Terminological logics for schema design and query processing in OODBs*

D. Beneventano°, S. Bergamaschi°, S. Lodi°, C. Sartori

Dipartimento di Elettronica, Informatica e Sistemistica
Università di Bologna - CIOC-CNR
°Facoltà di Ingegneria, Università di Modena

1 Introduction

The paper introduces ideas which make feasible and effective the application of Terminological Logic (TL) techniques for schema design and query optimization in Object Oriented Databases (OODBs).

Applying taxonomic reasoning and TL in database environment for traditional semantic data models led to a number of promising results for database schema design and other relevant topics, as query processing and data recognition. In particular, in [Bergamaschi and Sartori,1992] a general theoretical framework has been presented, which supports conceptual schema acquisition and organization by preserving *coherence* and *minimality* w.r.t. inheritance, exploiting the framework of terminological reasoning. Complex object data models, recently proposed in the area of OODBs, are more expressive than actually implemented TL languages in some aspects. For instance, most of the complex object data models introduce a distinction between objects with identity and values, which is not present in TL languages. Further, complex object models usually support additional type constructors, such as set and sequence. Most importantly, these models usually support the representation and management of cyclic classes. These problems have found a solution in [Bergamaschi and Nebel,1992; 1993], by the adoption of an extended TL, named ODL.

A real database specification always includes a set of rules, the so-called *integrity constraints*, which should guarantee data consistency. Constraints are expressed in various fashions, depending on the data model: e.g. subsets of first order logic, or inclusion dependencies and predicates on row values, or methods in OO environments. In particular OO methods are programs whose semantics cannot be inspected by an automatic reasoner. A first, necessary, improvement is to express at least a class of integrity constraints at schema level. Our proposal is to generalize the notion of a database schema including a declarative specification of a set of integrity constraints and to exploit this knowledge together with taxonomic reasoning for the different tasks of schema design and query optimzation. Let us examine separately the two aspects of schema design and query optimization.

*This research has been partially funded by the project M.U.R.S.T. 40%, "Metodi Formali e Strumenti per Basi di Dati Evolute".

2 Reasoning services in schema design

Provided that an adequate formalism to express integrity constraints is available, the following question arises: Is there any way to populate a database which satisfies the constraints supplied by a designer? Means of answering to this question should be embedded in automatic design tools, whose use is recommendable or often required in the difficult task of designing non–trivial database schemas.

Our proposal is to use the tableaux-calculus technique to guarantee schema consistency, therefore including state constraint consistency. Such a solution is actually a modification of existing algorithms for Description Logics [Schmidt-Schauss and Smolka,1991; Hollunder and Nutt,1990; Hollunder *et al.*,1990; Donini *et al.*,1991].

In order to substantially enhance OODBs with reasoning features, the next step should be the design of a front–end to the DB to validate insertions and updates, with respect to the extended schema description.

2.1 Examples

Let us consider the organizational structure of a company in order to explain the purpose of our constraint validation method. Assume the following: Employees have name and salary. Managers are employees and have a level composed of a qualification and a parameter. Repositories have a denomination, wich can be either a string or a structure composed by a repository name and an address; a repository stocks a set of at least one and at most five materials. Materials are described by a name and a risk. Departments have a denomination (string), and are managed by a manager. Warehouses have all the properties of departments and repositories.

The above description is expressed in our formalism, ODL extended, as follows:

$$\sigma(\texttt{Level}) = [\texttt{qualification}:\texttt{String}, \texttt{parameter}:\texttt{Int}]$$

$$
\begin{array}{rcl}
\sigma(\texttt{Employee}) & = & \triangle[\texttt{name}: \texttt{String}, \texttt{salary}: \texttt{Int}] \\
\sigma(\texttt{Manager}) & = & \texttt{Employee} \sqcap \triangle[\texttt{level}: \texttt{Level}] \\
\sigma(\texttt{Repository}) & = & \triangle[\ \texttt{denomination}: \texttt{String} \sqcup \\
 & & \qquad [\texttt{rname}: \texttt{String}, \\
 & & \qquad \texttt{address}: \texttt{String}], \\
 & & \quad \texttt{stock}: \{\texttt{Material}\}_{(1,5)}] \\
\sigma(\texttt{Department}) & = & \triangle[\texttt{denomination}: \texttt{String}, \\
 & & \texttt{managed-by}: \texttt{Manager}] \\
\sigma(\texttt{Warehouse}) & = & \texttt{Department} \sqcap \texttt{Repository} \\
\sigma(\texttt{Material}) & = & \triangle[\texttt{name}: \texttt{String}, \texttt{risk}: \texttt{Int}]
\end{array}
$$

Class and type descriptions use the tuple ([ ]) and set ({}) constructors, the latter with a cardinality interval. The $\triangle$ operator enforces a distinction between object classes, preceded by it, and value types. With respect to the formalism in [Bergamaschi and Nebel,1993], the general complement ($\neg$) and the union operator ($\sqcup$), considered in many works on complex object data models [Abiteboul and Kanellakis,1989] [Lecluse and Richard,1989], have been added.

As an example of integrity constraint, let us assert that an employee must earn less than his manager:

$$
\begin{array}{rl}
\sigma(\texttt{Technician}) = & \texttt{Employee} \sqcap \\
 & \triangle[\texttt{works-in}: \texttt{Department}] \sqcap \\
 & (\triangle\texttt{salary} < \triangle\texttt{works-in}. \\
 & \quad \triangle\texttt{managed-by}.\ \triangle\ \texttt{salary}).
\end{array}
$$

As a further example, if the class shipment is introduced, the following integrity constraint can be specified on it: for all shipments it must hold that if the risk of the material is greater than 3 then its urgency must be greater than 10, that is: "**for all** $x \in \texttt{Shipment}$ **if** x is of type $\texttt{Shipment} \sqcap (\triangle\texttt{item}.\ \triangle\ \texttt{risk} > 3)$ **then** x is of type $\texttt{Shipment} \sqcap (\triangle\texttt{urgency} > 10)$". The constraint can be embedded in the class description, obtaining the following type description for `Shipment`:

$$
\begin{array}{l}
\sigma(\texttt{Shipment}) = \\
\quad \triangle[\texttt{urgency}: \texttt{Int}, \texttt{item}: \texttt{Material}]\ \sqcap \\
\quad (\neg(\triangle\texttt{item}.\ \triangle\ \texttt{risk} > 3))\ \sqcup \\
\quad (\triangle\texttt{urgency} > 10))
\end{array}
$$

The coherence checking completion rules, devised by TL researchers, are a suitable starting point also to solve the corresponding problem in OODBs, as shown in [Beneventano *et al.*,1994].

3 Reasoning services in query optimization

The purpose of semantic query optimization is to use semantic knowledge (e.g. integrity constraints) for transforming a query into an *equivalent* one that may be answered more efficiently than the original version.

In database environment, semantic knowledge is usually expressed in terms of IC rules, that is *if then* rules on the attributes of a *database schema* (i.e., roughly a Tbox of a Terminological Knowledge Representation System (TKRS)). Informally, *semantic equivalence* means that the transformed query has the same answer as the original query on all databases satisfying the IC rules. The notion of semantic query optimization for relational databases was introduced in the early 80's by King [King,1981a; 1981b]; Hammer and Zdonik [Hammer and Zdonik,1980] independently developed very similar optimization methods. During the last decade, many efforts have been made to improve this technique and to generalize it to deductive databases [Shenoy and Ozsoyoglu,1989; Siegel *et al.*,1992; Chakravarthy *et al.*,1990]. More recently, some efforts have been made to perform semantic query optimization in OODBs [Chan,1992; Jeusfeld and Staudt,1993; Pang *et al.*,1991; Buchheit *et al.*,1994; Beneventano *et al.*,1993; Bergamaschi and Nebel,1993]. The main point is that OODBs provide a very rich type (class) system able to directly represent a subclass of integrity constraints in the database schema. By exploiting schema information as, for instance, inheritance relations between types (classes), it is possible to perform semantic query optimization.

In order to develop a theory of semantic query optimization, we propose a theoretical framework (in term of *subsumption*) which includes the main query transformation criteria proposed in the database literature and is based on inclusion statements between concepts, recently proposed in [Donini *et al.*,1993]. This new perspective perfectly fits the usual database viewpoint. In fact, actual database schemata are given in terms of *base classes* (i.e. primitive concepts); further knowledge is expressed as IC rules. In particular, structural class descriptions are expressed as rules where the antecedent is a name of the class and the consequent is the class description. More generally, rules allow the expression of integrity constraints with an antecedent and a consequent which are types of the formalism. Since query languages for OODBs are more expressive than our formalism we, following [Buchheit *et al.*,1994], ideally introduce a separation of a query into a *clean* part, that can be represented as a type in our formalism, and a *dirty* part that goes beyond the type system expressiveness. Semantic optimization will be performed only over the clean part of a given query. The clean part of a query, in the following referenced as query, corresponds to the so–called conjunctive queries or single operand queries [Kim,1989] in OODBs and is a *virtual class* (i.e. a defined concept).

The chosen strategy for optimization is the following. Prior to the evaluation of any query, we *compile*, once at all, the given schema (classes + IC rules), giving rise to an enriched schema obtained by adding (*all the new*) *isa relationships* which are logically implied by the original schema. The compilation process is based on the generation of the *semantic expansion* in *canonical form* (i.e. a form which permits to abstract from different syntactical representation of semantically equivalent types) of the schema types. Following the approach of [Shenoy and Ozsoyoglu,1989; Siegel *et al.*,1992] for semantic query expansion, the semantic expansion of a type, say EXP(S) permits to incorporate any possible restriction which is not present in the original type but is *logically implied* by the type and by the schema. EXP(S) is based on the iteration of this simple transformation: if a type implies the antecedent of an IC rule then the consequent of that rule can be added. Logical implications between these types (the type to be expanded and the antecedent of a rule) are evaluated by means of the *subsumption computation* [Brachman and Schmolze,1985; Bergamaschi and Sartori,1992; Bergamaschi and Nebel,1993].[1]

At run time, we add to the compiled schema the query Q and activate the process again for Q, obtaining EXP(Q), with possible new *isa* relationships is obtained. If new *isa* relationships are found, it is possible *to move the query down* in the schema hierarchy. The main points of our optimization strategy are:

1. The *most specialized query* among the equivalent queries EXP(Q) is computed. During the transformation, we compute also, and substitute in the query at each step, the *most specialized classes* satisfying the query.
2. A filtering activity (*constraint removal*) is performed by detecting the *eliminable* factors of a query, that is, the factors logically implied by the query.

3.1 Examples

Let us extend the schema of the previous section with the class dangerous-shipment, which has the same structure of shipment. The following integrity constraint can be specified on it: for all shipments it must hold that if the risk of the material is greater than 3 then its urgency must be greater than 10 and it must belong to the class dangerous-shipment. The constraint can be embedded in the class description, obtaining the following type description for `Shipment`:

$$\begin{aligned}\sigma(\texttt{Shipment}) &= \triangle[\texttt{urgency}:\texttt{Int}, \texttt{item}:\texttt{Material}] \\ &\sqcap(\neg(\triangle\texttt{item}.\ \triangle\,\texttt{risk} > 3)) \sqcup \\ &(\texttt{DShipment} \sqcap \triangle\texttt{urgency} > 10))\end{aligned}$$

Let us give two simple query optimization examples related to our schema.
Q: "Select all shipments involving a material with risk greater than 8"

$$Q = \texttt{Shipment} \sqcap (\triangle\texttt{item}.\ \triangle\,\texttt{risk} > 8)$$

From the rule on Shipment, we derive:

$$\begin{aligned}\mathrm{EXP}(Q) &= \texttt{DShipment} \sqcap \\ &(\triangle\texttt{item}.\ \triangle\,\texttt{risk} > 8) \sqcap \\ &(\triangle\texttt{urgency} > 10)\end{aligned}$$

The query is optimized by obtaining the *most specialized generalization* of the classes involved in the query itself. Furthermore, the factor ($\triangle\texttt{urgency} > 10$) can be added if some advantageous access structure is available for it.

Another rewriting rule proposed in [Shenoy and Ozsoyoglu,1989; Siegel *et al.*,1992] is the *constraint removal*, i.e., removal of implied factors. We formalize constraint removal by subsumption. As an example, consider the query:
Q: "Select all the shipments involving a material with risk greater than 8 and urgency grater than 5":

$$Q = \underbrace{\texttt{Shipment} \sqcap (\triangle\texttt{item}.\ \triangle\,\texttt{risk} > 8)}_{S} \sqcap \underbrace{(\triangle\texttt{urgency} > 5)}_{S'}$$

In the schema with rules S is subsumed by S', as $explo(S)$ is subsumed by S' in the schema without rules. Thus, S' can be eliminated from Q.

[1] The subsumption is similar to the *refinement* or *subtyping* adopted in OODBs [Cardelli,1984; Lecluse and Richard,1989].

References

[Abiteboul and Kanellakis, 1989] S. Abiteboul and P. Kanellakis. Object identity as a query language primitive. In *SIGMOD*, pages 159–173. ACM Press, 1989.

[Beneventano *et al.*, 1993] D. Beneventano, S. Bergamaschi, S. Lodi, and C. Sartori. Using subsumption in semantic query optimization. In A. Napoli, editor, *IJCAI Workshop on Object-Based Representation Systems - Chambery, France*, August 1993.

[Beneventano *et al.*, 1994] D. Beneventano, S. Bergamaschi, S. Lodi, and C. Sartori. Reasoning with constraints in database models. In S. Bergamaschi, C. Sartori, and P. Tiberio, editors, *Convegno su Sistemi Evoluti per Basi di Dati*, June 1994.

[Bergamaschi and Nebel, 1992] S. Bergamaschi and B. Nebel. Theoretical foundations of complex object data models. Technical Report 5/91, CNR, Progetto Finalizzato Sistemi Informatici e Calcolo Parallelo, Roma, January 1992.

[Bergamaschi and Nebel, 1993] S. Bergamaschi and B. Nebel. Acquisition and validation of complex object database schemata supporting multiple inheritance. *Applied Intelligence: The International Journal of Artificial Intelligence, Neural Networks and Complex Problem Solving Technologies*, 1993. to appear.

[Bergamaschi and Sartori, 1992] S. Bergamaschi and C. Sartori. On taxonomic reasoning in conceptual design. *ACM Transactions on Database Systems*, 17(3):385–422, September 1992.

[Brachman and Schmolze, 1985] R.J. Brachman and J.G. Schmolze. An overview of the KL-ONE knowledge representation system. *Cognitive Science*, 9(2):171–216, 1985.

[Buchheit *et al.*, 1994] M. Buchheit, M. A. Jeusfeld, W. Nutt, and M. Staudt. Subsumption between queries to object-oriented database. In *EDBT*, pages 348–353, 1994.

[Cardelli, 1984] L. Cardelli. A semantics of multiple inheritance. In *Semantics of Data Types - Lecture Notes in Computer Science N. 173*, pages 51–67. Springer-Verlag, 1984.

[Chakravarthy *et al.*, 1990] U. S. Chakravarthy, J. Grant, and J. Minker. Logic-based approach to semantic query optimization. *ACM Transactions on Database Systems*, 15(2):162–207, June 1990.

[Chan, 1992] Edward P.F. Chan. Containment and minimization of positive conjunctive queries in oodb's. In *Principles of Database Systems*, pages 202–11. ACM, 1992.

[Donini *et al.*, 1991] F.M. Donini, M. Lenzerini, D. Nardi, and W. Nutt. The complexity of concept languages. In J. Allen, R. Fikes, and E. Sandewall, editors, *KR '91 - 2nd Int. Conf on Principles of Knowledge Representation and Reasoning*, pages 151–162, Cambridge - MA, April 1991. Morgan Kaufmann Publishers, Inc.

[Donini *et al.*, 1993] F.M. Donini, A. Schaerf, and M. Buchheit. Decidable reasoning in terminological knowledge representation systems. In *13th International Joint Conference on Artificial Inteligence*, Chambery - France, September 1993.

[Hammer and Zdonik, 1980] M.M. Hammer and S. B. Zdonik. Knowledge based query processing. In *6th Int. Conf. on Very Large Databases*, pages 137–147, 1980.

[Hollunder and Nutt, 1990] Bernhard Hollunder and Werner Nutt. Subsumption algorithms for concept languages. Technical report, Research Report RR-90-4, Deutsche Forschungszentrum fuer Kuenstliche Intelligenz GmbH, Kaiserslautern, Germany, April 1990.

[Hollunder *et al.*, 1990] Bernhard Hollunder, Werner Nutt, and M. Schmidt-Schauss. Subsumption algorithms for concept languages. In *Proc. ECAI-90, Stockholm, Sweden*, 1990.

[Jeusfeld and Staudt, 1993] M. Jeusfeld and M. Staudt. Query optimization in deductive object bases. In Freytag, Maier, and Vossen, editors, *Query Processing for Advanced Database System*. Morgan Kaufmann Publishers, Inc., June 1993.

[Kim, 1989] W. Kim. A model of queries for object-oriented database systems. In *Int. Conf. on Very Large Databases*, Amsterdam, Holland, August 1989.

[King, 1981a] J. J. King. *Query optimization by semantic reasoning.* PhD thesis, Dept. of Computer Science, Stanford University, Palo Alto, 1981.

[King, 1981b] J. J. King. Quist: a system for semantic query optimization in relational databases. In *7th Int. Conf. on Very Large Databases*, pages 510–517, 1981.

[Lecluse and Richard, 1989] C. Lecluse and P. Richard. Modelling complex structures in object-oriented databases. In *Symp. on Principles of Database Systems*, pages 362–369, Philadelphia, PA, 1989.

[Pang *et al.*, 1991] H. H. Pang, H. Lu, and B.C. Ooi. An efficient semantic query optimization algorithm. In *Int. Conf. on Data Engineering*, pages 326–335, 1991.

[Schmidt-Schauss and Smolka, 1991] M. Schmidt-Schauss and G. Smolka. Attributive concept descriptions with unions and complements. *Artificial Intelligence*, 48(1), 1991.

[Shenoy and Ozsoyoglu, 1989] S. Shenoy and M. Ozsoyoglu. Design and implementation of a semantic query optimizer. *IEEE Trans. Knowl. and Data Engineering*, 1(3):344–361, September 1989.

[Siegel *et al.*, 1992] M. Siegel, E. Sciore, and S. Salveter. A method for automatic rule derivation to support semantic query optimization. *ACM Transactions on Database Systems*, 17(4):563–600, December 1992.