Les incertitudes de la classification incertaine

Roland Ducournau
L.I.R.M.M. – 161, rue Ada – 34392 Montpellier cedex
(ducournau@lirmm.fr)

29 avril 1996

Résumé

Les systèmes de représentation des connaissances par objets comme SHIRKA ont introduit une classification incertaine où la relation entre une classe et une instance peut être qualifiée de *sûre*, *possible*, ou *impossible*. L'objectif limité de cet article est de questionner, de façon abstraite, cette classification incertaine, en plongeant ces systèmes dans le cadre sémantique des *systèmes classificatoires* qui provient lui-même d'une abstraction de celui des *logiques terminologiques*.

Mots-clés : classification, logique terminologique, représentation des connaissances par objets, système classificatoire, intension, extension.

Des langages de programmation à objets aux logiques terminologiques (LT), les différentes approches fondées plus ou moins sur la notion d'objet trouvent un cadre sémantique commun dans la notion de *système classificatoire* (SC). Un système classificatoire est constitué par une *hiérarchie* de *classes*, définies en *intension* par un ensemble de *propriétés*, et munies d'une interprétation *extensionnelle* [Duc96]. Un SC peut ainsi être considéré comme un système de déduction, donc une logique, avec une sémantique directe en théorie des modèles, sur le modèle de celles établies depuis quelques années dans l'approche des LT [Neb90]. La justification de “classificatoire” est double : ces systèmes offrent une “classification des espèces”, au sens de la hiérarchie de classes, et ils se prêtent à une “classification des individus”, au sens du mécanisme d'inférence qui reconnaît qu'un individu appartient à une classe. C'est à ce mécanisme d'inférence que cet article est consacré.

Cette approche classificatoire a été reconnue et mise en œuvre à la charnière des années 70 et 80 dans les langages dits de *frames* [FK85] qui ont donné naissance à deux courants principaux. L'un des critères discriminant ces deux courants est le caractère *définitionnel* ou *descriptif* des classes. Les logiques terminologiques (LT) ont introduit la distinction entre concept (classe) *défini* ou *primitif* [Neb90] : pour une classe définie, la description de la classe constitue une *condition nécessaire et suffisante* (CNS) d'appartenance d'un objet à l'extension de la classe. En revanche, pour une classe primitive, la description n'est qu'un ensemble de conditions nécessaires. Des classes définies autorisent la classification automatique des instances que pratiquent les LT sous le nom de *réalisation*. De leur côté, les systèmes de représentation des connaissances par objets

(RCO), dont la sémantique des classes est purement descriptive, pratiquent souvent, depuis SHIRKA [Rec88] et ses successeurs TROPES [Mar93, ER95] ou FROME [Dek94], une classification que l'on peut qualifier d'*incertaine* puisqu'elle attribue à la relation classe-instance une modalité parmi *sûre*, *possible* et *impossible*.

Chacune de ces approches présente une limitation importante : pour les LT, l'appartenance à une classe primitive ne peut en général pas se déduire, alors que la RCO ne peut effectuer presqu'aucune inférence automatique valide.

L'objectif limité de cet article est de questionner, de façon abstraite, cette classification incertaine, en plongeant ces systèmes de RCO dans le cadre sémantique des SC qui provient lui-même d'une abstraction de celui des LT.

La première partie de l'article introduit le cadre général des SC et de leur sémantique formelle. La seconde partie pose le problème de la classification d'instances et donne deux versions, extensionnelle et intensionnelle, de la classification incertaine, avant d'examiner les questions de correction et de complétude qui ne manquent pas de se poser. L'article se termine par une comparaison avec des travaux connexes et ouvre quelques perspectives.

1 Les systèmes classificatoires

Le problème de la classification se pose dans un cadre de représentation par objets, au sens large, défini par une *hiérarchie* de *classes* décrites en *intension* et interprétées en *extension*. Cette hiérarchie constitue une première composante, *intensionnelle*, ce que les LT nomment la *TBox* ; il lui correspond une seconde composante, *extensionnelle*, la *ABox* des LT dans la définition restreinte de [Neb90] ou de [BDS93]. L'ensemble constitue un *système classificatoire* (SC).

1.1 La composante intensionnelle : les classes

La composante intensionnelle est formée d'une hiérarchie de classes, décrites par des attributs, et de diverses relations entre ces classes.

1.1.1 La hiérarchie et son interprétation extensionnelle

$\mathcal{X}$ est un ensemble fini d'*entités génériques* ou *classes*, ordonnées par une relation de *spécialisation* $\prec$ qui est un ordre partiel strict (et $\preceq$ est l'ordre partiel associé). Comme c'est l'habitude dans les approches objet, on parlera de *sous-classe* ou de *super-classe* en référence à cette relation.

Cette hiérarchie s'interprète dans un cadre extensionnel dont nous présentons d'abord un aperçu intuitif et relativement informel : Ω est un ensemble d'*entités individuelles* ou *objets*, qui peuvent être associés aux classes par l'application $Ext : \mathcal{X} \longrightarrow 2^{\Omega}$ telle que $\alpha \preceq \beta \Longrightarrow Ext(\alpha) \subseteq Ext(\beta)$. $Ext(\alpha)$ est appelé l'*extension* de la classe α : c'est l'ensemble des objets qui "tombent sous le concept" α. Lorsque $o \in Ext(\alpha)$, o est une *instance de* α, et α est *une* (et non pas *la*) *classe de* o. Inversement, l'application $Isa : \Omega \longrightarrow 2^{\mathcal{X}}$ associe à tout objet o les classes dont il est instance, *ses* classes. La relation de *subsomption extensionnelle* est un *préordre* sur $\mathcal{X}$, noté $\trianglelefteq$, induit par l'inclusion des extensions : $\alpha \trianglelefteq \beta \stackrel{\text{def}}{\Longleftrightarrow} Ext(\alpha) \subseteq Ext(\beta)$. Enfin, il existe dans $\mathcal{X}$ un élément maximal $\top$, d'extension Ω, ainsi qu'un élément minimal, $\bot$, d'extension vide. Ce cadre extensionnel naïf trouvera dans la suite deux réalisations différentes : la première dans la composante extensionnelle qui décrit les instances effectives, la seconde dans la sémantique formelle du SC.

1.1.2 La description intensionnelle des classes

Chaque classe est accompagnée d'une "description" *en intension*, par une collection de "propriétés" appelées *attribut*. $\mathcal{P}$ est un ensemble d'attributs et, $\forall \alpha \in \mathcal{X}$, $\mathcal{P}_\alpha$ représente l'ensemble des attributs de α. L'*héritage de propriétés* consiste d'abord en une inclusion des ensembles d'attributs associés à des classes qui se spécialisent : $\alpha \prec \beta \Longrightarrow \mathcal{P}_\alpha \supseteq \mathcal{P}_\beta$.

Une application $Dom : \mathcal{P} \times \mathcal{X} \longrightarrow 2^{\mathcal{D}}$ (où $\mathcal{D}$ est un domaine quelconque contenant au moins Ω et 2^Ω) associe à chaque attribut d'une classe un *domaine*. On notera aussi Dom_P (resp. Dom_α) la fonction obtenue en fixant P (resp. α).

Le cadre intensionnel est relié au cadre extensionnel comme suit. D'abord, chaque objet possède l'ensemble des attributs décrits dans ses classes :
$Attr(o) \stackrel{\text{def}}{=} \bigcup_{\alpha \in Isa(o)} \mathcal{P}_\alpha$. Ensuite, à tout attribut P qu'il possède ($P \in Attr(o)$), il est associé une *valeur*, notée $P(o)$, dont $Dom(P, \alpha)$ représente les valeurs admissibles. si $o \in Ext(\alpha)$ et $P \in \mathcal{P}_\alpha$, alors $P(o) \in Dom(P, \alpha)$. L'*héritage de propriétés* n'est pas seulement l'héritage des attributs : c'est aussi l'héritage de leur domaine et une contrainte d'inclusion en cas de *redéfinition* de ces domaines : si $\alpha \prec \beta$ et $P \in \mathcal{P}_\beta$, alors $Dom_P(\alpha) \subseteq Dom_P(\beta)$.

A ce stade, on peut introduire la notion d'*intension* d'une classe, notée $Int(\alpha)$ et constituée par le couple $(\mathcal{P}_\alpha, Dom_\alpha)$. Les intensions sont munies d'une relation d'ordre $\sqsupseteq$ similaire à l'inclusion comme suit :

$$Int(\alpha) \sqsupseteq Int(\beta) \stackrel{\text{def}}{\Longleftrightarrow} \mathcal{P}_\alpha \supseteq \mathcal{P}_\beta \ \& \ \forall P \in \mathcal{P}_\beta : Dom_P(\alpha) \subseteq Dom_P(\beta)$$

Int est donc (non strictement) monotone. $\alpha \prec \beta \Longrightarrow Int(\alpha) \sqsupseteq Int(\beta)$. La relation de *subsomption intensionnelle* est un *préordre* sur $\mathcal{X}$, noté $\trianglerighteq$, induit par l'inclusion des intensions : $\alpha \trianglerighteq \beta \stackrel{\text{def}}{\Longleftrightarrow} Int(\alpha) \sqsupseteq Int(\beta)$. Les opérations ensemblistes usuelles s'étendent aux intensions en introduisant les opérateurs $\sqcap$ et $\sqcup$: $Int(\alpha) \sqcup Int(\beta) \stackrel{\text{def}}{=} (\mathcal{P}_\alpha \cup \mathcal{P}_\beta, Dom_{\alpha\cap\beta})$, avec

$$Dom_{\alpha\cap\beta}(P) = \begin{cases} Dom_\alpha(P) \cap Dom_\beta(P) & \text{si } P \in \mathcal{P}_\alpha \cap \mathcal{P}_\beta \\ Dom_\alpha(P) & \text{si } P \in \mathcal{P}_\alpha - \mathcal{P}_\beta \\ Dom_\beta(P) & \text{si } P \in \mathcal{P}_\beta - \mathcal{P}_\alpha \end{cases}$$

Dans cette définition, si jamais l'un des $Dom_\alpha(P) \cap Dom_\beta(P)$ est vide (si P n'a aucune valeur admissible), on pose alors que $Int(\alpha) \sqcup Int(\beta) = (\mathcal{P}, \emptyset) = Int(\bot)$. $\sqcap$ se définit de manière duale.

1.1.3 Relation d'incompatibilité

Il faut ajouter à ce cadre général une relation d'incompatibilité, notée $\alpha \nabla \beta$, qui peut se définir extensionnellement par $Ext(\alpha) \cap Ext(\beta) = Ext(\bot)$, aussi bien qu'intensionnellement par $Int(\alpha) \sqcup Int(\beta) = Int(\perp)$. La définition intensionnelle implique la définition extensionnelle, mais ne lui est pas équivalente : il peut donc être nécessaire de déclarer explicitement certaines relations d'incompatibilité, en plus des déclarations intensionnelles implicites. La relation d'incompatibilité est héritable : si deux classes sont incompatibles, les sous-classes de l'une sont incompatibles avec (les sous-classes de) l'autre.

1.1.4 L'expression du système classificatoire

En pratique, une classe α est introduite au moyen de ses super-classes directes $Super(\alpha)$, explicitement désignées ou, par défaut, $\{\top\}$, et par son *intension propre*, notée $Int^*(\alpha)$, par laquelle sont ajoutés des attributs ou restreints des domaines. Alors, $Int(\alpha) = (\bigsqcup_{\beta \in Super(\alpha)} Int(\beta)) \sqcup Int^*(\alpha)$. On suppose bien sûr que le domaine de tout attribut *hérité* sans redéfinition s'obtient par intersection des domaines hérités. Il s'agit donc soit d'un simple héritage du domaine, soit de ce que [DHH+95] appelle une résolution correcte de conflit de valeur, sur l'aspect domaine de l'attribut P.

Pour que l'expression des classes soit complète, il faut aussi exprimer le domaine des attributs. Le problème est que Dom n'est pas donné en extension, mais en intension. Or le seul moyen que nous avons pour définir un ensemble, c'est les classes. En première approximation, $Dom(\alpha, P)$ peut donc être donné par le *type* de l'attribut, et par ses cardinalités admissibles, grâce à deux fonctions $type : \mathcal{X} \times \mathcal{P} \longrightarrow \mathcal{X}$ et $card : \mathcal{X} \times \mathcal{P} \longrightarrow 2^{\mathbf{N}}$, et l'ensemble des valeurs admissibles, $Dom(\alpha, P)$ est défini par : $\{A \subseteq Ext(type(\alpha, P)) \mid |A| \in card(\alpha, P)\}$.

Outre l'introduction intensionnelle des classes, par l'explicitation de leurs attributs et de la relation de spécialisation, il peut être aussi nécessaire d'expliciter les relations extensionnelles qui ne se déduisent pas de l'intension. On notera ainsi ∇_a la relation d'incompatibilité exprimée explicitement, pour affirmer par exemple que, dans le domaine modélisé, il n'existe pas d'objets de `[forme ronde]` et de `[couleur rouge]`.

1.2 La composante extensionnelle : les instances

1.2.1 Description du monde

La composante extensionnelle est une *description du monde* notée $\mathcal{W}$ et constituée de la description d'entités individuelles, ou instances. C'est une réalisation particulière de l'interprétation extensionnelle du SC. Cette description du monde doit donc respecter toutes les contraintes formulées dans l'exposé de cette interprétation. La description du monde peut être vue comme un ensemble fini d'assertions, analogue à une base de données relationnelle, ces assertions pouvant être de deux sortes :

- $(\alpha\ o) \in \mathcal{X} \times \Omega$ affirme l'existence de l'*objet* $o \in Ext(\alpha)$;
- $(P\ o\ o') \in \mathcal{P} \times \Omega \times \Omega$ associe à l'attribut P de l'objet o la valeur o'.

La seconde catégorie pose le problème des valeurs manquantes : la valeur de l'attribut P de l'objet o, notée $P(o)$, est constituée par éventuellement plusieurs assertions élémentaires : $(P\ o\ v_i), 1 \leq i \leq n$, et $v = \{v_i \mid 1 \leq i \leq n\}$.

L'ensemble de ces assertions s'interprétera comme une inclusion, $P(o) \supseteq v$: c'est une hypothèse du *monde ouvert* sur la valeur des attributs.

Les instances s'expriment tout aussi bien dans un style objet — c'est-à-dire en faisant apparaître des entités avec des attributs —, par un ensemble $\mathcal{O}$ d'instances o, appartenant à des classes $Isa_{\mathcal{W}}(o)$ et munies d'un ensemble d'attributs $Attr_{\mathcal{W}}(o)$ dont la valeur est notée $P_{\mathcal{W}}(o)$:

$$Isa_{\mathcal{W}}(o) \stackrel{\text{def}}{=} \{\beta \in \mathcal{X} \mid (\alpha\ o) \in \mathcal{W}\ \&\ \alpha \preceq \beta\} \tag{1}$$

$$Attr_{\mathcal{W}}(o) \stackrel{\text{def}}{=} \{P \mid (P\ o\ o') \in \mathcal{W}\} \tag{2}$$

$$P_{\mathcal{W}}(o) \stackrel{\text{def}}{=} \{o' \mid (P\ o\ o') \in \mathcal{W}\} \tag{3}$$

Ces fonctions caractérisent ce qui a été explicitement affirmé de l'objet, sans aucune inférence, à part la fermeture par $\preceq$ pour $Isa_{\mathcal{W}}$. La fonction $Ext_{\mathcal{W}}$, qui désigne les instances connues d'une classe, se définit symétriquement.

1.2.2 Les objectifs d'un système classificatoire

La construction de la hiérarchie elle-même mise à part, un système classificatoire peut être utilisé, au niveau extensionnel, de diverses façons :

l'héritage permet de connaître *déductivement* les attributs d'un objet — pas leurs valeurs, au moins dans un premier temps, mais leur présence et leur "intension" —, étant données ses classes ;

la classification permet de reconnaître *inductivement* un objet comme instance d'une classe, compte-tenu de ses attributs connus ; des assertions $\{(\alpha\ o)\ (P\ o\ v)\}$, il est possible de déduire que $v \in Ext(type(\alpha, P))$, mais il est aussi possible d'induire une classe β, éventuellement sous-classe de α, telle que $v \in Ext(type(\beta, P))$ et $o \in Ext(\beta)$.

le filtrage détermine les instances d'une classe ou, plus généralement, les objets qui vérifient certaines propriétés qui peuvent s'exprimer par l'intension d'une classe : dans les LT, toute requête est considérée comme une classe dont il ne reste plus qu'à collecter les instances, une fois qu'elle a été correctement insérée dans la hiérarchie. C'est aussi ce que fait [Dek93] dans FROME.

la construction d'instance [Gir95] consiste à valuer les attributs d'un objet, éventuellement en en créant d'autres qu'il faut récursivement construire et classer.

Mais la distinction de ces différents problèmes est artificielle : il s'agit d'abord d'un problème uniforme de déduction. Ce n'est qu'ensuite que l'on peut introduire des spécificités comme une classification ou un filtrage incertains.

1.3 Propriétés spécifiques des sc

[Euz93] propose un catalogue de propriétés structurelles des systèmes classificatoires. Ces propriétés sont examinées plus en détail dans [Duc96] et nous n'en reprendrons ici qu'une seule, utilisée plus loin : l'*exhaustivité* d'un SC se caractérise par le fait que tout objet est instance d'un (de) *puits* (c'est-à-dire un sommet sans successeur) de la relation $\preceq$ ($\perp$ mis à part). C'est le cas des *espèces naturelles* : il n'existe pas de pur mammifère, qui ne soit pas membre d'une sous-catégorie.

Dans les systèmes de RCO comme SHIRKA ou TROPES, la relation d'incompatibilité se déduit de deux autres propriétés spécifiques, l'*exclusivité* (deux classes incomparables sont incompatibles), ou l'*univocité* (mono-instanciation).

2 Sémantique des sc

La *théorie des modèles* offre un cadre général pour établir une sémantique formelle ensembliste : le but est en particulier de déterminer les inférences *valides*, c'est-à-dire valables pour tous les *modèles* du système considéré.

La représentation des connaissances part souvent du principe qu'elle représente une partie idéalisée du monde réel, et considère qu'il existe un *domaine modélisé*, c'est-à-dire un modèle privilégié. C'est par exemple la position de [Euz94]. Toutes les propositions qui sont vraies dans ce modèle particulier sans être valides traduisent alors une incapacité du système de représentation (ou/et de ses utilisateurs) à représenter fidèlement le domaine modélisé et lui seul. Mais, un SC est d'abord un système générique qui ne peut pas s'appuyer sur l'existence d'un domaine modélisé pour la réalisation de ses algorithmes : pour être corrects, ils doivent correspondre à des inférences valides. Aussi la prise en compte du domaine modélisé en tant que modèle privilégié ne paraît possible que dans le choix d'un SC muni de propriétés spécifiques, ou dans des extensions *ad hoc*. A cette *incomplétude intensionnelle* s'ajoute une *incomplétude extensionnelle* liée au fait que l'état du domaine modélisé, c'est-à-dire l'*état du monde*, n'est pas complètement connu.

2.1 Une sémantique en théorie des modèles

Le cadre général et les notations sont empruntés, avec quelques variantes, à [Neb90]. Précisons qu'il ne s'agit pas d'une sémantique complète, mais d'un schéma général où ne figurent que les équations communes à tous.

2.1.1 Modèle d'un sc

Une *structure sémantique* [1] d'un SC est un quadruplet $(\Delta, \mathcal{E}_{\mathcal{X}}, \mathcal{E}_{\mathcal{P}}, \mathcal{I})$ formé d'un domaine arbitraire Δ, d'une fonction d'interprétation des classes $\mathcal{E}_{\mathcal{X}} : \mathcal{X} \longrightarrow 2^{\Delta}$, d'une fonction d'interprétation des attributs $\mathcal{E}_{\mathcal{P}} : \mathcal{P} \longrightarrow (\Delta \rightarrow 2^{\Delta})$, et d'une fonction d'interprétation des objets, injective, $\mathcal{I} : \mathcal{O} \longrightarrow \Delta$. $\mathcal{O}$ est l'ensemble de toutes les instances introduites dans la description du monde.

1. Les notions de *structure sémantique*, *modèle* et *interprétation* sont souvent synonymes : nous avons pris le parti de désigner par *structure sémantique* un "type", constitué d'un domaine et des quelques fonctions associées à ce domaine, par *modèle* une structure sémantique vérifiant des propriétés particulières, et par *interprétation* les fonctions (ou leurs images) qui associent aux structures du langage des éléments du domaine.

Un *modèle* $\mathcal{M}$ d'un système classificatoire est une structure sémantique $(\Delta, \mathcal{E}_\mathcal{X}, \mathcal{E}_\mathcal{P}, \mathcal{I})$ où :

$$\alpha \preceq \beta \Longrightarrow \mathcal{E}_\mathcal{X}[\alpha] \subseteq \mathcal{E}_\mathcal{X}[\beta] \tag{4}$$

$$\alpha \nabla_a \beta \Longrightarrow \mathcal{E}_\mathcal{X}[\alpha] \cap \mathcal{E}_\mathcal{X}[\beta] = \emptyset \tag{5}$$

$$o \in \mathcal{E}_\mathcal{X}[\alpha], P \in \mathcal{P}_\alpha \Longrightarrow \begin{cases} \mathcal{E}_\mathcal{P}[P](o) \subseteq \mathcal{E}_\mathcal{X}[type(\alpha, P)] \\ |\mathcal{E}_\mathcal{P}[P](o)| \in card(\alpha, P) \end{cases} \tag{6}$$

$$o \notin \bigcup_{P \in \mathcal{P}_\alpha} \mathcal{E}_\mathcal{X}[\alpha] \Longrightarrow \mathcal{E}_\mathcal{P}[P](o) \text{ pas défini} \tag{7}$$

$$(\alpha\ o) \in \mathcal{W} \Longrightarrow \mathcal{I}[o] \in \mathcal{E}_\mathcal{X}[\alpha] \tag{8}$$

$$(P\ o\ o') \in \mathcal{W} \Longrightarrow \mathcal{I}[o'] \in \mathcal{E}_\mathcal{P}[P](\mathcal{I}[o]) \tag{9}$$

Pour une classe définie, on aura une équation supplémentaire du style[2] de :

$$\mathcal{E}_\mathcal{X}[\alpha] = \bigcap_{P \in \mathcal{P}_\alpha} \{o \mid \mathcal{E}_\mathcal{P}[P](o) \subseteq \mathcal{E}_\mathcal{X}[type(\alpha, P)]\ \&\ |\mathcal{E}_\mathcal{P}[P](o)| \in card(\alpha, P)\}$$

Si la classe est primitive, on a seulement l'inclusion, qui se déduit des formules précédentes.

2.1.2 Subsomption, incompatibilité, incohérence et inconsistance

On peut maintenant définir formellement diverses notions extensionnelles dont la présentation n'a été qu'intuitive :

- les relations de subsomption et d'incompatibilité extensionnelle $\trianglelefteq$ et ∇ se définissent en symétrisant les formules 4 et 5 et en les quantifiant universellement sur tous les modèles du SC :

$$\alpha \trianglelefteq \beta \overset{\text{def}}{\Longleftrightarrow} \forall \mathcal{M} : \mathcal{E}_\mathcal{X}[\alpha] \subseteq \mathcal{E}_\mathcal{X}[\beta] \tag{10}$$

$$\alpha \nabla \beta \overset{\text{def}}{\Longleftrightarrow} \forall \mathcal{M} : \mathcal{E}_\mathcal{X}[\alpha] \cap \mathcal{E}_\mathcal{X}[\beta] = \emptyset \tag{11}$$

- une classe *incohérente* est une classe dont l'extension est vide dans tous les modèles : c'est donc une classe subsumée extensionnellement par $\perp$;
- une description du monde est *inconsistante* si elle n'a aucun modèle.

2.1.3 La notion de conséquence

On distingue deux sortes de conséquence dans les descriptions du monde :

- le modèle $\mathcal{M}$ satisfait la description d, noté $\models_\mathcal{M} d$, qui correspond aux versions symétriques des clauses 8 et 9 ci-dessus ;

$$\models_\mathcal{M} (\alpha\ o) \Longleftrightarrow \mathcal{I}[o] \in \mathcal{E}_\mathcal{X}[\alpha] \tag{12}$$

$$\models_\mathcal{M} (P\ o\ v) \Longleftrightarrow \mathcal{I}[v] \in \mathcal{E}_\mathcal{P}[P](\mathcal{I}[o]) \tag{13}$$

2. Il y a plusieurs manières de définir une classe : celle-ci est absolue, sans référence aux super-classes, mais uniquement à l'intension de la classe. La définition habituelle est relative et s'exprime en fonction des super-classes (directes) et de l'intension propre.

– si tous les modèles de $\mathcal{W}$ satisfont d, noté $\mathcal{W} \models d$, d est une *conséquence valide* de $\mathcal{W}$: $\mathcal{W} \models d \iff \forall \mathcal{M} : \models_{\mathcal{M}} d$. Si $\mathcal{W}$ est *inconsistant*, toute description en est une conséquence valide, comme en logique classique.

Une description du monde $\mathcal{W}$ est *fermée* par $\models$ ssi $\mathcal{W} = \{d \mid \mathcal{W} \models d\}$.

La sémantique qui précède a vidé de toute signification formelle les notations intuitives comme $Isa(o)$, $Ext(\alpha)$, $Attr(o)$ ou $P(o)$. Il faut en fait choisir, entre trois notions différentes, suivant que l'on considère la description du monde $\mathcal{W}$, un modèle $\mathcal{M}$ donné, ou enfin tous les modèles de $\mathcal{W}$.

2.2 L'interprétation de l'incomplétude extensionnelle

La sémantique des systèmes à objets proposée ici fait l'hypothèse du *monde ouvert*, comme le font les LT, mais à l'opposé de l'hypothèse habituelle du *monde clos* de la programmation logique. Non seulement tous les objets ne sont pas connus, mais les objets connus ne le sont que par certains attributs dont les valeurs sont elles-mêmes incomplètes : c'est une incomplétude extensionnelle.

Il faut quand même supposer que chaque description du monde incomplète représente un état du monde complet, mais incomplètement connu. On peut considérer une suite croissante (au sens ensembliste) de descriptions consistantes du monde $\mathcal{W}_i, 1 \leq i \leq n$: le fait que les descriptions soient consistantes et la suite croissante permet de supposer qu'elles décrivent toutes le même monde. Il est probable que pour toute théorie à peu près bien formée, les modèles de $\mathcal{W}_i$ sont des modèles de $\mathcal{W}_j$, $\forall j < i$. C'est en fait une définition de monotonie.

Le problème est que le SC est utilisé, à un moment donné, pour représenter, non pas n'importe quel modèle, mais un certain *état du monde* (du domaine modélisé) qui n'est connu qu'incomplètement au travers de la description du monde $\mathcal{W}$. Si l'on assimile cet état du monde à un (sous-ensemble de) [3] modèle(s) de $\mathcal{W}$, toute suite croissante de descriptions $\{\mathcal{W}_i\}_{i \geq 0}$, avec $\mathcal{W} = \mathcal{W}_0$, n'est pas réalisable : seules celles qui conservent ceux que l'on appellera les *modèles du problème* sont possibles.

La difficulté réside dans le fait que ces modèles du problème ne sont pas connus, pas plus du système que de l'utilisateur : c'est d'ailleurs le problème à résoudre. Le raisonnement valide que le système peut mettre en œuvre ne suffit bien sûr pas pour les atteindre. Or, que peut-il faire de plus ? Si l'on exclut un raisonnement hypothétique ou non monotone, rien, si ce n'est déléguer au monde extérieur (l'utilisateur) le soin de compléter ses connaissances pour de nouvelles inférences. Dans ce cadre, l'un des rôles du SC pourrait être de présenter à l'utilisateur des hypothèses plus plausibles que d'autres.

3. Le nombre de modèles du problème dépend du genre de problème traité : pour un diagnostic, il n'y a vraisemblablement qu'un modèle, à moins de considérer des hypothèses alternatives. En revanche, pour un problème de conception ou de satisfaction de contraintes (CSP), il peut y avoir plusieurs solutions alternatives. Enfin, s'il y a plusieurs modèles, on peut chercher à les calculer tous (diagnostic) ou un seul (conception).

3 La classification d'instances

Jusque là, le seul raisonnement réellement supporté par le SC est l'héritage, c'est-à-dire déduire de l'appartenance d'un objet à une classe les valeurs admissibles de ses attributs. Mais l'un des objectifs des SC est de reconnaître un objet comme instance d'une classe, même s'il est un peu artificiel d'isoler ce problème particulier du problème plus général de la déduction dans un SC. On suppose donc donné un *objet à classer*.

De façon générale, la classification d'instances se heurte aux difficultés soulevées par plusieurs catégories d'incomplétude :

- les objets, leurs attributs et les valeurs de ces attributs peuvent être partiellement inconnus, donc incomplets (incomplétude extensionnelle) ;
- le processus de classification se fait syntaxiquement, par des comparaisons pas forcément complètes des intensions (incomplétude sémantique) ;
- le langage du SC ne permet pas d'exprimer exactement le domaine modélisé (incomplétude syntaxique) ;
- les classes peuvent être ou ne pas être des CNS d'appartenance pour les objets (incomplétude intensionnelle).

3.1 Classification, version extensionnelle

Une approche sémantique de la classification conduit à définir trois catégories de classes relativement à l'objet à classer o. Une classe β peut être :

valide : c'est une classe de o dans tous les modèles de $\mathcal{W}$;

satisfaisable : c'est une classe de o dans au moins un modèle de $\mathcal{W}$;

insatisfaisable : ce n'est une classe de o dans aucun modèle de $\mathcal{W}$.

Si la description du monde est consistante, les classes valides sont satisfaisables. Dans les LT, la classification d'instance consiste à calculer tout ou partie des classes valides, suivant la complétude de la classification, sans s'intéresser aux classes satisfaisables ou insatisfaisables.

Mais comme seuls les modèles du problème intéressent réellement un utilisateur, il faudrait définir les ensembles de classes correspondants, obtenus par une quantification sur le sous-ensemble de modèles : on obtiendrait des classes *solution valide*, *solution* ou *non solution*. Cette définition est un peu vaine puisqu'il n'y a aucun moyen d'appréhender formellement ces ensembles de classes, mais cela permet de les appréhender intuitivement.

3.2 Version intensionnelle = classification incertaine

Tout énoncé au niveau des modèles doit se traduire, *in fine*, au niveau de l'intension, ce qui conduit à déterminer plusieurs catégories de classes relativement à o. Précisons que ces catégories intensionnelles sont, en pratique, définies par un algorithme. Intuitivement et approximativement, une classe β peut être :

impossible, si o viole au moins une de ses contraintes :

$$\exists P \in \mathcal{P}_\beta \cap Attr_\mathcal{W}(o) : P_\mathcal{W}(o) \notin Dom_P(\beta) \tag{14}$$

possible, si o n'en viole aucune contrainte :

$$\forall P \in \mathcal{P}_\beta \cap Attr_\mathcal{W}(o) : P_\mathcal{W}(o) \in Dom_P(\beta) \tag{15}$$

probable, si o en vérifie toutes les contraintes :

$$\forall P \in \mathcal{P}_\beta : P_\mathcal{W}(o) \in Dom_P(\beta) \tag{16}$$

sûre, si c'est une classe valide que le système sait reconnaître comme telle, par exemple une classe *définie* et *probable*.

Dans ces définitions, les comparaisons entre la valeur et le domaine de l'attribut sont doublement approximatives : d'une part, la relation entre les valeurs de P et $type(\beta, P)$ peuvent elles-mêmes être qualifiées de sûre, probable, possible ou impossible ; d'autre part, la cardinalité peut n'être que provisoire et toutes les valeurs pas encore connues. De plus, la formulation utilisée laisse entendre qu'il s'agit d'une comparaison effective entre la valeur et le domaine. En réalité, il peut aussi s'agir d'une comparaison entre le domaine de l'attribut pour l'objet, tel qu'il résulte de ses classes connues, avec le domaine de l'attribut dans β : il peut en effet exister $\alpha \in Isa_\mathcal{W}(o)$ tel que $Dom(\alpha, P) \subseteq Dom(\beta, P)$ ou $Dom(\alpha, P) \cap Dom(\beta, P) = \emptyset$, ce qui permet de conclure malgré l'incomplétude extensionnelle. Enfin, des classes de o peuvent aussi être incompatibles avec β, ce qui assure l'impossibilité. Mais cette approximation nous suffira ici.

La détermination de ces ensembles est à la base de plusieurs systèmes de représentation des connaissances par objets (SHIRKA, TROPES, FROME), avec les correspondances exprimées par le tableau suivant :

SC	SHIRKA	TROPES90	TROPES95	FROME
sûre		sûre		sûre
probable	sûre		possible	
possible	possible	possible	inconnue	possible
impossible	impossible	impossible	impossible	impossible

Cette classification incertaine pourrait concerner tout aussi bien les LT. C'est une classification interactive : l'utilisateur reste chargé du rattachement effectif de l'objet à une classe.

Les qualificatifs de *sûr* et *possible* et leur interprétation nécessitent quelques précisions. En SHIRKA qui a semble-t-il introduit ces notions, leur signification était claire : une classe *sûre* est une classe à laquelle il est sûr que l'objet peut être rattaché, une classe *satisfaisable* donc [Rec88]. En revanche, une classe *possible* n'est qu'une classe qui n'est pas immédiatement reconnue comme *impossible*. Or [Mar93] (TROPES90 dans le tableau) interprète ensuite le qualificatif

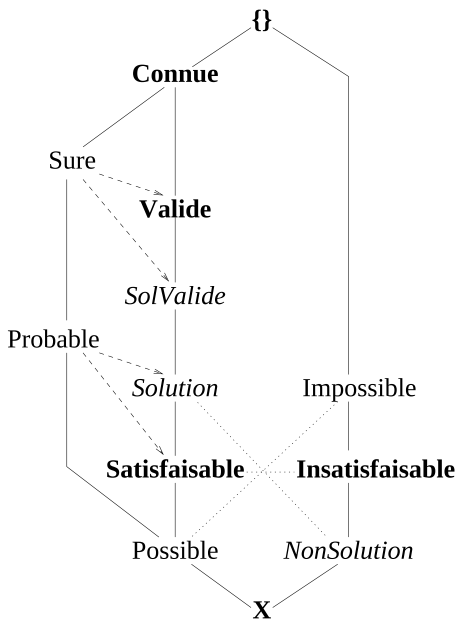

FIG. 1 - *Pour une description du monde consistante, l'inclusion des ensembles de classes résultant des classifications extensionnelle (en gras) et intensionnelle (en roman), avec les solutions, inaccessibles, en italique. Les flèches en pointillés désignent les alternatives de définition.*

de *sûr* par *valide* — “il est sûr que l'objet est instance de la classe”, comme s'il s'agissait de classification automatique sur des classes définies —, provoquant ainsi une certaine confusion. De son côté, dans FROME, [Dek94] utilise *sûr* pour des classes dont les conditions suffisantes sont vérifiées, ce qui en justifie l'usage. Enfin [ER95] clarifie la situation en adoptant pour SHIRKA et TROPES (TROPES95 dans le tableau) les qualificatifs de *possible* et *inconnu*, à la place de *sûr* et *possible*.

3.3 Correction et complétude

Pourtant, quelque soit le vocabulaire, SHIRKA et ses successeurs pèchent par une confusion des niveaux sémantique et syntaxique : dans [Rec88], une classe *sûre* est définie intensionnellement comme une classe *probable*, mais expliquée extensionnellement comme une classe *satisfaisable*, sans que la correction soit jamais questionnée.

Reste en effet à savoir dans quelle mesure l'interprétation intuitive de ces qualificatifs de *sûr* et *probable* est correcte. Une classe sûre est-elle bien valide, une classe probable satisfaisable ? Notons d'abord que la question de la correction ne présente pas vraiment de difficulté pour les classes possibles et impossibles : la complétude est en revanche difficile à atteindre. Soulignons aussi qu'il n'y a pas de définition complète, abstraite et générale, des classes possibles, probables, sûres et impossibles : ce ne sont que des cas d'espèces, fonction de l'expressivité des systèmes.

3.3.1 Les classes valides

Deux catégories d'inférence permettent de déduire qu'une classe primitive est *valide* :

- d'abord, l'inférence de type, élémentaire, qui permet de classer un objet lorsqu'il est valeur d'un attribut :

$$\{(\alpha\ o)\ (P\ o\ o')\} \models (type(\alpha, P))\ o') \tag{17}$$

- ensuite, les éventuelles propriétés structurelles spécifiques du SC : ainsi, si le SC est exhaustif (cf. paragraphe 1.3) et qu'il n'y a qu'un puits possible, c'est une classe *valide* [4]. Plus généralement, les super-classes communes à tous les puits possibles sont valides et si une classe sûre n'a qu'une seule sous-classe directe possible, cette sous-classe est sûre.

Les classes sûres sont constituées de ces sous-ensembles de classes valides, effectivement calculables. Le détail de ces classes sûres est relativement secondaire : ce qui compte est qu'il en existe. Il y a toujours moyen de faire un peu de classification automatique, même avec des classes primitives.

3.3.2 Les classes satisfaisables

Une classe *probable* est-elle bien *satisfaisable* ? Si o peut avoir une classe α probable et une classe β seulement possible et que ces deux classes soient incompatibles ($\alpha\ \nabla_a\ \beta$), le tout sans que la description du monde soit inconsistante, alors rien n'empêche que o soit la valeur d'un attribut d'un objet dont le type est β — $(P\ o'\ o), (\gamma\ o') \in \mathcal{W}$ et $type(\gamma, P) = \beta$ — ce qui rendrait β valide (par 17 et α insatisfaisable). Bien sûr, la validité de β peut être moins immédiate. On voit bien pourquoi une classe possible, voire probable, ne peut pas être considérée comme satisfaisable avec certitude : rien n'empêche qu'une inférence valide quelconque ne provoque l'incompatibilité de l'objet avec la classe.

Le problème s'aggrave si l'on se place dans un cadre de SC moins restreint, en autorisant des contraintes entre les attributs d'un objet : ce peut être des contraintes de type CSP [Gen95] ou des *sous-rôles* comme dans les LT [Neb90].

3.3.3 Complétude sémantique

Pour que ces notions aient des chances d'être correctes, il faudrait en fait les définir, non pas à partir de la description du monde $\mathcal{W}$, mais à partir de l'ensemble des conséquences valides de $\mathcal{W}$, $\{d \mid \mathcal{W} \models d\}$, pour obtenir une description du monde fermée par $\models$. Dans la description du monde résultante, il est probable que tout ce qui est possible est satisfaisable, et les classes probables peuvent alors être considérées, arbitrairement, comme des solutions.

4. Notons que soit le puits est satisfaisable, soit il ne l'est pas. Dans le second cas, la description du monde est inconsistante mais, dans les deux cas, le puits est valide.

Il faut donc assurer la complétude sémantique des déductions, au moins pour les classes valides[5] : si la correction s'avère délicate, la complétude n'était même pas revendiquée jusqu'ici, entraînant la coexistence de classes caractérisées au niveau sémantique (valide, satisfaisable, insatisfaisable) et au niveau syntaxique (sûre, probable, possible, impossible). Or, en un retournement inattendu, la correction de ces notions syntaxiques semble nécessiter finalement la complétude : on ne garderait donc plus que les classes valides, satisfaisables et insatisfaisables (identiques, respectivement aux classes sûres, possibles et impossibles), auxquelles s'ajouteraient les classes probables. Rien n'empêche bien sûr, pour un SC particulier, d'essayer de relever ce défi de la complétude, mais le cas général est désespéré.

Cette discussion montre en fait que la modalité la moins incertaine de la classification incertaine à la SHIRKA présente beaucoup d'incertitude. Que son qualificatif soit sûr, probable ou possible, il est probablement correct en SHIRKA où la seule façon de rendre une classe valide semble correspondre à la règle 17 : cette règle est assez élémentaire pour entraîner une propagation immédiate qui assure la fermeture par $\models$ à tout instant. En revanche, il est à craindre que des systèmes plus complexes comme TROPES (ou FROME, mais [Dek94] ne se sert pas de classes *probables*) ne soient mis en défaut sur ce point, surtout si l'on y rajoute des contraintes comme le fait [Gen95].

Finalement, le caractère priviligié des classes *probables* résulte uniquement de la maîtrise de l'incomplétude extensionnelle : vis-à-vis de la classe, l'objet à classer est parfaitement connu. Aucun retournement n'est à craindre d'un attribut dont le dévoilement viendrait contredire l'attachement de l'objet à une classe. Mais le retournement peut très bien venir d'ailleurs !

4 Comparaisons, conclusions et perspectives

[Euz93] a introduit la notion de *système classificatoire* dans une acception encore plus générale, puisqu'elle repose sur un *critère de catégorisation* abstrait, qui correspond à l'inclusion des intensions. Nous en proposons ici un cas particulier où l'intension est formée d'un ensemble d'attributs, ce qui offre un cadre plus précis pour comparer LT et RCO [CDE+95].

La classification incertaine ne doit pas être confondue avec la classification pratiquée par des SC qui gèrent des connaissances approximatives ou incertaines comme le font [GG96] et quelques systèmes *ad hoc* [DK87, Gra88]. Cependant, le précis étant un cas particulier de l'imprécis, et le certain un cas particulier de l'incertain, cette classification imprécise et incertaine doit se ramener, dans le cas particulier de données précises et certaines, à une sémantique aussi fondée que celle qui a été proposée ici.

Nous avons montré dans cet article comment la classification incertaine à la SHIRKA se comprend comme une approximation intensionnelle d'une clas-

5. Mais la complétude sur les valides pourrait nécessiter la complétude sur les satisfaisables. Considérons deux puits possibles, dont un insatisfaisable : l'autre est donc valide. Or, si l'on a la complétude sur les classes (in)satisfaisables, la correction des classes possibles et probables suit : il suffit d'inclure dans leur définition la satisfaisabilité.

sification fondamentalement extensionnelle. Si l'on veut profiter du caractère informatif et de la sécurité des classes probables, et non pas en rester à la seule alternative du possible et de l'impossible, il est nécessaire d'en prouver la correction, dont nous avons montré qu'elle risque de reposer sur la complétude de la classification extensionnelle. Il importe en particulier d'appliquer ces arguments aux systèmes existants (SHIRKA, TROPES, FROME) pour en vérifier la correction. Une perspective à plus long terme est d'appliquer cette classification incertaine aux LT, pour s'affranchir de la barrière des classes primitives.

Références

[BDS93] M. Buchheit, F.M. Donini, and A. Schaerf. Decidable reasoning in terminologic knowledge representation systems. *JAIR*, 1:109–138, 1993.

[CDE+95] B. Carré, R. Ducournau, J. Euzenat, A. Napoli, and F. Rechenmann. Classification et objets : programmation ou représentation. In *Actes des 5e journées nationales du PRC-GDR Intelligence Artificielle*, pages 213–237. Technea, 1995.

[Dek93] L. Dekker. La réification des filtres en langage FROME. In Habib and Oussalah [HO93].

[Dek94] L. Dekker. FROME *: représentation multiple et classification d'objets avec points de vue*. Thèse d'informatique, Université de Lille, 1994.

[DHH+95] R. Ducournau, M. Habib, M. Huchard, M-L. Mugnier, and A. Napoli. Le point sur l'héritage multiple. *Technique et Science Informatiques*, 14(3):309–345, 1995.

[DK87] J.-M. David and J.-P. Krivine. Utilisation de prototypes dans un système expert de diagnostic : le projet DIVA. In *Actes 7e Journées Internationales "Les systèmes experts et leurs applications"*, Avignon, mai 1987.

[Duc96] R. Ducournau. Des langages à objets aux logiques terminologiques : les systèmes classificatoires. Research Report 96-, L.I.R.M.M., Montpellier, 1996.

[ER95] J. Euzenat and F. Rechenmann. Shirka, 10 ans, c'est Tropes. In A. Napoli, editor, *Actes des journées Langages et Modèles à Objets, LMO'95*, Nancy, octobre 1995. INRIA.

[Euz93] J. Euzenat. Définition abstraite de la classification et son application aux taxonomies d'objets. In Habib and Oussalah [HO93].

[Euz94] J. Euzenat. Classification dans les représentations par objets : produits de systèmes classificatoires. In *Actes de la conférence Reconnaissance des Formes et Intelligence Artificielle*, 1994.

[FK85] R. Fikes and T. Kehler. The role of frame-based representation in reasoning. *Communications of the ACM*, 28(9):904–920, September 1985.

[Gen95] J. Gensel. *Contraintes et représentation des connaissances par objets. Application au modèle Tropes.* Thèse d'informatique, Université Grenoble I, 1995.

[GG96] M. Gonzalez-Gomez. *Représentation des connaissances approximatives et classification approchée d'instances dans un langage de schémas : le système CAIN.* Thèse d'informatique, Université d'Aix-Marseille III, 1996.

[Gir95] P. Girard. *Construction hypothétique d'objets complexes.* Thèse d'informatique, Université Grenoble I, 1995.

[Gra88] C. Granger. An Application of Possibility Theory to Object Recognition. *Fuzzy Sets and Systems*, 28(4):351–362, 1988.

[HO93] M. Habib and M. Oussalah, editors. *Actes des journées Représentation Par Objets, RPO'93*, La Grande Motte, juin 1993. EC2.

[Mar93] O. Mariño. *Raisonnement classificatoire dans une représentation à objets multi-points de vue.* Thèse d'informatique, Université Grenoble I, 1993.

[Neb90] B. Nebel. *Reasoning and revision in hybrid representation systems.* Lecture Notes in Computer Science. Springer-Verlag, Berlin, 1990.

[Rec88] F. Rechenmann. SHIRKA : système de gestion de bases de connaissances centrées-objet. Manuel de référence. Technical report, IMAG – INRIA Rhône-Alpes, Grenoble, 1988.