TLP **10** (3): 243–250, 2010. © Cambridge University Press 2010
doi:10.1017/S1471068410000086 First published online 11 May 2010

Introduction to the TPLP special issue, logic programming in databases: From DATALOG *to semantic-web rules*

GIORGIO ORSI and LETIZIA TANCA
Dipartimento di Elettronica e Informazione,
Politecnico di Milano,
Piazza Leonardo da Vinci 32,
20133 Milano, Italy
(*e-mail:* {orsi,tanca}@elet.polimi.it)

1 Introduction

Much has happened in data and knowledge base research since the introduction of the relational model in Codd (1970) and its strong logical foundations influence its advances ever since. Logic has been a common ground where Database and Artificial Intelligence research competed and collaborated with each other for a long time (Abiteboul *et al.* 1995). The product of this joint effort has been a set of logic-based formalisms, such as the Relational Calculus (Codd 1970), Datalog (Ceri *et al.* 1990), Description Logics (Baader *et al.* 2007), etc., capturing not only the structure but also the semantics of data in an explicit way, thus enabling complex inference procedures.

These formalisms have been the theoretical foundations on top of which several technologies were born. We refer, in particular, to Relational Database Management Systems (RDBMS) and to their extensions for the storage of XML and RDF data, but also to the plethora of Datalog and ASP engines, which are often the backbone of many successful software products.

The rule-based paradigm has accompanied the relational model for almost all its life. Rules found in the database setting are mostly of the form $B \rightarrow H$, where the *body* B and the *head* H are conjunctions of function-free atoms, possibly with existentially quantified variables in the head. By means of such rules it is possible to express, for instance, a very general class of database integrity constraints, the rules of deductive databases, and ontological knowledge of various kinds.

Recently, with the advent of the Semantic Web and of ontologies, the issue has been addressed of how the study of ontologies and their semantics can help solving typical database problems, through a better understanding of knowledge representation. However, while ontologies provide solid grounds for representing the database at the schema level, the application logic cannot be easily expressed, hence the opportunity to combine ontologies with logical rules as it has been done with databases. Moreover, the presence of data from the Web constitutes an un-foreshadowed challenge in terms of scale and diversity (de Virgilio *et al.* 2010), and the Semantic Web technology encourages people to formally specify,

and perform reasoning on, large-scale RDF datasets on the Web. Here, rule-based approaches constitute a declarative and intuitive alternative for the specification of inference.

Another interesting application of the conjugation between the rule-based and the ontological formalisms is in Natural Language Processing. The well-established research on Logic Grammars (Abramson and Dahl 1989) has produced a number of developments where, on the one hand, logic programming concepts like unification are adopted to compute type inclusion relationships, and on the other hand, syntactico-semantic grammars are coupled with a knowledge base implemented in Description Logic to support effective parsing by means of a semantic double-check. Other applications are the inference of knowledge bases from their description in natural language, quick disambiguation and semantic role compatibility checkup, discovering implicit parts of sentences.

Since this special issue contains papers which conjugate databases and logic-based methods, with special emphasis on rules and semantics, some brief considerations on the differences between the basic modeling assumptions of the database area and the area of knowledge representation in general are in order. An interesting and detailed discussion on these topics can be also found in (Horrocks and Patel-Schneider 2007).

The most famous logic-programming-language for databases is Datalog (Ceri *et al.* 1990), along with its variations, while when referring to inference in the Semantic Web we mostly refer to Classical Logic, that is, mainly Description Logic and its variations.

One of the most important differences between the two worlds is the "open" nature of the Web, versus the "closed" nature of databases. In Classical Logic unstated information does not assume a truth value: that is, when an assertion is not found as a known fact, nothing can be said about its truth value. On the other hand, in the database realm the facts that have neither been asserted nor inferred are considered as false. The first attitude is known as the *Open World Assumption (OWA)*, while the second is the *Closed World Assumption (CWA)*, and each of them is perfectly coherent with the framework in which it is assumed.

The CWA (Reiter 1987) can be seen as an inference rule that, given a set of sentences S and an atom A, if A does not follow from S (i.e., cannot be inferred from S), derives $\neg A$. The CWA accounts for the way database people see the database as a mirror of the real world. Indeed, though we can reasonably allow for a database to be incomplete, that is, not to contain *all* the facts which are true in the world, most database applications can perfectly accommodate the much more restrictive hypothesis that *what is not recorded must be considered as false*. Indeed, in information systems—where databases are most used—it is reasonable to assume that all relevant information is actually available. The result of this assumption allows for a much simpler treatment of negation, in that not only what is explicitly asserted as false is so. An important consequence of the CWA is the so-called *minimal model semantics* of databases. Since, from a proof-theoretic point of view, the CWA implies that facts that cannot be proven must be considered as false, then *the* model of the database consists of all the facts that are true in *all* the worlds satisfying S, that is, a minimal model.

The CWA is actually compatible with other kinds of nonmonotonic reasoning, like Circumscription (McCarthy 1980), which, roughly speaking, formalizes the common-sense assumption that things are as expected unless otherwise specified. In the (wider) Logic Programming realm, the CWA is coherent with the Negation As Failure (NAF) inference rule, used to infer that p is not true from failure to derive p. Note that saying that p is not true can be different from the statement $\neg p$: this has to do with the completeness of the inference algorithm and thus with the formal logic system. Negation as failure has been an important feature of logic programming since its earliest days; in Prolog, it is usually implemented using Prolog's extra-logical constructs.

When we turn to the logics needed to formalize the Semantic Web, we see that in principle there is no need to assume that a certain (although ample) collection of information sources should contain *all* the information which is true; thus the paradigm followed by Classical Logic is more appropriate for web modeling since, when a fact F is not inferrable from S, it does not exclude interpretations of S which contain F. This allows for the possibility that, coming into play another information source which entails F, we should not fall into contradiction.

Reconciliation attempts between the two attitudes have been made, for instance, by taking an *epistemic* view of the database content: in (Lifschitz 1991; Donini *et al.* 1992), where the epistemic operators provide a clean way to express the difference between the description of the external world, and that of the database itself, that is, *what the database knows*. Thus, of a certain fact we can ask whether it is *known to the database*, mimicking the semantics of a database query. This is a fascinating research issue which, again, falls at the border among the artificial intelligence, logic programming, and database communities.

The "closed" view adopted in the database world also has two more aspects, namely the *unique name assumption*, which states that individuals with different names are different, and the *domain closure assumption*, which comes in different flavors but basically states that there are no other individuals than those in the database. Both assumptions do not favor the richness of expressiveness needed for the web, and thus are to be rejected in that context. By contrast, they prove to be very practical in the database domain, where unambiguous answers to "for all" queries and queries involving negation can be provided, based on the three assumptions above. Very interesting developments on the problem of uniqueness have come recently into play in the setting of the so-called Web of Data vision (Berners-Lee 2006), where identification becomes an issue since the idea is to extend the Web with semantic data by publishing various open datasets as RDF on the Web and by setting RDF links between data items (identified by means of URIs) from different data sources.

The above problems are part of the wider question of *incomplete information*: for instance, in the open perspective of the web we would like to be able to assert that an employee belongs to a department, without being obliged to name this department explicitly. One way to (partially) solve the problem in relational databases is the introduction of *null values* (Zaniolo 1982), whose treatment still produces a lot of research because as yet considered unsatisfactory; research somewhat reconciling

these perspective uses a technique called *chase*, which amounts to repairing violations of integrity constraints expressed in the form of rules (TGDs) (Maier *et al.* 1979; Arenas *et al.* 1999), starting from a database, until a fixpoint is reached, pretty much as in a forward-chaining logic programming deduction procedure. One of the main difficulties behind all these approaches is the fact that such a fixpoint may be infinite, and several works are proposed which reduce the constraint set to classes of constraints for which the chase terminates (Baget *et al.* 2010).

Another example of incomplete information is given by disjunction: we might want to state that John has gone out either with Jane or Sara, but asserting such disjunctive information is impossible in the relational database model, and requires appropriate extensions. Disjunctive information management is also a difficult task in relation to negation and the CWA. Indeed, suppose that a disjunctive sentence $P \vee Q$ holds in a database: then by the CWA we will be able to derive $\neg P$ and also $\neg Q$, which obviously leads to inconsistency. Disjunctive logic programming (J. Minker 2002) is a well-established research field which gives fundamental contributions in the database field. On the robust formal basis of Disjunctive Datalog (Leone *et al.* 1997), the DLV system has been built, whose highly expressive language includes Disjunctive Datalog with Stable Model Semantics (Gelfond and Lifschitz 1988) extended with aggregates, weak constraints, functions, various kinds of collections. DLV is fully declarative, and interoperable with databases as well as with ontologies.

Among other important differences of the two approaches we mention the question of infinity, which in its turn is strictly related to the meaning of database instances. In the traditional context of relational databases, a database (instance) is a finite set of finite relations, i.e., the totality of all tuples that can appear in a database is finite. Thus, since a database instance can be viewed as an interpretation of the first-order theory defined by the database schema (plus possibly a deductive program) and the integrity constraints, only finite models for the database schema are admissible. In the Classical paradigm, no assumption is made as to the interpretations that are acceptable for a theory, thus infinite models are not ruled out. Moreover, the idea that an instance is an interpretation leads to identification between information and interpretation (which is the basis of the so-called Herbrand model semantics of Datalog), whereas an ontology is seen as a theory which admits many possible interpretations.

Despite all the differences in their semantic foundations, these research communities persevere in coming together to expand the possibilities of the languages they adopt. In particular, while the Semantic Web community is willing to trade the expressivity of their languages for tractability and certainty of the answers, Database people strive for a greater expressivity, still keeping reasonable tractability.

From the Semantic-Web perspective the need for extending the knowledge representation formalisms derive from some shortcomings of Description Logics such as the limitation of the data model to represent hierarchical structures, that prevents some important constructs, and the impossibility to model interesting nonmonotonic extensions such as integrity constraints and closed world reasoning in general. For this reason, the Semantic-Web community and especially the DL community have

put huge efforts in the definition of formalisms combining Description Logics formulae and Datalog-style rules.

Since the combination of Description Logics with rules gets easily undecidable, early efforts concentrated more on the definition of decidable combinations with little regard of the tractability of the related inference procedures. The main cause of undecidability is the interaction of recursion with existential variables; we have seen that existential variables are a precious tool for expressing integrity constraints, thus the first attempt to achieve decidability has been the elimination of recursion from the formalism. CARIN was the first system to introduce this restriction but then, recursion was partially re-introduced by forcing role atoms to contain at least one explicitly named constant (*Role Safety* Levy and Rousset 1998). An extension of this principle lead to the so-called *DL-Safety* where the rules applicability was restricted to named constants only; the most recognized attempts in such a direction are those of $\mathscr{AL}$-LOG (Donini *et al.* 1998) and DL-*safe Rules* (Motik *et al.* 2005).

However, decidability is not always enough to make a modeling formalism useful, especially when dealing with large data sets. Moreover, the application areas that mostly benefit from such modeling languages and the related inference procedures are exactly those requiring the analysis of enormous datasets such as Computational Biology, Computational Chemistry and the Semantic Web. Recent approaches, focused on the achievement of tractable decision procedures especially for what concerns problems related to Query Answering, maintain the necessity of a *tree-like data model*, disallowing those combinations of DL constructs and rules that can destroy the hierarchical shape of the knowledge bases. A foundational attempt in this direction is that of *ELP Rules* (Krötzsch *et al.* 2008).

Still, an important aspect of a modeling language is the possibility to model constraints and exceptions. Despite a large body of literature from the database community on such topics, the DL community developed its own techniques usually built on Disjunctive Logic Programs. As an example, the work of (Lukasiewicz 2007) uses DL knowledge bases to filter inconsistent models. In the $\mathscr{DL}$-LOG (Rosati 2006) family, DL predicates are interpreted under Open-World semantics while the Datalog predicates are interpreted under Closed-World semantics. Early approaches, leveraging on these results, are now targeting more general combinations (Motik and Rosati 2010).

On the other hand, the database community is now recognizing the use of languages that can describe expressive constraints over the data. Recent works on Database Theory lead to a better understanding of problems such has Query Answering under Expressive Constraints and Incomplete Databases (Calvanese *et al.* 2008; Calí *et al.* 2009). The major consequence is the possibility to reuse the highly optimized database technologies to break into application areas which are currently the kingdom of Semantic-Web technologies.

While important advances have been made w.r.t. a better general understanding of the basic semantic features which sometimes connect, sometimes differentiate the paradigms we have analyzed, rule-driven computation and inference have been actually implemented inside real systems (Gottlob *et al.* 2004; Abiteboul *et al.* 2008),

where declarative programming provides easy documentation and maintenance, while a well-founded formal semantics guarantees reliability.

This special issue contains three rigorously reviewed articles addressing problems that span from Query Answering to Data Mining. All these contributions have their roots in the foundational formalisms of Data and Knowledge Bases such as Logic Programming, Description Logic and Hybrid Logics, representing a clear example of the effort that the Database and the Semantic-Web communities are producing to bridge the various schools of thinking in modern Data and Knowledge Management.

The paper *Inductive Logic Programming in Databases: from* DATALOG *to* $\mathcal{DL}+\text{LOG}^{\neg\vee}$ by Francesca A. Lisi investigates how the formal semantics of description logics ontologies can be of help for solving typical database problems, through a better understanding of Knowledge Representation aspects related to databases. In particular, the paper addresses the problems of *view and constraint definition* for a database whose schema is represented by means of an ontology. The paper shows how these common database problems can be reformulated in terms of Inductive Logic Programming (ILP) problems and how they might benefit from the expressive and deductive power of the KR formalism $\mathcal{DL}+\text{LOG}^{\neg\vee}$, an extension of the $\mathcal{DL}+\text{LOG}$ formalism introduced by Rosati in 1999. This paper establishes important connections between a formalism such as ILP that has been historically concerned with the induction of rules from samples, and modeling languages such as Description Logics and Datalog, proposing ILP as a powerful supporting tool for ontology modeling in the Semantic-Web.

Another contribution in the direction of rules and ontology combination is represented by the paper *The role of semantics in mining frequent patterns from knowledge bases in description logics with rules*, by Joanna Józefowska, Agnieszka Ławrynowicz and Tomasz Łukaszewski. In this contribution, a new method for mining frequent patterns in a language that combines ontologies and DL-safe rules is proposed. The authors show that such a setting is important for the practical application of data mining to the Semantic Web. Differently from Lisi's paper, here the authors focus on the relationship between the semantics of the representation formalism and the task of frequent pattern discovery. The main contribution is an algorithm exploiting the semantics of the combined knowledge base along with a proof-of-concept implementation for data mining tasks. The paper constitutes an advancement in the field of semantic-web mining since it shows that exploiting the semantics of the chosen representation formalism is key to the design and the application of (onto-)relational frequent pattern discovery methods.

A completely different perspective on the combination of Datalog with Semantic Web technologies is represented by the paper *Querying Incomplete Data over Extended ER Schemata*, by Andrea Calí and Davide Martinenghi. The paper applies recent results in Database Theory for querying conceptual schemas expressed by means of the Extended Entity-Relationship model (EER), enriched with cardinality constraints, disjointness assertions, and is-a relations among both entities and relationships. In this setting, the authors considered the problem of querying incomplete databases by making direct use of their conceptual representations. Based on previous decidability results, the paper provides a query answering algorithm

based on the rewriting of the initial query into a recursive Datalog query encoding the information about the ER schema. The paper constitutes an interesting contribution since ontologies are often used as a replacement for classical conceptual models like the Entity-Relationships because of the possibility to query them. This is extremely important in settings such as Model Comprehension and Schema Evolution, but the complexity of Description Logics ontologies has always been an obstacle to a wide employment of such formalisms in real situations. This paper tries to bridge this gap by extending the Entity-Relationship model, which for years has played a fundamental role in database design especially due to its simplicity. The renaissance of this model might be of use to support the emerging trends in data exchange and information integration when dealing with inconsistent and incomplete data.

Acknowledgments

We are grateful to Annalisa Bossi, who, as the previous editor-in-chief of TPLP, supported the guest editors during the first phase of the preparation of this special issue. We also like to thank all the reviewers, whose comments and criticisms have very much helped us in the production of this special issue.

Many thanks go to all the participants of the March 2010 DATALOG 2.0 Workshop[1], since the various interventions provided us with an up-to-date understanding of the current status of the research on Datalog and its extensions, along with many related implementations and applications. We are especially grateful to V. Dahl for her insights on Datalog applications for Natural Language Processing and to B. Motik for his presentation on Datalog-based formalisms in the Semantic-Web community.

References

ABITEBOUL, S., HULL, R. AND VIANU, V., Eds. 1995. *Foundations of Databases*. Addison-Wesley Longman Publishing Co., Inc.

ABITEBOUL, S., OMAR, O. AND MILO, T. 2008. The active xml project: An overview. *VLDB Journal 17,* 5, 1019–1040.

ABRAMSON, H. AND DAHL, V. 1989. *Logic Grammars*. Springer-Verlag.

ARENAS, M., BERTOSSI, L. AND CHOMICKI, J. 1999. Consistent query answers in inconsistent databases. In *Proc. of PODS*. 68–79.

BAADER, F., CALVANESE, D., MCGUINNESS, D. L., NARDI, D. AND PATEL-SCHNEIDER, P. F. 2007. *The Description Logic Handbook*. Cambridge University Press.

BAGET, J., LECLERE, M. AND MUGNIER, M. 2010. Walking the decidability line for rules with existential variables. In *Proc. of KR*. (in press).

BERNERS-LEE, T. 2006. Linked data. URL: http://www.w3.org/DesignIssues/LinkedData.html.

CALÍ, A., GOTTLOB, G. AND LUKASIEWICZ, T. 2009. A general datalog-based framework for tractable query answering over ontologies. In *Proc. of PODS*. 77–86.

CALVANESE, D., GIACOMO, G. D. AND LENZERINI, M. 2008. Conjunctive query containment and answering under description logic constraints. *ACM Transactions on Computational Logic 9,* 3.

[1] http://datalog20.org/

Ceri, S., Gottlob, G. and Tanca, L. 1990. *Logic Programming and Databases*. Springer.

Codd, E. F. 1970. A relational model of data for large shared data banks. *Communications of the ACM 13,* 6, 377–387.

de Virgilio, R., Giunchiglia, F. and Tanca, L. 2010. *Semantic Web Information ManagementA Model-Based Perspective*. Springer.

Donini, F. M., Lenzerini, M., Nardi, D. and Schaerf, A. 1998. $\mathscr{AL}$-log: Integrating datalog and description logics. *Journal of Intelligent Information Systems 10,* 3, 227–252.

Donini, F. M., Lenzerini, M., Nardi, D., Schaerf, A. and Nutt, W. 1992. Adding epistemic operators to concept languages. In *Proc. of KR*. 342–353.

Gelfond, M. and Lifschitz, V. 1988. The stable model semantics for logic programming. In *Proc. of ICLP/SLP*. 1070–1080.

Gottlob, G., Koch, C., Baumgartner, R., Herzog, M. and Flesca, S. 2004. The lixto data extraction project: back and forth between theory and practice. In *Proc. of PODS*. 1–12.

Horrocks, I. and Patel-Schneider, P. F. 2007. A comparison of two modelling paradigms in the semantic web. *Journal of Web Semantics 5,* 4, 240–250.

Krötzsch, M., Rudolph, S. and Hitzler, P. 2008. ELP: Tractable rules for OWL 2. In *Proc. of ISWC*. 649–664.

Leone, N., Rullo, P. and Scarcello, F. 1997. Disjunctive stable models: Unfounded sets, fixpoint semantics and computation. *Information and Computation 135,* 2, 69–112.

Levy, A. Y. and Rousset, M. C. 1998. Combining horn rules and description logics in carin. *Artificial Intelligence 104,* 1–2, 165–209.

Lifschitz, V. 1991. Nonmonotonic databases and epistemic queries. In *Proc. of IJCAI*. 381–386.

Lukasiewicz, T. 2007. A novel combination of answer set programming with description logics for the semantic web. In *Proc. of ESWC*. 384–398.

Maier, D., Mendelzon, A. O. and Sagiv, Y. 1979. Testing implications of data dependencies. *ACM Transactions on Database Systems 4,* 4, 455–469.

McCarthy, J. 1980. Circumscription – A form of non-monotonic reasoning. *Artificial Intelligence 13,* 1–2, 27–39.

Minker, J. and Seipel, D. 2002. Disjunctive logic programming: A survey and assessment. In *Computational Logic: Logic Programming and Beyond*, A. C. Kakas and F. Sadri, Eds. Lecture Notes in Computer Science, vol. 2408. Springer, 472–511.

Motik, B. and Rosati, R. 2010. Reconciling description logics and rules. *Journal of ACM* (in press).

Motik, B., Sattler, U. and Studer, R. 2005. Query answering for owl-dl with rules. *Journal of Web Semantics 3,* 1, 41–60.

Reiter, R. 1987. *Readings in Non-monotonic Reasoning*. Morgan Kaufmann, Chapter On Closed World Data Bases, 300–310.

Rosati, R. 2006. $\mathscr{DL}+log$: A tight integration of description logics and disjunctive datalog. In *Proc. of KR*. 68–78.

Zaniolo, C. 1982. Database relations with null values. In *Proc. of PODS*. 27–33.