AGOSTINO DOVIER
4/12/2000

2000 Joint Conference on Declarative Programming (AGP'00)

La Habana (Cuba), December 4-6, 2000
Universidad de La Habana

2000 Joint Conference on Declarative Programming (AGP'00)

La Habana (Cuba), December 4-6, 2000
Universidad de La Habana

PRELIMINARY CONFERENCE PROGRAM

Monday, December 4th
Morning

9.00

Registration

9.30-10.30

Invited Lecture L1: Logic Programming Revisited: Logic Programs as Inductive Definitions
Maurice Bruynooghe

10.30-11.30

SESSION: Abstract Interpretation 1
Chairman: Luciano García

10.30

Designing semantics by domain complementation.
R Giacobazzi and I. Mastroeni.

11.00

Efficient Implementation of General Negation Using Abstract Interpretation.
S. Munoz, J. J. Moreno and M. Hermenegildo

11.30

Break

12.00-13.00

SESSION: Semantics 1
Chairman: Giorgio Levi

12.00

Correct Answers for First Order Logic.
G. Amato

12.30

Operational Semantics for Reexecution-based Analysis of Logic Programs with Delay Declarations.
B. Le Charleir, S. Rossi and A. Cortesi

13.00

Lunch

Monday, December 4th
Afternoon

15.00-16.00

SESSION: Applications and Implementations 1
Chairman: Maurizio Martelli

15.00

An Exercise with Dynamic Knowledge Representation.
J. J. Alferes, L. M. Pereira, H. Przymusinska, T. C. Przymusinski and P. Quaresma

15.30

An Interpreter of Transaction-Frame Logic.
J.A. Carsi and J.H. Canos

16.00-17.00

SESSION: Extensions
Chairman: Annalisa Bossi

16.00

Mixin-based modules for logic programming.
D. Ancona and V. Mascardi

16.30

Dynamic Logic Programming with Multiple Dimensions.
J. A. Leite, J. J. Alferes and L. M. Pereira

17.00

Break

17.30-18.30

SESSION: Theory and Foundations 1
Chairman: Nicoletta Cocco

17.30

Modularity in lambda calculus.
A. Salibra

18.00

Instructing equational set-reasoning with Otter.
A. Formisano, E. G. Omodeo and M. Temperini

Tuesday, December 5th
Morning

9.00 -10.00

Invited Lecture L2: To be announced
Thomas Eiter

10.00-11.30

SESSION: Program Analysis and related topics 1
Chairman: Juanjo Moreno Navarro

10.00

Logic programs as specifications in the inductive verification of logic programs.
M. Comini, R. Gori and G. Levi

10.30

Probabilistic Confinement in a Declarative Framework.
A. Di Pierro, C. Hankin and H. Wiklicky

11.00

An Algorithm for Computing the Data Flow Lattice to Exploit Independent AND-Parallelism in Logic Programming.
C. L. ALonso, C. R. Vela, R. Varela, J. Puente and E. Cabal

11.30

Break

12.00-13.30

SESSION: Databases
Chairman: Pasquale Rullo

12.00

Model-Checking Based Query Retrieval.
A. Dovier, N. Lavarini and E. Quintarelli

12.30

Search and Optimization Problems in Datalog.
S. Greco and D. Sacca

13.00

Lunch

Tuesday, December 5th
Afternoon

15.00-15.45

Tutorial : Constraint Handling Rules.
Thom Fruehwirth

15.45-16.45

SESSION: Concurrent Languages
Chairman: Antonio Brogi

15.45

A Timed Linda Language.
F. de Boer, M. Gabbrielli and M.C. Meo

16.15

Modeling concurrent systems specified in a Temporal Concurrent Constraint language
M. Falaschi, A. Policriti and A. Villanueva

16.45

Break

17.15-18.45

SESSION: Agents

Chairman: Miguel Katrib
17.15

On the interplay between reactivity and rationality.
A. Brogi, S. Contiero and F. Turini

17.45

Formal ReSpecT.
A. Omicini and E. Denti

18.15

Agent Coordination by Constraint optimisation.
Sascha Ossowski

Wednesday, December 6th
Morning

9.00-10.00

Invited Lecture L3: To be announced
James B. Lipton.

10.00-11.30

SESSION: Program Analysis and related topics 2

Chairman: Roberto Giacobazzi
10.00

A Performance Analysis of the BEAM Memory Manager.
R. Lopes and V. Costa

10.30

The Def-inite Approach to Dependency Analysis.
S. Genaim and M. Codish

11.00

On the Number of Rule Applications in Constraint Programs.
T. Fruehwirth

11.30

Coffee-Break

12.00-13.00

SESSION: Applications and Implementations 2

Chairman: Olga Padrón
12.00

A Tabling Engine for the Yap Prolog System,
R. Rocha, F. Silva and V. S. Costa

12.30

Design for AJACS, yet another Java Constraint Programming framework
L. Ferreira and S. Abreu

13.00

Lunch

Wednesday, December 6th
Afternoon

15.00-16.00

SESSION: Abstract Interpretation 2

Chairman: Jim Lipton
15.00

An Abstract Interpretation approach to Termination of Logic Programs.
R. Gori

15.30

Refining and Compressing Abstract Model Checking.
A. Dovier, R. Giacobazzi and E. Quintarelli

16.00-17.00

SESSION: Theory and Foundations 2

Chairman: Alberto Oliart
16.00

A graphical approach to map reasoning.
A. Formisano, E. G. Omodeo and M. Simeoni

16.30

Sistemas de Tipos Puros Extendidos y con Tipos en Forma Normal.
B. Ruiz

17.00

Break

17.30-18.30

SESSION: Semantics 2

Chairman: Luciano García
17.30

DLP as an Abstraction of LO: On the Relations between Disjunctive and Linear Logic Programming
M. Bozzano, G. Delzanno and M. Martelli

18.00

Semantics of Input-Consuming Logic Programs.
A. Bossi, S. Etalle, S. Rossi

Wednesday, December 6th
Evening

21.00

Social Dinner

Overview

	Monday, 4	Tuesday, 5	Wednesday, 6
9:00	Registration	Invited Lecture L2 *Thomas Eiter*	Invited Lecture L3 *James B. Lipton*
9:30	Invited Lecture L1 *Maurice Bruynooghe*		
10:00		SESSION Program Analysis and related topics 1	SESSION Program Analysis and related topics 2
10:30	SESSION Abstract Interpretation 1		
11:00			
11:30	Break		
12:00	SESSION Semantics 1	SESSION Databases	SESSION Applications and Implementations 2
12:30			
13:00	Lunch		
15:00	SESSION Applications and Implementations 1	Tutorial *Thom Fruehwirth*	SESSION Abstract Interpretation 2
15:45		SESSION Concurrent Languages	
16:00	SESSION Extensions		SESSION Theory and Foundations 2
16:45		Break	
17:00	Break		Break
17:15		SESSION Agents	
17:30	SESSION Theory and Foundations 1		SESSION Semantics 2
18:30			
18:45			
21:00			Social Dinner

AGP'00: Abstracts of Accepted Papers

Logic programs as specifications in the inductive verification of logic programs

by Marco Comini, Roberta Gori, Giorgio Levi

Abstract:

In this paper we define a new verification method based on an assertion language able to express properties defined by the user through a logic program. We first apply the verification framework defined in [CominiGLV99] to derive sufficient inductive conditions to prove partial correctness. Then we show how the resulting conditions can be proved using program transformation techniques.

An Abstract Interpretation approach to Termination of Logic Programs

by Roberta Gori

Abstract:

In this paper we define a semantic foundation for an abstract interpretation approach to universal termination and we develop a new abstract domain useful for termination analysis. Based on this approximation we define a method which is able to detect classes of goals which universally terminate (with a fair selection rule). We also define a method which is able to characterize classes of programs and goals for which depth-first search is fair.

Designing semantics by domain complementation

by Roberto Giacobazzi, Isabella Mastroeni

Abstract:

We characterize the symmetric structure of Cousot's hierarchy of semantics in terms of a purely algebraic manipulation of abstract domains. We consider domain complementation in abstract interpretation as a formal method for systematically deriving complementary semantics of programming languages. We prove that under suitable hypothesis the semantics abstraction commutes with respect to domain complementation. This result allows us to prove that angelic and demonic/infinite semantics are complementary and provide a minimal decomposition of all natural-style trace-based, relational, denotational, Dijkstra's predicate transformer and Hoare's axiomatic semantics. We apply this construction to the case of concurrent constraint programming, characterizing well known semantics as abstract interpretation of maximal traces of constraints.

Correct Answers for First Order Logic

by Gianluca Amato

Abstract:

Working within a semantic framework for sequent calculi developed in [AmatoL00], we propose a couple of extensions to the concepts of correct answers and correct resultants which can be applied to the full first order logic. With respect to previous proposals, this is based on proof theory rather than model theory. We motivate our choice with several examples and we show how to use correct answers to reconstruct an abstraction which is widely used in the static analysis of logic programs, namely groundness. As an example of application, we present a prototypical top-down static interpreter for properties of groundness which works for the full intuitionistic first order logic.

Especificación de una gramática para la resolución de referencias temporales

by Estela Saquete, Patricio Martinez-Barco

Abstract:
Este trabajo trata sobre el reconocimiento de expresiones temporales y la resolución de su referencia temporal. Se presenta una propuesta de los módulos que son necesarios para llevar a cabo estas tareas sobre un dominio restringido. Trataremos artículos de periódicos en castellano, por tanto, todas las referencias temporales serán en español. Para la identificación y reconocimiento de expresiones temporales nos basamos en una gramática de expresiones temporales y para la resolución de la correferencia en un diccionario, con la información necesaria para hacer el cálculo de las fechas en base a las expresiones reconocidas. En la evaluación de nuestra propuesta se han obtenido resultados satisfactorios en los casos estudiados.

Modularity in lambda calculus

by Antonino Salibra

Abstract:
We prove that the lattice of lambda theories is not modular and that the variety (equational class) of lambda abstraction algebras, introduced to algebraize the untyped lambda calculus as an alternative to combinatory logic, is not congruence modular.

Dynamic Logic Programming with Multiple Dimensions

by Joao Alexandre Leite, Jose Julio Alferes, Luis Moniz Pereira

Abstract:
According to the recently introduced notion of Dynamic Logic Programming, knowledge is given by a set of theories (encoded as logic programs) representing different states of the world. Different states may represent different time periods, different hierarchical instances, or even different domains. The mutual relationships extant between different states are used to determine the semantics of the combined theory composed of all individual theories. Although suitable to encode one of the possible representational dimensions (e.g. time, hierarchies, domains,...), Dynamic Logic Programming cannot deal with more than one such dimensions simultaneously because it is only defined for linear sequences of states. In this paper, we extend Dynamic Logic Programming to allow for collections of states represented by arbitrary acyclic digraphs, introducing the notion of Multi-dimensional Dynamic Logic Programming. Within this more general theory, we will also show how some particular state structures can be quite useful to model and reason about dynamic multi-agent systems.

Model-Checking Based Query Retrieval

by Agostino Dovier, Nico Lavarini, Elisa Quintarelli

Abstract:
The handling of semistructured data is a crucial task for accessing information distributed over the Web. Typical approaches for solving a query w.r.t. a semistructured Database need to find subgraphs of the instance of a Database that match (e.g., they are isomorphic or bisimilar) with a part of the query. This approach is in general unpractical due to NP-completeness of the method. In this paper we point out that semistructured databases can be naturally seen as Kripke Transition Systems. Moreover, we show that for a large class of queries, modal formulae (more precisely, temporal formulae of CTL) are the natural logic interpretations. This allows to implement query retrieval for semistructured data as an instance of the model-checking problem for CTL, that can be solved in linear time.

Sistemas de Tipos Puros Extendidos y con Tipos en Forma Normal

by Blas Ruiz

Abstract:
Extended Pure Type Systems (\gamma PTS) are an extension of PTS and other theories with dependent strong sums and a hierarchy among universes determined by a relation \gamma between sorts. In this paper we study different formulations for \gamma PTS systems that are correct with respect to the original formulation providing \beta normal types. We establish some consequences on the completeness for one of these systems: a direct proof of the condensation property and its use in the solution to the Expansion Postponement problem. We will show that completeness is a consequence of the inductiveness on a relation between context and subject pairs. KEYWORDS: lambda calculi with types, pure type systems, proof--assistant systems.

A Performance Analysis of the BEAM Memory Manager

by Ricardo Lopes, Vitor Costa

Abstract:
One of the holy grails of logic programming has been to design computational models that could be executed efficiently and would allow for both a reduction of the search space and for exploiting all the available parallelism in the application. These goals have motivated the design of the Extended Andorra Model. We report on a detailed study of the memory management techniques used on our sequencial implementation of the EAM, the BEAM (not to be confused with the Erlang BEAM virtual machine). In this report we address questions like how effective are the techniques the BEAM uses to recover space, reuse and garbage collection and how garbage collection affects performance.

Refining and Compressing Abstract Model Checking

by Agostino Dovier, Roberto Giacobazzi, Elisa Quintarelli

Abstract:
For verifying systems involving a wide number or even an infinite number of states, standard model checking needs approximating techniques to be tractable. Abstract interpretation offers an appropriate framework to approximate models of reactive systems in order to obtain simpler models, where properties of interest can be effectively checked. In this work we study the impact of domain refinements in abstract interpretation based model checking. We consider the universal fragment of the branching time temporal logic CTL* and we characterize the structure of temporal formulae that are verified in new abstract models obtained by refining an abstract domain by means of reduced product and disjunctive completion, or by simplifying the domain by their inverse operations of complementation and least disjunctive bases.

Probabilistic Confinement in a Declarative Framework

by Alessandra Di Pierro, Chris Hankin, Herbert Wiklicky

Abstract:
We show how to formulate and investigate security notions in the context of declarative programming. We concentrate on a particular class of security properties, namely the one called in [VolpanoSmith98a] *confinement* properties. Our reference language is concurrent constraint programming. We use a probabilistic version of this language to highlight via simple programs examples the difference between probabilistic and non-deterministic confinement as pointed out in the work by Volpano and Smith [VolpanoSmith98b,VolpanoSmith98c] in the context of imperative languages. The different role played by variables in imperative and constraint programming hinders a direct translation of the notion of confinement into our declarative setting. Therefore, we introduce the notion of ``identity confinement" which is more appropriate for constraint languages. Finally, we present an approximating probabilistic semantics which can be used as a base for the analysis of confinement properties.

Semantics of Input-Consuming Logic Programs

by Annalisa Bossi, Sandro Etalle, Sabina Rossi

Abstract:
Input-consuming programs are logic programs with an additional restriction on the selectability (actually, on the resolvability) of atoms. This class of programs arguably allows to model logic programs employing a dynamic selection rule and constructs such as delay declarations: as shown also in a previous paper, a large number of them are actually input-consuming.
In this paper we show that -- under some syntactic restrictions -- the S-semantics of a program is correct and fully abstract also for input-consuming programs. This allows us to conclude that for a large class of programs employing delay declarations there exists a model-theoretic semantics which is equivalent to the operational one.

<u>Generación de subontologias de dominios concretos aplicadas a tareas de PLN</u>
by María Pilar Escobar,Maximiliano Saiz-Noeda

Abstract:
Una de las características fundamentales que presenta la base de datos léxica WordNet, es la gran cantidad y diversidad de sentidos de las palabras y de las relaciones semánticas entre ellos. En muchos casos estas características de WordNet son inmanejables en las tareas de PLN (Procesamiento del Lenguaje Natural). Es por ello que en este artículo presentamos un método que restringe la información de WordNet para un dominio concreto, demostrando que esta información obtenida es de suma utilidad para las tareas de PLN, como veremos para la tarea de la resolución de la anáfora.

<u>The Def-inite Approach to Dependency Analysis</u>
by Samir Genaim, Michael Codish

Abstract:
We propose a new representation for the domain of Definite Boolean functions. The key idea is to view the set of models of a Boolean function as an incidence relation between variables and models. This enables two dual representations: the usual one, in terms of models, specifying which variables they contain; and the other in terms of variables, specifying which models contain them. We adopt the dual representation which provides a clean theoretical basis for the definition of efficient operations on Def in terms of classic ACI1 unification theory. Our approach illustrates in an interesting way the relation of Def to the well-known set-Sharing domain which can also be represented in terms of sets of models and ACI1 unification. From the practical side, a prototype implementation provides promising results which indicate that this representation supports efficient groundness analysis using Def formula.

<u>A Denotational Semantics for Timed Linda</u>
by F. de Boer, M. Gabbrielli, M.C. Meo

Abstract:
We introduce a Timed Linda language (T-Linda) which is obtained by a natural timed interpretation of the usual constructs of the Linda model: action-prefixing is interpreted as the next-time operator and the basic Linda actions (in, out and rd) are assumed to take one time-unit. Additionally, T-Linda includes a simple primitive which allows one to specify time-outs. Parallel execution of processes follows the scheduling policy of interleaving, however maximal parallelism is assumed for actions depending on time. We define the operational semantics of T-Linda by means of a transition system and we then define a denotational model which is correct w.r.t the input/output notion of observables.

An Exercise with Dynamic Knowledge Representation
by J. J. Alferes, L. M. Pereira, H. Przymusinska, T. C. Przymusinski, and P. Quaresma

Abstract:
In [ALP+00] we proposed a comprehensive solution to the problem of knowledge base updates. Given the original knowledge base KB and a set of update rules represented by the updating knowledge base KB', we defined a new updated knowledge base KB*=KB+KB' that constitutes the update of the knowledge base KB by the knowledge base KB'. In [APPP99] we introduced a fully declarative, high-level language for knowledge updates called LUPS that describes transitions between consecutive knowledge states and can therefore be viewed as a language for dynamic knowledge representation.
The main objective of the present paper is to show that the dynamic knowledge representation paradigm introduced in [ALP+00] and the associated language LUPS, defined in [APPP99], constitute natural, powerful and expressive tools for representing dynamically changing knowledge. We do so by demonstrating the applicability of the dynamic knowledge representation paradigm and the language LUPS to several broad knowledge representation domains, for each of which we provide an illustrative example. In [APP+99], we presented a preliminary exploration in the application of this paradigm to examples in the domain of action. Here we show the application to other domains, and also add some more insights in its application to the actions domain. All of the examples have been run and tested under our implementation of the LUPS language.

On the Number of Rule Applications in Constraint Programs
by Thom Fruehwirth

Abstract:
We automatically predict the maximal number of rule applications, i.e. worst-case derivation lengths of computations, in rule-based constraint solver programs written in the CHR language. The derivation lengths are derived from rankings used in termination proofs for the respective programs. We are especially interested in rankings that give us a good upper bound, we call such rankings tight. Based on test-runs with randomized data, we compare our predictions with empirical results by considering constraint solvers ranging from Boolean and arithmetic to arc-consistent and path-consistent constraints.

An Interpreter of Transaction-Frame Logic
by J.A. Carsi, J.H. Canos

Abstract:
In this paper we present the architecture and implementation of an interpreter of \em{Transaction-Frame Logic} (TF-logic) that supports most of its characteristics and is currently available for HP-UX workstations. TF-logic allows the definition of object societies and deal with state changes declaratively. TF-logic can be used in a great variety of applications including logic programming, databases and artificial intelligence. To our knowledge, this is the first implementation of TF-logic. We describe the interpreter's architecture and implementation details, as well as its use to define a framework for schema evolution in object-oriented databases.

Mixin-based modules for logic programming

by Davide Ancona, Viviana Mascardi

Abstract:
In this paper we show how it is possible to define a rather rich language of mixin modules suitable for combining together large logic programs without changing the underlying logic.
The type and reduction rules for the language are presented in a somehow informal way, whereas more emphasis is given to the usefulness of the constructs from the programming point of view and to the comparison with other proposals for modular logic programming found in the literature.

DLP as an Abstraction of LO: On the Relations between Disjunctive and Linear Logic Programming

by Marco Bozzano, Giorgio Delzanno, Maurizio Martelli

Abstract:
In this paper we investigate the relationship between disjunctive logic programming as defined in {MRL91} and a subset of linear logic, namely Andreoli and Pareschi's LO {AP91}. We analyze the two languages both from a `top-down', operational perspective, and from a `bottom-up', semantical one. >From a proof-theoretical perspective, we show that, modulo a simple mapping between classical and linear connectives, LO can be viewed as a sub-structural fragment of DLP in which the structural rule of `contraction' is forbidden on the right-hand side of sequents. We also prove that LO is strictly more expressive than DLP in the propositional case. >From a semantical perspective, after recalling the definition of a bottom-up fixpoint semantics for LO we have given in our previous work {BDM00}, we show that DLP fixpoint semantics can be viewed as an abstraction of the corresponding LO semantics, defined over a suitable abstract domain. We study the properties of the resulting abstract interpretation, namely `correctness' and `completeness' of the abstraction {GR97a}. We prove completeness of the abstraction for an interesting class of LO programs encoding Petri-Nets, and we discuss the application of this framework for checking mutual exclusion properties in concurrent systems.

References:
{AP91} J.~Andreoli and R.~Pareschi. Linear Objects: Logical Processes with Built-In Inheritance. New Generation Computing, 9:445--473, 1991.
{BDM00} M.~Bozzano, G.~Delzanno, and M.~Martelli. A Bottom-up semantics for Linear Logic Programs. To be presented at the 2nd International Conference on Principles and Practice of Declarative Programming (PPDP 2000). Montreal, Canada, September, 20-22 2000.
{CC77} P.~Cousot and R.~Cousot. Abstract Interpretation: A Unified Model for Static Analysis of Programs for Construction or Approximation of Fix-Points. Proceedings of the 4th ACM POPL, pages 238--252, Los Angeles, California, 1977. ACM Press.
{GR97a} R.~Giacobazzi and F.~Ranzato. Completeness in Abstract Interpretation: A Domain Perspective. In M.~Johnson, editor, Proceedings of AMAST'97, volume 1349 of LNCS, pages 231--245. Springer-Verlag, 1997.
MRL91 J.~Minker, A.~Rajasekar, and J.~Lobo. Theory of Disjunctive Logic Programs. In J.~Lassez and G.~Plotkin, editors, Computational Logic. Essays in Honor of Alan Robinson, pages 613--639. The MIT Press, 1991.

A Tabling Engine for the Yap Prolog System

by Ricardo Rocha, Fernando Silva, Vítor Santos Costa

Abstract:

This paper addresses the design and implementation of YapTab, a tabling engine that extends the Yap Prolog system to support sequential tabling. The tabling implementation is largely based on the XSB engine, the SLG-WAM, however substantial differences exist since our final goal is to support parallel tabling execution. We discuss the major contributions in YapTab and outline the main differences of our design in terms of data structures and algorithms. Finally, we present some initial performance results for YapTab and compare with those for XSB.

Search and Optimization Problems in Datalog

by Sergio Greco, Domenico Sacca

Abstract:

In this paper we analyze capacity of \DA\ languages to express search and optimization problems. We show that \NP\ search problems can be formulated as \DAm\ queries under total stable model semantics whereas \NP\ optimization problems can be formulated as \DAm\ queries under total stable model semantics by using a {\sl max} (or {\sl min}) construct to select the model which maximizes (resp., minimizes) the result of a polynomial function applied to the answer relation. We analyze several restrictions of \DAm\ and show that they are able to capture significant classes of search and optimization queries.

Design for AJACS, yet another Java Constraint Programming framework

by Ligia Ferreira, Salvador Abreu

Abstract:

This article introduces AJACS (Another JAva Constraint programming System), a toolkit for Concurrent Constraint programming implemented in the Java language. It comes as a successor to our previous work in implementing Constraint Programming idioms in Java, GC, in that it represents an attempt to deal with some of GC's inadequacies in terms of performance whilst providing a setting which is adequate to express problems in a way that can be easily solved in a parallel execution environment, as provided by a concurrent programming setting.

We claim that AJACS provides a very flexible toolkit for use in general applications that may benefit from constraint programming techniques. We also claim that AJACS allows for the coding of CSP problems whose solution is amenable to a practically effortless parallelization.

On the interplay between reactivity and rationality

by Antonio Brogi, Simone Contiero, Franco Turini

Abstract:
Reactivity and rationality are two capabilities of primary importance in multi-agent systems. These two capabilities can be combined in different ways to obtain different behaviours of agents.
The aim of this paper is to provide a simple, logic-based formalization of the interplay between reaction and computation in multi-agent systems. We choose logic programming as the specification language of reactive agents, and we model the possible behaviours of a reactive agent by means of the least fixpoint of a continuous mapping over Herbrand interpretations.
Several different ways of combining reactivity and rationality are modeled and compared one another.

Efficient Implementation of General Negation Using Abstract Interpretation

by Susana Munoz, Juan Jose Moreno, Manuel Hermenegildo

Abstract:
While negation has been a very active area of research in logic programming, comparatively few papers have been devoted to implementation issues. Furthermore, the negation-related capabilities of current Prolog systems are limited. We recently presented a novel method for incorporating negation in a Prolog compiler which takes a number of existing methods (some modified and improved) and uses them in a combined fashion. The method makes use of information providad by a global analysis of the source code. Our previous work focuses on the systematic description of the techniques and the reasoning about correctness and completeness of the method, but provided no experimental evidence to evaluate the proposal. In this paper, after proposing some extensions to the method, we provide experimental data which indicate that the method is not only feasible but also quite promising from the efficiency point of view. In addition, the tests have provided new insight as to how to improve th! e proposal further. Abstract interpretation techniques (in particular those included in the Ciao Prolog system preprocessor) have had a significant role in the success of the technique.

A graphical approach to map reasoning

by Andrea Formisano, Eugenio G. Omodeo, Marta Simeoni

Abstract:
Map reasoning is concerned with relations over an unspecified domain of discourse. Two limitations w.r.t. first-order reasoning are: only dyadic relations are taken into account; all map formulas are equations, having the same expressive power as first-order sentences in three variables. The map formalism inherits from the Peirce-Schr\"oder tradition, through contributions of Tarski and many others.
Algebraic manipulation of map expressions (equations in particular) is much less natural than developing inferences in first-order logic; it may in fact appear to be overly machine-oriented for direct hand-based exploitation.
The situation radically changes when one resorts to a convenient representation of map expressions based on labeled graphs. The paper provides details of this representation, which abstracts w.r.t. inessential features of expressions.

Formal techniques illustrating three uses of the graph representation of map expressions are discussed: one technique deals with translating first-order specifications into map algebra; another one, with inferring equalities within map calculus with the aid of convenient diagram-rewriting rules; a third one with checking, in the specialized framework of set theory, the definability of particular set operations. Examples of use of these techniques are produced; moreover, a possible approach to mechanization of graphical map-reasoning is outlined.
Key words. Formalized reasoning, algebra of dyadic relations, labeled multi-graphs, Peircean existential diagrams, graph transformation.

<u>*Instructing equational set-reasoning with Otter*</u>

by Andrea Formisano, Eugenio G. Omodeo, Marco Temperini

Abstract:
An experimentation activity in automated set-reasoning is reported. The methodology adopted is based on an equational re-engineering of ZF set theory within the ground formalism of \tarskiTimes~[tarski-givant]. On top of a kernel axiomatization of map algebra we develop a layered formalization of basic set-theoretical concepts. A first-order theorem prover is exploited to obtain automated certification and validation of this layered architecture.
Key words. Set reasoning, map algebra, first-order theorem proving.

<u>*An Algorithm for Computing the Data Flow Lattice to Exploit Independent AND-Parallelism in Logic Programming*</u>

by Cesar L. ALonso, Camino R. Vela, Ramiro Varela, Jorge Puente, Eugenia Cabal

Abstract:
In this paper we present an effective strategy to compute a partial order relation among the literals of a given query. This ordering express sufficient conditions in order to the exploit Independent AND Parallelism, that consists in the ordered evaluation of the literals in such a way that two literals are not evaluated in parallel if they share some free variable. To represent the partial ordering in such a way that an efficient strategy of joining partial solutions from the subgoals can be obtained, we introduce a structure named the Data Flow Lattice. The problem of determining the optimal partial ordering is proved to be NP-hard, therefore we propose an heuristic algorithm that requires polynomial time in the number of literals and variables of the query.

<u>*Algoritmos de conversion entre los calculos en Deduccion Natural y Lambek-Gentzen en Gramaticas Categoriales*</u>

by Jose Antonio Jimenez Millan, Antonio Frias Delgado

Abstract:
Trabajamos en el fragmento sin producto de las Gramaticas Categoriales (GC), las cuales dotan de un formalismo logico a las problemas sintacticos. En este campo se presentan dos algoritmos, en forma de matarreglas, referentes a las demostraciones de teoremas en estos formalismos: El primero obtiene un arbol de demostracion en el calculo de Lambek-Gentzen (LG) a partir de una deduccion en el calculo n Deduccion Natural (DN) de cierto teorema; mientras que el segundo algoritmo obtiene una deduccion en el calculo en DN para un teorema a partir de un arbol de demostracion en

LG para dicho teorema. Puesto que el calculo en DN no es el habitual, desarrollamos una version del DN, a la Prawitz, para el fragmento sin producto de las GC y, siguiendo de cerca a Prawitz, demostramos el principio de inversion para las deducciones en este calculo, y algunas de sus propiedades geometricas en las que se basan los demostradores automaticos de teoremas.
KEYWORDS: CATEGORIAL GRAMMARS, NATURAL DEDUCTION, LAMBEK CALCULUS, AUTHOMATIC THEOREM PROVING.

Formal ReSpecT

by Andrea Omicini, Enrico Denti

Abstract:

Logic tuple centres have shown that logic-based languages can be effectively exploited not only for building individual agents and enabling inter-agent communication in multi-agent systems, but also for ruling inter-agent communication so as to build social behaviours.
In this paper, we formally define the notion of logic tuple centre as well as the operational semantics of the logic-based language ReSpecT for the behaviour specification of logic tuple centres.
For this purpose, we exploit a general semantic framework for asynchronous distributed systems allowing a coordination medium to be formally denoted in a separate and independent way with respect to the whole coordinated system.
This shows that a logic-based coordination medium does not bound agents and coordination languages to be logic-based, but may instead enable agents of different sorts and technologies to be combined and coordinated in an effective way by exploiting a logic-based approach.

Operational Semantics for Reexecution-based Analysis of Logic Programs with Delay Declarations

by Baudouin Le Charleir, Sabina Rossi, Agostino Cortesi

Abstract:

We sketch a general semantics framework for logic programs with delay declarations. This is the basis towards the definition of an abstract interpretation that enhances any abstract domain for logic programs to the case of dynamic scheduling. These semantics explicitly express both deadlock information and qualified answers, and allow us to systematically apply well known refinement techniques based on reexecution.

Modeling concurrent systems specified in a Temporal Concurrent Constraint language

by M. Falaschi, A. Policriti, A. Villanueva

Abstract:

In this paper we present an approach to model concurrent systems specified in a temporal concurrent constraint language. Our goal is to construct a framework in which it is possible to apply the \emph{Model Checking} technique to programs spe\-ci\-fied in such language.
This work is the first step to the framework construction. We present a formalism to transform a specification into a `tcc` Structure. This structure is a graph representation of the program behavior.
Our basic tool is the *Timed Concurrent Constraint Programming* (`tcc`) framework defined by Saraswat *et al.* to describe reactive systems. With this language we take advantage of both the natural properties of the declarative paradigm and of the fact that the notion of time is built into the semantics of the programming language. In fact, on this ground it becomes reasonable to introduce the idea of applying the technique of Model Checking to a \emph{finite} time interval (introduced by the user). With this restriction we naturally force the space representing the behavior of the program to be finite and hence Model Checking algorithms to be applicable. The graph construction is a completely automatic process that takes as input the `tcc` specification.
{\bf Keywords:} Timed Concurrent Constraint programming, specification, Reactive systems, Model checking

Agent Coordination by Constraint optimisation

by Sascha Ossowski

Abstract:

This paper presents an instrumentation of a constructionist bottom-up approach to coordination in multiagent systems. Coordination scenarios of autonomous agents are modelled and formalised as a contraint optimisation problem. A distributed asynchronous algorithm is presented that computes potential outcomes of coordination in these scenarios. Finally, it is shown how this algorithm can be integrated into an operational coordination mechanism for a real-world domain.