Query Evaluation in Constraint Databases

(Tutorial Proposal)

David Toman

Department of Computer Science, University of Toronto
Toronto, Ontario M5S 1A4, Canada*

Abstract

One of the leading approaches to finite encodings of infinite relational databases is the use of constraints—quantifier-free characteristic formulas that describe sets of elements in a particular domain. We introduce an abstract notion of a constraint class to serve as a basis for finite encodings of (classes of) such infinite sets. We study the basic operations in terms of which the evaluation of queries over such representation can be defined together with techniques used in efficient query evaluation algorithms. We also introduce several areas where the use of constraints is very beneficial.

1 Introduction

Many modern applications of database technology require infinite domains of values and potentially infinite relations between such values to be handled by the database management system. The restriction to finite relations, common to virtually all relational systems, is not acceptable in many cases, e.g., for modeling time in temporal databases [6] or the spatial dimension in spatial databases [16]. On the other hand, computers can directly handle only finite amount of information. Thus *finite encodings* of classes of infinite databases are needed in these cases. The leading approach to finite encodings of infinite databases uses *constraints* (generalized tuples [14]) in place of the ground tuples. The constraints allow to represent whole sets of tuples using a single tuple—the constraint itself serves as a characteristic formula of such a set.

There are many published proposals, that extend the standard relational model with classes of constraints over various domains [5, 7, 12, 14, 18, 19, 20, 21, 22, 23, 26] and study the query evaluation over such representations. These approaches differ in

- the chosen class of infinite databases (abstract databases) to be represented using constraints. This choice is usually determined by specifying the language (syntax and semantics) of the constraints,
- the query language(s) over the class of abstract databases, and

*Current e-mail address: `david@cis.ksu.edu`

- the actual query evaluation algorithms that map the queries over the abstract databases into operations on the finite encodings—the concrete databases.

In this tutorial we focus on following issues:

1. we extract the common underlying properties of the constraint representation of abstract databases and show the relationship between many of the proposals and our general properties,
2. we define operations on the finite representation, in terms of which the query evaluation of query languages based on logic can be defined, and
3. we study techniques leading to efficient query evaluation algorithms.

Figure 1 captures the relationship between the abstract databases, the concrete databases (where infinite relations are represented using intervals—unary constraints over a linear order), the query languages, and the evaluation algorithms. Besides these goals, we also survey the known finite constraint encodings and present several practical applications of the constraint technology.

2 Organization of the Tutorial

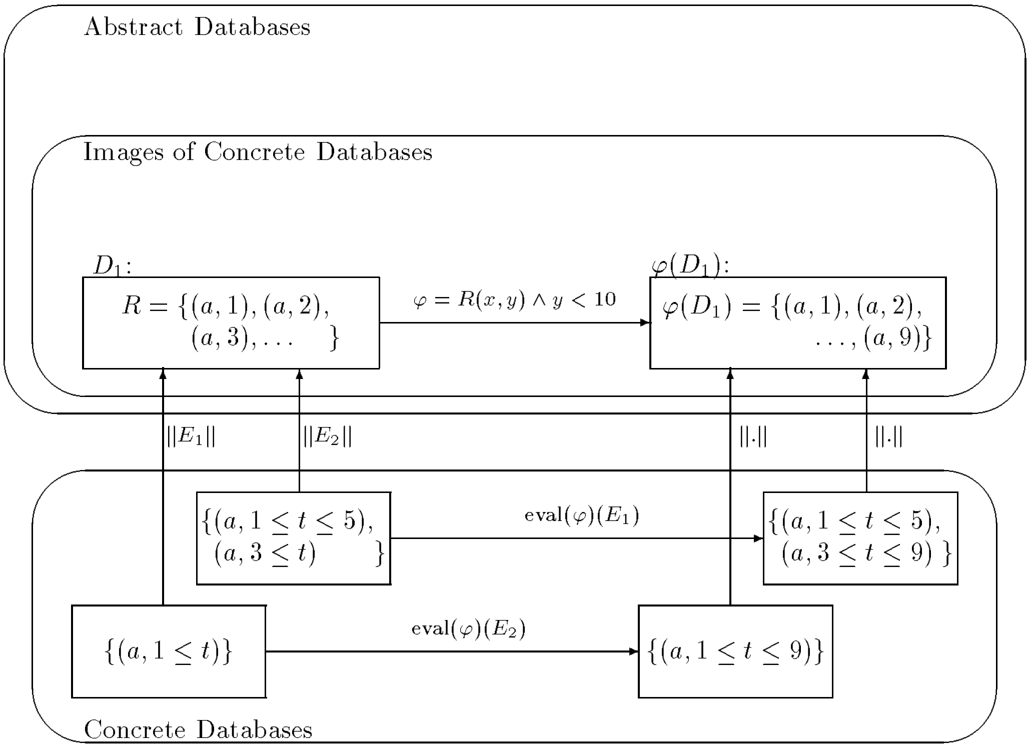
The tutorial consists of three parts:

2.1 Theoretical Foundations

In the first part of the tutorial we investigate the theoretical foundations on which the query evaluation in constraint databases is based:

- We define an abstract notion of *constraint class* [27] that will serve as the common basis of finite encoding of abstract databases.
- We instantiate this general definition with examples of constraint classes, e.g., the linear arithmetic constraints [14], the (gap-)order constraints [18], the periodicity constraints [26], and other.
- We study various query languages over the chosen class of abstract databases and their relation to the chosen encoding. In this section we study the first-order queries and Datalog (deductive) queries over such encodings.

We also introduce general conditions that have to be satisfied by the chosen class of constraints and the query language to guarantee correctness of the appropriate query evaluation algorithm. Special emphasis is placed of on the *closure* of query languages over a chosen finite representation and the notion of *genericity* of queries over such a representation.



E_1 and E_2 are two concrete databases that represent the same abstract database D_1. The $\|.\|$ function is the *semantics* mapping, that determines the meaning of the encodings (concrete databases). The eval mapping is the query evaluation algorithm.

Figure 1: Abstract and Concrete Databases.

2.2 Implementation Techniques

In the second part of the tutorial we present query evaluation algorithms that work over the finite encodings based on constraints. We focus on:

- Evaluation of first-order queries (quantifier elimination, duplicates and subsumption, free variables and domain independence).
- Evaluation of deductive queries (bottom-up algorithm, termination of queries, goal oriented query evaluation strategies based on MST transformation [17, 15] and resolution with tabling [25, 27]).
- Low-level implementation issues (indexing of constraint relations, storage techniques, access methods, compilation of queries, etc.).
- Other features of query languages (e.g., aggregation or dependencies) and their relation to the constraint representation of abstract databases.

We also survey the known constraint representations, including the complexity of query evaluation of both the first-order and deductive queries.

2.3 Applications of Constraint Databases

The last part of the tutorial will introduce several applications of constraint databases:

- Temporal Databases [6] where the constraints are used to represent possibly unbounded periods of time (cf. Figure 1).
- Spatial Databases [16] (e.g., using linear arithmetic constrains to define convex sets as an intersection of hyperplanes of the appropriate dimension), and
- Program analysis using abstract interpretation [1, 8] where the constraints encode the abstract properties we use to analyze the programs.

References

[1] Abramsky, S., Hankin, C. Abstract Interpretation of Declarative Languages. Ellis Horwood, 1987.

[2] Bancilhon F., Maier D., Sagiv Y., Ullman J. D. Magic sets and other strange ways to implement logic programs. In *Proc. ACM Symposium on Principles of Database Systems*, 1986.

[3] Bancilhon F., Ramakrishnan, R. An Amateur's Introduction to Recursive Query Processing Strategies. In *Proc. of ACM SIGMOD '86*, 16–52, 1986.

[4] Baudinet, M., Chomicki, J., Wolper, P. Temporal Deductive Databases. In *Temporal Databases: Theory, Design, And Implementation*, Tansel, A. et al. (editors). Benjamin Cummings 1993.

[5] Baudinet, M., Níezette, M., Wolper P. On the Representation of Infinite Temporal Data and Queries. *10th ACM Symposium on Principles of Database Systems,* pp 280–290, 1991.

[6] Chomicki J. Temporal Query Languages: a Survey. Proc. *International Conference on Temporal Logic*, July 1994, Bonn, Germany, Springer-Verlag (LNAI 827), pp. 506–534.

[7] Chomicki, J., Imieliński, T. Finite Representation of Infinite Query Answers, In *ACM Transactions on Database Systems*, pp. 181–223, (18) 2, 1993.

[8] Cousot P., Cousot R. Abstract Interpretation Frameworks. *J. of Logic and Computation*, 2–4, 1992.

[9] Gao, H., Warren, D. S. A Powerful Evaluation Strategy For CLP Programs. Proc. *Intl. Workshop on Principles and Practice of Constraint Programming*, 1993.

[10] Jaffar J., Maher, M. J. Constraint Logic Programming: A Survey. J. Logic Programming 1994, 19. 20:503–581.

[11] Kabanza, F., Stevenne, J-M., Wolper, P. Handling Infinite Temporal Data. *Journal of Computer and System Sciences*, (51) 1, 149–186, 1995.

[12] Kanellakis, P. C., Goldin, D. Q. Constraint Programming and Database Query Languages. In *Proc. 2nd TACS*, 1994.

[13] Kanellakis, P. C., Ramaswamy, S., Vengroff, D. E., Vitter, J. S. Indexing for Data Models with Constraints and Classes. Proc. *ACM Symposium on Principles of Database Systems,* pp 233–243, 1993.

[14] Kanellakis, P. C., Kuper, G. M., Revesz, P.Z. Constraint Query Languages, *Journal of Computer and System Sciences*, (51) 1, 26–52, 1995.

[15] Mumick, I. S. Query Optimization in Deductive and Relational Databases. PhD thesis, Stanford University, 1991.

[16] Paredaens, J., Van den Bussche, J., Van Gucht, D. Towards a Theory of Spatial Database Queries. In *ACM Symposium on Principles of Database Systems*, 1994.

[17] Ramakrishnan, R. Magic Templates. A spellbinding approach to logic programs. J. Logic Programming 1991, 11:189–216.

[18] Revesz, P. Z. A Closed Form for Datalog Queries with Integer Order. Proc. *3rd International Conference on Database Theory*, pp 187–201, Springer-Verlag LNCS 470, 1990.

[19] Revesz, P. Constraint Query Languages. PhD Dissertation, Brown University, 1991.

[20] Revesz, P. Z. A Closed Form Evaluation for Datalog Queries with Integer (Gap)-Order Constraints. *Theoretical Computer Science*, vol. 116, no. 1, 117–149, 1993.

[21] Revesz, P. Z. Datalog Queries of Set Constraint Databases. In *Proc. 5th International Conference on Database Theory*, 423–438, 1995.

[22] Revesz, P. Safe Stratified Datalog with Integer Order Programs. In *Proc. First International Conference on Constraint Programming*, Montanari U., Rossi F. eds., Springer-Verlag LNCS 976, Cassis, France, September 1995.

[23] Srivastava, D., Ramakrishnan, R., Revesz, P. Z. Constraint Objects. Proc. *Intl. Workshop on Principles and Practice of Constraint Programming*, 218–228, 1994.

[24] Swift, T., Warren, D. S. An Abstract Machine for SLG Resolution: Definite Programs. Proc. *1994 International Logic Programming Symposium*, MIT Press, 1994, 633–652.

[25] Swift, T., Warren, D. S., Chen, W. Operational semantics of SLG evaluation. Tech. report, SUNY at Stony Brook, 1994.

[26] Toman, D., Chomicki, J., Rogers D. S. Datalog with Integer Periodicity Constraints. Proc. *1994 International Logic Programming Symposium*, MIT Press, 1994, 189–203.

[27] Toman, D. Top-Down beats Bottom-Up for Constraint Based Extensions of Datalog. Proc. *1995 International Logic Programming Symposium*, MIT Press, 1995 98–112.

[28] Ullman J. D. Principles of Database and Knowledge-base Systems, Vol. 1,2. Computer Science Systems, 1989.

[29] Williams, H. P. Fourier–Motzkin Elimination Extension to Integer Programming Problems. In *Journal of Combinatorial Theory* (A) 21, pp 118-123, 1976.