Curriculum Vitae et Studiorum

Wolfgang Faber

June 28, 2010

Contents

1 Personal Data

Degrees:	Diplom-Ingenieur, Doctor technicae, Privatdozent (the first two roughly correspond to M.Sc., Ph.D.)
Date and Place of Birth:	8. April 1974, Vienna, Austria
Citizenship:	Austrian
Family Status:	unmarried, no children
Current Position:	Professore Associato (Associate Professor) at the University of Calabria, and Privatdozent (Affiliated Lecturer) at the Vienna University of Technology
Office and Mailing Address:	Department of Mathematics University of Calabria Via P. Bucci, cubo 30B, I–87030 Rende (CS), Italy
Office Phone:	+39 0984 496462
Office Fax:	+39 0984 496410
E-mail:	wf@wfaber.com
Homepage:	`http://www.wfaber.com`
Languages:	German (first language) English (fluent) Italian (fluent) French (high-school level)

2 Education

May 2006	Habilitation officially granted.
April 2006	Habilitation colloquium (for the venia docendi in "Information Systems"), I successfully defended my habilitation thesis "Contributions to Extensions of Answer Set Programming".
July 2002	Degree "Doctor technicae" (Ph.D.) officially granted.
June 2002	Graduation with honors. Thesis "Enhancing Efficiency and Expressiveness in Answer Set Programming Systems" advised by Prof. Th. Eiter.
October 1998 – June 2002	Doktoratsstudium, Studienzweig Informatik (PhD course in Computer Science) at the Vienna University of Technology.
June 1998	Graduation with honors. Thesis "Disjunctive Datalog with Strong and Weak Constraints: Representational and Computational Issues", advised by Prof. N. Leone.
October 1992 – June 1998	Diplomstudium Informatik (Diploma Study in Computer Science) at the Vienna University of Technology.
June 1992	Graduation with honors.
September 1984 – June 1992	Secondary education at Schottengymnasium, Vienna.
September 1980 – June 1984	Primary education at Volksschule Groß-Enzersdorf.

3 Grants and Awards

April 2004	Awarded an "APART" grant by the Austrian Academy of Sciences for 3 years of research. APART is one of the most prestigious grants in Austria and is awarded to only about 10 researchers (of all fields) per year.

4 Professional Experience

since December 2006	Professore Associato (Associate Professor) at the University of Calabria, Italy
since May 2006	Privatdozent (Affiliated Lecturer) at the Vienna University of Technology
October 2004 – November 2006	"APART" fellowship of the Austrian Academy of Sciences for 3 years of research. APART is one of the most prestigious grants in Austria and is awarded to only about 10 researchers (of all fields) per year.
October 1999 – November 2006	Universitätsassistent (corresponds to Assistant Professor) at the Knowledge Based Systems Group, Vienna University of Technology.
October 2002 – September 2003	Zivildiener (Social Servant, civil version of compulsory military service) at hospital Krankenanstalt Rudolfstiftung, Vienna[1].
July 1998 – September 1999	Research Assistant in the project P-11580-MAT *"Design and Implementation of a Query System for Disjunctive Deductive Databases"*, funded by FWF, the Austrian Science Fund.
March 1997 – June 1998	Student Fellow in the project P11580-MAT.
1995 – 1996	Studienassistent (Student Teaching Assistant) at the Institute of Computer Languages, Vienna University of Technology.
1994 – 1998	Tutor for the lab course "Logic-oriented Programming Languages".

I am also a co-founder of the company DLVSYSTEM s.r.l.

5 Teaching Experience

5.1 Students

- Axel Polleres
  - co-supervised (with Thomas Eiter) Master's thesis "The DLVK System for Planning with Incomplete Knowledge" 2001
  - Axel has been a postdoctoral researcher at the University of Innsbruck, Austria, the Universidad Rey Juan Carlos, Spain, and currently is a postdoctoral researcher at the National University of Ireland, Galway.
- Mario Alviano
  - co-supervised (with Nicola Leone) Master's thesis "Valutazione Efficiente di Aggregati Ricorsivi in Programmazione Logica" 2007

  - Mario has been awarded the prize for the best Italian Master's thesis on Artifiacial Intelligence by the Italian Association for Artificial Intelligence in September 2008.
  - Mario has started a PhD at the University of Calabria in 2008 under my supervision (together with Nicola Leone).

- Annamaria Bria
  - co-supervised (with Nicola Leone) Doctoral thesis "Normal Form Nested Programs" 2010
  - Annamaria is currently a postdoctoral researcher at the University of Calabria.

- Gennaro Frazzingaro
  - I am supervising Gennaro's Master's thesis (together with Giovambattista Ianni), expected to finish in 2010. Gennaro has done a 6-month internship at Apple in Cupertino and has been employed full-time after that.

I have supervised several other students, even though I have not been their official supervisor. Among these are Simona Perri (currently assistant professor at the University of Calabria), Tina Dell'Armi (project coordinator at the company Exeura and contractual lecturer at the University of Calabria), Giuseppe Ielpa (employee at the company ID Tech), Francesco Calimeri (assistant professor at the University of Calabria), Chiara Cumbo (employee at the company Exeura), Manuela Citrigno (software analyst at the University of Bologna), Francesco Ricca (assistant professor at the University of Calabria).

5.2 Lectures

- Lecture "Knowledge Management" at the University of Calabria, Italy (fully responsible, fall 2009)
- Lecture "Logic and Computability" at the University of Calabria, Italy (fully responsible, 2009)
- Lecture "Introduzione all'Informatica (Introduction to Computer Science)" at the University of Calabria, Italy (fully responsible, fall 2008)
- Lecture "Ragionamento Automatico (Automated Reasoning)" at the University of Calabria, Italy (fully responsible, fall 2008)
- Lecture "Planning nell'IA (Planning in AI)" at the University of Calabria, Italy (fully responsible, fall 2008)
- Lecture "Ragionamento Automatico (Automated Reasoning)" at the University of Calabria, Italy (fully responsible, fall 2007)
- Lecture "Logiche Nonmonotone (Nonmonotonic Logics)" at the University of Calabria, Italy (fully responsible, 2007)
- Lecture "Ragionamento Automatico (Automated Reasoning)" at the University of Calabria, Italy (fully responsible, 2006 (spring and fall))
- Lecture with Laboratory "Intelligenza Artificiale (Artificial Intelligence)" at University of Calabria, Italy (with Nicola Leone, Simona Perri, Francesco Calimeri, 2004).

- Lecture “AK der Artificial Intelligence 3 (Deduktive Datenbanken/Deductive Databases) 184.225 – 2.0 VU” at TU Vienna, Austria (fully responsible, 2004).
- Lecture “Deductive Databases 184.115 – 2.0 VO” at TU Vienna, Austria (fully responsible, 2001, 2002).
- Lecture “Wissensbasierte Systeme (Knowledge-based Systems) 184.178 – 2.0 VO” at TU Vienna, Austria (with Uwe Egly, Thomas Eiter, Hans Tompits, 2004).
- Lecture with Laboratory “Logikorientierte Programmierung (Logic Programming) 184.143 – 2.0 VL” at TU Vienna, Austria (with Uwe Egly, Hans Tompits, 2004).
- Laboratory course “Logikorientierte Programmierung (Logic Programming) 184.126 – 2.0 LU” at TU Vienna, Austria (with Thomas Eiter, 2001).
- Laboratory course “Einführung in Wissensbasierte Systeme (Introduction to Knowledge-based Systems) 184.176 – 1.0 LU” at TU Vienna, Austria (with Hans Tompits, 2003).
- Various seminars (1999 – 2004).
- Several student practica (1999 – 2004).
- Guest Lecture “Answer Set Based Planning” at University of Leipzig, Germany (2005).
- Guest Lecture “Recursive Aggregates in Disjunctive Logic Programs: Semantics, Complexity, and Computation” at University of Potsdam, Germany (2005).
- Guest Lecture “Answer Set Programming” at University of Messina, Italy (2002).
- Guest Lecture “ASP-based Planning” at University of Reggio Calabria, Italy (2002).
- Guest Lecture “Logic-based Planning” at University of Calabria, Italy (2002).
- Student advisor for a Logic Programming laboratory course (1994 – 1998).

6 Participation in Projects

- Logiche Descrittive Nonmonotone: Complessità e implementazioni (Nonmonotonic Description Logics: Complexity and implementation – LoDeN), PRIN Project
  2010 – 2012, funded by the Italian Ministry of Education, Universities, and Research
  Partners: University of Naples, University of Eastern Piedmont, University of Calabria, National Research Council Pisa
- Potenziamento e Applicazioni della Programmazione Logica Disgiuntiva (Enhancement and Applications of Disjunctive Logic Programming), PRIN Project
  2007 – 2009, funded by the Italian Ministry of Education, Universities, and Research
  Partners: University of Calabria, University of Naples, University of Messina
- Sistemi basati sulla logica per la rappresentazione di conoscenza: estensioni e tecniche di ottimizzazione (Logic-based Systems for Knowledge Representation: Extensions and Optimization Techniques), Italy-Austria Internationalization Project
  2004 – 2008, funded by the Italian Ministry of Education, Universities, and Research
  Partners: Vienna University of Technology, University of Calabria

- Advanced Information Integration (IST-2001-33570 INFOMIX)
  2002 – 2005, funded by the European Commission (Fifth Framework Programme)
  Partners: University of Calabria, University of Rome "La Sapienza", Vienna University of Technology, Rodan Systems S.A. (company based in Poland)

- Working Group on Answer Set Programming (IST 2001-37004 WASP)
  2002 – 2005, funded by the European Commission (Fifth Framework Programme)
  Partners: University of Messina, Helsinki University of Technology, Vienna University of Technology, University of Cyprus, University of Manchester, University of L'Aquila, University of Calabria, University of A Coruña, University of Trento, University of Leipzig, Frije Universiteit Brussel, University of Bath, University of Potsdam, University of Naples, Katholieke Universiteit Leuven, University Rey Juan Carlos, University of Reggio Calabria, Clausthal University of Technology

- Intelligent Content Management System (IST 2001-32429 ICONS)
  2001 – 2003, funded by the European Commission
  Partners: Rodan Systems S.A. (company based in Poland), Polish Academy of Sciences, CIES (company based in Calabria), InfoVide Sp. z o.o. (company based in Poland), Sema Group Belgium, University Paris 9 Dauphine, University of Ulster

- Answer Set Programming for Reactive Planning and Execution Monitoring (FWF P16536)
  2003 – 2006, funded by the Austrian Science Funds
  Vienna University of Technology

- A Declarative Planning System Based on Logic Programming (FWF P14781)
  2001 – 2003, funded by the Austrian Science Funds
  Vienna University of Technology

- Enhancing Nonmonotonic Systems to Deal with Quantitative Information (scientific collaboration Italy-Austria)
  1999 – 2000, funded by the Italian and Austrian governments
  Partners: Vienna University of Technology, University of Calabria

- Design and Implementation of a Query System for Disjunctive Deductive Databases (FWF P11580)
  1997 – 2000, funded by the Austrian Science Funds
  Vienna University of Technology

7 Research Achievements

Most of our research so far has been centered around Answer Set Programming (ASP). This paradigm has evolved from Logic Programming and Deductive Databases, and allows for expressing a mathematically precisely defined class of problems. In contrast to conventional Logic Programming, such as Prolog, ASP does not allow for function symbols (or only in a limited form); however, its language features include negation as failure, strong negation, and disjunctions in a semantically sound way.

Our work in this area, described in Sections 7.1, 7.2, and 7.3, consists of both theoretical and practical studies, the latter consisting of contributions to ASP systems, most notably the system `DLV`, and also of applications of ASP. However, we have also worked in areas which are closer to Databases, as reported in Section 7.4. Work focussing on applications is reported in Section 7.8. Yet another focus of our work has been on Reasoning about Action and Change, in particular Planning, as evidenced in Section 7.5. We have also studied Diagnostic Reasoning, described in Section 7.6. Finally, work which does not fit into these categories is described in Section 7.9.

7.1 Answer Set Programming Systems

Our work in this area is mostly centered around the system `DLV`, of which the author has been, and still is, a main architect and contributor. About this system in general, in [95, 4] we have described programming methodologies and usage scenarios for `DLV`, while in [16] we focus more on system internals and adopted techniques. We have presented the system on various occasions, where the focus has usually been on some particular novelty or feature of the system [78, 80, 82, 89, 93, 97, 98]. In the following, we will briefly describe some of the major contributions on certain aspects of ASP systems, and `DLV` in particular.

7.1.1 Grounding

Current competitive ASP systems use a two-stage architecture, where the first stage creates a ground version of the input program by instantiating variables and possibly simplifying the program, obtaining a program without variables, which is equivalent (with respect to answer sets) to the original program with variables. The second stage then computes the answer sets of the obtained program.

We have been involved in the implementation of the `DLV` grounding, in particular concerning program rewriting and simplification. In [94] we have described a general method for rewriting non-ground programs such that their instantiated version becomes smaller while preserving answer sets. This method is mostly based on introducing or pushing projections, which has been motivated both by related techniques in the areas of database systems and satisfiability planning.

Also the work on magic sets, which is discussed separately in Section 7.4.2, can be seen as an optimization technique for grounding.

7.1.2 Pruning Operators

In their second stages, ASP systems work on programs without variables, and compute their answer sets. These algorithms usually consist of a backtracking search, which is loosely based on the Davis-Putnam-Loveland-Logeman (DPLL) technique for SAT. However, there are many crucial differences, such as the need for a final model check in the presence of disjunctions, and a completely different way of computing deterministic consequences after having made a choice. In particular, DPLL has only one pruning operator, which is unit propagation, while in ASP there are many more and quite different pruning operators.

We have made several studies on pruning operators, their potential and impact. A fundamental work has been [91], in which we have introduced a new truth value, termed must-be-true. The rationale of this operator is that each true atom in an answer set must be supported by a rule. Hence, it makes sense to distinguish between true atoms which do not yet have a supporting rule (these are must-be-true) and those which do (which we just refer to as true). Based on this idea, we have developed several operators which exploit the additional truth value, implemented them in the system DLV, and showed by means of extensive benchmarks that it has an overall positive effect on efficiency, which is dramatic in many cases. A detailed description of this work can also be found in our doctoral dissertation [100].

In another body of work, published in [79, 18], we study the impact of the so-called well-founded operator, in comparison to the related Fitting operator. We could clarify this relationship, demonstrating that the two operators coincide on a certain class of programs. Given the fact that the well-founded operator is more costly to compute, it does not make sense to use it on this kind of programs. Furthermore, we could show that the programs on which the operators do not coincide, can be further divided into two classes – one in which the well-founded operator is tractable, while it is not in the other class. Thus, it is prohibitive to use the well-founded operator on the latter class of programs. We have created an implementation of the well-founded and Fitting operators in DLV based on these theoretical results, and have assessed experimentally that these changes are favorable in terms of efficiency.

7.1.3 Heuristics

Still in the second stage of ASP systems, which deal with variable-free programs, an important parameter of the employed backtracking search algorithms is the way how choices are made. The search process can be shrunk tremendously when making "good" choices. However, the notion of a good choice is far from clearly defined. Moreover, it is well-known that determining optimal choices (for which the computation will be shortest) is intractable. Therefore, all competitive systems make use of search heuristics, which are soft criteria, tending to yield good choices, which are close to optimal choices.

We have studied several search heuristics for ASP. In [84, 86], we have analyzed a number of look-ahead based criteria, where some techniques have been motivated by heuristics, which have been developed for boolean satisfiability, while others are genuine techniques, which are tailored for ASP, and DLV in particular. After having developed the rationales for the various criteria, we have conducted extensive benchmarks to assess their impact, leading to a conclusion which determined the basis of the currently used search heuristic of DLV.

In [63, 56, 53, 17], we have elaborated on this work, noting that the previously established criterion has weaknesses on a certain class of programs. After having made an analysis on the reasons, we have proposed an evolved criterion, which should behave better on these programs, while retaining the properties of the established heuristics on all other programs. Our experiments could confirm this intuition.

In [81] we have examined optimizations on the calculation of the heuristic criteria themselves. Due to the look-ahead nature of the adopted heuristics, we have noted that frequently considerable time is spent just in computing the heuristic values of the candidate atoms. Usually the overhead is outweighed by the cut in search space, but not always. We could identify some cases in which heuristic values are known to be equal to already computed values, thus allowing for safely omitting their computation. Moreover, we have examined a second-level heuristic criteria, which is not look-ahead based and very lightweight, which is used to identify a subclass of choices for which the sharper, but more expensive look-ahead heuristic will be calculated. We could demonstrate experimentally that each of these methods, as well as their combination, has benign effects on efficiency.

7.1.4 Backjumping

Backjumping is a technique which is frequently used in boolean satisfiability solvers, and which has also been examined in the realm of ASP, but only in frameworks which are a strict subset of the `DLV` language. Moreover, backjumping has in the past always been studied in conjunction with another technique known as clause learning. The basic idea of backjumping is that if backtracking is performed because of an inconsistency, it makes sense to go back to the choice, by which the earliest reason for the found inconsistency has been determined, as all other attempts will encounter the inconsistency again. In [59, 17], we have studied how backjumping without clause learning can be adapted and used in `DLV`, and whether it can improve its performance. To this end, we have defined an adapted and simplified reason calculus to determine and maintain reasons for literals, which are then used to identify the reasons of inconsistencies. A series of benchmarks could demonstrate that backjumping for ASP is effective also without clause learning.

7.2 Extensions of Answer Set Programming

We have studied several extensions of classical ASP by additional syntactic constructs. While these works usually include an implementation (in most cases involving `DLV`), they are not centered on classic ASP and mostly contain a study on the syntax and semantics of the extended language.

7.2.1 Inheritance

In [92, 24] we have introduced the notion of inheritance to disjunctive logic programs. In this setting, programs are contained in classes, which may appear in a class hierarchy. Classes which are below other classes in this hierarchy, inherit from the upper classes, and may override information in the upper classes. This formalism is hence suitable for defining ontologies and similar knowledge, and is particularly useful for encoding exceptions.

We have formally defined the syntax and semantics of the language, confronted it with related approaches from the literature, studied the complexity of the associated reasoning tasks, and finally described a translation to ASP, which we leveraged to produce an implementation on top of the system `DLV`.

7.2.2 Aggregates

The use of aggregate functions has been studied for a long time. Especially in databases, they are widely used and nowadays seen as a necessary feature of query languages. For these reasons, many attempts have been made to introduce them also in the language of logic programs, and particularly in Datalog and also ASP. However, it soon became clear that their semantics, and in particular its definition, is far from obvious in the presence of recursive definitions involving aggregates. Several semantics have been suggested in the past, but for many of these, examples showed that they admit highly unintuitive results.

However, for the class of stratified aggregates (i.e., aggregates, which do not occur in recursive definitions), virtually all proposals coincided, and matched the intuitive meaning of these programs.

In [75, 72], we elaborated on this, and extended the basic `DLV` language by a series of aggregates, which, however, are required to meet stratification. We could show that the introduction of these aggregates does not increase the complexity of reasoning, and have suggested and implemented an extension of the DLV algorithm to compute answer sets of these programs. Our theoretical results show that all programs with this kind of aggregates have equivalent programs without aggregates. However, we showed by means of examples that many problems can be encoded in a much more succinct and natural way by using stratified aggregates, and, moreover, benchmarks indicate that programs with aggregates can often solve problems much more efficiently than their aggregate-free equivalents.

In the sequel, we have embarked on examining the intended semantics for programs with recursive definitions involving aggregates. A first tentative in [73] proved to be easy to implement, but, as many of its predecessors, exhibited several unintended results.

After that, in [66], we managed to present a semantics, which is astonishingly simple to define, yet appears to be consistent with the intuitive semantics in all cases. In fact, this work has become a sort of standard reference on this topic, and is by many considered the definitive solution of this problem. The basic ASP semantics is defined as a fixpoint of interpretations with respect to a program reduction. In [66], we keep this schema and change only the definition of the reduct, which is indeed a relaxation, rather than an extension, of the original reduct. We could actually prove that our definition, involving the simpler reduct, is equivalent to the old definition, involving the more complicated reduct, on ASP without aggregates. We have therefore also given a simplified definition of ASP in this paper. The paper concludes by examining several properties, most prominently the computational complexities of reasoning tasks under the new semantics.

Based on these results, we have conducted further research in this direction. While our semantic definition in [66] is simple and gives intuitive results, it does not immediately point to an algorithm. In [58], we have defined an alternative criterion for the semantics of [66], which immediately suggests an implementation. In particular, the work defines the notion of unfounded sets for non-disjunctive programs with a certain class of aggregates: monotone and antimonotone aggregates. We could show that this notion generalizes unfounded sets of aggregate-free programs, and could give a characterization of answer sets by means of unfounded sets, which is in analogy to the aggregate-free case. Given that unfounded sets are in the computational core of all competitive answer set solvers, this result is very important for the creation of systems supporting ASP with aggregates. Since unfounded sets are also at the basis of an alternative, three valued semantics, known as the well-founded semantics, we could generalize this notion to programs with monotone and antimonotone aggregates. Still in [58], we have assessed the computational complexity of computational tasks associated with unfounded sets and the well-founded models.

While the definition of unfounded sets in [58] had been applicable only for certain types of aggregates and only for nondisjunctive programs, we have generalized the definition in [52] to deal with any type of aggregates. We could prove that the characterizations derived in [58] also hold in the more general framework using the new definition. These results thus suggest an implementation for programs with arbitrary aggregates. However, we could also show that the complexity of certain tasks increases in the more general setting; in particular it seems that the addition of disjunctions is similar to the addition of aggregates which are neither monotone nor antimonotone. Our results therefore indicate that the computational framework of DLV, which can deal with disjunctions, can also accommodate arbitrary aggregates.

In [50], we have elaborated on the notion of nonmonotone aggregates, that is, aggregates which are neither monotone nor antimonotone. While our previous work had shown that their presence increases the complexity of reasoning tasks in general, one can observe that many simple aggregates are nonmonotone, but do not justifiy the higher complexity. In particular, we could identify a class of nonmonotone aggregates, for which any program containing these can be rewritten into a program containing only monotone and antimonotone aggregates, where the rewriting requires only minor resources. Indeed, reasoning with programs containing this kind of nonmonotone aggregates has the same complexity as reasoning with monotone and antimonotone aggregates only.

7.2.3 Preferences

Another extension of ASP is defining preferences among the rules of programs. This topic has been studied extensively in the literature, and many proposals for semantics have been given. However, for many of these semantics, implementations are not available, making it hard to test the features of these semantics and assessing their practicality and differences. When implementations are available, they are often ad hoc, or limited to one particular semantics.

Motivated by these facts, we have introduced a new ASP technique in [87, 21], which we refer to as meta-interpretation (a term taken from a related popular Prolog programming technique). Meta-interpretation has allowed us to produce implementations of various semantics for preferential ASP in a uniform and portable way. Indeed, for some of the considered semantics, this work has provided the first implementation at all. The work contains a detailed description of the technique in general and the encodings for the various semantics.

In this area, we have also followed a quite different line of research. In the realm of standard ASP, a notion known as strong equivalence of programs has been developed. Intuitively, two programs are strongly equivalent, if they can be used interchangeably in any context. In [57, 15], we have studied this problem in the setting of ASP with preferences. We have given a definition of strong order equivalence, which masters several technicalities that came up with this generalization, for three prominent semantics. Subsequently, we have examined conditions under which two programs are strongly order equivalent under the respective semantics. It turned out that, somewhat surprisingly, the preference relations of two strongly order equivalent programs must be exactly equal. In addition, the programs without preferences must be strongly equivalent in the classical sense, and in addition the generating rules of the programs must be equal in any context. Especially the last criterion appears to be very restrictive, and one would assume that only a few programs are strongly order equivalent, which might make their identification very easy. However, this did not turn out to be true; indeed we could prove that the complexity of strong order equivalence checking is the same as for strong equivalence checking (without preferences).

7.3 Answer Set Programming Theory

While thematically close to topics discussed in Sections 7.1 and 7.2, a few works do not fit there, as they deal with theoretical aspects of core ASP.

In particular, in [71, 74] we have discussed several complexity issues on non-ground ASP, including the problem of checking whether an interpretation is an answer set. We have shown that for many language fragments, the complexity of this problem stays in the polynomial hierarchy, while in general it is NEXPTIME hard. Moreover, we have demonstrated that the complexities of various problems in the non-ground case strongly depend on how interpretations and answer sets are represented. Furthermore we have analyzed the class of programs for which the predicate arities are bound by a constant. We could show, that many reasoning tasks, the complexities of which are hard for EXPTIME or higher classes for arbitrary programs, are in the polynomial hierarchy for programs with bounded arities, hence easier to solve. An important consequence of this result is that for programs with bounded arities, one should be able to come up with better implementations than the current ASP systems, which involve a grounding stage.

Another work, [55] describes a system for testing strong equivalence of non-ground programs. While the implementation has followed an idea given previously in the literature, it has been the first functional system of its kind. Essentially it transforms two programs into a first-order formula in the Bernays-Schönfinkel class (formulas with a $\exists\forall$ quantifier pattern). The satisfiability of formulas of this class is known to be decidable, but only few theorem provers are guaranteed to terminate on them. One prover which is guaranteed to terminate on Bernays-Schönfinkel formulas is DARWIN, thus using it together with a program performing the transformation yields a strong equivalence tester for non-ground ASP.

7.4 Databases

We have also investigated in several database topics. Actually, ASP can be seen as a database query language, so most of the work described so far could also be associated to databases, in particular to deductive databases. However, the work described in this section has a much stronger link to databases than the others.

7.4.1 Data Integration

We have been involved in building a system for data integration, which is called INFOMIX. Data integration is a problem in which several independent data sources should be united to form a single (virtual) database. This is a problem which occurs frequently, for example most companies do not have one central database, but rather their data is dispersed over several independent databases. An even vaster setting is when also data from the WWW has to be considered. In INFOMIX two main issues in data integration are dealt with: Incompleteness and inconsistency of data. Incompleteness refers to the fact that some data might be missing from data sources, while inconsistency means that several data sources might contain information, which – though consistent in each data source – is inconsistent in combination.

Several semantics have been suggested for solving these kinds of problems; in INFOMIX we have implemented the loosely sound semantics in a GAV (global as view) setting. The resulting system can deal with various kinds of data sources, ranging from traditional relational databases over information retrieved from the WWW to very generic sources such as text files. The integration itself is internally done by a reduction to ASP, using `DLV` as a computational engine.

This activity sparked a number of related work in order to improve the efficiency of INFOMIX. In particular, several techniques such as those described in Section 7.4.2 have been motivated by INFOMIX. The system has been used successfully in a sample setting involving real-world data and queries involving a university information system, and has produced very satisfactory results. The system has been presented on various occasions: [54, 61, 62]. Features of `DLV` which are dedicated to or motivated by INFOMIX have been described in [67].

7.4.2 Magic Sets

An important optimization technique for Deductive Databases is referred to as Magic Sets. Originally defined in the setting of Datalog (without disjunction and without negation), it is a transformation technique, which speeds up bottom-up evaluation of queries, by limiting the bottom-up computation to a relevant fragment of the query program. Indeed, in this setting it has been shown that bottom-up computation with magic sets can reach or even outperform top-down computations. Several attempts have been made to generalize this technique to programs containing negations. However, these attempts have been confined to programs with stratified negation.

Motivated also by the use of ASP in Data Integration, as described in Section 7.4.1, where query answering is the main problem to solve, we have made an attempt to generalize Magic Sets to nondisjunctive programs containing negations, which are not necessarily stratified. The key point here is that, different from traditional Magic Sets, also a dataflow from rule bodies to rule heads may exist. In order to analyze the requirements, we have first studied the question of when a subprogram is query-equivalent to the full program, i.e. when a subprogram yields the same result as the full program. We have analyzed existing notions of modularity, but found that they do not guarantee query equivalence in general. To overcome this, we have defined a new notion of modularity, which takes body-to-head-dataflow into account and is based on the notion of "dangerous" rules. We could establish query equivalence for our notion of module in the case of programs which are guaranteed to have an answer set; for other programs, we can guarantee either query soundness or query completeness, depending on the reasoning mode. We then defined an elaboration of the Magic Set method, which we could prove to isolate a module according to our notion, which means that the Magic Set method is correct for programs having at least one answer set, otherwise either sound or complete. This result has been crucial for INFOMIX, in which the ASP programs are guaranteed to have at least one answer set. In [60, 65, 14] we define all of these notions in detail and settle also various associated questions on complexity. Moreover, we describe the application to data integration in detail, underlining the importance of our result.

We have also investigated in how Magic Sets can be defined for disjunctive programs, which do not contain negation. Some previous work on this problem existed, but was unsatisfactory in various respects. In [68], we have elaborated on this problem and defined a new method which overcomes many of the problems of the previous proposal. Moreover, we have also implemented our work and could show experimentally that it has positive effects on several problems.

In [69], we have also defined a Magic Set method for programs containing aggregates. The interesting part of all these extensions of Magic Sets is that they are defined in a way that they can be combined easily, thereby arriving at a general method for ASP with disjunctions, negation, and aggregates.

7.5 Reasoning about Actions and Planning

Planning is one of the oldest and most studied problems of Artificial Intelligence. In brief, the objective is to find a sequence of actions which will bring about a desired situation. This problem is a main task in autonomous robots (it should determine what to do to reach its goals), but also in software agents, which act in an artificial world. More technically, the problem to be solved consists in the description of an initial or current situation, a description of the world and how it evolves, and a goal to be achieved. The task is then to find a plan (often a sequence of actions) which achieves the goal starting from the initial situation. From a more general point of view, planning can be seen as one of several problems that deal with reasoning about actions and change.

7.5.1 Action Languages

In this area, we have worked on logic-based languages for representing action and change and planning. In particular, we have introduced a new action language, called $\mathcal{K}$, in [88, 20]. The distinguishing feature of this language is that it works on knowledge states rather then world states, as did virtually all of the previously suggested languages. This view is in line with the robot or agent metaphor: The agent will usually have some *knowledge* about the world, either acquired or a priori, and will in general not be omniscient, or fully aware about everything in the world of discourse. In particular, the knowledge of such an agent will usually be incomplete.

Planning with incomplete knowledge using the language $\mathcal{K}$ can be done in different modes: With *optimistic planning*, plans are just required to achieve the goal in some possible evolution of the world, while with *secure planning*, plans must achieve the goal in any case. In [88, 20], we have defined syntax and semantics of this language, and in [20] we have studied the complexities of optimistic planning, secure planning and secure plan checking in detail. Depending on the structure of the underlying planning domain, these problems range from NP-completeness to NEXPTIME-completeness.

Based on these studies, we have implemented the system $\texttt{DLV}^{\mathcal{K}}$ which allows for optimistic and secure planning for a wide range of planning domains. The implementation uses the system `DLV` as its computational core and is described in [85, 83, 23]. In the same work, we report on extensive benchmarks that have been performed which show the effectiveness and viability of $\texttt{DLV}^{\mathcal{K}}$, also with respect to related planning systems.

Building on this work, we have extended the basic language by introducing cost functions for actions, arriving at the language $\mathcal{K}^c$. Plans can then be valuated according to the actions occurring in it, possibly rejecting all but the plans with minimal cost, or rejecting all plans whose cost is larger than a given bound. In [76, 77, 22], we formally define syntax and semantics of $\mathcal{K}^c$, analyze the complexities of the respective planning tasks, and describe how to extend $\texttt{DLV}^{\mathcal{K}}$ in a modular way such that it can be used also for $\mathcal{K}^c$.

We have furthermore given a general overview of action languages, representing action domains, and planning with them in [3].

7.5.2 Plan Execution

We have also studied various other problems concerning reasoning about actions and change, which are closely related to planning. One of them concerns diagnosing the reasons for failures in plan execution. In this context, we assume that the executed plan is not a secure one (in many cases, a secure plan does not even exist, in others it may be hard to find), and that at some point an action can not be executed. This means that the world did not evolve in a favorable way, and it is necessary to recover from this failure in order to reach the goal nevertheless. In [103], we study how to determine points in time at which an action has caused the failure. While obtaining this information may serve several purposes, its most immediate application is when an agent tries to reuse a plan and wants to go back to some point from which the plan can be retried and has a chance to reach the goal. We have isolated and defined this problem, studied its complexity, and have created an implementation which can do these diagnoses.

Another problem which arises in the plan recovery context described above is that of plan reversal. If an agent wants to undo a portion of a plan, it needs another plan to do so. In [70, 102] we elaborate on this problem, study its complexity and describe several ways of solving it. The techniques usually involve on *offline* and an *online* part, where the offline part creates a library of plan reversals (which is intended to be done before the plan execution), which can then be used in the online part to efficiently determine the reverse plan for a given action sequence.

7.6 Diagnostic Reasoning

In diagnostic reasoning, the goal is to determine explanations for observations, given a theory and hypotheses. In [25], we have studied this problem in the setting of Answer Set

Programming. There are different options about the precise meaning of "explanation". If the explanation is to give a reason for the observations, we arrive at abductive diagnoses, while another option is to allow for explanations which are consistent with the observations. Moreover, often one is interested in parsimonious explanations, that is, preferring simple ones. We studied two modalities, single-error and subset minimal diagnoses. In [25], we provide a detailed analysis of these problems, and describe an implementation on top of the `DLV` system.

In [90], we studied the relationship between diagnostic reasoning and planning problems. It turns out that these are quite closely related.

7.7 Computational Complexity

As evidenced in Sections 7.1, 7.3, 7.4, 7.5, and 7.8, an important aspect of our work has been the study of computational complexity. These activities have been carried out in the respective settings of the analyzed problems, and are commented on in the respective sections.

7.8 Applications

In this section, we report on our research activity, which has been oriented towards applications. While also some other works fall into this class (especially the work described in Section 7.4.1), these had also another special focus beyond applications.

In the paper [51], we have studied the problem of reasoning about privacy. In particular, the problem treated is referred to as the "privacy preservation problem". The setting is an information system with several users, where some users may declare information to be private. For example, in a health care information system, patients may want to keep information about their diseases private. Other users should be able to see this information only when authorized. The problem in this setting is that unauthorized users may be able to determine the private information indirectly, by reasoning based on other informations. In the health care example, an unauthorized user may be able to infer the disease of a patient by querying for the medications. The "privacy preservation problem" assumes that the knowledge of users is modelled, and provides only those informations which do not allow a user to infer private information. We have formally defined this problem and suggested an encoding using default logic. We have also analyzed the complexity of the problem in detail.

In [96] we have elaborated on how to encode problems in school timetabling in ASP. In particular, these problems involve both hard (e.g., at most one lesson can be given in a room at any time) and soft constraints (e.g., teacher preferences about time) on the timetable, and can be encoded in an arguably elegant way in ASP. In particular, we have picked several types of problems from the school timetabling literature and have encoded them using ASP.

7.9 Various Activities

In the course of our research, especially when assessing system improvements as described in Section 7.1, the need of a flexible tool for evaluating resource consumption (e.g. time or space) of heterogeneous systems. Since a tool, which would meet our requirements had not been available, we have implemented our own, as described in [64]. It is available on the web for everyone to use.

8 Activities in the Scientific Community

8.1 Conference Chairing and Organization

- Program Chair of the 11th International Conference on Logic Programming and Non-monotonic Reasoning, (LPNMR 2011)
- Program Chair of the 25th Italian Conference on Computational Logic, (CILC 2010)
- General Chair of the Second Workshop on Answer Set Programming and Other Computing Paradigms (ASPOCP 2009)
- General Chair of the First Workshop on Answer Set Programming and Other Computing Paradigms (ASPOCP 2008)
- Local Organization of the Third International Symposium on Foundations of Information and Knowledge Systems (FoIKS 2004)
- Publicity Chair and Local Organization of the 6th International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'01)
- Publicity and Local Organization for the Joint German/Austrian Conference on Artificial Intelligence (KI-2001)

8.2 Program Committee Membership

- Nonmonotonic Reasoning at 30 (NMR@30 2010)
- 12th European Conference on Logics in Artificial Intelligence (JELIA 2010)
- 26th International Conference on Logic Programming (ICLP 2010)
- 3rd Workshop on Answer Set Programming and Other Computing Paradigms (ASPOCP 2010)
- The 16th RCRA International Workshop on Experimental Evaluation of Algorithms for Solving Problems with Combinatorial Explosion (RCRA 2009)
- First International Workshop on Logic-Based Context Interpretation: Modelling and Applications (Log-IC 2009)
- 2nd International Workshop on Software Engineering for Answer Set Programming (SEA'09)
- 10th International Conference on Logic Programming and Non-Monotonic Reasoning (LPNMR'09)
- 21st International Joint Conference on Artificial Intelligence (IJCAI-09)
- 2nd International Workshop on Logic and Search (LaSh 2008)
- Application Session of the 12th International Workshop on Non-Monotonic Reasoning (NMR 2008)
- 11th International Conference on the Principles of Knowledge Representation and Reasoning (KR2008)

- 12th Conference of the Spanish Association for Artificial Intelligence (CAEPIA'07)
- The 14th RCRA Workshop on Experimental Evaluation of Algorithms for Solving Problems with Combinatorial Explosion (RCRA 2007)
- 22nd National Conference on Artificial Intelligence (AAAI-07)
- 1st Workshop on Correspondence and Equivalence for Nonmonotonic Theories (CENT 2007)
- 1st Workshop on Software Engineering for Answer Set Programming (SEA 2007)
- 9th International Conference on Logic Programming and Non-Monotonic Reasoning (LPNMR 2007)
- 17th European Conference on Artificial Intelligence (ECAI-2006)
- Doctoral Consortium of the 22nd International Conference on Logic Programming (ICLP'06)
- Giornata di Lavoro: Analisi sperimentale e benchmark di algoritmi di Intelligenza Artificiale (RCRA'06 Italian Workshop on Experimental analysis and benchmarks for AI algorithms) 2006
- Answer Set Programming, Special Session of the 11th International Workshop on Non-Monotonic Reasoning (NMR 2006)
- 11th Conference of the Spanish Association for Artificial Intelligence (CAEPIA'05)
- INFOMIX Workshop on Data Integration 2005
- 8th International Conference on Logic Programming and Non-Monotonic Reasoning (LPNMR 2005)
- Giornata di Lavoro: Analisi sperimentale e benchmark di algoritmi di Intelligenza Artificiale (RCRA'05 Italian Workshop on Experimental analysis and benchmarks for AI algorithms) 2005
- Computational Aspects of Non-monotonic Reasoning, Subworkshop of the 10th International Workshop on Non-Monotonic Reasoning (NMR 2004)
- 8th European Conference on Logics in Artificial Intelligence (JELIA'02)

8.3 Invited Talks and Tutorials

- "Aggregates in Answer Set Programming," invited talk given at the 23rd Italian Conference on Computational Logic, July 2008
- "Disjunctive Logic Programming: Problem-Solving Techniques, Systems, and Applications," tutorial given at the Joint Conference on Declarative Programming APPIA-GULP-PRODE, September 2003

8.4 Talks (other than conferences and workshops)

- "Manifold Answer-Set Programs and Consequence Modules"
  given at the Comenius University in Bratislava, Slovakia, January 8, 2010
- "Aggregate in Logischen Programmen"
  given at the University of Leipzig, Germany, November 14, 2005
- "Answer Set Based Planning: Planning with Action Languages"
  given at the University of Leipzig, Germany, May 3, 2005
- "Recursive Aggregates in Disjunctive Logic Programs: Semantics, Complexity, and Computation"
  given at the University of Potsdam, Germany, May 4, 2005
- "Modularity, Magic Sets and their Application to Data Integration"
  given at Seminar "Nonmonotonic Reasoning, Answer Set Programming and Constraints" at Schloss Dagstuhl, April 25, 2005
- "Knowledge-based Planning and ASP"
  given at Seminar "Nonmonotonic Reasoning, Answer Set Programming and Constraints" at Schloss Dagstuhl, September 19, 2002
- "Logic-based Planning"
  given at the University of Calabria, Italy, May 31, 2002
- "ASP-based Planning"
  given at the University of Reggio Calabria, Italy, May 27, 2002
- "Answer Set Programming"
  given at the University of Messina, Italy, May 27, 2002

8.5 Peer Reviews

Peer Reviews for Journals and Collections

- ACM Transactions on Computational Logic
- Annals of Mathematics and Artificial Intelligence
- Artificial Intelligence
- Computational Logic: From Logic Programming into the Future (In honour of Bob Kowalski)
- Data & Knowledge Engineering
- IEEE Transactions on Knowledge and Data Engineering
- Information Systems
- Information Fusion
- International Journal of Approximate Reasoning
- Journal of Artificial Intelligence Research

- Journal of Algorithms in Logic, Informatics and Cognition
- Journal of Automated Reasoning
- Journal of Computer and System Sciences
- Theory and Practice of Logic Programming

Peer Reviews for Conferences and Workshops

- 5th Latin American Workshop On Non-Monotonic Reasoning (LANMR 2009)
- 25th International Conference on Logic Programming (ICLP'09)
- The 15th RCRA Workshop on Experimental Evaluation of Algorithms for Solving Problems with Combinatorial Explosion (RCRA 2008)
- 15th International Conference on Logic for Programming Artificial Intelligence and Reasoning (LPAR 2008)
- 11th European Conference on Logics in Artificial Intelligence (JELIA 2008)
- 24th International Conference on Logic Programming (ICLP'08)
- 18th European Conference on Artificial Intelligence (ECAI 2008)
- 23rd National Conference on Artificial Intelligence (AAAI-08)
- 8th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2008)
- 10th International Symposium on Practical Aspects of Declarative Languages (PADL'08)
- 4th International Workshop on Answer Set Programming (ASP'07)
- 30th German Conference on Artificial Intelligence (KI 2007)
- 16th EACSL Annual Conference on Computer Science and Logic (CSL'07)
- 23rd International Conference on Logic Programming (ICLP'07)
- 26th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS 2007)
- 22nd International Conference on Logic Programming (ICLP'06)
- 25th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS 2006)
- International Symposium on Foundations of Information and Knowledge Systems (FoIKS 2005)
- 12th International Conference on Logic for Programming Artificial Intelligence and Reasoning (LPAR 2005)
- 11th Conference of the Spanish Association for Artificial Intelligence (CAEPIA'05)
- Italian Conference on Theoretical Computer Science (ICTCS'05)

- 20th National Conference on Artificial Intelligence (AAAI-05)
- 18th International Joint Conference on Artificial Intelligence (IJCAI 2005)
- 5th International Conference on Knowledge Based Systems (KBCS-04)
- 9th European Conference on Logics in Artificial Intelligence (JELIA'04)
- 16th European Conference on Artificial Intelligence (ECAI 2004)
- 19th National Conference on Artificial Intelligence (AAAI-04)
- 23rd ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS 2004)
- 9th International Conference on the Principles of Knowledge Representation and Reasoning (KR 2004)
- 7th International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR-7)
- 17th International Joint Conference on Artificial Intelligence (IJCAI 2003)
- 19th International Conference on Automated Deduction (CADE-19)
- 18th National Conference on Artificial Intelligence (AAAI 2002)
- STarting Artificial Intelligence Researchers Symposium (STAIRS 2002)
- 9th International Workshop on Non-Monotonic Reasoning (NMR'2002)
- 18th International Conference on Logic Programming (ICLP'01)
- 6th International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'01)
- 17th International Joint Conference on Artificial Intelligence (IJCAI'01)
- First International Joint Conference on Automated Reasoning (IJCAR'01)
- Fourth International Conference on Multi-Agent Systems (ICMAS-2000)
- International Symposium on Foundations of Information and Knowledge Systems (FoIKS 2000)
- First International Conference on Computational Logic (CL2000)
- 7th International Conference on Principles of Knowledge Representation and Reasoning (KR 2000)
- 8th International Workshop on Non-Monotonic Reasoning (NMR'2000)
- 5th International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'99)
- 10th International Conference and Workshop on Database and Expert Systems Applications (DEXA'99)

9 Publications

Edited Books

[1] Thomas Eiter, Wolfgang Faber, and Mirosław Truszczyński, editors. *Logic Programming and Nonmonotonic Reasoning — 6th International Conference, LPNMR'01, Vienna, Austria, September 2001, Proceedings*, number 2173 in Lecture Notes in AI (LNAI). Springer Verlag, 2001.

Chapters in Books and Collections

[2] Wolfgang Faber, Nicola Leone, and Francesco Ricca. Answer set programming. In Benjamin W. Wah, editor, *Wiley Encyclopedia of Computer Science and Engineering*, volume 1, pages 149–162. John Wiley & Sons, Inc., January 2009.

[3] Thomas Eiter, Wolfgang Faber, Gerald Pfeifer, and Axel Polleres. Declarative planning and knowledge representation in an action language. In Ioannis Vlahavas and Dimitris Vrakas, editors, *Intelligent Techniques for Planning*, chapter 1, pages 1–34. Idea Group, Inc., 2005.

[4] Thomas Eiter, Wolfgang Faber, Nicola Leone, and Gerald Pfeifer. Declarative Problem-Solving Using the DLV System. In Jack Minker, editor, *Logic-Based Artificial Intelligence*, pages 79–103. Kluwer Academic Publishers, 2000.

Journal Articles

[5] Mario Alviano, Wolfgang Faber, and Nicola Leone. Disjunctive asp with functions: Decidable queries and effective computation. *Theory and Practice of Logic Programming*, 2010. Accepted for publication.

[6] Wolfgang Faber, Nicola Leone, and Gerald Pfeifer. Semantics and complexity of recursive aggregates in answer set programming. *Artificial Intelligence*, 2010. Accepted for publication.

[7] Annamaria Bria, Wolfgang Faber, and Nicola Leone. Normal form nested programs. *Fundamenta Informaticae*, 96(3):271–295, 2009.

[8] Wolfgang Faber, Gerald Pfeifer, Nicola Leone, Tina Dell'Armi, and Giuseppe Ielpa. Design and implementation of aggregate functions in the dlv system. *Theory and Practice of Logic Programming*, 8(5–6):545–580, 2008.

[9] Marco Maratea, Francesco Ricca, Wolfgang Faber, and Nicola Leone. Look-back techniques and heuristics in dlv: Implementation, evaluation and comparison to qbf solvers. *Journal of Algorithms in Cognition, Informatics and Logics*, 63(1–3):70–89, 2008.

[10] Thomas Eiter, Esra Erdem, and Wolfgang Faber. Undoing the Effects of Action Sequences. *Journal of Applied Logic*, 6(3):380–415, 2008.

[11] Thomas Eiter, Wolfgang Faber, Michael Fink, and Stefan Woltran. Complexity results for answer set programming with bounded predicate arities and implications. *Annals of Mathematics and Artificial Intelligence*, 51(2–4):123–165, 2007.

[12] Wolfgang Faber, Nicola Leone, Gerald Pfeifer, and Francesco Ricca. On look-ahead heuristics in disjunctive logic programming. *Annals of Mathematics and Artificial Intelligence*, 51(2–4):229–266, 2007.

[13] Thomas Eiter, Esra Erdem, Wolfgang Faber, and Ján Senko. A Logic-Based Approach to Finding Explanations for Discrepancies in Optimistic Plan Execution. *Fundamenta Informaticae*, 79(1–2):25–69, 2007.

[14] Wolfgang Faber, Gianluigi Greco, and Nicola Leone. Magic Sets and their Application to Data Integration. *Journal of Computer and System Sciences*, 73(4):584–609, 2007.

[15] Wolfgang Faber and Kathrin Konczak. Strong Order Equivalence. *Annals of Mathematics and Artificial Intelligence*, 47(1–2):43–78, June 2006.

[16] Nicola Leone, Gerald Pfeifer, Wolfgang Faber, Thomas Eiter, Georg Gottlob, Simona Perri, and Francesco Scarcello. The DLV System for Knowledge Representation and Reasoning. *ACM Transactions on Computational Logic*, 7(3):499–562, July 2006.

[17] Francesco Ricca, Wolfgang Faber, and Nicola Leone. A Backjumping Technique for Disjunctive Logic Programming. *AI Communications – The European Journal on Artificial Intelligence*, 19(2):155–172, 2006.

[18] Francesco Calimeri, Wolfgang Faber, Nicola Leone, and Gerald Pfeifer. Pruning Operators for Disjunctive Logic Programming Systems. *Fundamenta Informaticae*, 71(2–3), 2006.

[19] Wolfgang Faber, Nicola Leone, and Francesco Ricca. Solving Hard Problems for the Second Level of the Polynomial Hierarchy: Heuristics and Benchmarks. *Intelligenza Artificiale*, 2(3):21–28, September 2005.

[20] Thomas Eiter, Wolfgang Faber, Nicola Leone, Gerald Pfeifer, and Axel Polleres. A Logic Programming Approach to Knowledge-State Planning: Semantics and Complexity. *ACM Transactions on Computational Logic*, 5(2):206–263, April 2004.

[21] Thomas Eiter, Wolfgang Faber, Nicola Leone, and Gerald Pfeifer. Computing Preferred Answer Sets by Meta-Interpretation in Answer Set Programming. *Theory and Practice of Logic Programming*, 3:463–498, July/September 2003.

[22] Thomas Eiter, Wolfgang Faber, Nicola Leone, Gerald Pfeifer, and Axel Polleres. Answer Set Planning under Action Costs. *Journal of Artificial Intelligence Research*, 19:25–71, 2003.

[23] Thomas Eiter, Wolfgang Faber, Nicola Leone, Gerald Pfeifer, and Axel Polleres. A Logic Programming Approach to Knowledge-State Planning, II: the DLV^K System. *Artificial Intelligence*, 144(1–2):157–211, March 2003.

[24] Francesco Buccafurri, Wolfgang Faber, and Nicola Leone. Disjunctive Logic Programs with Inheritance. *Theory and Practice of Logic Programming*, 2(3), May 2002.

[25] Thomas Eiter, Wolfgang Faber, Nicola Leone, and Gerald Pfeifer. The Diagnosis Frontend of the `dlv` System. *AI Communications – The European Journal on Artificial Intelligence*, 12(1–2):99–111, 1999.

Articles in Refereed Conference and Workshop Proceedings

[26] Mario Alviano and Wolfgang Faber. Dynamic magic sets for super-consistent answer set programs. In *3rd Workshop on Answer Set Programming and Other Computing Paradigms (ASPOCP 2010)*, July 2010. Accepted for publication.

[27] Thomas Eiter, Wolfgang Faber, and Mushthofa Mushthofa. Space efficient evaluation of asp programs with bounded predicate arities. In *Twenty-Fourth AAAI Conference on Artificial Intelligence (AAAI-10)*, July 2010. Accepted for publication.

[28] Mario Alviano, Wolfgang Faber, and Nicola Leone. Well-founded semantics for logic programs with aggregates: Implementation and experimentation. In *17th RCRA International Workshop on Experimental Evaluation of Algorithms for Solving Problems with Combinatorial Explosion (RCRA 2010)*, June 2010.

[29] Wolfgang Faber and Stefan Woltran. Manifold answer-set programs for meta-reasoning. In Esra Erdem, Fangzhen Lin, and Torsten Schaub, editors, *Logic Programming and Nonmonotonic Reasoning — 10th International Conference (LPNMR 2009)*, volume 5753 of *Lecture Notes in Computer Science*, pages 115–128. Springer Verlag, September 2009.

[30] Annamaria Bria, Wolfgang Faber, and Nicola Leone. nfn2dlp and nfnsolve: Normal form nested programs compiler and solver. In Esra Erdem, Fangzhen Lin, and Torsten Schaub, editors, *Logic Programming and Nonmonotonic Reasoning — 10th International Conference (LPNMR 2009)*, volume 5753 of *Lecture Notes in Computer Science*, pages 477–482. Springer Verlag, September 2009.

[31] Wolfgang Faber and Stefan Woltran. Manifold answer-set programs for meta-reasoning. In *The IJCAI-09 Workshop on Nonmonotonic Reasoning, Action and Change*, July 2009.

[32] Annamaria Bria, Wolfgang Faber, and Nicola Leone. nfn2dlp: A compiler for normal form nested programs. In Marco Gavanelli and Fabrizio Riguzzi, editors, *Proceedings of the 24th Italian Conference on Computational Logic (CILC 2009)*, June 2009.

[33] Wolfgang Faber, Nicola Leone, Marco Maratea, and Francesco Ricca. Look-back techniques for ASP programs with aggregates. In Marco Gavanelli and Toni Mancini, editors, *The 15th RCRA Workshop on Experimental Evaluation of Algorithms for Solving Problems with Combinatorial Explosion (RCRA 2008)*, December 2008.

[34] Nicola Leone and Wolfgang Faber. The DLV project: A tour from theory and research to applications and market. In Maria Garcia de la Banda and Enrico Pontelli, editors, *Proceedings of the 24th International Conference on Logic Programming (ICLP'08)*, volume 5366 of *Lecture Notes in Computer Science*, pages 53–68. Springer Verlag, December 2008.

[35] Wolfgang Faber, Nicola Leone, Marco Maratea, and Francesco Ricca. Backjumping and look-back heuristics for disjunctive logic programming with aggregates. In *Proceedings of the 2nd International Workshop on Logic and Search (LaSh'08)*, November 2008.

[36] Annamaria Bria, Wolfgang Faber, and Nicola Leone. Normal form nested programs. In Steffen Hölldobler, Carsten Lutz, and Heinrich Wansing, editors, *Proceedings of the 11th European Conference on Logics in Artificial Intelligence (JELIA 2008)*, volume 5293 of *Lecture Notes in Computer Science*, pages 76–88. Springer Verlag, September 2008.

[37] Wolfgang Faber, Hans Tompits, and Stefan Woltran. Notions of Strong Equivalence for Logic Programs with Ordered Disjunction. In Gerhard Brewka and Jérôme Lang, editors, *Proceedings of the 11th International Conference on Principles of Knowledge*

Representation and Reasoning (KR 2008), pages 433–443. AAAI Press, September 2008.

[38] Wolfgang Faber, Gianluigi Greco, and Nicola Leone. Magic Sets for Data Integration. In Dieter Fox and Carla P. Gomes, editors, *Proceedings of the 23rd AAAI Conference on Artificial Intelligence (AAAI 2008)*, pages 1528–1531. AAAI Press, July 2008.

[39] Mario Alviano, Wolfgang Faber, and Nicola Leone. Compiling minimum and maximum aggregates into standard ASP. In Andrea Formisano, editor, *Proceedings of the 23rd Italian Conference on Computational Logic (CILC 2008)*, July 2008.

[40] Annamaria Bria, Wolfgang Faber, and Nicola Leone. Normal form nested programs. In Andrea Formisano, editor, *Proceedings of the 23rd Italian Conference on Computational Logic (CILC 2008)*, July 2008.

[41] Wolfgang Faber, Hans Tompits, and Stefan Woltran. Characterizing Notions of Strong Equivalence for Logic Programs with Ordered Disjunctions. In *Proceedings of the 3rd Multidisciplinary Workshop on Advances in Preference Handling*, September 2007.

[42] Wolfgang Faber, Nicola Leone, Marco Maratea, and Francesco Ricca. Looking back in dlv: Experiments and comparison to qbf solvers. In Stefania Costantini and Richard Watson, editors, *Proceedings of the 4th International Workshop on Answer Set Programming (ASP 2007)*, September 2007.

[43] Wolfgang Faber, Nicola Leone, Marco Maratea, and Francesco Ricca. Look-back techniques and heuristics in dlv: Implementation and evaluation. In Marco Gavanelli and Toni Mancini, editors, *The 14th RCRA Workshop on Experimental Evaluation of Algorithms for Solving Problems with Combinatorial Explosion*, July 2007.

[44] Mario Alviano, Wolfgang Faber, and Nicola Leone. Using unfounded sets for computing answer sets of programs with recursive aggregates. In Giacomo Fiumara, Massimo Marchi, and Alessandro Provetti, editors, *Proceedings of the Italian Conference on Computational Logic (CILC 2007)*, June 2007.

[45] Wolfgang Faber, Nicola Leone, Marco Maratea, and Francesco Ricca. Implementation and evaluation of look-back techniques and heuristics in dlv. In Giacomo Fiumara, Massimo Marchi, and Alessandro Provetti, editors, *Proceedings of the Italian Conference on Computational Logic (CILC 2007)*, June 2007.

[46] Wolfgang Faber, Nicola Leone, Marco Maratea, and Francesco Ricca. Experimenting with Look-Back Heuristics for Hard ASP Programs. In Chitta Baral, Gerhard Brewka, and John Schlipf, editors, *Logic Programming and Nonmonotonic Reasoning — 9th International Conference, LPNMR 2007, Tempe, AZ, USA, May 2007, Proceedings*, volume 4483 of *Lecture Notes in AI (LNAI)*, pages 110–122. Springer Verlag, May 2007.

[47] Wolfgang Faber and Nicola Leone. On the Complexity of Answer Set Programming with Aggregates. In Chitta Baral, Gerhard Brewka, and John Schlipf, editors, *Logic Programming and Nonmonotonic Reasoning — 9th International Conference, LPNMR 2007, Tempe, AZ, USA, May 2007, Proceedings*, volume 4483 of *Lecture Notes in AI (LNAI)*, pages 97–109. Springer Verlag, May 2007.

[48] Thomas Eiter, Esra Erdem, and Wolfgang Faber. On reversing actions: Algorithms and complexity. In *Twentieth International Joint Conference on Artificial Intelligence (IJCAI-07)*, pages 336–341, January 2007.

[49] Wolfgang Faber, Nicola Leone, Marco Maratea, and Francesco Ricca. Evaluating backjumping for disjunctive logic programming. In Marco Gavanelli and Toni Mancini, editors, *Atti della Giornata di Lavoro: Analisi sperimentale e benchmark di algoritmi per l'Intelligenza Artificiale*, 2006.

[50] Wolfgang Faber. Decomposition of Nonmonotone Aggregates in Logic Programming. In Michael Fink, Hans Tompits, and Stefan Woltran, editors, *Proceedings of the 20th Workshop on Logic Programming (WLP 2006)*, pages 164–171, Vienna, Austria, February 2006.

[51] Jürgen Dix, Wolfgang Faber, and V.S. Subrahmanian. The Relationship between Reasoning about Privacy and Default Logics. In Geoff Sutcliffe and Andrei Voronkov, editors, *Logic for Programming, Artificial Intelligence, and Reasoning, 12th International Conference, LPAR 2005*, volume 3835 of *Lecture Notes in Computer Science*, pages 637–650. Springer Verlag, December 2005.

[52] Wolfgang Faber. Unfounded Sets for Disjunctive Logic Programs with Arbitrary Aggregates. In Chitta Baral, Gianluigi Greco, Nicola Leone, and Giorgio Terracina, editors, *Logic Programming and Nonmonotonic Reasoning — 8th International Conference, LPNMR'05, Diamante, Italy, September 2005, Proceedings*, volume 3662 of *Lecture Notes in Computer Science*, pages 40–52. Springer Verlag, September 2005.

[53] Wolfgang Faber and Francesco Ricca. Solving Hard ASP Programs Efficiently. In Chitta Baral, Gianluigi Greco, Nicola Leone, and Giorgio Terracina, editors, *Logic Programming and Nonmonotonic Reasoning — 8th International Conference, LPNMR'05, Diamante, Italy, September 2005, Proceedings*, volume 3662 of *Lecture Notes in Computer Science*, pages 240–252. Springer Verlag, September 2005.

[54] Nicola Leone, Thomas Eiter, Wolfgang Faber, Michael Fink, Georg Gottlob, Luigi Granata, Gianluigi Greco, Edyta Kałka, Giovambattista Ianni, Domenico Lembo, Maurizio Lenzerini, Vincenzino Lio, Bartosz Nowicki, Riccardo Rosati, Marco Ruzzi, Witold Staniszkis, and Giorgio Terracina. Data Integration: A Challenging ASP Application. In Chitta Baral, Gianluigi Greco, Nicola Leone, and Giorgio Terracina, editors, *Logic Programming and Nonmonotonic Reasoning — 8th International Conference, LPNMR'05, Diamante, Italy, September 2005, Proceedings*, volume 3662 of *Lecture Notes in Computer Science*, pages 379–383. Springer Verlag, September 2005.

[55] Thomas Eiter, Wolfgang Faber, and Patrick Traxler. Testing Strong Equivalence of Datalog Programs – Implementation and Examples. In Chitta Baral, Gianluigi Greco, Nicola Leone, and Giorgio Terracina, editors, *Logic Programming and Nonmonotonic Reasoning — 8th International Conference, LPNMR'05, Diamante, Italy, September 2005, Proceedings*, volume 3662 of *Lecture Notes in Computer Science*, pages 437–441. Springer Verlag, September 2005.

[56] Wolfgang Faber, Nicola Leone, and Francesco Ricca. Heuristics for Hard ASP Programs. In *Nineteenth International Joint Conference on Artificial Intelligence (IJCAI-05)*, pages 1562–1563, August 2005.

[57] Wolfgang Faber and Kathrin Konczak. Strong Equivalence for Logic Programs with Preferences. In *Nineteenth International Joint Conference on Artificial Intelligence (IJCAI-05)*, pages 430–435, August 2005.

[58] Francesco Calimeri, Wolfgang Faber, Nicola Leone, and Simona Perri. Declarative and Computational Properties of Logic Programs with Aggregates. In *Nineteenth*

International Joint Conference on Artificial Intelligence (IJCAI-05), pages 406–411, August 2005.

[59] Wolfgang Faber, Nicola Leone, and Francesco Ricca. A Backjumping Technique for Disjunctive Logic Programming. In Marina De Vos and Alessandro Provetti, editors, *Answer Set Programming: Advances in Theory and Implementation (ASP'05)*, pages 216–230. Research Press International, July 2005.

[60] Wolfgang Faber, Gianluigi Greco, and Nicola Leone. Magic Sets and their Application to Data Integration. In *Proceedings of the Thirteenth Italian Symposium on Advanced Database Systems (SEBD-2005)*, pages 47–54, June 2005.

[61] Nicola Leone, Thomas Eiter, Wolfgang Faber, Michael Fink, Georg Gottlob, Gianluigi Greco, Edyta Kałka, Giovambattista Ianni, Domenico Lembo, Vincenzino Lio, Bartosz Nowicki, Riccardo Rosati, Marco Ruzzi, Witold Staniszkis, and Giorgio Terracina. Boosting Information Integration: The INFOMIX System. In *Proceedings of the Thirteenth Italian Symposium on Advanced Database Systems (SEBD-2005)*, pages 55–66, June 2005.

[62] Nicola Leone, Georg Gottlob, Riccardo Rosati, Thomas Eiter, Wolfgang Faber, Michael Fink, Gianluigi Greco, Giovambattista Ianni, Edyta Kałka, Domenico Lembo, Maurizio Lenzerini, Vincenzino Lio, Bartosz Nowicki, Marco Ruzzi, Witold Staniszkis, and Giorgio Terracina. The INFOMIX System for Advanced Integration of Incomplete and Inconsistent Data. In *Proceedings of the 24th ACM SIGMOD International Conference on Management of Data (SIGMOD 2005)*, pages 915–917, Baltimore, Maryland, USA, June 2005. ACM Press.

[63] Wolfgang Faber, Nicola Leone, and Francesco Ricca. Solving problems on the second level of the polynomial hierarchy. In Marco Cadoli, Marco Gavanelli, and Toni Mancini, editors, *Atti della Giornata di Lavoro: Analisi sperimentale e benchmark di algoritmi per l'Intelligenza Artificiale*, number CS-2005-03 in Computer Science Group Technical Reports, pages 115–116, Dipartimento di Ingegneria, Universita' di Ferrara, Italy, June 10 2005.

[64] Wolfgang Faber. A tool for benchmarking command-line systems. In Marco Cadoli, Marco Gavanelli, and Toni Mancini, editors, *Atti della Giornata di Lavoro: Analisi sperimentale e benchmark di algoritmi per l'Intelligenza Artificiale*, number CS-2005-03 in Computer Science Group Technical Reports, pages 113–114, Dipartimento di Ingegneria, Universita' di Ferrara, Italy, June 10 2005.

[65] Wolfgang Faber, Gianluigi Greco, and Nicola Leone. Magic sets and their application to data integration. In Thomas Eiter and Leonid Libkin, editors, *Proceedings of the Tenth International Conference on Database Theory (ICDT'05)*, volume 3363 of *Lecture Notes in Computer Science*, pages 306–320. Springer, January 2005.

[66] Wolfgang Faber, Nicola Leone, and Gerald Pfeifer. Recursive aggregates in disjunctive logic programs: Semantics and complexity. In José Júlio Alferes and João Leite, editors, *Proceedings of the 9th European Conference on Logics in Artificial Intelligence (JELIA 2004)*, number 3229 in Lecture Notes in AI (LNAI), pages 200–212. Springer Verlag, September 2004.

[67] Francesco Calimeri, Manuela Citrigno, Chiara Cumbo, Wolfgang Faber, Nicola Leone, Simona Perri, and Gerald Pfeifer. New dlv features for data integration. In José Júlio Alferes and João Leite, editors, *Proceedings of the 9th European Conference on Logics*

in Artificial Intelligence (JELIA 2004), number 3229 in Lecture Notes in AI (LNAI), pages 698–701. Springer Verlag, September 2004.

[68] Chiara Cumbo, Wolfgang Faber, Gianluigi Greco, and Nicola Leone. Enhancing the magic-set method for disjunctive datalog programs. In Bart Demoen and Vladimir Lifschitz, editors, *Proceedings of the 20th International Conference on Logic Programming (ICLP'04)*, number 3132 in Lecture Notes in Computer Science, pages 371–385. Springer Verlag, September 2004.

[69] Manuela Citrigno, Wolfgang Faber, Gianluigi Greco, and Nicola Leone. Efficient evaluation of disjunctive datalog queries with aggregate functions. In Elio Panegai and Gianfranco Rossi, editors, *Proceedings of the Italian Conference on Computational Logic (CILC 2004)*, pages 148–162, June 2004.

[70] Thomas Eiter, Esra Erdem, and Wolfgang Faber. Plan Reversals for Recovery in Execution Monitoring. In James Delgrande and Torsten Schaub, editors, *Proceedings of the 10th International Workshop on Non-Monotonic Reasoning (NMR 2004)*, pages 147–154, June 2004.

[71] Thomas Eiter, Wolfgang Faber, Michael Fink, Gerald Pfeifer, and Stefan Woltran. Complexity of Model Checking and Bounded Predicate Arities for Non-ground Answer Set Programming. In Didier Dubois, Christopher Welty, and Mary-Anne Williams, editors, *Proceedings Ninth International Conference on Principles of Knowledge Representation and Reasoning (KR 2004), June 2-5, Whistler, British Columbia, Canada*, pages 377–387. AAAI Press, June 2004.

[72] Tina Dell'Armi, Wolfgang Faber, Giuseppe Ielpa, Nicola Leone, and Gerald Pfeifer. System Description: DLV with Aggregates. In Vladimir Lifschitz and Ilkka Niemelä, editors, *Proceedings of the 7th International Conference on Logic Programming and Non-Monotonic Reasoning (LPNMR-7)*, LNCS, pages 326–330. Springer, January 2004.

[73] Tina Dell'Armi, Wolfgang Faber, Giuseppe Ielpa, Nicola Leone, and Gerald Pfeifer. Aggregate Functions in DLV. In Marina de Vos and Alessandro Provetti, editors, *Proceedings ASP03 - Answer Set Programming: Advances in Theory and Implementation*, pages 274–288, Messina, Italy, September 2003. Online at `http://CEUR-WS.org/Vol-78/`.

[74] Thomas Eiter, Wolfgang Faber, Michael Fink, Gerald Pfeifer, and Stefan Woltran. Complexity of Answer Set Checking and Bounded Predicate Arities for Non-ground Answer Set Programming. In Marina de Vos and Alessandro Provetti, editors, *Proceedings ASP03 - Answer Set Programming: Advances in Theory and Implementation*, pages 69–83, Messina, Italy, September 2003. Online at `http://CEUR-WS.org/Vol-78/`.

[75] Tina Dell'Armi, Wolfgang Faber, Giuseppe Ielpa, Nicola Leone, and Gerald Pfeifer. Aggregate Functions in Disjunctive Logic Programming: Semantics, Complexity, and Implementation in DLV. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI) 2003*, pages 847–852, Acapulco, Mexico, August 2003. Morgan Kaufmann Publishers.

[76] Thomas Eiter, Wolfgang Faber, Nicola Leone, Gerald Pfeifer, and Axel Polleres. Answer Set Planning under Action Costs. In Sergio Flesca, Sergio Greco, Giovambattista

Ianni, and Nicola Leone, editors, *Proceedings of the 8th European Conference on Logics in Artificial Intelligence (JELIA)*, number 2424 in Lecture Notes in Computer Science, pages 186–197, Cosenza, Italy, September 2002.

[77] Thomas Eiter, Wolfgang Faber, Nicola Leone, Gerald Pfeifer, and Axel Polleres. The DLV^K Planning System: Progress Report. In Sergio Flesca, Sergio Greco, Giovambattista Ianni, and Nicola Leone, editors, *Proceedings of the 8th European Conference on Logics in Artificial Intelligence (JELIA)*, number 2424 in Lecture Notes in Computer Science, pages 541–544, Cosenza, Italy, September 2002. (System Description).

[78] Nicola Leone, Gerald Pfeifer, Wolfgang Faber, Francesco Calimeri, Tina Dell'Armi, Thomas Eiter, Georg Gottlob, Giovambattista Ianni, Giuseppe Ielpa, Christoph Koch, Simona Perri, and Axel Polleres. The DLV System. In Sergio Flesca, Sergio Greco, Giovambattista Ianni, and Nicola Leone, editors, *Proceedings of the 8th European Conference on Logics in Artificial Intelligence (JELIA)*, number 2424 in Lecture Notes in Computer Science, pages 537–540, Cosenza, Italy, September 2002. (System Description).

[79] Francesco Calimeri, Wolfgang Faber, Nicola Leone, and Gerald Pfeifer. Pruning Operators for Answer Set Programming Systems. In *Proceedings of the 9th International Workshop on Non-Monotonic Reasoning (NMR'2002)*, pages 200–209, April 2002.

[80] Francesco Calimeri, Wolfgang Faber, Nicola Leone, Simona Perri, and Gerald Pfeifer. DLV - Declarative Problem Solving using Answer Set Programming. In *Proceedings of the Seventh Congress of the Italian Association for Artificial Intelligence AI*IA 2001*, Bari, Italy, 2001.

[81] Wolfgang Faber, Nicola Leone, and Gerald Pfeifer. Optimizing the Computation of Heuristics for Answer Set Programming Systems. In Thomas Eiter, Wolfgang Faber, and Mirosław Truszczyński, editors, *Logic Programming and Nonmonotonic Reasoning — 6th International Conference, LPNMR'01, Vienna, Austria, September 2001, Proceedings*, number 2173 in Lecture Notes in AI (LNAI), pages 288–301. Springer Verlag, September 2001.

[82] Tina Dell'Armi, Wolfgang Faber, Giuseppe Ielpa, Christoph Koch, Nicola Leone, Simona Perri, and Gerald Pfeifer. System Description: `DLV`. In Thomas Eiter, Wolfgang Faber, and Mirosław Truszczyński, editors, *Logic Programming and Nonmonotonic Reasoning — 6th International Conference, LPNMR'01, Vienna, Austria, September 2001, Proceedings*, number 2173 in Lecture Notes in AI (LNAI), pages 409–412. Springer Verlag, September 2001.

[83] Thomas Eiter, Wolfgang Faber, Nicola Leone, Gerald Pfeifer, and Axel Polleres. System Description: The DLV^K Planning System. In Thomas Eiter, Wolfgang Faber, and Mirosław Truszczyński, editors, *Logic Programming and Nonmonotonic Reasoning — 6th International Conference, LPNMR'01, Vienna, Austria, September 2001, Proceedings*, number 2173 in Lecture Notes in AI (LNAI), pages 413–416. Springer Verlag, September 2001.

[84] Wolfgang Faber, Nicola Leone, and Gerald Pfeifer. Experimenting with Heuristics for Answer Set Programming. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI) 2001*, pages 635–640, Seattle, WA, USA, August 2001. Morgan Kaufmann Publishers.

[85] Thomas Eiter, Wolfgang Faber, Nicola Leone, Gerald Pfeifer, and Axel Polleres. The DLV^K Planning System. In Alessandro Cimatti, Héctor Geffner, Enrico Giunchiglia, and Jussi Rintanen, editors, *IJCAI-01 Workshop on Planning under Uncertainty and Incomplete Information*, pages 76–81, August 2001.

[86] Wolfgang Faber, Nicola Leone, and Gerald Pfeifer. A Comparison of Heuristics for Answer Set Programming. In *Proceedings of the 5th Dutch-German Workshop on Nonmonotonic Reasoning Techniques and their Applications (DGNMR 2001)*, pages 64–75, April 2001.

[87] Thomas Eiter, Wolfgang Faber, Nicola Leone, and Gerald Pfeifer. Computing Preferred and Weakly Preferred Answer Sets by Meta-Interpretation in Answer Set Programming. In Alessandro Provetti and Son Tran Cao, editors, *Proceedings AAAI 2001 Spring Symposium on Answer Set Programming: Towards Efficient and Scalable Knowledge Representation and Reasoning*, pages 45–52, Stanford, California, USA, March 2001. AAAI Press.

[88] Thomas Eiter, Wolfgang Faber, Nicola Leone, Gerald Pfeifer, and Axel Polleres. Planning under Incomplete Knowledge. In John Lloyd, Veronica Dahl, Ulrich Furbach, Manfred Kerber, Kung-Kiu Lau, Catuscia Palamidessi, Luís Moniz Pereira, Yehoshua Sagiv, and Peter J. Stuckey, editors, *Computational Logic - CL 2000, First International Conference, Proceedings*, number 1861 in Lecture Notes in AI (LNAI), pages 807–821, London, UK, July 2000. Springer Verlag.

[89] Thomas Eiter, Wolfgang Faber, Christoph Koch, Nicola Leone, and Gerald Pfeifer. DLV – A System for Declarative Problem Solving. In Chitta Baral and Mirosław Truszczyński, editors, *Proceedings of the 8th International Workshop on Non-Monotonic Reasoning (NMR'2000)*, Breckenridge, Colorado, USA, April 2000.

[90] Thomas Eiter, Wolfgang Faber, Nicola Leone, Gerald Pfeifer, and Axel Polleres. Using the `dlv` System for Planning and Diagnostic Reasoning. In François Bry, Ulrich Geske, and Dietmar Seipel, editors, *Proceedings of the 14th Workshop on Logic Programming (WLP'99)*, pages 125–134. GMD – Forschungszentrum Informationstechnik GmbH, Berlin, January 2000. ISSN 1435-2702.

[91] Wolfgang Faber, Nicola Leone, and Gerald Pfeifer. Pushing Goal Derivation in DLP Computations. In Michael Gelfond, Nicola Leone, and Gerald Pfeifer, editors, *Proceedings of the 5th International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'99)*, number 1730 in Lecture Notes in AI (LNAI), pages 177–191, El Paso, Texas, USA, December 1999. Springer Verlag.

[92] Francesco Buccafurri, Wolfgang Faber, and Nicola Leone. Disjunctive Logic Programs with Inheritance. In Danny De Schreye, editor, *Proceedings of the 16th International Conference on Logic Programming (ICLP'99)*, pages 79–93, Las Cruces, New Mexico, USA, November 1999. The MIT Press.

[93] Wolfgang Faber, Nicola Leone, and Gerald Pfeifer. dlv: a DLP System for AI. In Evelina Lamma and Paola Mello, editors, *Sixth Congress of the Italian Association for Artificial Intelligence (AIIA '99)*, pages 511–514, Bologna, Italy, September 1999. Pitagora Editrice Bologna.

[94] Wolfgang Faber, Nicola Leone, Cristinel Mateis, and Gerald Pfeifer. Using Database Optimization Techniques for Nonmonotonic Reasoning. In INAP Organizing Committee, editor, *Proceedings of the 7th International Workshop on Deductive Databases*

and Logic Programming (DDLP'99), pages 135–139. Prolog Association of Japan, September 1999.

[95] Thomas Eiter, Wolfgang Faber, Georg Gottlob, Christoph Koch, Nicola Leone, Cristinel Mateis, Gerald Pfeifer, and F. Scarcello. The DLV System. In Jack Minker, editor, *Workshop on Logic-Based Artificial Intelligence, Washington, DC*, College Park, Maryland, June 1999. Computer Science Department, University of Maryland. Workshop Notes.

[96] Wolfgang Faber, Nicola Leone, and Gerald Pfeifer. Representing School Timetabling in a Disjunctive Logic Programming Language. In Uwe Egly and Hans Tompits, editors, *Proceedings of the 13th Workshop on Logic Programming (WLP'98)*, pages 43–52, Vienna, Austria, October 1998.

[97] Robert Bihlmeyer, Wolfgang Faber, Christoph Koch, Nicola Leone, Cristinel Mateis, and Gerald Pfeifer. `dlv` – An Overview. In Uwe Egly and Hans Tompits, editors, *Proceedings of the 13th Workshop on Logic Programming (WLP'98)*, pages 65–67, Vienna, Austria, October 1998.

[98] Simona Citrigno, Thomas Eiter, Wolfgang Faber, Georg Gottlob, Christoph Koch, Nicola Leone, Cristinel Mateis, Gerald Pfeifer, and Francesco Scarcello. The `dlv` System: Model Generator and Application Frontends. In F. Bry, B. Freitag, and D. Seipel, editors, *Proceedings of the 12th Workshop on Logic Programming (WLP'97), Research Report PMS-FB10*, pages 128–137, München, Germany, September 1997. LMU München.

Theses

[99] Wolfgang Faber. Contributions to Extensions of Answer Set Programming, 2005. Cumulative habilitation thesis, Technische Universität Wien.

[100] Wolfgang Faber. *Enhancing Efficiency and Expressiveness in Answer Set Programming Systems*. PhD thesis, Institut für Informationssysteme, Technische Universität Wien, 2002.

[101] Wolfgang Faber. Disjunctive Datalog with Strong and Weak Constraints: Representational and Computational Issues. Master's thesis, Institut für Informationssysteme, Technische Universität Wien, 1998.

Reports

[102] Thomas Eiter, Esra Erdem, and Wolfgang Faber. Undoing the Effects of Action Sequences. Technical Report INFSYS RR-1843-04-05, Institut für Informationssysteme, Technische Universität Wien, December 2004.

[103] Thomas Eiter, Esra Erdem, and Wolfgang Faber. Diagnosing Plan Execution Discrepancies in a Logic-Based Action Framework. Technical Report INFSYS RR-1843-04-03, Institut für Informationssysteme, Technische Universität Wien, August 2004.

[104] Wolfgang Faber, Nicola Leone, and Gerald Pfeifer. Computing Consistent Preferred Answer Sets. Technical Report DBAI-TR-99-28a, Institut für Informationssysteme, Technische Universität Wien, Austria, April 1999.

Other Publications

[105] Nicola Leone, Wolfgang Faber, Annamaria Bria, Francesco Calimeri, Gelsomina Catalano, Susanna Cozza, Tina Dell'Armi, Gianluigi Greco, Giovambattista Ianni, Giuseppe Ielpa, Marco Maratea, Claudio Panetta, Simona Perri, Francesco Ricca, Francesco Scarcello, Giorgio Terracina, Gerald Pfeifer, Thomas Eiter, and Georg Gottlob. DLV: An advanced system for knowledge representation and reasoning. *ALP Newsletter*, 20(3/4), 2007.

[106] Martin Brain, Wolfgang Faber, Marco Maratea, Axel Polleres, Torsten Schaub, and Roman Schindlauer. What should an asp solver output? a multiple position paper. In Marina De Vos and Torsten Schaub, editors, *Proceedings of the 1st International Workshop on Software Engineering for Answer Set Programming (SEA'07)*, May 2007.

[107] Wolfgang Faber. Report on the Dagstuhl Seminar on Nonmonotonic Reasoning, Answer Set Programming and Constraints. *ALP Newsletter*, 18(4), 2005.

[108] Wolfgang Faber and Stefan Woltran. KR 2004 — Konferenzbericht. *GAI Journal*, 23(2):24–27, 2004.

[109] Wolfgang Faber and Hans Tompits. WLP'98 — Bericht ber den Workshop. *GAI Journal*, 17(4):26–27, 1998.

Web Pages

[110] Wolfgang Faber and Gerald Pfeifer. `DLV` homepage, since 1996. `http://www.dlvsystem.com/`.

[111] Wolfgang Faber. `DLVi` homepage, 1999. `http://www.dlvsystem.com/inheritance/`.