Discovering Description Logic Assertions from Database Schemes

A. Bonifati, L. Palopoli, D. Saccà, D. Ursino
DEIS, Univ. della Calabria, 87036 Rende, Italy
email:{bonifati,palopoli,sacca,ursino}@si.deis.unical.it

1 Introduction

The problem of understanding the intensional semantics of databases is an essential task for dealing with data management and utilization in the most appropriate way. Recent papers point out the need of a more generalized exploitation of intensional knowledge in, e.g., accessing integrated and cooperative data resource systems [13, 12, 4, 9, 1] query optimization and view maintenance [8, 5], structuring of data warehouses and data constraints [7]. To describe and manipulate intensional knowledge about data, [4] proposes a logic formalism largely based on description logics to express interscheme properties in cooperative database systems, whereas the Information Mainfold [12] uses an extension of description logics with Datalog-like rules for the integration purposes. A difficulty that is encountered while reasoning about intensional semantics of pre-existing databases is that many useful properties are hidden within database schemes and, as such, cannot be immediately utilized. The aim of this paper is to show how such useful properties can be extracted from database schemes applying a process which can be looked at as a special knowledge discovery process.

In order to represent and manipulate intensional knowledge, we utilize a logic language, called PDL, which is a simple extension of the DL used in [4] along two directions: first, besides expressing structural properties of classes of objects, most notably, class inclusions, our formalism allows also to express synonymies and homonymies between class names (we will call such properties nominal properties); second, each logic assertion is associated with a factor, i.e., a real number between 0 and 1, used to measure the strength of that assertion, in a way that will be explained shortly.

The adoption of such description logic like formalism was motivated by two reasons. First, Description Logics is well-suited for representing complex semantic properties of represented domains, and indeed we focus on complex semantic properties of database schemes. Second, DL features a precise formal inference system, which we took advantage of as the basis for constructing our derivation rule system.

The approach we are presenting here consists of two main phases. (Phase 1), the preprocessing phase, which consists, in turn, of two steps: Step 1, which includes an algorithm for extraction of synonymies and an algorithm for extraction of homonymies between database scheme objects. Step 2, including an algorithm for the extraction of the so-called exact properties (properties whose derivation is always valid) and conditional ones (properties whose validity depends on some coefficients). The pre-processing phase takes in input an initial set of synonymy or inclusion properties. These, and the corresponding strength factors, are supplied by database experts. During this phase a certain number of algorithms are applied to derive some properties (with the corresponding strength factors) which will constitute Synonymy, Homonymy and Inclusion Dictionaries. For space limitations, we are not going to illustrate Phase 1, which is however described into details in the full paper [3]. (Phase 2), which is devoted to discovering more complex properties and consists in (1) focusing on relevant database objects and (2) discovery of new properties about relevant objects. The focusing step is important since, in general, there exists a virtually infinite number of properties which could be extracted, but, clearly, we will be wishing to derive only the most relevant ones.

2 Description logics for intensional knowledge discovery

During the last decades there has been an enormous growth in the amount of data stored in database systems. For many organizations stored information nowadays represents an important patrimony which should be efficiently utilized and managed. Recently, to indicate the systematic organization of such data collections within structured information depository, the term Data Warehousing was coined. Because of their overwhelming size, information stored in data

warehouses cannot be properly queried using traditional techniques, as very often they are not able to produce data reporting of reasonably small size. Therefore, the necessity arises to employ special techniques capable to assist, in an intelligent and semi-automatic way, operators in retrieving useful information from large databases. These techniques should be able to yield new knowledge (called data patterns) by complex and structured inspection of available data, which can not be produced via simple querying. The cultural area focusing on devising such data extraction techniques is called Knowledge Discovery in Databases (KDD) and is attracting a growing interest in the database community [6]. The computational core of KDD processes consists in data mining algorithms, which perform the actual extraction of interesting pattern from data.

In this paper, we concentrate on such core phase and present some techniques by which interesting patterns can be extracted from database schemes rather than from extensional data. We have coined the term *Intentional Data Mining* to refer to this process of extracting new information from database schemes. The associated Knowledge Discovery process is similarly called *Intentional Knowledge Discovery* (IKDD).

To reason about database intensional properties, we use a variant of description logics, which we call PDL, which extends the logics presented in [4]. We implicitly assume that our input database schemes are represented in PDL. This is not an actual limitation, since translations exist from all data models [4]. However, for the sake of presentation, we shall often refer to Entity Relationship database schemes.

PDL differs from DL in the following points: (1) each PDL assertion is associated to an inclusion coefficient, which is a real number between 0 and 1; an assertion $L_1 \leq L_2$ with inclusion coefficient $W_{L_1L_2}$ will be represented as a triplet $\langle L_1, L_2, W_{L_1L_2}\rangle_{Inc}$, where the suffix Inc tells us the triplet is relative to an inclusion property; the triplet $\langle L_1, L_2, W_{L_1L_2}\rangle_{Inc}$ indicates that, of 100 elements of L_2, $W_{L_1L_2} \times 100$ are also elements of L_1; (2) besides inclusion properties, PDL allows to represent also what we call nominal properties, which express synonymy and homonymy characterizations of class names. Nominal properties are associated with synonymy/homonymy coefficients. $\langle L_1, L_2, V_{L_1L_2}\rangle_{Syn}$, tells that a synonymy is believed to exist between the name L_1 and the name L_2 with a confidence equal to $V_{L_1L_2} \times 100$ percent [1]. Homonymies are represented similarly, their plausibility coefficient being represented by the letter Z. Note that both kinds of nominal properties make sense only between classes belonging to different input database schemes. Note that, in PDL, relationships can have more than two roles. Thus, as in [4], we shall have entity expressions of the form $\forall R[U].T_1 : C_1, ..., T_n : C_n$ and $\exists R[U].T_1 : C_1, ..., T_n : C_n$ whose formal semantics is the following: given an interpretation I, $(\forall R[U].T_1 : C_1, ..., T_n : C_n)^I = \{a \mid \forall r \in R^I.(r[U] = a) \Rightarrow (r[T_1] \in C_1^I \wedge ... \wedge r[T_n] \in C_n^I)\}$ and $(\exists R[U].T_1 : C_1, ..., T_n : C_n)^I = \{a \mid \exists r \in R^I.(r[U] = a) \wedge (r[T_1] \in C_1^I \wedge ... \wedge r[T_n] \in C_n^I)\}$. Informally speaking, $(\forall R[U].T_1 : C_1, ..., T_n : C_n)^I$ (resp. $(\exists R[U].T_1 : C_1, ..., T_n : C_n)^I$) denotes the values a such that for each (resp., there exists a) tuple t in R having a as the value of its role U, the values of t on the role T_i belongs to C_i, $1 \leq i \leq n$.

[1] So the meaning of Synonymy and Homonymy coefficients is different from that of Inclusion one

3 Discovering Assertions from Database Schemes

The discovery phase extracts properties involving, in general, complex class expressions. The extraction of such complex properties can be divided into two steps. (Step 1) most interesting classes are singled out on the basis of a weight assigned by an algorithm which uses dictionaries constructed in the preprocessing phase. (Step 2) new properties involving interesting classes identified in Step 1 are derived.

In particular, Step 1 uses the information stored in the synonymy and inclusion dictionaries. The process consists in associating an interest weight to objects (entities and relationships), denoted by the function $I(.)$. To single out interesting objects, an interest threshold value relative to relationships is used. All objects with interest coefficient greater than such a threshold are considered interesting. For space limitation, we are not going to detail on the formulae we use to obtain interest coefficients and threshold (see [3] for details). So, we focus next on Step 2.

3.1 Step 2

As already stated, the general form of PDL assertions extracted by our method correspond to formulae $L_1 \leq L_2$, to which an inclusion factor f is associated to form a triplet $\langle L_1, L_2, f\rangle_{Inc}$. Here, both L_1 and L_2 are class expressions. The preprocessing phase extracts properties where both L_1 and L_2 are simple entity symbols. The method we are presenting next derives new and more complex properties. The method works by case analysis: each of the following subsections will be devoted to the illustration of one of such inference cases. We will provide the justification for only one of derived coefficients for lack of space. Analogous considerations are valid for all other coefficients.

Expressions containing $\sqcap$ and $\sqcup$. Assume that the assertions $\langle A, C, W_{AC}\rangle_{Inc}$ and $\langle B, C, W_{BC}\rangle_{Inc}$ were extracted during the preprocessing phase. To establish the inclusion coefficient of the assertions

$(A \sqcap B) \dot{\leq} C$ and $(A \sqcup B) \dot{\leq} C$, we reason as follows: first of all, note that there are two extreme situations to be considered:

- A and B are included either ways into one another, i.e. their simmetric difference has the smallest value, in which case the derived coefficients are $W_{(A \sqcap B,C)} = min(W_{AC}, W_{BC})$ and $W_{(A \sqcup B,C)} = max(W_{AC}, W_{BC})$.
- A and B have minimal intersection, i.e. their simmetric difference has the greatest value, in which case we derive the coefficients $W_{(A \sqcap B,C)} = max(0, W_{AC} + W_{BC} - 1)$ and $W_{(A \sqcup B,C)} = min(1, W_{AC} + W_{BC})$

To derive the general coefficient form, we chose to take the mean value between those extremal ones; thus, derived coefficients will be:

$$W_{(A \sqcap B,C)} = \frac{min(W_{AC}, W_{BC}) + max(0, W_{AC} + W_{BC} - 1)}{2}$$
$$W_{(A \sqcup B,C)} = \frac{max(W_{AC}, W_{BC}) + min(1, W_{AC} + W_{BC})}{2}$$

Expressions containing $\exists$ Assume that relationship R is interesting according to weights computed in Step 1. Assume, moreover, that R is connected to the entity C_1 through the role T_1 and to the entity E through the role U. Finally, assume that the assertion $\langle \exists R[U].T_1 : C_1, E, W_{C_1 \exists E} \rangle_{Inc}$ is stored in the inclusion dictionary. The algorithm considers several cases, which are illustrated next.

- **There exists a subset property between an entity C_2 and an entity C_1** Suppose the assertion $\langle C_2, C_1, W_{C_2 C_1} \rangle_{Inc}$ has been stored in the inclusion dictionary. Then, the algorithm infers $\langle \exists R[U].T_1 : C_2, E, W_{C_2 \exists E} \rangle_{Inc}$ where: $W_{C_2 \exists E} = 1 - min\left(1, \left[(1 - W_{C_1 \exists E}) \times \left(\frac{1}{W_{C_2 C_1}}\right)^{\alpha}\right]\right)$ and α is a suitable coefficient which is used to normalize $W_{C_2 C_1}$ in the formula. Such an expression for $W_{C_2 \exists E}$ is justified by the following observations:
  1. $W_{C_1 \exists E}$ indicates the set of instances of E which take part in the relation R and whose role T_1 is an instance of C_1;
  2. $(1 - W_{C_1 \exists E})$ indicates the set of instances of E which take part in the relation R and whose role T_1 is not an instance of C_1;
  3. $min\left(1, \left[(1 - W_{C_1 \exists E}) \times \left(\frac{1}{W_{C_2 C_1}}\right)^{\alpha}\right]\right)$ represents all the instances of E which take part in the relation R and whose role T_1 is not an instance of C_2. Their number is, obviously, greater than the instances in point 2 above because C_2 is a subset of C_1. Note that α is a suitable threshold which influences the variation of the coefficient value w.r.t. corresponding variation of $W_{C_2 C_1}$.
  4. $1 - min\left(1, \left[(1 - W_{C_1 \exists E}) \times \left(\frac{1}{W_{C_2 C_1}}\right)^{\alpha}\right]\right)$ represents all the instances of E which take part to relation R and whose role T_1 is an instance of C_2; this is really $W_{C_2 \exists E}$.
- **The entity F includes the entity E** Suppose that the assertion $\langle E, F, W_{EF} \rangle_{Inc}$ belongs to the inclusion dictionary. Then, the algorithm infers that $\langle \exists R[U].T_1 : C_1, F, W_{C_1 \exists F} \rangle_{Inc}$. The Inclusion Coefficient $W_{C_1 \exists F}$ is, in this case, simply equal to $W_{C_1 \exists E} \times W_{EF}$.
- **Entity C_1 is included in entity C_2** Suppose that the assertion $\langle C_1, C_2, W_{C_1 C_2} \rangle_{Inc}$ belongs to the inclusion dictionary. Then, the algorithm derives $\langle \exists R[U].T_1 : C_2, E, W_{C_2 \exists E} \rangle_{Inc}$ where: $W_{C_2 \exists E} = 1 - [(1 - W_{C_1 \exists E}) \times (W_{C_1, C_2})^{\alpha}]$.
- **The entity C_1 is substituted with a complex expression which is a subset of C_1** Suppose that the property $\langle expr(C_1), C_1, W_{expr} \rangle_{Inc}$ is stored in the inclusion dictionary, where $expr(C_1)$ is any structured expression denoting a subset of C_1 as, for example, $(C_2 \sqcap C_3)$ or $(C_2 \sqcup C_3)$ or $(\exists R_1.T_4 : C_4)$. Then, the assertion $\langle \exists R[U].T_1 : expr(C_1), E, W_{expr2} \rangle_{Inc}$ is inferred, where $W_{expr2} = 1 - min\left(1, \left[(1 - W_{C_1 \exists E})\left(\frac{1}{W_{expr}}\right)^{\alpha}\right]\right)$
- **R has more than one role**

  Assume we have already derived the assertions $\langle \exists R[U].T_1 : C_1, E, W_{C_1 \exists E} \rangle_{Inc}$ and $\langle \exists R[U].T_2 : C_2, E, W_{C_2 \exists E} \rangle_{Inc}$, where T_1 and T_2 are roles of the same relation R. Suppose we want to determine the inclusion coefficient associated to the assertion $\langle \exists R[U].T_1 : C_1, T_2 : C_2, E, W_{expr} \rangle_{Inc}$. Note that PDL expressions such as the one above, where two roles occur in the selection part, are equivalent to intersection expressions, as those discussed above and, therefore, we can use the same reasoning to infer that $W_{expr} = \frac{min(W_{C_1 \exists E}, W_{C_2 \exists E}) + max(0, W_{C_1 \exists E} + W_{C_2 \exists E} - 1)}{2}$
- **E is not an entity symbol** Suppose E is a complex expression (i.e., not a symbol), A and C are simple or complex expressions. Suppose that $\langle A, C, W_{AC} \rangle$ and $\langle E, C, W_{EC} \rangle$ have been derived. Now, if $W_{AC} < W_{EC}$, the coefficient W_{AE} in $\langle A, E, W_{AE} \rangle$ can be evaluated. Again, we consider two extreme situations:
  - the two subsets A and E have minimal intersection, in which case coefficient W_{AE} is $W_{AE} = \frac{max(0, W_{AC} + W_{EC} - 1)}{max(W_{AC}, W_{EC})}$;
  - the two subsets A and E are included into one another in either ways, in which case we infer $W_{AE} = \frac{min(W_{AC}, W_{EC})}{max(W_{AC}, W_{EC})}$

Again, we use the mean value $W_{AE} = \frac{max(0, W_{AC}+W_{EC}-1)+min(W_{AC}, W_{EC})}{2 \times max(W_{AC}, W_{EC})}$

Expressions containing $\forall$ or negation In this cases assertions are derived using rules in a similar way as we have done for $\exists$. Such rules are not reported here due to space limitations.

4 Applications

Extracting assertions has many applications. These include designing information integration layers on top of existing database systems, such as mediators [13, 12], and more in general the development of tools for supporting integrated access to cooperative information systems [4, 9], query optimization and view maintenance [8, 5], structuring and maintenance of warehouses and constraints [7].

As an example, consider the following schemes representing the Production (denoted SP) and Administration (denoted SO) Departments of an organization, respectively:

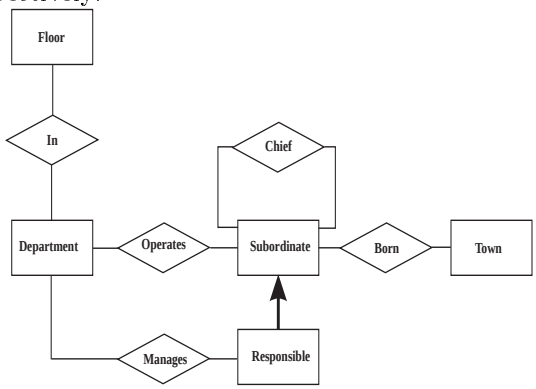

Figure 1 Scheme SP: the Production Department database

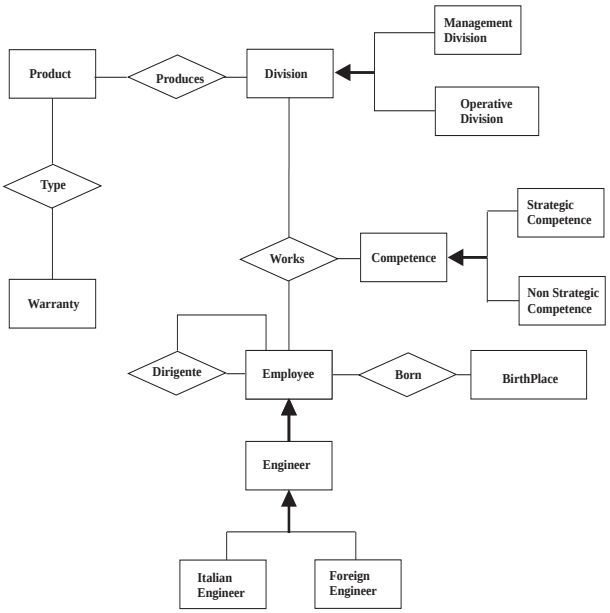

Figure 2 Scheme SO: the Administration Department database

Suppose to have in input the following information, previously supplied by the DBAs or by the preprocessing phase:

$\langle \text{Engineer}, \text{Employee}, 0.5 \rangle_{\text{Inc}}$
$\langle \text{Engineer}, \text{Subordinate}, 0.35 \rangle_{\text{Inc}}$
$\langle \text{Responsible}, \text{Employee}, 0.35 \rangle_{\text{Inc}}$
$\langle \text{BirthPlace}, \text{Town}, 0.95 \rangle_{\text{Inc}}$
$\langle \exists \text{Born[NL].IN} : \text{Employee}, \text{BirthPlace}, 0.98 \rangle_{\text{Inc}}$
$\langle \exists \text{Manager[ID].DI} : \text{Employee}, \text{Employee}, 0.3 \rangle_{\text{Inc}}$
$\langle \exists \text{Works[IL].DL} : \text{Division} \sqcap \text{ManagementDiv}, \text{Employee}, 0.7 \rangle_{\text{Inc}}$
$\langle \exists \text{Works[IL].LC} : \text{Competence} \sqcap \text{StrategicalComp}, \text{Employee}, 0.5 \rangle_{\text{Inc}}$
$\langle \exists \text{Born[NL].IN} : \text{Subordinate}, \text{Town}, 0.94 \rangle_{Inc}$

then, applying some of the rules described previously we can deduce:

$\langle \exists \text{Born[NL].IN} : \text{Engineer}, \text{BirthPlace}, 0.92 \rangle_{\text{Inc}}$
$\langle \exists \text{Born[NL].IN} : \text{Engineer}, \text{Town}, 0.874 \rangle_{\text{Inc}}$
$\langle \exists \text{Born[NL].IN} : \text{Subordinate}, \text{BirthPlace}, 0.99 \rangle_{\text{Inc}}$
$\langle \exists \text{Born[NL].IN} : (\text{Engineer} \sqcap \text{Responsible}), \text{BirthPlace}, 0.38 \rangle_{\text{Inc}}$
$\langle \exists \text{Born[NL].IN} : (\text{Engineer} \sqcup \text{Responsible}), \text{BirthPlace}, 0.96 \rangle_{\text{Inc}}$
$\langle \exists \text{Born[NL].IN} : (\exists \text{Manager[ID].DI} : \text{Employee}), \text{BirthPlace}, 0.78 \rangle_{\text{Inc}}$
$\langle \exists \text{Works[IL].DL} : (\text{Division} \sqcap \text{ManagDivi}), \text{LC} : (\text{Competence} \sqcap \text{StratCompe}),$
$\text{Employee}, 0.35 \rangle_{\text{Inc}}$
$\langle \exists \text{Born[NL].IN} : (\text{Engineer} \sqcap \text{Responsible}),$
$\exists \text{Born[NL]} : \text{IN} : \text{Subordinate}, 0.38 \rangle_{\text{Inc}}$

These and other properties can be automatically derived and given to an integration/abstraction algorithm as the inputs which returns an integrated global scheme as the output. This algorithm, which cannot be illustrated here for space limitation, is reported in

[10]. The integrated global scheme supplied by this algoritm in this case would be the following:

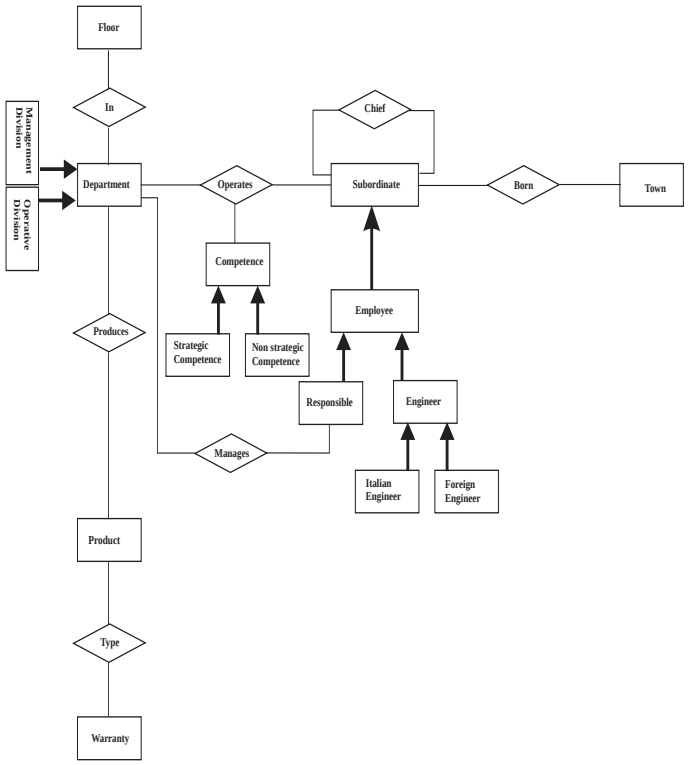

Figure 3. Global Scheme.

References

[1] S. Abiteboul, Querying Semi-Structured Data, *Proc. ICDT*, 1997, 1–18.

[2] C. Batini, M. Lenzerini, S. B. Navathe, A comparative analysis of methodology for database schema integration, *ACM CS* 18(4), 323-364, 1986.

[3] A. Bonifati, L. Palopoli, D. Saccá and D. Ursino, Discovering Description Logic Assertions from Database Schemes, Manuscript, 1997. Available from the authors.

[4] T. Catarci, M. Lenzerini, Representing and using interschema knowledge in cooperative information systems, *Journal of Intelligent and Cooperative Information Systems*, 2(4), 375-398, 1993.

[5] S. Chaudhuri, R. Krishnamurthy, S. Potamianos, K. Shim, Optimizing queries with materialized views, *ICDE '95.*

[6] U. Fayyad, G. Piatetsky-Shapiro, P. Smyth, The KDD process for extracting useful knowledge from volumes of data, *Comm. ACM*, 39(11), 1996, 27–34

[7] A. Gupta, I.S. Mumick, K.A. Ross, Adapting materialized views afetr redefinitions, *Proc. SIGMOD Conf. '95.*

[8] A.M. Keller, J. Basu, *A predicate caching scheme for client-server database architectures*, *Proc. PDIS*, 1994.

[9] R. Krishnamurthy, W. Litwin, W. Kent, Language features for interoperability of databases with schematic discrepancies, *Proc. ACM SIGMOD Conf.*, 1991.

[10] M. La Camera, L. Palopoli, S. Rotundo, D. Saccá and D. Ursino, Deriving properties from database schemes for automatic integration of large information systems, 1997. Manuscript. Available from the authors.

[11] W. Litwin, L. Mark, N. Roussopulos, Interoperability of multiple autonomous databases, *ACM CS*, 22, 1990.

[12] A. Levy, A. Rajaraman, J. Ordille, Querying heterogeneous information sources using source descriptions, *Proc. VLDB*, 1996.

[13] J.D. Ullman, Information integration using logical views, *Proc. ICDT*, 1997, 19–40.