Dagstuhl Report on

Disjunctive Logic Programming and Databases: Non-Monotonic Aspects

Compiled by
Chandrabose Aravindan
Universität Koblenz-Landau
Fachbereich Informatik
Rheinau 1, D-56075 Koblenz, Germany.
<http://www.uni-koblenz.de/~arvind/>
<arvind@informatik.uni-koblenz.de>

This report covers the seminar on *Disjunctive Logic Programming and Databases: Non-monotonic Aspects*, held at Dagstuhl, Germany during July 1–5, 1996. This seminar was organized by Jürgen Dix, Donald Loveland, Jack Minker, and David Warren to meet the growing interest in this area of research. It brought together about forty researchers from various countries.

Dagstuhl, a place being developed exclusively for research activities in Computer Science, provides an excellent atmosphere for researchers to meet and exchange ideas. During the five days of the seminar, there were vaious talks, panel discussions, and system demonstrations. This report summarizes all these activities. An on-line version of this report is available at the URL <http://www.uni-koblenz.de/ag-ki/dag9627/>.

1 A Note from the Organizers

Jürgen Dix
<http://www.uni-koblenz.de/~dix/>
<dix@informatik.uni-koblenz.de>

Fachbereich Informatik
Universität Koblenz-Landau
Rheinau 1, D-56075 Koblenz, Germany

Non-monotonic reasoning plays a fundamental role in knowledge representation, diagnosis, reasoning about action, logic programming and deductive

databases. In this seminar, we investigated into the interaction and merging of non-monotonic deduction and logic programming-techniques.

In recent years, much research has been accomplished in this area: see the LPNMR conferences, the workshops organized at Logic Programming conferences (ICLP '94, JICSLP '96) by J. Dix, L. M. Pereira and T. Przymusinski on *Non-Monotonic Extensions of Logic Programming* (appeared as Springer LNAI 927) or many papers devoted to these issues in the classical LP-conferences like ICLP and ILPS.

It was also recognized that there are some deep interactions with the following four subareas where non-monotonicity plays an important role:

1. Model-theory of Non-monotonic LP-semantics,
2. Proof-theory of PTTP-like calculi,
3. Deductive Databases, and
4. Efficient Implementations of semantics using LP-techniques.

These fields correspond to the research-areas of the organizers and the invited participants of the seminar.

Besides regular talks, we also had four Panels on *Automated Deduction* (headed by Don Loveland), *Implementations* (headed by David Warren), *Applications* (headed by Luís Pereira) and finally on *Projects-Descriptions* (headed by Teodor Przymusinski).

We believe that closer cooperation between the four areas mentioned above will greatly help to understand disjunctive semantics and also bring efficient implementations of general non-monotonic reasoning systems on its way.

2 Abstracts of the talks

Following is the abstracts of the talks given at the seminar. Wherever possible the on-line home page and email addresses of the speakers are included for reaching them easily. Many speakers have provided an on-line reference to a paper related to their talks. These abstracts and other information about this Dagstuhl seminar are available on the web at the URL `<http://www.uni-koblenz.de/ag-ki/dag9627/>`.

José J. Alferes
<http://www-ia.di.fct.unl.pt/~jja/>
<jja@uevora.pt>

Departamento de Mathemática
Universidade de Évora
P-7000 Évora
Portugal

A non-monotonic logic program semantics and its implementation

See the entry for Luís Moniz Pereira

Chandrabose Aravindan
<http://www.uni-koblenz.de/~arvind/>
<arvind@informatik.uni-koblenz.de>

Fachbereich Informatik
Universität Koblenz-Landau
Rheinau 1, D-56075 Koblenz, Germany.

Abduction and negation in disjunctive logic programming

In this talk, we presented an abductive framework for disjunctive logic programming that captures the semantics of negation in disjunctive programs. Based on this framework, a new inference rule for negation, referred to as negation by failure to explain, was introduced for positive programs. This view of negation was shown to be equivalent to the semantics provided by (Extended) Generalized Closed World Assumption. This relationship between abduction and negation provides a clear methodology to develop algorithms for minimal model reasoning. To demonstrate this, we showed how a theorem prover based on restart model elimination calculus can be modified for abductive reasoning and thus for minimal model reasoning.

This report is available on-line at `http://www.uni-koblenz.de/local/fb4/publications/GelbeReihe/RR-9-96.ps.gz`

Chitta Baral
<http://cs.utep.edu/chitta/>
<chitta@cs.utep.edu>

Department of Computer Science
University of Texas at El Paso

El Paso, Texas 79968, U.S.A.

A Calculus for Disjunctive Logic Programs

Proof procedures for logic programs are complex and usually defined in terms of complicated structures like trees (refer for example to SLDNF or SLINF trees). We would like to define a calculus for disjunctive logic programs similar to the SLDNF calculus introduced by Lifschitz for logic programs without disjunctions. As in the case of SLDNF we believe that this calculus will be significantly much simpler than the current proof procedures for disjunctive programs and will allow us to separate the questions on the procedures from the questions on the proofs. In the presentation we reviewed our initial investigation on the development of this calculus.

Joint work with Jorge Lobo

Peter Baumgartner
<http://www.uni-koblenz.de/~peter/>
<peter@informatik.uni-koblenz.de>

University of Koblenz
Institute for Computer Science
Rheinau 1, D-56075 Koblenz, Germany.

Hyper Tableaux. Part 1: Proof Procedure and Model Generation

This paper introduces a variant of clausal normal form tableaux that we call "hyper tableaux". Hyper tableaux keep many desirable features of analytic tableaux while taking advantage of the central idea from (positive) hyper resolution, namely to resolve away all negative literals of a clause in a single inference step. Another feature of the proposed calculus is the extensive use of universally quantified variables. This enables new efficient forward-chaining proof procedures for full first order theories as variants of tableaux calculi.

This report is available on-line as `http://www.uni-koblenz.de/universitaet/fb/fb4/publications/GelbeReihe/RR-8-96.ps.gz`

Alexander Bochman
<bochman@bimacs.cs.biu.ac.il>

Rh. Kiryat Sefer 18/12
Rishon Le-Zion 75220
Israel

Biconsequence relations for general logic programs

We suggest a general logical formalism for Logic Programming (called biconsequence relations) based on a four-valued inference. We show that it forms a proper setting for representing logic programs of a most general kind and for describing logics and semantics that characterize their behavior. In this way we also extend the connection between Logic and Logic Programming beyond positive programs. A uniform representation of various semantics for logic programs is presented. The main conclusion from this representation is that the distinction between these semantics can be largely attributed to the difference in their underlying (monotonic) logical systems. Moreover, in most cases the difference can even be reduced to that of the language, that is, to the difference in the logical connectives allowed for representing derivable information. In addition, it allows us to see a reasoning about logic programs as a most simple kind of non-monotonic reasoning in general.

Stefan Brass
<http://www.informatik.uni-hannover.de/home/sb/www/brass.html>
<sb@informatik.uni-hannover.de>

Universität Hannover
Institut für Informatik
Lange Laube 22, D-30175 Hannover, Germany

Model-theoretic Characterization and Computation of the Static Semantics

(Joint work with Jürgen Dix and Teodor C. Przymusinski) The autoepistemic logic of beliefs (AEB) extends propositional logic by a modal operator B for "is believed". The static semantics proposed by Przymusinski interprets B(F) as "F is true in all minimal models" (minimal with a fixed interpretation of the belief literals). Then not(F) is interpreted as a shorthand for B(F). We give a fixpoint construction for computing the models of the least static expansion for a class of theories which extends the class of disjunctive logic programs. The algorithm works by computing the minimal models and then iteratively eliminating those which are based on belief interpretations that cannot be represented as the "intersection" of some set of minimal models. The construction is based on a model-theoretic characterization via Kripke-structures.

This report is available on-line at `ftp://ftp.informatik.uni-hannover.de/papers/1996/BDP96a.ps.gz`

A simple prototype implementation is available from `ftp://ftp.informatik.uni-hannover.de/software/static/static.html`

Gerhard Brewka <brewka@informatik.uni-leipzig.de>

Institut für Informatik
Universität Leipzig
Augustusplatz 10-11, D-04109 Leipzig, Germany

Well-Founded Semantics for Default Logic

Default logic is one of the most popular approaches to model defeasible reasoning. Nevertheless, there are a number of problems with Reiter's original semantics that have led to the investigation of alternative approaches. In particular, Baral/Subrahmanian and Przymusinska/Przymusinski have investigated generalizations of well-founded semantics for normal logic programs to default logic. These generalizations have a number of interesting properties. Unfortunately, it turns out that in many realistic situations they are unable to draw any defeasible conclusions at all - which can hardly be viewed as satisfactory. We show how this difficulty can be solved by varying the fixed point operator underlying the semantics. We define a range of different semantics. All of them are correct wrt. safe conclusions under Reiter's semantics, i.e. those conclusions with the same proof in all extensions. For the strongest semantics we have also completeness in the case of coherent default theories, i.e. default theories with at least one extension. The logics differ in the effort spent for determining potential conclusions. We show that our approach also leads to new semantics for normal and extended logic programs. Moreover, we define prioritized versions of the logics.

François Bry
<http://www.pms.informatik.uni-muenchen.de/>
<bry@informatik.uni-muenchen.de>

Computer Science Institute
Ludwig-Maximilians-Universität München
Oettingenstr. 67, D-80538 München, Germany

Towards a constructive definition of the compositional semantics

Non monotonic negation plays an important role in logic programming for several reasons. First, it is a widespread form of negation in various forms of common-sense reasoning as well as in databases. Second, in spite of several attractive proposals, it has not been yet fully satisfactorily formalized, for

some requirements that are natural from the viewpoint of applications are not fulfilled by the proposed semantics. Reconsidering the issue with a strong bias towards applications, we proposed a ***"compositional semantics"*** *in [1]. The compositional semantics aims at accommodating natural program composition requirement - called modularity requirement in [3] - and database integrity constraints. In a same formal framework, the compositional semantics accounts for disjunctive logic programs and database integrity constraints. In this presentation, the compositional semantics as well as the considerations and results that led to its definition are recalled. It is first shown that the procedural semantics of (negation-free) logic programs can be formalized without calling for refutation proofs. That is, as opposed to what is usually assumed, classical logic negation is not needed for formalizing negation-free logic programs. Then it is shown that this gives rise to propose a notion of "local inconsistency" the appropriate one for formalizing negation and integrity constraints in logic programs and deductive databases. A model theory is introduced that accommodates local inconsistencies by means of so-called "weak" and "intended models". It is shown how this semantics generalises the completion [2] and stable semantics [4]. Then it is investigated how a fixpoint like, constructive definition of the intended models of the compositional semantics can be given. Thanks to the locality of inconsistencies, it is possible to generalize the usual bottom-up construction for stratified logic programs to logic programs and deductive databases with unrestricted forms of negation. It is investigated how the fixpoint like, constructive definition extends to disjunctive logic programs and to deductive databases including integrity constraints.*

References:

[1] F. Bry A Compositional Semantics for Logic Programs and Deductive Databases. Proceedings of Joint International Conference and Symposium on Logic Programming, MIT Press, 1996.

[2] K. Clark Negation as Failure. Logic and Databases 293-322, Plenum Press, New York, 1978.

[3] J. Dix A Classification-Theory of Semantics of Normal Logic Programs: I. Strong Properties, II. Weak Properties. Fundamentae Informatica. Vol. 22, No. 3, 227-288, 1995.

[4] M. Gelfond and V. Lifschitz. The Stable Model Semantics for Logic Programming. Proceedings of the International Conference and Symposium on Logic Programming, MIT Press, 1988.

Weidong Chen
<http://www.seas.smu.edu/~wchen/>
<wchen@seas.smu.edu>

Department of Computer Science & Engineering
Southern Methodist University
Dallas, Texas 75275-0122, U.S.A.

Programming with Logical Queries, Bulk Updates and Hypothetical Reasoning

This talk presents a language of update programs that integrates logical queries, bulk updates and hypothetical reasoning in a seamless manner. There is no syntactic or semantic distinction between queries and updates. Update programs extend logic programs with negation in both syntax and semantics. They allow bulk updates in which an arbitrary update is applied simultaneously for all answers of an arbitrary query. Hypothetical reasoning is naturally supported by testing the success or failure of an update. We describe an alternating fixpoint semantics of update programs and show that it can express all nondeterministic database transformations.

This report is available on-line at `http://www.seas.smu.edu/~wchen/papers/updates.ps.gz`

C. V. Damásio
<http://www-ia.di.fct.unl.pt/~cd/>
<cd@fct.unl.pt>

Departamento de Informática
Universidade Nova de Lisboa
P-2825 Monte da Caparica, Portugal

A non-monotonic logic program semantics and its implementation

See the entry for Luís Moniz Pereira

Hendrik Decker
<hendrik@zfe.siemens.de>

Siemens ZFE T SE 4
München, Germany

Is Disjunction in Logic Programming Inclusive or Exclusive or What?

In the field of disjunctive logic programming, it is very common to speak of ***"exclusive"*** *and* ***"inclusive"*** *interpretations of disjunction. By several examples, we argue that it may be misleading or even counter-intuitive to speak of inclusivity or exclusivity with regard to the meaning of disjunction in the head of clauses. Likewise, it is misleading to associate the minimal model based GCWA with exclusive disjunction. For instance, the database* $\{ p \vee q,\ q \vee r,\ r \vee s\}$ *has minimal models* $\{p, r\}$, $\{q, r\}$, $\{q, s\}$ *The second model interprets the second clause of the database as an inclusive disjunction and the other ones as exclusive. Since the middle model is preferred by many semantics, it exemplifies that it is doubtful to speak of an inclusive or exclusive semantics of disjunction at all. Another example: The database* $\{ p \vee q,\ q \vee r,\ r \vee p \}$ *has minimal models* $\{p, q\}$, $\{q, r\}$, $\{r, p\}$ *each of which is interpreting exactly one of the clauses inclusively and the others exclusively. The example shows that, in general, it does not make much sense to say that the GCWA "leads to an exclusive interpretation of Or", or even that "exclusive interpretation of Or leads to minimal models". The latter is not true, as can be seen when attempting to interpret each disjunction in the second database above as exclusive: There is no consistent all-exclusive interpretation. Also, to say that "the GCWA leads to a semantics which is as exclusive as possible" does not necessarily seem to be clear in general. For instance, the GCWA and the WGCWA as well as minimal and maximal models coincide and are the same for the two logically equivalent databases* $\{p \vee p\}$ *and* $\{p \vee false\}$ *[example by Ch. Sakama] where the meaning of disjunction on the left is necessarily inclusive, and the meaning of disjunction on the right is necessarily exclusive. The examples above suggest that it is more accurate to just speak of minimal models or maximal models (i.e., least Herbrand models of the 'maximal version' of a program), for distinguishing different semantics of disjunctive logic programs. The terminology of "inclusive" and "exclusive" is appropriate only in some simple cases.*

Jürgen Dix
<http://www.uni-koblenz.de/~dix/>
<dix@informatik.uni-koblenz.de>

Universität Koblenz-Landau
Rheinau 1, D-56075 Koblenz, Germany

Computation of Non-Ground Disjunctive Well-Founded Semantic

with Constraint Logic Programming

See the entry for Frieder Stolzenburg

Thomas Eiter
<eiter@vexpert.dbai.tuwien.ac.at>

Expressiveness of partial disjunctive models on databases

In this talk, we investigate the expressive power and complexity of partial model semantics for disjunctive deductive databases. In particular, partial stable, regular model, maximal stable (M-stable), and least undefined stable (L-stable) semantics for function-free disjunctive logic programs are considered, for which the expressiveness of queries based on possibility and certainty inference is determined. The analysis pays particular attention to the impact of syntactical restrictions on programs in the form of limited use of disjunction and negation. It appears that the considered semantics capture complexity classes at the lower end of the polynomial hierarchy. In fact, each class of the up to its third level is captured by some semantics using appropriate syntactical restrictions. Partial stable models have exactly the same expressive power and complexity as total stable models, while a higher degree of expressiveness is obtained by the semantics which minimize undefinedness. In particular, L-stable semantics has the highest expressive power. An interesting result in this course is that, in contrast with total stable models, negation is for partial stable models more expressive than disjunction. The results of this paper complement and extend previous results, and contribute to a more complete picture of the computational aspects of disjunctive logic programming and databases, which supports in choosing an appropriate setting that fits the needs in practice.

This report is available on-line at `ftp://ftp.dbai.tuwien.ac.at/pub/papers/eiter/cdtr9583.ps.Z`

The results are from joint research with Nicola Leone (TU Vienna) and Domenico Sacca (Univ. Calabria). This work and more results have been presented at the LID '96 (Logic in Databases) workshop.

Ulrich Furbach
<http://www.uni-koblenz.de/~uli/>
<uli@informatik.uni-koblenz.de>

University of Koblenz
Institute for Computer Science

Rheinau 1, D-56075 Koblenz, Germany

Calculi for Disjunctive Logic Programming

For disjunctive logic programs there are several proposals for defining interpreters, like the nearHorn-Prolog-Family, SLI-Resolution, SLO-Resolution, Model Tree construction or Restart Model Elimination. There have been also different approaches to assign least fixpoints to disjunctive logic programs. This paper proves that there exists an efficient bottom up proof procedure, namely hyper tableaux, which can be understood as a direct implementation of some of the well known fixpoint iteration techniques. We relate the hyper tableaux calculus to state semantics and to model trees. Furthermore we show by a simple syntactic transformation that hyper tableaux can be understood as an extension of SLO-resolution.

Heinrich Herre and Gerd Wagner
<herre@informatik.uni-leipzig.de>

Universität Leipzig
Institut fuer Informatik
Augustusplatz 10-11, D-04109 Leipzig, Germany

Stable Generated Models of Generalized Logic Programs

We present a general definition of stable generated models for extended generalized logic programs (EGLP) which a) subsumes the notion of stable model for normal logic programs (in the sense of Gelfond-Lifschitz) and contains the answer set semantics for extended disjunctive logic programs; and b) allowing for arbitrary (open) formulas in the body and the head of a rule (i.e. does not depend on the presence of any specific connective, nor any specific syntax of rules). The intuition underlying our notion of stability are the properties of groundedness and supportedness which capture the concept of rule-governed generation of factual knowledge. EGLP proves as a powerful general-purpose AI-reasoning system with a clear preferential semantics in the framework of partial logic and thus provides a foundation for practical knowledge-based reasoning. The talk was presented by ***H. Herre****.*

Nicola Leone
<nik@si.deis.unical.it>

Information Systems Department
Technical University of Vienna

A-1040 Vienna, Austria

Disjunctive Stable Models: Unfounded Sets, Fixpoint Semantics and Computation

Disjunctive Deductive Databases (DDDBs) — function-free disjunctive logic programs where negation may occur in the rule bodies — have been recently recognized as a very powerful tool for knowledge representation and common-sense reasoning. We focus on stable model semantics, which, for the time being, is the most widely acknowledged semantics for DDDBs. A new notion of unfounded set for disjunctive programs is first presented. Then, two declarative characterizations of stable models in terms of unfounded sets are given. The former shows that the set of stable models coincides with the family of unfounded-free models (i.e., a model is stable iff it contains no unfounded atom). The second proves that stable models can be equivalently defined by a property of their false literals, as a model is stable if and only if the set of its false literals coincides with its greatest unfounded set. The well-founded W_P operator is then generalized to disjunctive logic programs and a fixpoint semantics for disjunctive stable models is provided. Exploiting the above theoretical results, an algorithm for the computation of stable models is provided. Both soundness and completeness of the proposed method are proved. The computational complexity of the main computational problems related to the presented concepts is also analyzed.

This is a joint research with Pasquale Rullo and Francesco Scarcello. Part of the results presented in this talk appear in the proceedings of the conference ILPS'95.

Jorge Lobo
<jorge@eecs.uic.edu>

Department of EECS
University of Illinois at Chicago
851 S. Morgan st., Room 1120 SEO
Chicago, Illinois 60607-7053, U.S.A.

A Calculus for Disjunctive Logic Programs

See the entry for Chitta Baral

Norman McCain
<mccain@cs.utexas.edu>

University of Texas at Austin
Department of Computer Sciences
Taylor Hall, Austin, TX 78712-1188, U.S.A.

Causal Explanation and Disjunctive Default Logic

In this talk I report on joint work with Hudson Turner. We define two logical systems — one non-modal and one modal — in which the central syntactic and semantic notions are those of a causal theory and a causally explained interpretation, respectively. Roughly, we say that a propositional interpretation I is causally explained by a causal theory T if every proposition that is true in I is caused to be true in I according to T. Given a language whose atoms are formed from fluents subscripted by times, we identify the world histories that are possible according to a causal theory T with the interpretations that are causally explained by T. Of our two logical systems, the modal one, which we call CEL (for Causal Explanation Logic), is more expressive. Under a modal encoding due to Hector Geffner (AAAI-90), it is shown to properly extend the non-modal one. We conclude by presenting a theorem, due to Hudson Turner, which demonstrates a close connection between CEL and disjunctive default logic.

Jack Minker
<http://www.cs.umd.edu/~minker/>
<minker@cs.umd.edu>

Department of Computer Science and
Institute for Advanced Computer Studies
University of Maryland
College Park, MD 20742, U.S.A.

Logic and Databases: a 20 Year Retrospective

At a workshop held Toulouse, France in 1977, Gallaire, Minker and Nicolas stated that ***logic and databases*** *was a field in its own right. This was the first time that this designation was made. The impetus for this started approximately twenty years ago in 1976 when I visited Gallaire and Nicolas in Toulouse, France, which culminated in a workshop held in Toulouse, France in 1977. It is appropriate, then to provide an assessment as to what has been achieved in the twenty years since the field started as a distinct discipline. In this retrospective I shall review developments that have taken place in the field, assess the contributions that have been made, consider the status of*

implementations of deductive databases and discuss the future of work in this area.

This report is available on-line from `http://www.cs.umd.edu/~minker/`

Gerd Neugebauer
`<http://www.uni-koblen.de/~gerd/>`
`<gerd@informatik.uni-koblenz.de>`

Fachbereich Informatik
Universität Koblenz-Landau
Rheinau 1, D-56075 Koblenz, Germany

Deductive Techniques for Information-Management-Systems

Logic programming techniques can be used as a basis for an information management system. For this purpose it is necessary to provide means to incorporate heterogeneous sources of external information into a logic programming system. This talk presents the approach used by the author for this purpose. It also features the angles and pitfalls with which we are faced when we work on a real application.

This report is available on-line at `http://www.uni-koblenz.de/~gerd/Papers/glue.ps.gz`
The slides are available from the URL `http://www.uni-koblenz.de/~gerd/ftp/GLUE/Slides/dagstuhl-96.ps.gz`

Ilkka Niemelä
`<http://www.uni-koblenz.de/~ini/>`
`<ini@informatik.uni-koblenz.de>`

University of Koblenz
Department of Computer Science
Rheinau 1, D-56075 Koblenz, Germany

Towards Implementing Disjunctive Logic Programs using Hyper Tableaux

Circumscriptive (minimal model) reasoning is identified as a key computational task in implementing leading semantics of disjunctive logic programs. Hyper tableaux, which combine hyper resolution and analytic tableaux, are put forward as a basis for a circumscriptive proof method. A new technique for extending tableau calculi to handle circumscriptive inference is proposed. The technique is based on a novel characterization of minimal models and it en-

ables implementation of circumscriptive inference with low space complexity as the branches can be constructed "one at a time". Using this technique first-order parallel circumscription with fixed and varying predicates with respect to Herbrand models can be treated. The method is sound in the general case and complete when no function symbols are allowed.

There are two reports related to the talk:

[1] A Tableau Calculus for Minimal Model Reasoning. Appears in Proceedings of the Fifth Workshop on Theorem Proving with Analytic Tableaux and Related Methods, Palermo, Italy, May 15-17, 1996, Springer-Verlag, 1996 `http://www.uni-koblenz.de/universitaet/fb4/publications/GelbeReihe/RR-5-96.ps.gz`

[2] Implementing Circumscription Using a Tableau Method. Appears in Proceedings of the European Conference on Artificial Intelligence, Budapest, Hungary, Aug 12-16, 1996, John Wiley, 1996 `http://www.uni-koblenz.de/universitaet/fb4/publications/GelbeReihe/RR-6-96.ps.gz`

Luigi Palopoli
<palopoli@si.deis.unical.it>

Tractable disjunctive logic programming

Implementing disjunctive logic programming is a highly complex task. Several tractable subsets of disjunctive logic programming have been proposed, but all of them implies significant limitations in expressive power. In this paper we devise an algorithm which always runs in polynomial time and realizes non-deterministic reasoning for an interesting class of disjunctive logic programs with function symbols. We show that the functional fragment of this class captures all polynomial time queries.

The paper appears in the Proc. of the European Conf. on Artificial Intelligence, Budapest, August 1996.

Christian Papp
<papp@uaic.ro>

Universitatia "Al. J. Cuza"
Jasi, Romania

Hypothesis of complete and partially complete information

While default and autoepistemic logic have received a great interest from researchers, CWA and its generalizations were less considered in the literature. Only CWA and GCWA were iterated on stratified programs. In

this paper, we present two families of logics which can be obtained from the non-monotonic logics used to reason with positive programs (CWA, GCWA, WGCWA, and EGCWA) and their 3-valued versions by performing two kinds of operations: iteration and stable extension. Iteration is suitable in the context of stratified programs. Stable extensions are defined recursively and do not make use if modal operators. A parametrical definition of several known semantics for normal and disjunctive logic programs is this way obtained.

David Pearce
<pearce@dfki.uni-sb.de>

German Research Centre for Artificial Intelligence (DFKI)
Stuhlsatzenhausweg 3
D-66123 Saarbrücken, Germany.

A new logical characterisation of stable models

This paper relates inference in extended logic programming with non-classical, non-monotonic logics. We define a non-monotonic logic, called ***equilibrium logic****, based on the least constructive extension, N2, of the intermediate logic of "here-and-there". We show that on logic programs equilibrium logic coincides with the inference operation associated with the stable model and answer set semantics of Gelfond and Lifschitz. We thereby obtain a very simple characterisation of answer set semantics as a form of minimal model reasoning in N2, while equilibrium logic itself provides a natural generalisation of this semantics to arbitrary theories. We discuss briefly some consequences and applications of this result.*

Luís Moniz Pereira
<http://www-ia.di.fct.unl.pt/~lmp/>
<lmp@fct.unl.pt>

Departamento de Informática
Universidade Nova de Lisboa
P-2825 Monte da Caparica, Portugal

A non-monotonic logic program semantics and its implementation

A semantics for non-monotonic logic programming including explicit negation, integrity constraints, and belief revision was presented. Correct top-down procedures for this semantics and the belief revision methods were presented too, as well as their implementation in prolog, plus examples of

application.

Joint work with J. J. Alferes and C. V. Damásio

Halina Przymusinska
<halina@cs.ucr.edu>

California State Polytechnic University
U.S.A.

Towards a theory of elaboration tolerance: Logic Programming approach

We present an attempt at mathematical investigation of software development process in the context of declarative logic programming. We introduce notions of specification and specification constructor which are developed from natural language description of the problem. Generalization of logic programs, called ***lp-functions*** *are introduced to represent these specifications. We argue that the process of constructing lp-function by certain types of mathematical results which we call* ***representation theorems*** *and present some such theorems to illustrate the idea.*

Joint work with Michael Gelfond, University of Texas at El Paso
<mgelfond@cs.utep.edu>

Teodor C. Przymusinski
<teodor@cs.ucr.edu>

Super-Logic Programs

In order to demonstrate that a class of programs can be justifiably called an ***extension of logic programs*** *one has to establish that: the proposed* ***syntax of such programs*** *is sufficiently general to include the class of normal logic programs; the proposed* ***syntax of queries*** *about such programs is sufficiently expressive to include the class of standard queries about normal logic programs; the proposed* ***semantics of such programs*** *is sufficiently powerful to precisely determine answers to the above queries and the answers, when restricted to normal logic programs, coincide with those provided by one of the well-established semantics of normal logic programs. In addition to these minimal and necessary requirements one should be able to argue that: the proposed class of programs is significantly more expressive than the class of normal programs; the proposed semantics constitutes an intuitively natural extension of the semantics of normal programs; there is a clear and well-understood relationship between the proposed class of programs and their*

semantics and other, well-established and broader classes of non-monotonic formalisms; there exists a reasonably simple, at least in principle, procedural mechanism allowing the computation of the proposed semantics which can be (or already is) implemented as a system prototype. In my talk I proposed a specific class of such extended logic programs which will be called ***super logic programs*** *or just* ***super-programs****. I argued that the class of super-programs satisfies all of the above conditions, and, in addition, is sufficiently flexible to allow various application-dependent extensions and modifications.*

Based on joint work with J.J. Alferes, S. Brass, J. Dix and L.M. Pereira

Dave Reed
<http://stats.dickinson.edu/math/reed/reed.html>
<reedd@dickinson.edu>

Dept. of Math and Computer Science
Dickinson College
Carlisle, PA 17013, U.S.A.

Incorporating Default Negation into Bidirectional Reasoning

This talk addressed the use of negation as failure in the context of a SATCHMO-style theorem prover. SATCHMO (Manthey and Bry, 1988) is an elegant model-generation theorem prover, with one variant that combines the backward chaining of Prolog with forward chaining to implement bidirectional reasoning. Previous work (Loveland, Reed, and Wilson, 1995) has shown that relevancy testing can be incorporated into SATCHMO (SATCHMORE - SATCHMO with RElevancy), leading to much greater control over the forward chaining. Although SATCHMO (SATCHMORE) already incorporates classical negation, it is natural to consider the incorporation of default negation, especially the use of negation as failure as in Prolog. Unfortunately, the use of negation as failure in the backward chaining phase of SATCHMO can lead to inconsistent result. This is due to the fact that not all positive information is known in the backward chaining phase, since new facts may be asserted via forward chaining. However, if the clause set is locally stratified so that negative goals have their definitions contained entirely in the backward chaining component, then negation as failure is safe and behaves as expected.

Konstantinos Sagonas
<http://www.cs.sunysb.edu/~kostis/>

<kostis@CS.SunySB.Edu>

Department of Computer Science
University at Stony Brook
Stony Brook, NY 11790, U.S.A.

An Abstract Machine for Computing the Well-Founded Semantics

In this talk we will present an implementation of the well-founded semantics in the SLG-WAM of XSB. To compute the well-founded semantics, the SLG-WAM adds to an efficient tabling engine three operations — negative loop detection, delaying and simplification — which serve to detect, to break and to resolve the cycles through negation that may arise in evaluating normal programs. We describe fully the addition of these operations to our tabling engine, and demonstrate the efficiency of our implementation in two ways. First, we present a theorem that bounds the need for delaying to those literals which are not dynamically stratified for a fixed-order computation. Secondly, we present performance results that indicate that the overhead of delaying and simplification to Prolog — or tabled — evaluations is minimal.

This is joint work with Terrance Swift and David S. Warren. An extended abstract of this paper appears under the same title in the proceedings of the 1996 Joint International Conference and Symposium on Logic Programming (published by The MIT Press).

John S. Schlipf
<john.schlipf@uc.edu>

University of Cincinatti
Dept. of Computer Science Cincinatti, OH 45221-0008, U.S.A.

Exploring closure properties in building stable models

One way to grow stable models is to make tentative assignments of truth values as in the Davis-Putnam procedure. There it is always desirable to make all possible unit clause reductions before guess; here the analogue seems to be to close under on analogue of the well-founded semantics. For example, the system of Niemelä and Simons seems to do this. Unfortunately, due to non-monotonicity, the correctness of the procedure may not be obvious. We show this can be justified easily by studying the correct operator, in this case $ST_p(I) + I$, where $ST_p(I)$ is the least model of the GL-transform of P w.r.t. I, and $+$ is the lub in the knowledge ordering. (the same operator was also

used by Seipel).

Dietmar Seipel
<http://www-info1.informatik.uni-wuerzburg.de/staff/seipel.html>
<seipel@informatik.uni-wuerzburg.de>

Universität Würzburg
Institut für Informatik
Am Hubland, D-97074 Würzburg, Germany

Model Generation for Disjunctive Deductive Databases

This talk investigates two fixpoint approaches for minimal model reasoning with disjunctive deductive databases P. The first one, called model generation*, is based on an operator $T_i(P)$ defined on sets of Herbrand* interpretations*, whose least fixpoint is logically equivalent to the set of minimal Herbrand models of the database. The second approach, called* state generation*, uses a fixpoint operator $T_s(P)$ based on* hyperresolution*. It operates on disjunctive Herbrand* states *and its least fixpoint is the set of logical consequences of P, the so-called* minimal model state *of the database. We establish a useful correlation between hyperresolution by $T_s(P)$ and model generation by $T_i(P)$. Then we investigate the problem of* continuity *of the two operators $T_s(P)$ and $T_i(P)$. It is known that the operator $T_s(P)$ is continuous, and so it reaches its least fixpoint in at most omega steps. On the other hand, the question of whether $T_i(P)$ is continuous has been open. We show by a counterexample that $T_i(P)$ is not continuous. Nevertheless, we prove that it converges towards its least fixpoint in at most omega steps too, as follows from the relationship that we show exists between hyperresolution and model generation. This convergence result can be used to show that various kinds of computations that are related to model generation, e.g. the computation of perfect and stable models for normal-disjunctive deductive databases, reach their least fixpoints in at most omega steps. Finally, based on an alternating fixpoint computation with $T_i(P)$, we define a new well-founded semantics for normal-disjunctive deductive databases. This semantics, which we call* alternating disjunctive well-founded semantics ADWFS*, generalizes several well-known semantics - including the perfect model and the well-founded semantics - from the normal to the normal-disjunctive case. Furthermore,* ADWFS *is* consistent *for all normal-disjunctive databases without integrity constraints.*

This report is available on-line from `http://karna.cs.umd.edu:3264/projects/dddb.html`

The research on model generation and state generation was done in cooperation with *Jack Minker* and *Carolina Ruiz* from the University of Maryland at College Park, USA.

A research report on *Model Generation and State Generation for Disjunctive Logic Programs* is available at `http://karna.cs.umd.edu:3264/projects/dddb.html`

Frieder Stolzenburg
`<http://www.uni-koblenz.de/~stolzen/>`
`<stolzen@informatik.uni-koblenz.de>`

Universität Koblenz-Landau
Rheinau 1, D-56075 Koblenz, Germany

Computation of Non-Ground Disjunctive Well-Founded Semantic with Constraint Logic Programming

Impressive work has been done in the last years concerning the meaning of negation and disjunction in logic programs, but most of this research concentrated on propositional programs only. While it suffices to consider the propositional case for investigating general properties and the overall behaviour of a semantics, we feel that for real applications and for computational purposes an implementation should be able to handle first-order programs. In this talk we present a theoretical framework by defining a calculus of program transformations that apply directly to rules with variables and function symbols. Our main results are that (1) this calculus is confluent for arbitrary programs, (2) for finite ground programs it is equivalent to a terminating calculus introduced by Brass/Dix, and (3) it approximates a generalization of D-WFS for arbitrary programs. We achieve this by transforming program rules into ***rules with equational constraints*** *thereby using methods and techniques from* ***constraint logic programming****. In particular,* ***disconnection-methods*** *play a crucial role. This work might become the basis for a general combination of two paradigms: constraint logic programming and non-monotonic reasoning.*

This is a joint work with Jürgen Dix. A related report is available on-line at `http://www.uni-koblenz.de/~stolzen/ papers/nmelp96.ps.gz`

Terrance Swift
`<http://www.cs.sunysb.edu/>`

<tswift@cs.sunysb.edu>

Computer Science Dept.
SUNY Stony Brook
Stony Brook, NY 11790-44400, U.S.A.

Taking I/O Seriously: Resolution Reconsidered for DISK I/O

Modern compilation techniques can give Prolog programs, in the best cases, a speed comparable to C. However, Prolog has proven to be unacceptable for data-oriented queries for two major reasons: its sometimes poor termination and complexity properties for Datalog, and its tuple-at-a-time strategy. A number of tabling frameworks and systems have addressed the first problem, most notably the XSB system which has achieved Prolog speeds for tabled programs. However, tabling systems such as XSB continue to use the tuple-at-a-time paradigm. As a result, these systems may have poor complexity for aggregate optimization problems, and are not amenable to a tight interconnection with disk-resident data. However, in a tabling framework, the difference between tuple-at-a-time behavior and set-at-a-time can be viewed as one of scheduling. Accordingly, we define a set-at-a-time tabling strategy and prove it ***iteration equivalent*** *to a form of semi-naive magic evaluation. That is, we extend the well-known asymptotic results of Seki by proving that each iteration of the tabling strategy produces the same information as semi-naive magic. Further, this set-at-a-time scheduling is amenable to implementation in an engine that uses Prolog compilation. We describe both the engine, which is freely available, and its performance, which is comparable with the depth-first strategy* ***even for in-memory Datalog queries****. Because of its performance, and its fine level of integration of Prolog with a database-style search, the set-at-a-time engine appears as an important key to linking logic programming and deductive databases.*

Mirek Truszczynski
<http://www.cs.engr.uky.edu/~mirek/>
<mirek@cs.engr.uky.edu>

Computer Science Department
University of Kentucky
Lexington, KY 40506-0046, U.S.A.

Extremal problems in logic programming

We discuss the following problem: given a class of (disjunctive) logic programs C, determine the maximum number of stable models (answer sets) of a program from C. We establish the maximum for the class of all logic

programs with at most n clauses, and for the class of all logic programs of size at most n. We also characterize the programs for which the maxima are attained. We obtain similar results for the class of all disjunctive logic programs with at most n clauses, each of length at most m, and for the class of all disjunctive logic programs of size at most n. Our results on logic programs have direct implication for the design of algorithms to compute stable models. Several such algorithms, similar in spirit to the Davis-Putnam procedure, are described in the paper.

This report is available on-line at `ftp://www.cs.engr.uky.edu/~ftp/cs/manuscripts/extremal.ps`

Adnan H. Yahya
`<http://www.pms.informatik.uni-muenchen.de/mitarbeiter/yahya/>`
`<yahya@informatik.uni-muenchen.de>`

Institut für Informatik
Ludwig-Maximilians Universität
Oettingerstr. 67, D-80538 Munich, Germany

A Goal-Driven Approach to Efficient Query Processing in Disjunctive Databases.

In query answering, approaches based on model generation tend to explore a much larger search space than is strictly needed. Top-down processing has a more focused search space which can, in many cases, result in more efficient query answering procedures. In this work a strong connection between model generation and clause derivability is established that allows the use of a (minimal) model generating procedure for evaluating queries in a top-down fashion. The advanced procedure requires no extensive rewriting of the input theory or the introduction of new predicates. Rather, it is based a certain duality principle for interpreting logical connectives of the theory. The dual theory is achieved by reversing the direction of the implication sign in all clauses representing the theory and the negation of the query. It is shown that a generic (minimal) model generating procedure operating on the transformed theory is equivalent to top-down query answering in the original theory. We explain the reasoning behind the advanced approach and show that it can be utilized for refined query answering by specifying the conditions under which the query becomes derivable from the theory.

This report is available on-line at http://www.pms.informatik.uni-muenchen.de/publikationen/PMS-FB/PMS-FB-1996-12.ps.gz

Jia-Huai You
<http://web.cs.ualberta.ca/~you/>
<you@cs.ualberta.ca>

GSB 615
Dept. Computing Science
University of Alberta
Edmonton, Alberta
Canada T6G 2H1

Disjunctive Logic Programming by Constrained Monotonic Inferences

We present a novel view of non-monotonic reasoning based on monotonic inferences constrained by a simple notion of priority constraint. More important, we show that these type of constrained inferences can be specified in a knowledge representation language where a theory consists of a collection of logic programming-like inference rules and a priority constraint among the rules. We demonstrate its applications to CWA, default reasoning, and circumscription. In particular we show that the answer set semantics of disjunctive logic programs can be represented by constrained monotonic inferences.

This report is available on-line from http://web.cs.ualberta.ca/~you/papers/pub.html
The talk is based on joint work with Drs. Xianchang Wang and Li Yan Yuan.

Li-Yan Yuan
<http://www.cs.ualberta.ca/~yuan/>
<yuan@cs.ualberta.ca>

Department of Computing Science
University of Alberta
Edmonton, Canada T6G 2H1

Accessibility Relation and Disjunctive Logic Programs

By studying the accessibility relations in logic of belief, we reveal that almost all prominent declarative semantics of disjunctive logic programs can be characterized in terms of standard Kripke structures with different accessibility relations. Based on our understanding of belief, we impose a simple but

very intuitive restriction on the accessibility relation for belief: "An agent, being in world s, considers t as a possible belief world if and only if every belief held in t can be justified by facts and beliefs held in s." Under this restriction, we are able to define three interesting classes of Kripke structures via various accessibility relations, and we further show that almost all prominent declarative semantics of disjunctive programs can be characterized by these three difference classes of Kripke structures. This result not just extends the expressive power of the Kripke structure but also clearly demonstrates that the non-monotonic reasoning is a reasoning about implicit belief and therefore can be characterized by various logics of belief.

This report is available on-line at `http://web.cs.ualberta.ca/~yuan/papers/dag96.ps`

3 Abstracts of the Panel discussions

Following is the abstracts of the panels held at the Dagstuhl seminar. Three panels of general interest were held: automated deduction, implementations of LP systems and applications of LP systems. Along with the abstract, complete contact information of the corresponding organizers is also provided.

PANEL I: Automated Deduction

Organized by: Donald W. Loveland
`<dwl@cs.duke.edu>`

Computer Science Dept.
Duke University
Box 90129
Durham, NC 27708-0129, U.S.A.

The panel on Automated Deduction, with members Wolfgang Bibel, Ulrich Furbach, Miroslaw Truszczynski, David S. Warren and Donald Loveland, moderator, led a discussion that included the full audience, on the status and future directions for automated deduction, logic programming and deductive databases. The panel noted the decreased funding in the U.S. for these fields, and discussed reasons why this is occurring. Also addressed were steps possible to improve the visibility of the fields and the most promising directions

for research in these areas.

PANEL II: Implementations

Organized by: David Warren
<warren@CS.SunySB.EDU>

Dept. of Computer Science
State University of New York at Stony Brook
Stony Brook, NY 11794, U.S.A.

The panel on implementations was organized by David Warren and included Weidong Chen, Carlos Damasio, Ilkka Niemelä, Dietmar Seipel, and Mirek Truszczynski as panelists. After opening remarks from the panelists, the discussion was thrown open to the whole audience. The issues under discussion included: What has been achieved in the area of NMR implementations? E.g., what do we know now that we didn't know recently? What are the strengths and weaknesses of current implementations? What are the most difficult problems now facing the implementers? How do logic programming, NMR, and automated deduction fit together? Are there implementation issues dividing them? Are there things that implementers can do to make their systems more used?

PANEL III: Applications

Organized by: Luís Moniz Pereira
<http://www-ia.di.fct.unl.pt/~lmp/>
<lmp@fct.unl.pt>

Departamento de Informática
Universidade Nova de Lisboa
P-2825 Monte da Caparica, Portugal

In this panel we examined promising application areas for LP and discussed the requirements put on the evolution of the language and its implementation. Several invited panelists made opening statements (Pearce, Decker, Furbach, Pereira), followed by general discussion.

In addition to these three panels, a special session on project descriptions and system implementations that included system demonstrations was orga-

nized by Teodor Przymusinski <teodor@cs.ucr.edu>. Following successful demonstrations of various LP systems by L.M. Pereira & his colleagues, I. Niemelä, P. Baumgartner, D. Seipel, F. Bry & A.H. Yahya, S. Brass, and D.S. Warren & his colleagues, it was decided to set up a web page to collect information on all such LP systems. The artificial intelligence research group at Koblenz, headed by Ulrich Furbach <uli@informatik.uni-koblenz.de>, volunteered to do that job, and the corresponding web page is at the URL <http://www.uni-koblenz.de/ag-ki/LP/>.