d_0$,

Apparier (w, d_i)
si d_i D_t alors
A_i (w) = c_i w $\{c_i\}$
sinon
Pour tout v_i d_i
si y_i (w) = v_i alors Apparier (w, d_i')
Fin pour tout
Fin si
Fin.

7.2.2 Limites de l'approche inductive

7.2.2.1 Apprentissage automatique = perte d'information

Toutes les techniques issues de l'apprentissage, qu'elles soient empiriques ou analytiques, présentent un certain nombre de limites inhérentes à l'approche. Pour l'induction à partir d'exemples, c'est-à-dire à partir de la *représentation en extension* des classes à apprendre (des descriptions de spécimens), l'algorithme va dériver une *représentation en intension* (une caractérisation) des classes par des concepts. Il en résulte des définitions différentielles (ou diagnoses) permettant de délimiter les concepts les uns des autres.

Cette représentation en intension s'accompagne d'une *généralisation* des exemples, de façon à prendre en compte des individus autres que les exemples eux-mêmes. Les généralisations peuvent être obtenues à l'aide de diverses techniques mais quelle que soit celle choisie, on va perdre de l'information contenue dans les exemples. C'est à la fois l'avantage de l'approche et son inconvénient car on risque d'éliminer une information utile. Tout l'art consiste à déterminer quelles sont les informations utiles qui doivent apparaître dans la définition en intension des concepts. Malheureusement, pour certaines applications (dont celles en biologie), il est impossible de prédire à l'avance quelles sont les informations importantes qu'il faut conserver.

7.2.2.2 Gestion de l'inconnu en phase de consultation

Considérons l'arbre d'identification de la figure 7.3. En phase de consultation, le système expert de reconnaissance d'éponges va d'abord demander à l'utilisateur comment est l' “extrémité des dents”. Supposons que ce dernier ne soit pas en mesure de répondre (la réponse est «inconnu»). L'inférence suit les deux branches “en-lancette” et “élargies” puis combine les réponses aux feuilles de l'arbre. Dans la branche “élargies”, nous obtenons un diagnostic partiel (Paradisconema avec 1 exemple). Dans la branche “en-lancette”, le système expert demande ensuite quelle est la forme du corps. L'utilisateur répond “conique”. Le système expert conclut alors qu'il s'agit de Coscinonema (0.5) ou de Paradisconema (0.5), ce qui dénote que ces deux conclusions sont également possibles. Ce diagnostic incertain est obtenu en combinant les exemples aux deux feuilles de l'arbre que nous avons atteint au cours de la consultation comme indiqué dans la figure 7.4 :

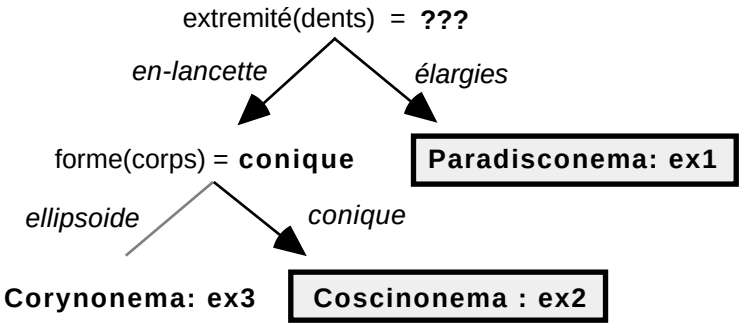


Fig. 7.4 : Consultation de l'arbre de décision de la figure 7.3

Pourtant lorsque nous regardons la forme du corps de ex1, nous nous apercevons que celle-ci est “ellipsoide”. Il y a donc une incohérence avec les informations fournies par l'utilisateur puisque le cas courant a la forme du corps “conique”. Le cas courant ne peut donc pas être une Paradisconema. Le système expert n'a pas fait cette conclusion car l'information sur la forme du corps de ex1, présente dans les données originales, a été délaissée suite à la phase d'induction. Ce problème se retrouve dans tout raisonnement à partir de connaissances générales (acquises par apprentissage ou non).

On pourrait objecter que lorsque les exemples d'apprentissage présentent des configurations de valeurs inconnues analogues à celles rencontrées durant la consultation, nous obtiendrions le bon résultat. En fait, pour de nombreuses applications du monde réel, il serait absurde de constituer une base d'exemples comprenant toutes les configurations d'inconnu possibles. Il semble aussi fallacieux de présenter comme un enrichissement de la connaissance l'accumulation de non-information !

7.2.2.3 Rigidité de la consultation interactive

Mis à part le problème des réponses inconnues, le raisonnement à partir d'arbres de décision présente d'autres limites. Le raisonnement est trop dirigiste et l'utilisateur est contraint de répondre aux questions dans un ordre pré-déterminé et rigide. Si ce mode de consultation est adapté pour des utilisateurs naïfs, il ne l'est plus lorsque l'utilisateur final est un expert du domaine. En effet, l'expert se lasse vite de ce jeu des questions-réponses alors qu'il estime pouvoir fournir directement l'information discriminante. Il est éventuellement prêt à répondre à 2 ou 3 questions complémentaires si cette information est insuffisante pour conclure, mais il veut rester maître de la consultation et entend suivre son propre raisonnement plutôt que la progression “artificielle” de la déduction.

De plus, le problème de tous les systèmes experts à base de règles de production (ou d'un arbre de décision) est qu'ils sont incapables de court-circuiter leur mécanismes habituels devant un cas particulier alors que des experts humains prennent parfois une décision brusque simplement parce que par exemple la

situation présente leur rappelle une situation grave *analogue* rencontrée dans le passé : à ce moment, ils n'ont pas besoin de cerner progressivement une hypothèse explicative comme ils le font d'habitude. La remémoration des faits fondée sur des ressemblances frappantes ou airs de famille [Wittgenstein, 1953] ayant une importance primordiale dans l'intellect humain, il nous a semblé intéresssant de pouvoir étudier ce type de raisonnement analogique en phase d'identification d'une nouvelle observation.

7.3 Le raisonnement par cas

7.3.1 Généralités

Le raisonnement par cas (“case-based reasoning”) est le nom donné aux techniques de résolution de problèmes qui font appel aux expériences passées plutôt qu'à un corpus de connaissances synthétisées [Bareiss, 1989]. La distinction essentielle entre le raisonnement par cas et d'autres méthodes automatiques de raisonnement est qu'un nouveau problème est résolu en reconnaissant sa similitude avec des problèmes résolus précédemment, puis en transférant leurs solutions.

Certains auteurs affirment que le raisonnement par cas est une forme de raisonnement analogique qui se place dans le cadre strict d’un domaine. Les recherches analogiques se situent dans le contexte plus global de trouver les analogies entre différents domaines [Burstein, 1989], [Hall, 1989].

D'autres auteurs définissent plus formellement le raisonnement par analogie comme étant un processus de démonstration du quatrième terme à partir des trois premiers [Bourrelly & Chouraqui, 1985]. Il enchaîne deux phases, comparaison et transfert, prenant appui sur la reconnaissance implicite d'une dépendance entre les éléments constituant la seconde paire de l'analogie (figure 7.5) :

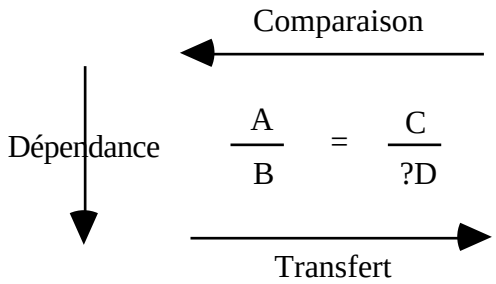


Fig. 7.5 : L'analogie selon Bourrelly et Chouraqui (1985)

L'exemple de la figure 7.6 permet d'illustrer ce point de vue [Vogel, 1988]. Les symptômes de l'incident à diagnostiquer sont mis en rapport avec des symptômes déjà rencontrés pour permettre le transfert du diagnostic correspondant sur l'incident actuel :

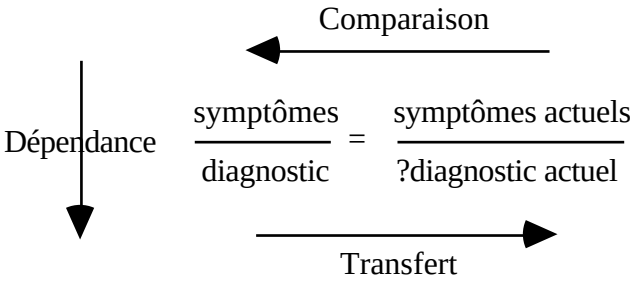


Fig. 7.6 : L'explication sur anomalies connues

Enfin, d'autres auteurs plus synthétiques [Aamodt & Plaza, 1994] englobent dans le terme "raisonnement par cas" l'ensemble des méthodes de raisonnement fondées sur les exemples, les instances, la mémoire, les cas, l'analogie (exemplar-based, instance-based, memory-based, case-based, analogy-based reasoning). Toutes ces méthodes partagent les mêmes traitements qui sont les tâches à réaliser pour obtenir un système de raisonnement par cas :

1) **Rechercher** le(s) cas le(s) plus similaire(s),
2) **Réutiliser** l'information et la connaissance du (des) cas pour résoudre le problème,
3) **Réviser** la solution proposée,
4) **Retenir** la partie de cette expérience qui pourrait être utile à la résolution d'un nouveau problème.

Les expériences passées (les "cas") qui sont utilisées au cours du raisonnement peuvent avoir été acquises par le système ou avoir été fournies au départ. Par contraste, les autres formes de résolution de problèmes, comme l'induction ou le raisonnement à base de règles, dérivent la solution à partir d'une caractérisation générale d'un groupe de problèmes ou à partir d'un ensemble de connaissances encore plus générales.

Les travaux de recherches menés au cours de ces dernières années ont montré que différentes classes de problèmes peuvent être traitées à l'aide de techniques de raisonnement par cas. Outre les problèmes de classification (voir en particulier [Kolodner J.R & Kolodner R.M., 1985], [Kibler & Aha, 1987], le raisonnement par cas a été utilisé pour des problèmes de planification [Kolodner, 1987], [Simpson, 1985], de raisonnement légal à partir de la jurisprudence [Ashley, 1987], [Bain, 1986], pour une gestion dynamique de la mémoire [Kolodner, 1983a], [Kolodner, 1983b], la reconnaissance de la parole [Bradshaw, 1987], la prononciation de mots [Stanfill & Waltz, 1986], [Lehnert,

1987], la détermination des structures secondaires de protéines [Zhang *et al.*, (à paraître)], etc..

7.3.2 Notre procédure de raisonnement par cas : CaseWork

En nous plaçant dans le cadre de la définition sous forme de tâches du raisonnement par cas [Aamodt & Plaza, 1994], CaseWork effectue les deux premiers traitements, c'est-à-dire Rechercher et Réutiliser. Les deux autres tâches (Réviser et Retenir) font partie de la procédure de validation qui est assurée par l'expert dans notre méthodologie d'acquisition des connaissances (voir § 2.4). Notons aussi que la réutilisation n'est qu'une *copie* du résultat (le nom du concept associé à l'attribut Classe du cas similaire) et qu'il n'y a pas d'*adaptation* de la solution proposée par transformation ou dérivation, telle qu'elle est expliquée dans [Carbonell, 1986].

Pour notre exemple, au lieu de raisonner sur le cas courant avec un arbre de décision, le système part directement de la base de cas de références. Nous utilisons une technique de base analogue à celle utilisée dans le système d'induction KATE (optimisation du gain d'information) mais, au lieu d'engendrer complètement une structure statique d'arbre de décision puis d'oublier les exemples d'apprentissage, nous raisonnons directement sur les exemples pour engendrer dynamiquement un chemin dans un arbre (fictif et implicite) qui correspond au cas courant. Les autres branches de l'arbre, qui n'ont pas d'intérêt pour le cas courant, ne sont pas développées.

Ce module de raisonnement par cas permet de mieux traiter le problème des réponses inconnues en phase de consultation et d'avoir un outil flexible totalement guidé par l'utilisateur. En effet, en phase de construction de l'arbre de décision, les critères sont ordonnés à chaque nœud en fonction de leur pouvoir discriminant comme on peut le voir sur la figure 7.7.

En phase d'induction, à chaque nœud de l'arbre, seul le premier critère (celui qui a le meilleur gain) est utilisé pour construire l'arbre globalement optimal en terme d'efficacité (cet arbre cache la forêt des autres arbres possibles !). Pour la figure ci-dessous, c'est la forme du corps qui est choisie à la racine pour générer l'arbre de décision (le gain d'information est égal à 1). La forme homogène de l'arbre (bien équilibré) traduit cette efficacité.

Pour le raisonnement par cas, aucune structure d'arbre n'est générée. Il suffit que l'utilisateur réponde «inconnu» à un nœud correspondant à la question associée au premier critère pour que le système remplace ce critère non renseigné par son successeur ayant un pouvoir de discrimination juste inférieur, et ainsi de suite jusqu'à épuisement de la liste des critères si l'utilisateur n'a aucune information à apporter en réponse aux questions posées (ce qui n'est pas réaliste

!)[2]. Pour la figure 7.7, si l'utilisateur ne connait pas la forme du corps de l'éponge, le système lui posera la question sur la longueur du rayon distal des pinules dermaux.

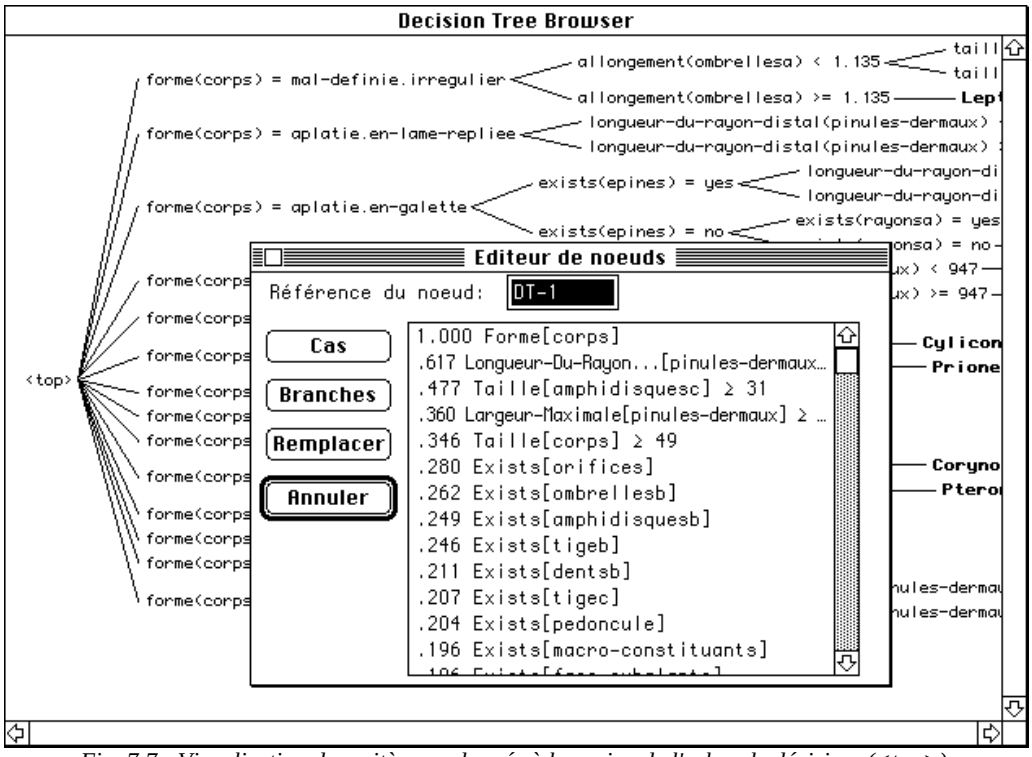


Fig. 7.7 : Visualisation des critères ordonnés à la racine de l'arbre de décision (<top>)

Pour notre exemple simple, cela donne la figure 7.8 suivante :

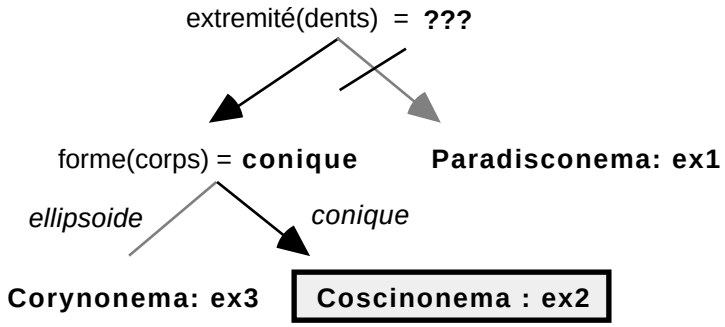


Fig. 7.8 : Une procédure de raisonnement par cas

[2] La procédure de remplacement se poursuit jusqu'à ce que l'on atteigne un critère avec un gain d'information égal à zéro. Si c'est le cas, chaque branche correspondant à la liste des valeurs possibles de cet attribut est parcourue en récupérant les exemples conformes à la valeur. Le gain d'information est réévalué sur chaque sous-arbre et les conclusions apportées sont pondérées en fonction de leur fréquence d'apparition et du nombre d'exemples correspondants.

Le fait que l'utilisateur ne sache pas répondre à la question sur l'extrémité des dents provoque le remplacement de ce critère par le second le plus discriminant : on arrive ainsi à déterminer totalement le nouvel individu en utilisant toute l'information disponible dans la base de cas.

En fait, les deux critères “extrémité(dents)” et “forme(corps)”, au vu du tableau de la figure 7.2 (et non de la figure 7.5), ont un gain d'information identique : ils discriminent les exemples avec la même probabilité. Théoriquement, rien ne justifiait donc le choix du premier critère pour la consultation au lieu du second car la mesure du gain d'information ne tient pas compte du contenu du message véhiculé par le critère.

Néanmoins, il est tout à fait possible de tenir compte d'un ordre sur les critères à utiliser en fonction d'une sémantique donnée lorsque ceux-ci ont le même pouvoir de discrimination : par exemple, l'expert peut indiquer dans le modèle descriptif une priorité d'utilisation liée à la facilité d'observer l'attribut (il est plus facile d'observer visuellement la forme du corps que l'extrémité des dents au microscope). Cette connaissance explicite supplémentaire peut donc être mise à contribution en phase de détermination pour améliorer la robustesse de la consultation.

7.3.3 Formalisation

La procédure de raisonnement par cas que nous avons développée dans CaseWork peut être décrite par l’algorithme suivant. Il explicite les deux tâches qui permettent de retrouver les cas “similaires” : Rechercher et Réutiliser. La méthode utilise toujours le gain d’information comme mesure de discrimination. Aucun nœud n’est construit, les critères choisis ne servent qu’à indexer les cas :

Algorithme :

Début :
E = Ω,

Rechercher (w, E, Y)
si Critèred'Arrêt (E) alors
w {c_i} ; Réutiliser
sinon
Y = ConstruireEspace (E) ; Récupèrer les attributs pertinents
s = OrdonnerCritères (E, Y) ; Ordonner les attributs
E_i = Sélectionner (w, s, E)
Rechercher (w, E_i, Y)
Fin si

Fin.

Sélectionner (w, s, E)

A = Meilleure_division (E, s)

si GainInformation (A) = 0 alors
    partition = R (E)
    Pour tout E_i partition
        Rechercher (w, E_i, Y)
    Fin Pour tout
Fin si

si A (w) = alors
    Sélectionner (w, s\A, E) ; "\" est le symbole d'exception
Fin si

Pour tout w_i E
    $E_i = \{w_i / A (w_i) = y_i (w) = v_i\}$; comparer les anciens cas avec celui
Fin pour tout ; à identifier et les sélectionner

retourner E_i

7.3.4 Comparaison des deux approches

L'approche "raisonnement par cas" pour la détermination correspond à une méthode d'**identification par comparaison** des descriptions. Il n'y a pas généralisation des exemples sous la forme d'un arbre de décision. Elle s'oppose en cela à l'approche déductive d'utilisation d'un arbre sous la forme d'une clé d'identification qui représente une classification artificielle préexistante. Notre procédure de raisonnement par cas peut être comprise comme une recherche *multi-accès orientée* et *séquentielle mono-critère :*

1) Elle est orientée car la recherche est guidée par la quantité d'information véhiculée par chaque attribut, de manière à aboutir à une identification rapide. Elle est multi-accès du fait des possibilités de remplacement d'un critère par un autre lorsque l'utilisateur ne sait pas répondre au premier.

2) Elle est séquentielle et mono-critère du fait que la recherche ne se base pas sur une combinaison de critères associés (disjonctions de critères en parallèle) à un moment donné de la procédure de consultation, mais sur une séquence ordonnée dans le temps d'un seul critère à la fois (conjonction de critères en série).

La comparaison basée sur le choix d'un seul critère à un moment donné est analytique. Elle est aussi qualifiée de **monothétique** [Pankhurst, 1991] ou mono dimensionnelle [Fenelon, 1981]. La mesure d'entropie utilise la liste des attributs disponibles à chaque étape pour évaluer leurs différents pouvoirs de séparation des exemples conditionnellement aux différentes classes présentes. De ce fait, on peut qualifier le gain d'information comme une mesure de discrimination **inter-classe**. Dans cette méthode, on s'intéresse à la distribution relative des exemples par rapport aux valeurs possibles de chaque attribut, c'est-à-dire que l'on compte les exemples par rapport aux attributs. Ici, ce sont les attributs qui sont comparés entre eux, ce qui revient à travailler sur la définition en intension des concepts.

D'autres procédures de détermination comparent les exemples entre eux, c'est-à-dire à partir de la représentation en extension des classes. Ces méthodes sont **polythétiques** car elles étudient toutes les configurations possibles d'appariement des exemples à chaque étape de la division. Ce sont des méthodes synthétiques d'analyse multi-dimensionnelle des données [Bertier & Bouroche, 1981]. Elles se basent sur une mesure de proximité entre les exemples appartenant à une même classe comme par exemple pour la recherche des *k plus proches voisin* en analyse discriminante [Celeux *et al.*, 1989]. Ces mesures de ressemblance tenant compte de l'homogénéité des descriptions à l'intérieur d'une même classe sont des mesures **intra-classe**. Une distance est calculée pour évaluer la similarité entre les exemples qui sont appariés deux à deux. Cette approche tient compte de tous les attributs à la fois parce qu'elle compte les identités et les différences au niveau des valeurs prises par les attributs par rapport aux exemples : il s'agit d'un comptage des attributs par rapport aux exemples.

L'avantage d'une procédure de raisonnement par cas par rapport à une procédure de détermination déductive (ou associative) est son incrémentalité. Casework prend en compte tous les cas qui sont actuellement dans la base. Contrairement à KATE, il n'est plus nécessaire de passer par une phase de mise à jour et d'engendrer un nouvel arbre lorsqu'on rajoute un nouvel exemple.

Sa difficulté est que justement, elle contraint à travailler sur la quasi-totalité de l'information disponible, ce qui peut s'avérer d'une lourdeur insurmontable pour les méthodes polythétiques. En ce qui concerne notre méthode monothétique, nous n'avons jusqu'à présent pas rencontré de problèmes d'efficacité même pour des applications volumineuses dans d'autres domaines que la biologie. Le nombre de cas considérés se réduit très rapidement au fur et à mesure des questions et le temps de calcul du meilleur critère, compte tenu de l'efficacité de la méthode, n’est pas une contrainte d’utilisation.

Donc, le raisonnement par cas peut se substituer avantageusement à l'induction pour la phase de consultation interactive. En revanche, il ne permet pas

d'acquérir des connaissances explicites sur le domaine à partir de la base de cas. Le processus inductif extrait des connaissances à partir des données d'apprentissage sous la forme d'un arbre de décision ou d'une base de règles. Ces connaissances peuvent être exploitées d'une part pour mieux comprendre le domaine d'application, d'autre part pour déterminer si le domaine a été bien formalisé. La présence de conclusions incertaines aux feuilles de l'arbre peut indiquer que le domaine a été mal défini et qu'il faut rajouter de nouveaux critères pour décrire les données. Les autres avantages respectifs des deux approches sont décrits dans [Manago *et al.*, 1993] (voir annexe 6).

Les deux technologies de l'induction et du raisonnement par cas sont donc complémentaires. L'induction permet d'acquérir et de valider une base de connaissances. Le raisonnement par cas permet de maintenir l'application et améliore la qualité des consultations. Ces deux technologies sont en cours d'intégration dans le cadre du projet européen INRECA en collaboration avec Irish Medical System (Irelande), tecInno et l'Université de Kaiserslautern (Allemagne). La combinaison des deux approches permettra de mieux répondre aux besoins des utilisateurs en améliorant la robustesse de la détermination.