

Abstracts of the Benelux Workshop on Computational Logic

*Edited by Henk Vandecasteele and
Maurice Bruynooghe*

Report CW 290, 25 May 2000



Katholieke Universiteit Leuven
Department of Computer Science
Celestijnenlaan 200A – B-3001 Heverlee (Belgium)

Abstracts of the Benelux Workshop on Computational Logic

*Edited by Henk Vandecasteele and
Maurice Bruynooghe*

Report CW 290, 25 May 2000

Department of Computer Science, K.U.Leuven

Sponsored by Flemish Research Network for Declarative Methods in
Informatics

Organising committee:

Maurice Bruynooghe (KUL)
Jan Paredaens (UIA)
Jan Van den Bussche (LUC)
Henk Vandecasteele (KUL)

List of Participants

Aerts Kris (Kris.Aerts@cs.kuleuven.ac.be)
Apt R. Krzysztof (K.R.Apt@cwi.nl)
Areces Carlos (carlos@wins.uva.nl)
Arieli Ofer (Ofer.Arieli@cs.kuleuven.ac.be)
Blockeel Hendrik (Hendrik.Blockeel@cs.kuleuven.ac.be)
Brand Sebastian (Sebastian.Brand@cwi.nl)
Bruynooghe Maurice (Maurice.Bruynooghe@cs.kuleuven.ac.be)
Calders Toon (calders@uia.ua.ac.be)
Claassen Arvid (aclaass@uia.ua.ac.be)
De Mot Emmanuel (Emmanuel.DeMot@cs.kuleuven.ac.be)
De Schreye Danny (Danny.DeSchreye@cs.kuleuven.ac.be)
De Sitter Anneleen (adsitt@uia.ua.ac.be)
De Vos, Marina (marinadv@tinfl.vub.ac.be)
Dekeyser Stijn (dekeyser@uia.ua.ac.be)
Demoen Bart (Bart.Demoen Bart@cs.kuleuven.ac.be)
Denecker Marc (Marc.Denecker@cs.kuleuven.ac.be)
Deville Yves (yde@info.ucl.ac.be)
Driessens Kurt (Kurt.Driessens@cs.kuleuven.ac.be)
Etalle Sandro (etalle@cs.unimaas.nl)
Geerts Floris (floris.geerts@luc.ac.be)
Gennari Rosella (gennari@cwi.nl)
Goethals Bart (bart.goethals@luc.ac.be)
Gyssens Marc (marc.gyssens@luc.ac.be)
Haesevoets Sofie (sofie.haesevoets@luc.ac.be)
Jacobs Nico (Nico.Jacobs@cs.kuleuven.ac.be)
Jakobovits, Hadassa (hadassa@tinfl.vub.ac.be)
Janssen Micha (mja@info.ucl.ac.be)
Janssens Gerda (Gerda.Janssens@cs.kuleuven.ac.be)
Keyngnaert Peter (Peter.Keyngnaert@cs.kuleuven.ac.be)
Kosala Raymondus (Raymondus.Kosala@cs.kuleuven.ac.be)
Kuijpers Bart (bart.kuijpers@luc.ac.be)
Mazur Nancy (Nancy.Mazur@cs.kuleuven.ac.be)
Neven Frank (frank.neven@luc.ac.be)
Paredaens Jan (pareda@uia.ua.ac.be)
Pelov Nikolay (Nikolay.Pelov@cs.kuleuven.ac.be)
Poupaert Eric (ep@info.ucl.ac.be)
Raeymaekers Stefan (Stefan.Raeymaekers@cs.kuleuven.ac.be)
Ramon Jan (Jan.Ramon@cs.kuleuven.ac.be)
Raskin Jean-Francois (Jean-Francois.Raskin@ulb.ac.be)
Riche Jacques (Jacques.Riche@cs.kuleuven.ac.be)
Serebrenik Alexander (Alexander.Serebrenik@cs.kuleuven.ac.be)
Van Gucht Dirk (vgucht@cs.indiana.edu)
Van Hentenryck Pascal (pvh@info.ucl.ac.be)

Van Houteghem Hans (hans.vanhauteghem@uia.ac.be)
Van Laer Wim (Wim.VanLaer@cs.kuleuven.ac.be)
Van Nuffelen Bert (Bert.VanNuffelen@cs.kuleuven.ac.be)
Van den Bussche Jan(jan.vandenbussche@luc.ac.be)
Vandecasteele Henk (Henk.Vandecasteele@cs.kuleuven.ac.be)
Vandeurzen Luc (luc.vandeurzen@luc.ac.be)
Vanhoof Wim (Wim.Vanhoof@cs.kuleuven.ac.be)
Verbaeten Sofie (Sofie.Verbaeten@cs.kuleuven.ac.be)
Verdoolaege Sven (Sven.Verdoolaege@cs.kuleuven.ac.be)
Warren S. David (warren@cs.sunysb.edu)
Wijssen Jef (Jef.Wijssen@umh.ac.be)

Program

1 Thursday May 25

- 12h00-13h00 lunch (sandwiches)
- 13h00-14h00 *David S. Warren (Stony Brook)*, "When Databases Met Logic Programming"
- 14h00-15h00 *Jef Wijssen (Mons)*, "Mining good roll-up levels in data cubes"
- 15h00-15h30 coffee break
- 15h30-16h30 *Sofie Verbaeten (KUL)*, "Termination Analysis of Logic Programs"
- 16h30-17h30 *Jean-Francois Raskin (ULB)*, "Logics, Automata and Classical Theories for Deciding Real Time"
- 17h30-19h30 time to explore Center Parcs or to relax
- 19h30 dinner

2 Friday May 26

- 9h00-10h00 *Pascal Van Hentenryck (UCL)*, "OPL: A modeling language for mathematical and constraint programming"
- 10h00-10h30 coffee break
- 10h30-11h30 *Frank Neven (LUC)*, "Current Trends in Logical Query Languages for Structured or Semi-Structured Data"
- 11h30-12h30 *Carlos Areces (UvA)*, "Dealing with Structure, Logically"
- 12h30-14h00 lunch
- 14h00-15h00 *Ofer Arieli (KUL)*, "Four-valued logics for reasoning with uncertainty"
- 15h00-16h00 *Hadassa Jakobovits (VUB)*, "Semantics for argumentation frameworks, with applications to logic programming and dialogue"
- 16h00- 16h30 coffee
- 16h30-17h30 *Dirk Van Gucht (Indiana University)*, "Recent results in the theory of geometric query languages"

When Databases met Logic Programming

David S. Warren

Stony Brook

e-mail: warren@cs.sunysb.edu

Abstract. In the mid 1980's the communities of Logic Programming and Theoretical Databases began to have serious interactions, and the database point of view began to have significant impact on Logic Programming research (and vice versa.) In this talk I will describe my view of this collision of viewpoints and how this resulted in new advances in Logic Programming in several areas, and led to an explosion of interest in Nonmonotonic Reasoning.

References

[Ullman] J.Ullman Principles of Knowledge Base Systems volume 2.

[Przymusinski] T. Przymusinski Survey paper(s).

[sbprolog] <http://www.cs.sunysb.edu/~sbprolog>.

Mining Good Roll-Up Levels in Data Cubes

Jef Wijsen

Université de Mons-Hainaut

e-mail: Jef.Wijsen@umh.ac.be

Abstract. In recent years, the growing interest in decision-support databases has given rise to a number of new research areas, including data warehousing, online analytical processing (OLAP), and data mining. In these applications, data is conceptually stored not in tables, as in the relational model, but in multidimensional matrices called *data cubes*. Dimensions are usually provided with a lattice of sensible groupings (for example, days are grouped into weeks). A typical class of decision-support queries, called roll-up queries, concerns aggregating and summarizing data along one or more dimensions. Information is generally lost in this roll-up operation. In recent work [4], we characterized this information loss by an entropy-like measure, and we investigated the problem of determining roll-ups for which the information loss is below a specified threshold value. It can be argued that the data cubes resulting from such roll-ups are interesting from a decision-support point of view.

The work on roll-up dependencies continues our earlier research [1, 2, 3] in the area of temporal constraints, where attention is focused on the time dimension. The concept of time granularity, however, is more sophisticated than the roll-up lattices typically encountered in decision-support systems. This gives rise to a number of particular, temporal reasoning and data mining problems.

References

- [1] J. Wijsen. Reasoning about qualitative trends in databases. *Information Systems*, 23(7):469–493, 1998.
- [2] J. Wijsen. Temporal FDs on complex objects. *ACM Trans. on Database Systems*, 24(1):127–176, 1999.
- [3] J. Wijsen. Trends in databases: Reasoning and mining. *IEEE Trans. on Knowledge and Data Engineering*, In press.
- [4] J. Wijsen, R. T. Ng, and T. Calders. Discovering roll-up dependencies. In *Proc. ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, pages 213–222, San Diego, CA, 1999.

Termination Analysis of Logic Programs

Sofie Verbaeten*

Department of Computer Science, K.U.Leuven,
Celestijnenlaan 200A, 3001 Leuven, Belgium.
e-mail: sofie.verbaeten@cs.kuleuven.ac.be

Abstract. Many papers address the problem of proving termination of definite logic programs executed under SLD-resolution (see [DD94] for a survey). In this talk, I first recall the framework of [DVB92] for proving termination of SLD-resolution with the left-to-right selection rule (called LD-termination). I then discuss how adding a tabulation mechanism, more precisely how SLG-resolution of [CW96], improves the termination behaviour. Two notions of termination of definite logic programs executed under mixed tabled/non-tabled execution with the left-to-right selection rule are introduced: quasi-termination and the stronger notion of LG-termination. I show how the necessary and sufficient condition of [DVB92] for LD-termination is extended to conditions for quasi- and LG-termination. Finally, I point out how the termination proofs can be automated by extending the automatic, constraint-based framework for proving LD-termination of [DDV].

References

- [CW96] W. Chen and D.S. Warren. Tabled Evaluation with Delaying for General Logic Programs. *Journal of the ACM*, 43(1):20–74, 1996.
- [DD94] D. De Schreye and S. Decorte. Termination of logic programs: the never-ending story. *Journal of Logic Programming*, 19 & 20:199–260, 1994.
- [DDV] S. Decorte, D. De Schreye, and H. Vandecasteele. Constraint-based automatic termination analysis for logic programs. *ACM TOPLAS*. to appear.
- [DVB92] D. De Schreye, K. Verschaeetse, and M. Bruynooghe. A framework for analysing the termination of definite logic programs with respect to call patterns. In *Proceedings FGCS*, pages 481–488, ICOT Tokyo, 1992. ICOT.
- [VS99a] S. Verbaeten and D. De Schreye. Constraint-based termination proofs for Prolog with tabling. In S. Etalle, editor, *Proceedings Benelog*. Computer Science, Universiteit van Maastricht, number CS 99-07, 1999.
- [VS99b] S. Verbaeten and D. De Schreye. Termination analysis of tabled logic programs using mode and type information. In A. Middeldorp and T. Sato, editors, *Proceedings FLOPS*, pages 163–178. Springer-Verlag, LNCS 1722, 1999.
- [VSS99] S. Verbaeten, K. Sagonas, and D. De Schreye. Modular termination proofs for Prolog with tabling. In G. Nadathur, editor, *Proceedings PPDP*, pages 342–359. Springer-Verlag, LNCS 1702, 1999.

*Research Assistant of the Fund for Scientific Research - Flanders (Belgium)(F.W.O.).

Logics, Automata and Classical Theories for Deciding Real Time

Jean-François Raskin

Département d'Informatique

Faculté des Sciences

Université Libre de Bruxelles

e-mail: Jean-Francois.Raskin@ulb.ac.be

webpage: www.ulb.ac.be/di/ssd/jfr

Abstract. In the theory underlying the automatic verification of reactive systems a class of languages, called “the omega regular languages”, plays a central role. The omega regular languages are identified by formalisms from *modal logic*, like the linear temporal logic, from *automata theory*, like Büchi automata, and by *classical theories*, like the first and second order monadic logics. This class of languages and the formalisms that identify it have been intensively studied since the early seventies. A large number of interesting results are known about those formalisms, for example, they are all closed under all boolean operations and fully decidable, i.e. their satisfiability and validity problems are decidable. Those formalisms are able to express *qualitative* temporal properties about executions of reactive systems, e.g. proper ordering between events. On the other hand, they can not be used to express *quantitative* temporal properties, for example a constraint on the amount of time between two events.

The formalisms that are able to specify quantitative *real-time properties* are called *real-time formalisms*. The real-time formalisms define classes of *real-time languages*. The properties of those languages are still not fully understood. Several real-time formalisms have been proposed in the literature but those formalisms have drawbacks: undecidability problems, non closure under negation, etc. In this PhD thesis, we try to overcome some of those problems by proposing a new family of real-time formalisms.

The *logics, automata and classical theories* that we propose here are (fully) decidable and closed under all boolean operations. The *expressiveness* of those formalisms is studied and compared to the expressiveness of other real-time formalisms proposed in the literature. In particular, we show that those new formalisms identify a class of real-time languages that we propose to call, by analogy with the untimed case, the *real-time omega regular languages*. The problem of complete axiomatization of the logics that identify those languages is studied and solved.

Published material:

- Jean-François Raskin. Logics, Automata and Classical Theories for Deciding Real Time. Thèse de doctorat. Institut d'Informatique, Facultés Universitaires Notre-Dame de la Paix, Avril 1999.
- Jean-François Raskin, Pierre-Yves Schobbens and Thomas A. Henzinger. Ax-

ioms for Real-Time Logics. To appear in *Theoretical Computer Science*, 2000. A preliminary version of this paper appeared in *Proceedings of the 9th International Conference on Concurrency Theory, CONCUR'98*, Lecture Notes in Computer Science 1466, Springer, 219-236, 1998.

- Jean-François Raskin and Pierre-Yves Schobbens. The Logic of Event Clocks: Decidability, Complexity and Expressiveness. In the *Journal of Automata, Languages and Combinatorics*, 4-3, 247-282, 1999.
- Thomas A. Henzinger, Jean-François Raskin and Pierre-Yves Schobbens. The Regular Real-Time Languages. In *Proceedings of the 25th International Conference on Automata, Languages, and Programming, ICALP'98*, Lecture Notes in Computer Science 1443, Springer, 580-591, 1998.
- Thomas A. Henzinger, Jean-François Raskin and Pierre-Yves Schobbens. Temporal Logics, Automata, and Classical Theories for Defining Real-Time Languages. Published as MPI-I-1999-03-003, technical report of the Max-Planck Institute for Computer Science, Saarbrücken, Germany. Also published as a Report of the Computer Science Department of the University of California at Berkeley, UCB/CSD-99-1073, U.S.A., October 1999. (71 pages). Submitted for publication.
- Jean-François Raskin and Pierre-Yves Schobbens. Real-Time Logics : Fictitious Clock as an Abstraction of Dense Time. In *Proceedings of Tacas'97: Tools and Algorithms for the Construction and Analysis of Systems*, Lecture Notes in Computer Science 1217, Springer, 165-182, 1997.
- Jean-François Raskin and Pierre-Yves Schobbens. State Clock Logic: a Decidable Real-Time Logic. In *Proceedings of Hart'97: Hybrid and Real-Time Systems*, Lecture Notes in Computer Science 1201, Springer, 31-47, 1997.

The OPL Optimization Programming Language

Pascal Van Hentenryck
Université catholique de Louvain
2, Place Sainte-Barbe
B-1348, Louvain-la-Neuve (Belgium)
e-mail: pvh@info.ucl.ac.be

Abstract. OPL is a modeling language for solving mathematical programming and combinatorial optimization problems. It is the first modeling language to combine high-level algebraic and set notations from modeling languages with a rich constraint language and the ability to specify search procedures and strategies which are the essence of constraint programming. In addition, OPL models can be controlled and composed using OPL Script, a script language that simplifies the development of applications that solve sequences of models, several instances of the same model, or a combination of both as in column-generation applications. This tutorial surveys the functionalities and application of OPL and OPL Script.

References

- [1] P. Van Hentenryck, I. Lustig, L. Michel, and J-F. Puget The OPL optimization programming language. 245 pages, MIT press Cambridge (Mass), 1999.

Current Trends in Logical Query Languages for XML

Frank Neven

LUC

e-mail: frank.neven@luc.ac.be

Abstract. There is a lot of hype and excitement around XML today. In brief, XML is the universal data format proposed by the W3C and is becoming the *lingua franca* for data exchange between applications over open protocols. All sorts of scenarios are possible: the XML data may be materialized (warehouse); or it may be generated on demand (relational db); it may be a small piece of XML data (e.g., one purchase order); or a huge collection of data (e.g., biological data). The only invariant is that all participants exchange data in the same format, XML. This opens up possibilities for third partners to step in and offer enhancement services: warehouse data, integrate it in order to add value, archive it, etc. Of course, when more XML data becomes available on the Web, applications will need to query, store, integrate, retrieve, and in general manage data in this new format. In this talk, I will focus on one such aspect: querying XML. Query languages for XML differ from more conventional ones, say for the relational model, in that they have to deal with data of irregular or highly flexible structure. For this reason, they usually consist of a pattern language part and a constructing part. The former specifies, usually by means of path expressions, which data elements should be extracted; while the second part indicates how the selected elements should be combined to form the output. Two paradigms of querying have emerged: declarative SQL-like querying inspired by query languages for semi-structured data (XML-QL, RXL); and a more procedural query language introduced by W3C based on template rules (XSLT). I will give examples of these query languages, will discuss their expressiveness and shortcomings, and suggest improvements. Time permitting, I will address the issue of type checking. That is, given an input schema (represented by a Document Type Definition, DTD), an output schema, and a query, is it the case that every XML document conforming to the input schema is transformed to an XML document corresponding to the output schema.

References

Most relevant book:

- S. Abiteboul, P. Buneman, and D. Suciu. *Data on the Web : From Relations to Semistructured Data and XML*. Morgan Kaufmann, 1999.

XSL(T):

- G. Bex, S. Maneth, and F. Neven. A formal model for an expressive fragment of XSLT. <http://www.luc.ac.be/~fneven/pubs.html>

- J. Clark. XSL Transformations version 1.0 (november 1999). <http://www.w3.org/TR/xslt>.
- S. Maneth and F. Neven. Structured document transformations based on XSL. *Workshop on Data Bases and Programming Languages*, 1999.

XML-QL and RXL:

- A. Deutsch, M. Fernandez, D. Florescu, A. Levy, D. Maier, and D. Suciu. Querying XML Data. *IEEE Data Engineering Bulletin*, 22(3):10–18, 1999.
- M. Fernandez, D. Suciu, and W.-C. Tan. SilkRoute: trading between relations and XML. *International Conference on the World Wide Web*, 2000.

Type checking and type inference:

- T. Milo, D. Suciu, and V. Vianu. Type checking for XML transformers. *International Symposium on Principles of Database Systems*, 2000.
- Y. Papakonstantinou and V. Vianu, DTD Inference for Views of XML Data. *International Symposium on Principles of Database Systems*, 2000.

Expressiveness:

- F. Neven. *Design and Analysis of Query Languages for Structured Documents — A Formal and Logical Approach*. Doctor's thesis, Limburgs Universitair Centrum (LUC), 1999.
- F. Neven and T. Schwentick. Expressive and efficient pattern languages for tree-structured data. *International Symposium on Principles of Database Systems*, 2000.
- F. Neven and T. Schwentick. On the power of tree walking automata. *International Colloquium on Automata, Languages and Programming*, 2000.

Dealing with Structure, Logically

Carlos Areces

ILLC, University of Amsterdam

e-mail: carlos@wins.uva.nl

<http://www.illc.uva.nl/~carlos>

Abstract. The central theme of work within the Computational Logic group is computing with content. Content can be represented in many ways, ranging from simple keywords to lightweight semantic analyses to deep ones. For a computational logician, the key challenge is to understand the balance between the richness of representations and the computational efficiency of constructing representations and reasoning with them. Building on ILLC's traditional strengths in modal logic and natural language semantics, the group's strategy is a broad spectrum of foundational, experimental, and applied work with an emphasis on developing small, dedicated logical techniques and lean natural language processing tools.

The central theme of my thesis is Logic Engineering, and in particular the analysis of description and hybrid logics. What is logic engineering? In line with the general interest of the computational logic group, I have been studying modal logics as a means to devise dedicated languages for a given reasoning task. I am interested in understanding how one should go about developing 'made-to-fit' logics. Description logics are a family of knowledge representation languages that are actively being used in areas as diverse as deductive databases, image retrieval, modeling and web maintainance. The description logic community has long stressed the importance of 'structuring' knowledge about a given domain by means of an appropriate formal language. For instance, the difference between terminological and assertional information is a hallmark of the field, as well as the development of specific techniques for particular reasoning tasks, and the use of 'lean' logical systems. Description logicians pay special attention to the complexity of reasoning methods, and the implementation of decision algorithms. But the results are scattered. Hybrid logics, in a sense the 'mathematical brother' of description logics, let us formally analyze issues such as expressiveness power, and in this way provide a suitable logical framework for exploring description logics.

Available material. My papers are available at my web page. Below, I list those that are most related to the topics I will touch on in the presentation. At <http://www.wins.uva.nl/~carlos/hybrid> you will find an extensive bibliography on hybrid logics. The description logic web site can be found at <http://dl.kr.org>.

1. On Hybrid Logics
 - (a) Hybrid Logics. Characterization, Interpolation and Complexity, C. Areces, P. Blackburn and M. Marx. To appear in the *Journal of Symbolic Logic*.
 - (b) Complexity of Hybrid Temporal Logics, C. Areces, P. Blackburn and M. Marx. To appear in the *Logic Journal of the IGPL*.
2. On Description Logics
 - (a) Prefixed Resolution: A Resolution Method for Modal and Description Logics, C. Areces, H. de Nivelle and M. de Rijke. In *Proceedings of CADE'99*. Trento, pages 187–201 1999.
 - (b) Expressiveness Revisited, C. Areces and M. de Rijke. In E. Franconi, G. De Giacomo, R. MacGregor, W. Nutt, C. Welty, editors, *Proceedings of DL'98*, Trento, pages 35–43, 1998.
3. On Applications
 - (a) Feature Interaction as a Satisfiability Problem, C. Areces, W. Bouma and M. de Rijke. In *Proceedings of MASCOTS'99*, IEEE PMSS, 1999.
 - (b) Spatial Reasoning for Image Retrieval, M. Aiello, C. Areces, and M. de Rijke. In *Proceedings of DL'99*, Linköping, 1999.

Four-valued Logics for Reasoning with Uncertainty

Ofer Arieli

Department of Computer Science K.U.Leuven
Celestijnenlaan 200A, B-3001 Heverlee, Belgium
e-mail: arieli@cs.kuleuven.ac.be

Abstract. In his well-known paper “How computer should think” (Contemporary Aspects of Philosophy, Oriel Press, pages 30–56, 1977) Belnap argues that four valued semantics is a very suitable setting for computerized reasoning. In this talk we vindicate this thesis by showing that the *logical* role that the four-valued structure has among Ginsberg / Fitting *bilattices* is similar to the role that the two-valued algebra has among Boolean algebras. Specifically, we provide several propositions that show that the most useful bilattice-valued logics can actually be characterized as four-valued inference relations. In addition, we show that everything that can be done using three values is also possible in the four-valued setting, and even more efficiently, while the converse is not true. The outcome is an evidence that the four-valued setting provides a powerful framework, which is particularly useful for computing some standard non-monotonic methods and paraconsistent techniques.

This is a joint work with Arnon Avron.

Some of my related publications:

1. O.Arieli, A.Avron. *Reasoning with logical bilattices*. Journal of Logic, Language, and Information. Vol.5, No.1, pages 25–63, 1996.
2. O.Arieli, A.Avron. *The value of the four values*. Artificial Intelligence, Vol.102, No.1, pages 97–141, 1998.
3. O.Arieli, A.Avron. *A model theoretic approach to recover consistent data from inconsistent knowledge-bases*. Journal of Automated Reasoning, Vol.22, No.3, pages 263–309, 1999.
4. O.Arieli, A.Avron. *General patterns for nonmonotonic reasoning: From basic entailments to plausible relations*. The Logic Journal of the IGPL, Vol.8, No.2, pages 119-148, 2000.

A full list of publications and some postscript files of the papers are available at:
<http://www.math.tau.ac.il/~ofer>

Dialogues as games on argumentation frameworks

H. Jakobovits

Dept. of Computer Science

Free University of Brussels, VUB

e-mail: hadassa@tinf1.vub.ac.be

webpage: <http://tinf2.vub.ac.be/hadassa/>

Abstract. We provide a formalism for the study of dialogues, where a dialogue is a two-person game, initiated by the proponent who defends a proposed thesis. We examine several different winning criteria and propose an example dialogue type, where a dialogue type is determined by a set of positions, an attack relation between positions and a legal-move function. We propose a proof theory, where a proof theory is determined by a dialogue type and a winning criterion. We supply a declarative semantics which corresponds to the proof theory, this conciliating declarative and dialectic semantics. We suggest an application to logic-programming semantics.

References

- [JV96a] H. Jakobovits and D. Vermeir. Contradiction in argumentation frameworks. In *Proceedings of the IPMU conference*, pages 821-826, 1996.
- [JV96b] H. Jakobovits and D. Vermeir. R-stable models for logic programs. In Dino Pedreschi and Carlo Zaniolo, editors, *Logic in Databases*, number 1154 in Lecture Notes in Computer Science, pages 233-244. Springer Verlag, 1996.
- [JV99a] H. Jakobovits and D. Vermeir. Dialectic semantics for argumentation frameworks. In *Proceedings of the seventh international conference on artificial intelligence and law*, pages 53-62. ACM, June 1999.
- [JV99b] H. Jakobovits and D. Vermeir. Robust semantics for argumentation frameworks. *Journal of Logic and Computation*, 6(2):215-261, 1999.

Recent results in the theory of geometric query languages

Dirk Van Gucht

Indiana University

e-mail: vgucht@cs.indiana.edu

Abstract. Geometric information appears ubiquitously in databases, geographic information systems and solid data modeling systems just being two examples. For at least two decades, researchers and developers have worked on data models, query languages and systems to provide good support for such databases.

In traditional models, geometric data is modeled using a pre-selected set of geometric data types (points, lines, polygons etc). More recently, data models and query languages have been proposed wherein geometric data are specified using *constraints*. For example, if the database consists of the unit circle, then this database would be specified using the constraint $x^2 + y^2 = 1$. Modeling geometric data this way permits for the elegant introduction of declarative query languages based on first order logic.

During the last five years, considerable research has been done to understand the strengths and weaknesses of modeling and querying geometric databases using constraint-theory techniques.

In this talk, I will give a brief overview of the accomplishments and challenges of constraint-based geometric databases. I also plan to single out how a particular limitation of first order query languages, i.e. its inability to express whether a geometric object is topologically connected, has led to richer query languages wherein such problems can be decided. A key technical idea, that of cylindrical decomposition of certain geometric object, will be discussed in this regard.