Abstraction on Object Based Knowledge Representations

Stéphane Le Peutrec and **Sophie Robin**
IRISA - Campus de Beaulieu
35042 Rennes
Tel: 99.84.74.91 - Fax: 99.84.71.71
E-mail : {lepeutre,robin}@irisa.fr

Abstract. In this paper, we present our study of abstraction in object based knowledge representations. We distinguish two kinds of abstractions: **forgetting abstractions** which abstract objects by forgetting information and **synthesis abstractions** which abstract objects by synthesizing information. Finally, we propose to use these abstractions to generate concise answers to knowledge bases queries -formalized by object based knowledge representations.

1 INTRODUCTION

In object oriented data models, a query is traditionally expressed as a class description which is classified in the class taxonomy [1, 2, 3, 5]. The answer is simply the set of all objects that are instances of the class equivalent to the query.

However, the answer may be too voluminous and provide too much details than the user desires. Some works [8, 14] have developed reasoning mechanisms based on knowledge abstraction in order to generate concise answers to database queries. The proposed idea is to transform initial information by forgetting certain details which are not relevant.

We propose to extend abstraction mechanisms to knowledge bases formalized by an object based knowledge representation with a class/ instance approach. The aim is to allow users to consult in a synthetic way a large knowledge base. Our system must be able to generate concise answers by forgetting irrelevant details, but also by synthesizing relevant information. In a first time, our approach consists in transforming an initial query in an abstract query and in a second time to abstract in the same way the answer of the initial query.

Queries are expressed as class descriptions so we first present the different ways to abstract objects and then we describe generations of abstract answers.

2 OBJECT ABSTRACTIONS

An object O is defined by a conjonction of properties. $O \equiv (a_1 : d_1, ..., a_n : d_n)$. a_i is a slot, it specifies the name of a property. d_i is the domain of a_i. An object is a class or an instance. Classes, organized into hierarchies, ordered by the specialization relation, modelise generic objects or concepts. Instances modelise specific objects, they are representative of certain classes.

In the problem solving area, abstraction is defined by F. Giunchiglia and T. Walsh as a total function transforming a problem in a simpler problem [9].

Let $abs(O)$ be the set of all possible abstract objects of O. For all $O' \in abs(O)$, there exists a total function f such that $f(O) = O'$. We distinguish two kinds of abstractions: **forgetting abstractions** which abstract objects by forgetting information and **synthesis abstractions** which abstract objects by synthesizing information.

2.1 Forgetting abstractions

If an object O_2 is an abstraction obtained by forgetting information of an object O_1 then O_2 description is included in O_1 description. There are two ways to forget information which could be applied to a slot $(a_i : d_i)$. We can forget the whole slot description $(a_i : d_i)$ or a part of the domain description (d_i), so a new domain d_i' is obtained. If d_i is an object, d_i' is a forgetting abstraction of d_i. But if the type of d_i is simple, like an integer interval for instance, then d_i is included in d_i'.

The relation between an abstract object and an initial object, induced by a forgetting abstraction, is a particular case of o-subsumption -subsumption in object based representations- [11]. An object $O_2 \equiv (a_1 : d_1, ..., a_n : d_n)$ o-subsumes an object $O_1 \equiv (b_1 : e_1, ..., b_m : e_m)$ if for all slot a_i, there exists one slot b_j such that $(a_i : d_i) \geq_{a_i} (b_j : e_j)$. $\geq_{a_i}$ is a own partial order relation of a_i. The o_subsumption relation is a partial order relation defined as a conjunction of the partial order relations $\geq_{a_i}$.

Let $\geq_{abs_{forg}}$ be the abstraction relation by forgetting information.

Definition 1 *$O_1 \equiv (b_1 : e_1, ..., b_m : e_m)$, $O_2 \equiv (a_1 : d_1, ..., a_n : d_n)$ are two objects.*

$$O_2 \geq_{abs_{forg}} O_1 \Leftrightarrow \quad \forall\ (a_i : d_i) \in O_2,\ \exists\ (b_j : e_j) \in O_1,\ (a_i : d_i) \geq_{slot_{forg}} (b_j : e_j)$$

$(a_i : d_i) \geq_{slot_{forg}} (b_j : e_j)$ means that a_i description is an abstraction of b_j description obtained by forgetting information.

ECAI 96. *12th European Conference on Artificial Intelligence*
Edited by W. Wahlster
Published in 1996 by John Wiley & Sons, Ltd.

Definition 2 *$(a_i : d_i)$ and $(b_j : e_j)$ are two slot descriptions.*

$$(a_i : d_i) \geq_{slot_{forg}} (b_j : e_j) \Leftrightarrow \begin{array}{c} a_i = b_j, \\ \textit{if } a_i \textit{ type is simple} \\ \textit{then } d_i \supseteq e_j \\ \textit{else } d_i \geq_{abs_{forg}} e_j \end{array}$$

The relations $\geq_{abs_{forg}}$ and $\geq_{slot_{forg}}$ are both partial order relations. They are reflexive, antisymetric and transitive.

Examples :
Let the class *Child* defined by
Child $\equiv$ (firstname : *String*, name : *String*, father : *Man*, mother : Woman).
If the class *Person* is a forgetting abstraction of the classes *Man* and *Woman*, the class defined by
(firstname : *String*, name : *String*, father : *Person*, mother : *Person*)
is a forgetting abstraction of the class *Child*.

Let us consider the two following instances :
Child1 $\equiv$ (firstname : *Peter*, name : *Smith*, father : *Man1*, mother : Woman1),
and
Man1 $\equiv$ (firstname : *John*, name : *Smith*, job : *Bookseller*).
The following instances are three possible abstractions of the instance *Child1* :
(firstname : *Peter*), (name : *Smith*), (father : (job : *Bookseller*)).

These definitions imply that all the surperclasses and all the components of an object are forgetting abstractions of it. These abstractions are similar in spirit to predicate abstractions that have been discussed by J.D. Tenenberg [12, 13].

The notion of point of view is traditionally used to make a partition of an object slot set [4, 7, 6, 10]. Each subset groups together slots which are known by a given expert. Any object which is obtained by applying a point of view to an object, is a forgetting abstraction of it.

2.2 Synthesis abstractions

If an object O_2 is an abstraction obtained by synthesizing information of an object O_1 then O_2 description is obtained by applying functions to O_1 description.

To be more precise, if an object O_2 is a synthesis abstraction of an object O_1 then for all slot a, its description is obtained by applying a function *synt* to a slot subset $\{b_k, ..., b_l\}$ of O_1. The function *synt* maps the semantics of $b_k, ..., b_l$ onto that of a. So a function *synt* is expressed by the term :

$$synt : (b_k : d_k) \times ... \times (b_l : d_l) \rightarrow (a : d)$$

If d type is simple, it is calculated from $d_k, ..., d_l$ domains. Otherwise d is an object and then it is a synthesis abstraction of the object formed by the conjunction of the domains $d_k, ..., d_l$.

Functions *synt* induce a partial order relation, written $\geq_{slot_{synt}}$, between a slot and a slot set. Let $\geq_{abs_{synt}}$ be the abstraction relation by synthesizing information.

Definition 3 *$O_1 \equiv (b_1 : e_1, ..., b_m : e_m)$, $O_2 \equiv (a_1 : d_1, ..., a_n : d_n)$ are two objects.*

$$O_2 \geq_{abs_{synt}} O_1 \Leftrightarrow \quad \forall (a_i : d_i) \in O_2,\ \exists (b_k : e_k), ..., (b_l : e_l) \in O_1, \quad (a_i : d_i) \geq_{slot_{synt}} ((b_k : e_k), ..., (b_l : e_l))$$

$(a_i : d_i) \geq_{slot_{synt}} ((b_k : e_k), ..., (b_l : e_l))$ means that a_i description is an abstraction of the conjunction of $b_k, ..., b_l$ descriptions by synthesizing information.

Definition 4 *$(a : d)$ is a slot description and $\{(b_1 : d_1), ..., (b_m : d_m)\}$ is a slot description set.*

$$\begin{array}{c} (a : d) \geq_{slot_{synt}} ((b_1 : d_1), ..., (b_m : d_m)) \Leftrightarrow \\ \exists synt : (b_1 : t_1) \times ... \times (b_m : t_m) \rightarrow (a, t), \\ d \subseteq t, \forall d_i\ d_i \subseteq t_i, \\ synt((b_1 : d_1), ..., (b_m : d_m)) = (a : d) \end{array}$$

The relations $\geq_{abs_{synt}}$ and $\geq_{slot_{synt}}$ are both partial order relations. They are reflexive, antisymetric and transitive.

Example :
The class
blood-preasure-measure $\equiv$ (1^{st}-number : *integer*, 2^{nd}-number : *integer*, date : *date*)
can be synthetized by the class
blood-preasure $\equiv$ (value : {*normal, high-blood-preasure, low-blood-preasure*})
using the total function
f : (1^{st}-number : *integer*) $\times$ (2^{nd}-number : *integer*) $\rightarrow$ (value : {*normal, high-blood-preasure, low-blood-preasure*})

In the object based representation, there isn't any relation between objects which may be interpreted as a synthetis abstraction relation. We propose to introduce two kinds of relations : **synthesis slot relations** and **synthesis class relations**. A synthesis slot relation links a slot set $\{b_k, ..., b_l\}$ to a slot a. It means that $(a_i : d_i) \geq_{slot_{synt}} ((b_k : e_k), ..., (b_l : e_l))$. A synthesis class relation links a class set $\{C_1, ..., C_n\}$ to a class C. It means that the class C is a synthesis abstraction of the class C' formed by the conjunction of $C_1, ..., C_n$, i.e $C \geq_{abs_{synt}} C'$.

3 ABSTRACT ANSWER GENERATIONS

A query answer is the set of objects which are instances of the class equivalent to the query. In order to generate abstract answers, we have to generate abstract instances.

In a first time, we propose to abstract a query Q in another query Q' by a total function f such that $f(Q) = Q'$. Q' may be a forgetting or a synthesis abstraction of Q. In a second time, the total function f is used to abstract the initial query answer.

The instanciation class Q_i of any instance i of the query Q answer is a subclass of Q. So Q is a forgetting abstraction of Q_i, i.e there exists a total function f_i such that $f_i(Q_i) = Q$. Then any instance i of the query Q answer can be abstracted by the total function $f \circ f_i$. The result of $f \circ f_i(i)$ is an instance of Q'.

An abstract answer must interest the user, i.e it must be composed of relevant information. There are many different abstractions of one object. To determine which abstractions the user would like, we need more information than the initial query description. For example, it will be possible to indicate which slots must absolutely appear in the abstract answer, or on the contrary which slots mustn't in any case appear in the abstract answer. It will also be possible to choose in the class taxonomy, some classes which would be interesting abstractions of the query, or to indicate different points of view which would give interesting restrictions on query slots.

4 CONCLUSION

To allow users to consult in a synthetic way a large knowledge base, our system must be able to generate concise answers by forgetting irrelevant details, but also by synthesizing relevant information. These reasoning mechanisms are typically based on knowledge abstractions. In the object oriented knowledge representation area, we distinguish two kinds of object abstractions: forgetting abstractions and synthesis abstractions.

To generate an abstract answer, we first abstract the query in a new query by a total function and in a second time this total function is used to abstract the initial query answer.

The kind-of, point of view and composition relations between classes may be interpreted as forgetting abstraction relations. These relations are used to provide query forgetting abstractions. In the object based representation, there isn't any relation between classes which may be interpreted as a synthetis abstraction relation. So we propose to introduce two new relations to modelise this kind of abstractions.

REFERENCES

[1] S. Abiteboul and A. Bonner, 'Objects and views', in *Proceedings of the ACM/SIGMOD international conference on the management of Data*, pp. 238–247, Denver, Colorado, (1991).

[2] H.W. Beck, S.K. Gala, and S.B. Navathe, 'Classification as a query processing technique in the candide data model', in *Proceedings of the 5th international conference on Data Engineering*, pp. 572–581, Los Angeles,CA, (1989).

[3] D. Beneventano and S. Bergamaschi, 'Using subsumption in semantic query optimization', in *Proceedings of the IJCAI Workshop on Object-Based Representation Systems*, pp. 19–31, Chambéry, (1993).

[4] D. G. Bobrow and M. Stefik, *The LOOPS Manual : A data and Object Oriented Programming System for Interlisp*, Knowledge-Based VLSI Design Group Memo KB-VLSI-81-13, Xerox PARC, Palo Alto, California, 1983.

[5] A. Borgida, R.J. Brachman, D.L. McGuinness, and L.A. Resnick, 'Classic: A structural data model for objects', in *Proceedings of the ACM/SIGMOD International Conference on the Management of Data, 18(2)*, pp. 58–67, Portland, Oregon, (1989).

[6] B. Carré and J.M. Geib, 'The point of view notion for multiple inheritance', in *Proceedings of OOPSLA-ECOOP'90*, pp. 312–321, Ottawa, Canada, (1990).

[7] H.E. Davis, *Views: Multiple perspectives and structured objetcs in a knowledge representation language*, Master's thesis, MIT, 1987.

[8] T. Ellman, 'Approximation and abstraction techniques for generating consice answers to database queries', Technical report, Departement of computer science, Rutgers university, (1994).

[9] F. Giunchiglia and T. Walsh, 'A theory of abstraction', *Artificial Intelligence*, **57**, 323–389, (1992).

[10] O. Mariño, F. Rechemmann, and P. Uvietta, 'Multiple perspectives and classification mechanism in object-oriented representation', in *ECAI '90*, pp. 425–430, (1990).

[11] A. Napoli and R. Ducournau, 'Subsumption in object-based representations', in *Proceedings ERCIM Workshop on theretical and pratical aspects of knowledge representation*, pp. 1–9, (1992).

[12] Josh D. Tenenberg, 'Preserving consistency across abstraction mappings', in *IJCAI87*, pp. 1011–1014, Los Altos, CA, (1987). Morgan Kaufmann.

[13] Josh D. Tenenberg, 'Abstracting first order theories', in *Change of Representation and Inductive Bias*, ed., Paul Benjamin, Kluwer, Boston, Mass, (1990).

[14] W. W. Cohen, 'Learning from textbook knowledge: A case study', in *In Proceedings of the AAAI'*, pp. 743–748, (1990).