Datalog+/-: A New Family of Languages for Ontology Querying

Andrea Calì[2,3] Georg Gottlob[1,3] Thomas Lukasiewicz[1]
Andreas Pieris[1]

[1]Computing Laboratory
University of Oxford, UK

[2]Department of Information Systems and Computing
Brunel University, UK

[3]Oxford-Man Institute of Quantitative Finance
University of Oxford, UK

andrea.cali@brunel.ac.uk
{georg.gottlob, thomas.lukasiewicz, andreas.pieris}@comlab.ox.ac.uk

We briefly report on Datalog$^{\pm}$, a family of recently introduced variants of Datalog. In Datalog$^{\pm}$ languages, Datalog is extended by allowing features such as existential quantifiers, the equality predicate, and the truth constant false to appear in rule heads. At the same time, the resulting language is syntactically restricted, so as to achieve decidability and in some relevant cases even tractability.

Datalog (see, e.g., [1]) has been used as a paradigmatic database programming and query language for over three decades. Rules in Datalog$^{\pm}$ are so-called *tuple-generating dependencies (TGDs)*, i.e., Datalog (Horn) rules with the possibility of having existentially quantified variables in the head. For example, the rule $person(X) \rightarrow \exists Y\, father(X, Y)$ (with the universal quantifiers omitted) expresses the fact that every person has a father. Existential quantification in Datalog$^{\pm}$ rules allows us to overcome the shortcomings of Datalog as an ontology language [15].

In the following, we will consider the ontological query answering problem as that of answering Boolean conjunctive queries (BCQs) against a database instance D (a set of ground facts) under an ontology Σ constituted by rules. The (decision) problem is to calculate whether q is entailed by D and Σ, written as $D \cup \Sigma \models q$, or, equivalently, whether q has positive answer against D and Σ. Notice that considering BCQs is without loss of generality regarding complexity.

Unfortunately, already for sets Σ of TGDs alone, most basic reasoning and query answering problems are undecidable. In particular, given an ontology constituted by a set Σ of TGDs, and a set of ground facts D, checking whether $D \cup \Sigma \models q$ is undecidable when q is a ground fact [3]. Worse than that, there exist a *fixed* set of TGDs and a BCQ, and only the database is given as input,

where undecidability still holds [4]. Languages in the Datalog$^{\pm}$ family have various restrictions on the form of the rule bodies, and this allows for decidability of query answering, and in some cases tractability in *data complexity*, i.e., the complexity when the only input is the instance D, while all the rest is considered fixed.

Guardedness [2] is a well-known restriction of first-order logic that ensures decidability. Based on this notion, *guarded TGDs (GTGDs)* have been introduced [6, 4]. In a guarded TGD, the rule body is required to have an atom that contains as arguments all body variables of the rule. This class of TGDs ensures polynomial-time data complexity of query answering. The more restricted class of *linear TGDs (LTGDs)* is the class of TGDs having a single body-atom. Such a class has even better computational properties than GTGDs; in fact, linear TGDs are *first-order rewritable*, which means that any set Σ of LTGDs, and any BCQ q, can be transformed into a first-order query q_Σ such that $D \models q_\Sigma$ iff $D \cup \Sigma \models q$, for every database D. This property, introduced in [13] in the context of description logics, is essential if D is a very large database, which is the usual case. It means that query answering can be deferred to a standard query language such as (basic, non-recursive) SQL. GTGDs can be enriched by *stratified negation*, a simple non-monotonic form of negation often used in the context of Datalog [6]. GTGDs are extended by *weakly-guarded sets of TGDs* [4], where the guardedness condition for rule bodies is somewhat relaxed.

Stickiness, a completely different paradigm for decidable and tractable query answering, was introduced in [9]. The class of *sticky sets of TGDs* is defined in [9] by a syntactic criterion that is easily testable. Sticky sets of TGDs are first-order rewritable. Extensions of sticky sets of TGDs are studied in [11].

Negative constraints are rules whose head is the truth constant *false*, denoted by $\bot$. It turns out, as shown in [6], that negative constraints can be added to TGDs without any increase of complexity. The reason is that checking whether a rule ρ: $body \rightarrow \bot$ is satisfied by a database D, given a Datalog$^{\pm}$ program Σ, is tantamount to showing that $D \cup \Sigma \not\models body$, i.e., to the evaluation of a BCQ.

Equality-generating dependencies (EGDs) have a body of the same form as TGDs, but having in their head an equality between two variables appearing in the body. Unfortunately, as is well-known in database theory, query answering becomes undecidable even when putting together some extremely week forms of TGDs and EGDs such as *inclusion dependencies* and *functional dependencies* [14]. It is interesting for real-world scenarios to study the interaction of TGDs with a very simple, nevertheless very useful class of EGDs, namely *key dependencies* (or simply *keys*). Semantic and syntactic conditions ensuring that keys are usable without destroying decidability and tractability have been recently studied in [4, 6, 9].

The Datalog$^{\pm}$ family has several applications—we mention just a few of them, referring the reader, for instance, to [5, 7]. LTGDs and keys (combined with negative constraints) have been used to model and query various extensions of Entity-Relationship schemas [8, 10]. In spite of their restricted syntax, linear TGDs prove to be very useful for ontology modeling and querying: LTGDs with key dependencies and negative constraints are capable of properly capturing the whole *DL-Lite family* [13, 16]. The *F-Logic Lite* ontology language, introduced

and studied in [12], can also be modeled in the Datalog$^{\pm}$ framework [4].

References

[1] Serge Abiteboul, Richard Hull, and Victor Vianu. *Foundations of Databases*. Addison-Wesley, 1995.

[2] Hajnal Andréka, Johan van Benthem, and István Németi. Modal languages and bounded fragments of predicate logic. *J. Philosophical Logic*, 27:217–274, 1998.

[3] Catriel Beeri and Moshe Y. Vardi. The implication problem for data dependencies. In *Proc. of ICALP*, pages 73–85, 1981.

[4] Andrea Calì, Georg Gottlob, and Michael Kifer. Taming the infinite chase: Query answering under expressive relational constraints. In *Proc. of KR*, pages 70–80, 2008.

[5] Andrea Calì, Georg Gottlob, Michael Kifer, Thomas Lukasiewicz, and Andreas Pieris. Ontological reasoning with F-Logic Lite and its extensions. In *Proc. of AAAI*, 2010.

[6] Andrea Calì, Georg Gottlob, and Thomas Lukasiewicz. A general Datalog-based framework for tractable query answering over ontologies. In *Proc. of PODS*, pages 77–86, 2009.

[7] Andrea Calì, Georg Gottlob, Thomas Lukasiewicz, Bruno Marnette, and Andreas Pieris. Datalog+/-: A family of logical knowledge representation and query languages for new applications. In *Proc. of LICS*, pages 228–242, 2010.

[8] Andrea Calì, Georg Gottlob, and Andreas Pieris. Tractable query answering over conceptual schemata. In *Proc. of ER*, pages 175–190, 2009.

[9] Andrea Calì, Georg Gottlob, and Andreas Pieris. Advanced processing for ontological queries. *Proc. of VLDB*, 3(1):554–565, 2010.

[10] Andrea Calì, Georg Gottlob, and Andreas Pieris. Query answering under expressive Entity-Relationship schemata. In *Proc. of ER*, pages 347–361, 2010.

[11] Andrea Calì, Georg Gottlob, and Andreas Pieris. Query answering under non-guarded rules in Datalog+/-. In *Proc. of RR*, pages 1–17, 2010.

[12] Andrea Calì and Michael Kifer. Containment of conjunctive object meta-queries. In *Proc. of VLDB*, pages 942–952, 2006.

[13] Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati. Tractable reasoning and efficient query answering in description logics: The *DL-Lite* family. *J. Autom. Reasoning*, 39(3):385–429, 2007.

[14] A. K. Chandra and M. Y. Vardi. The implication problem for functional and inclusion dependencies. *SIAM J. Comput.*, 14:671–677, 1985.

[15] Peter F. Patel-Schneider and Ian Horrocks. A comparison of two modelling paradigms in the semantic web. *J. Web Semantics*, 5(4):240–250, 2007.

[16] Antonella Poggi, Domenico Lembo, Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, and Riccardo Rosati. Linking data to ontologies. *J. Data Semantics*, 10:133–173, 2008.