Query Processing and Optimization for Logic programs with Certainty Constraints

Jinzan Lai and **Nematollaah Shiri**
Dept. of Computer Science and Software Engineering
Concordia University, Montreal, Quebec
Email: {la_jin, shiri@cse.concordia.ca}

Abstract

Numerous logic frameworks have been proposed for modeling uncertainty and reasoning with such data. While different in syntax, the approaches of these frameworks have been classified into "annotation based" (AB) and "implication based" (IB). In this paper, we present a unified framework which allows evaluating programs in either approach. It extends existing query processing techniques to handle certainty constraints and uses heuristics to further improve the performance. Our experiments indicate that the proposed techniques yield useful tools for uncertainty reasoning.

Introduction

Uncertainty reasoning has been identified as an important research in AI and database research (Abiteboul ; Lakshmanan and Shiri 2001a; Shapiro 1983). Many real life applications require to represent uncertain knowledge and reason with it efficiently. Standard logic programming and deductive databases have been a primary choice for modeling uncertainty, which resulted in numerous frameworks. On the basis on which uncertainty is associated with the facts and rules in a program, the approaches of these frameworks are classified into implication-based (IB) and annotation-based (AB) (Lakshmanan and Shiri 2001b).

While it has been noted that in general the AB approach is more expressive than the IB for allowing certainty annotations, (Shiri 2005) establishes equivalence of the two approaches when rules are generalized and extended with certainty constraints (CC), which are essentially built-in predicates of the form $wt(A)\,\theta\,wt(B)$ or $wt(A)\,\theta\,\sigma$, where A and B are atoms in the rule body, $wt(A)$ denotes A's certainty, σ is a certainty value, and θ is a comparison operator. This yields the so-called generalized IB (GIB) and GAB frameworks. Transformation algorithms have also been introduced between the two generalized frameworks.

In this work, we study query processing and optimization in the context of GAB and GIB, obtained by extending those proposed in (Shiri and Zheng 2004; 2008). We also use heuristics to improve efficiency of handling CC's during query processing. We developed a running prototype and performed experiments using different program types and/or data sizes. The system prototype allows evaluating IB and AB programs in a single environment. Our experimental results indicate the overhead of handling CC's is negligible compared to the overall computation.

Unified Query Processing Scheme

(Shiri 2005) introduced the notion of "certainty constraint." When GIB and GAB are extended with CC's, they yield EGIB and EGAB frameworks, resp, and their equivalence in expressive power is established. Motivated by this result, we propose a unified query processing scheme to evaluate programs in these frameworks. The key point in our development of this unified evaluation scheme is the design of a single suitable internal representation (IR) for EGIB and EGAB programs, which is used by the query processing module.

We also developed modules to parse every EGIB or EGAB program and transform it into its internal representation. This transformation module is shown in Figure 1. As can be seen, there are two paths from input programs to IR: one for EGIB programs and the other for EGAB. In addition, this module can be used to transform an EGIB program into its equivalent EGAB program, and vice versa. This transformation is the basis for equivalence of these two frameworks.

Query processing and unification in our context are more complex due to presence of CC's in rule bodies, as this requires calling certainty constraint checkers, after the certainties associated with regular subgoals in the rule bodies are determined. Our query processing algorithms are obtained by extending the ones proposed in the context of the parametric framework (Lakshmanan and Shiri 2001b; Shiri and Zheng 2004; 2008), to handle CC's. We introduced certainty constraint checker (CC-Checker) in the query processing algorithms, which as the name suggests is used for for dealing with CC's.

The query processing algorithm invokes the CC–Checker on the fly while it evaluates normal subgoals in the rule body and terminates immediately when the CC–Checker fails. We apply the "select–before–join" principle. For every predicate participating in a join, a "select" operation is performed, which prunes useless tuples from a relation before the join operation on consecutive subgoals. The second type of certainty constraints is verified next. The join process fails if

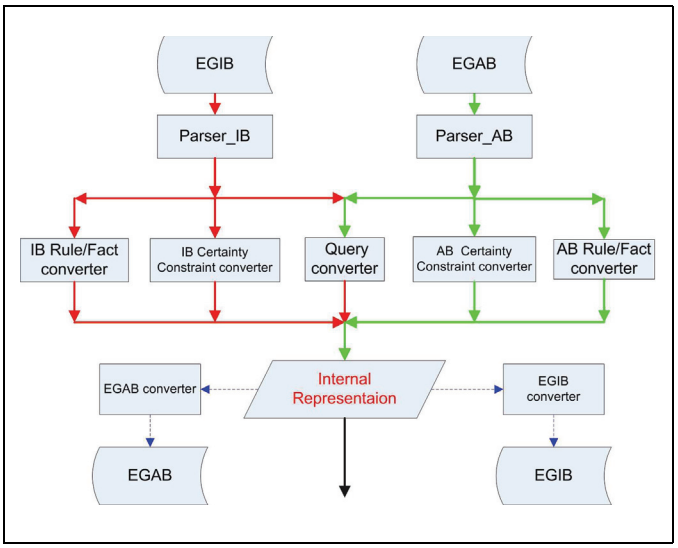

Figure 1: EGAB and EGIB Transformation Module

the verification fails.

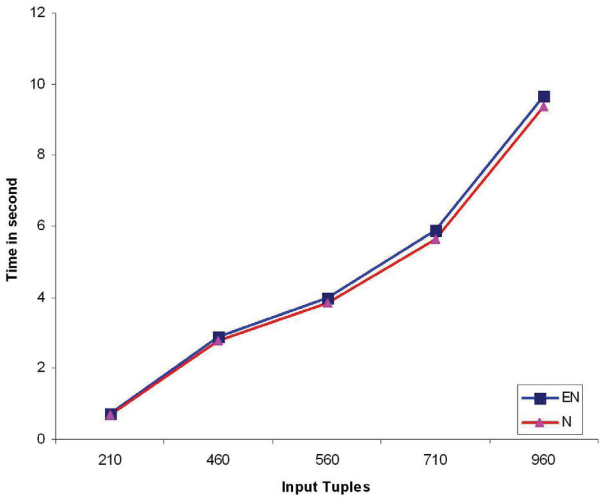

Figure 2: Overhead of CC-Checker evaluation

Experiments

We used two classes of EGIB programs in our experiments, obtained from standard programs, and extended them to include CC's. For this, we considered 9 EDB data sets: A_n, B_n, C_n, F_n, S_n, $T_{n,m}$, and $U_{n,m}$, in different sizes and structures adapted from (Shiri and Zheng 2008). These data sets were originally proposed to evaluate standard Datalog programs. The results shown in Figure 2 indicate that the overhead of handling certainty constraints is reasonable. Figure 3 show different evaluation schemes enjoy different efficiency gains, depending on the complexity of the data.

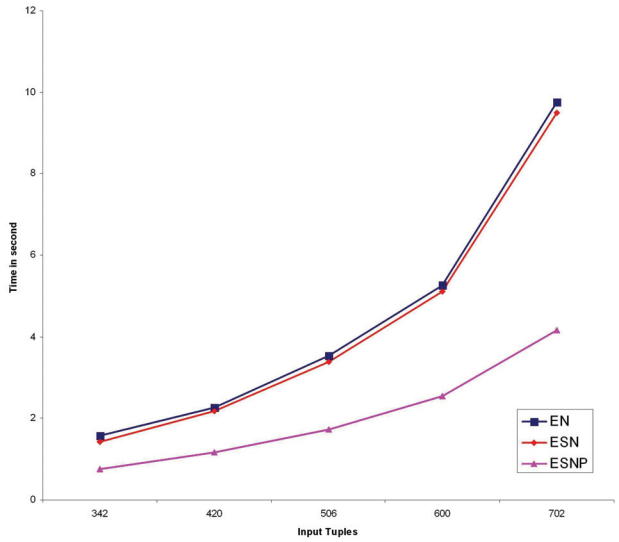

Figure 3: Performance of different evaluation schemes

Conclusion

We developed query processing and optimization techniques to evaluate logic programs with certainty constraints. This provides a unified framework for evaluating logic programs with uncertainty in AB and IB approaches. Our results indicate effectiveness of the proposed techniques.

Acknowledgments: This work was partially supported by NSERC Canada and by Concordia University.

References

Abiteboul, S. et al. 2005. the lowell database research self-assessment. *Commun. ACM* 48(5):111–118.

Lakshmanan, L., and Shiri, N. 2001a. Logic programming and deductive databases with uncertainty: A survey. In *Encyclopedia of Computer Science and Technology, Vol. 45, pp 155–176, Marcel Dekker, Inc., New York.*

Lakshmanan, L., and Shiri, N. 2001b. A parametric approach to deductive databases with uncertainty. *IEEE Trans. on Knowledge and Data Engineerin* 13(4):554–574.

Shapiro, E. 1983. Logic programs with uncertainties: a tool for implementing expert systems. In *Proc. IJCAI'83*, 529–532,William Kaufmann.

Shiri, N., and Zheng, Z. 2004. Challenges in fixpoint computation with multisets. In *Proc. 3rd Int'l Symp. Foundations of Info. and Knowledge Sys., LNCS 2942*, 273–290.

Shiri, N., and Zheng, Z. 2008. Optimizing fixpoint evaluation of logic programs with uncertainty. In *Proc. 13 CSI Int'l Comp. Conf. (CSICC), Comm. in Computer and Information Science, Elsevier, Kish, Persian Gulf, Iran.*

Shiri, N. 2005. Expressive power of logic frameworks with certainty constraints. In *18th Int'l FLAIRS Conference, Special Track on Uncertainty Reasoning*, 759–765.