

# The Disjunctive Datalog System DLV: from Research to Industrial Applications

Nicola Leone

Department of Mathematics

University of Calabria

[leone@unical.it](mailto:leone@unical.it)

Joint work with

W. Faber, G. Pfeifer, T. Eiter, G. Gottlob, G. Grasso, ....

and about 20 other contributors

# Roadmap

- Some Historical Notes
- Main features of the System
- A Flavour of the DLV Language
  - Disjunctive Datalog with Stable Model Semantics
  - Aggregates, Weak Constraints, Functions and Lists
- DLV<sup>DB</sup> for the execution “on DBMS”
- Spin-off Companies
- Industry-level Applications
- Conclusions

# **Some Historical Notes**

# DLV History (1)

**1992** Algorithm for computation of the well-founded model of V-free Datalog programs

- [\_, Rullo, Information Systems, 1992]

**1993** Algorithm for computation of the stable models of V-free Datalog programs

- [\_, Romeo, Rullo, Sacca`, Logidata 93]

**1995** Algorithm for computation of the stable models of general disjunctive Datalog programs

- [\_, Rullo, Scarcello, ILPS 95][\_, Rullo, Scarcello, Inf&Comp 97]

**1996** (november) DLV project starts in Vienna

# DLV History (2)

1997 (july, lpmr) 1st Release of DLV

- First implementation of a Stable Model System for Disjunctive Datalog
- 3COL: Smodels 0.27 secs, DLV killed after 2 hours!

1998 (june, KR) 5th Release of DLV

- Competitive with Smodels and DeRes
- More efficient on Deductive Database Applications

1998 --> 2007 29 Releases

- A lot of improvements in all modules
- Many linguistic extensions (aggregates, weak constraints,....)

# DLV History (3)

2007 (Ipnmr) DLV wins in the ASP System Competition

- 1st in the Disjunctive Category
- 1st in the MGS (“Royal”) Category
- DLV widely used in academy

2010 DLV exploited also in industry

- DLV-based products for Knowledge Management
- Commercial Applications
- 2 Spin-off Companies

# **Main Features of the DLV System**

# Advanced Knowledge Modeling Capabilities (1)

- Language:
  - Disjunctive Datalog with Stable Model Semantics
  - Extension with aggregates, weak constraints, functions, lists, sets...
- High Expressiveness:
  - Captures  $\Sigma^P_2$  ( $\text{NP}^{\text{NP}}$ )
  - Able to represent complex problems not (polynomially) translatable to SAT or CSP

# Advanced Knowledge Modeling Capabilities (2)

## ➤ Full Declarativeness:

- Rules ordering and goals ordering is immaterial
- Computation is sound and complete
- Termination is always guaranteed

## ➤ Front ends for AI applications

- Planning
- Diagnosis
- Ontology representation and reasoning
- ...

# Solid Implementation

- Database Optimization Techniques
  - Join Ordering Methods
  - Magic Sets
  - Indexing
- Search Optimization Methods:
  - Heuristics
  - Backjumping
  - Pruning Operators
- Based on an in-depth complexity analysis

# Interoperability

- Relational DBMSs
  - Powerful reasoning on top of data stored in databases
- Semantic Web Reasoners
  - Integrate ontologies and rules
- C++ programs
  - Call (application specific) C++ functions from DLP programs
- JAVA programs
  - Call DLV from JAVA programs

# **A Flavour of DLV Language**

# **Disjunctive Datalog with Stable Model Semantics**

# Disjunctive Datalog Programs

Rule:  $a_1 \vee \dots \vee a_n \text{ :- } b_1, \dots, b_k, \text{ not } b_{k+1}, \dots, \text{ not } b_m$

Constraint:  $\text{ :- } b_1, \dots, b_k, \text{ not } b_{k+1}, \dots, \text{ not } b_m$

Program: A finite Set  $P$  of rules and constraints.

- $a_i$   $b_i$  are atoms
- variables are allowed in atoms' arguments

$\text{mother}(P,S) \vee \text{father}(P,S) \text{ :- } \text{parent}(P,S).$

A program with variables is a shorthand for its ground instantiation

# Formal Semantics for Positive Programs

Suppose programs are ground (variable-free) and Positive (not - free)

**Interpretation I of P:** set of atoms of P.

- Atom **q** is true w.r.t. I if **q** is in I; otherwise it is false.
- Literal **not q** is true w.r.t. I if **q** is not in I; otherwise it is false.

**Model:** Interpretation satisfying all rules of P.

**Stable Model:** Minimal model of P (w.r.t. set inclusion).

# Semantics for Programs with Negation

Consider general programs (with NOT)

**Reduct  $P^I$**  of program  $P$  w.r.t. interpretation  $I$ :

- Delete all rules with a false literal (w.r.t.  $I$ ) in the body;

**Stable Model** of  $P$ : interpretation  $I$  such that  
 $I$  is a minimal model of the reduct  $P^I$ .

Note the novel definition of reduct [Faber, Leone, Pfeifer '04]

- yields precisely the same stable models as [Gelfond&Lifschitz '91]
- Simpler: Only one condition, rules not altered
- Uniform treatment of positive and negative literals
- Works fine with aggregates and other kind of literals



# Some Linguistic Extensions

- Weak Constraints
- Aggregate Literals
- Functions, Sets, and Lists

# Weak Constraints: *Exams Scheduling*

**Input:** a number of courses (`course(X)`),  
the number of common students in each pair of courses (`commonStudents(X,Y,N)`)

**Problem:** Assign course exams to 3 time slots avoiding overlapping of exams  
of courses with common students

$r_1$ : `assign(X,s1) ∨ assign(X,s2) ∨ assign(X,s3) :- course(X).`

$s_1$ : `:- assign(X,S), assign(Y,S), commonStudents(X,Y,N), N>0.`

## What can we do if there is no solution?

If overlapping is unavoidable, then reduce it “As Much As Possible”  
Find an approximate solution

$r_2$ : `assign(X,s1) ∨ assign(X,s2) ∨ assign(X,s3) :- course(X).`

$w_2$ : `:-~ assign(X,S), assign(Y,S), commonStudents(X,Y,N), N>0. [N:]`

Scenarios (models) minimizing the total number of “lost” exams are preferred.

# Aggregate Literals: *Team Building*

- p<sub>1</sub>** The team consists of a certain number of employees
- p<sub>2</sub>** At least a given number of different skills must be present in the team
- p<sub>3</sub>** The sum of the salaries of the employees working in the team must not exceed the given budget
- p<sub>4</sub>** The salary of each individual employee is within a specified limit

$\text{in}(I) \vee \text{out}(I) \text{ :- emp}(I, Sx, Sk, Sa).$

$\text{:- nEmp}(N), \text{ not \#count}\{ I : \text{in}(I) \} = N.$

$\text{:- nSkill}(M), \text{ not \#count}\{ Sk : \text{emp}(I, Sx, Sk, Sa), \text{in}(I) \} \geq M.$

$\text{:- budget}(B), \text{ not \#sum}\{ Sa, I : \text{emp}(I, Sx, Sk, Sa), \text{in}(I) \} \leq B.$

$\text{:- maxSal}(M), \text{ not \#max}\{ Sa : \text{emp}(I, Sx, Sk, Sa), \text{in}(I) \} \leq M.$

# Data-Complexity of Cautious Reasoning (aggregate-free programs)

	$\{\}$	$\text{not}_s$	$w$	$w, \text{not}_s$	$\text{not}$	$w, \text{not}$
$\{\}$	P	P	P	P	coNP	$\Delta^P_2$
$V_h$	coNP	coNP	$\Delta^P_2$	$\Delta^P_2$	coNP	$\Delta^P_2$
$V$	coNP	$\Pi^P_2$	$\Delta^P_3$	$\Delta^P_3$	$\Pi^P_2$	$\Delta^P_3$

N. Leone, G. Pfeifer, W. Faber, T. Eiter, G. Gottlob, S. Perri, F. Scarcello  
 The DLV System for Knowledge Representation and Reasoning  
**ACM Transactions on Computational Logic**. 7(3), 499-562, 2006

# Data-Complexity of Cautious Reasoning (programs with aggregates)

=>

	$\{ \}$	not	$\vee$	not, $\vee$
$\{ \}$	P	coNP	coNP	$\Pi^P_2$
M, A <sub>s</sub> , N <sub>s</sub>	P	coNP	$\Pi^P_2$	$\Pi^P_2$
M, A, N <sub>s</sub>	coNP	coNP	$\Pi^P_2$	$\Pi^P_2$
N	$\Pi^P_2$	$\Pi^P_2$	$\Pi^P_2$	$\Pi^P_2$
M, A, N	$\Pi^P_2$	$\Pi^P_2$	$\Pi^P_2$	$\Pi^P_2$

Aggregates: M: Monotone; A: Antimonotone; N: Nonmonotone; -<sub>s</sub> : Stratified

[ W.Faber, N.Leone, G. Pfeifer, Semantics and Complexity of Recursive Aggregates in Answer Set Programming, **Artificial Intelligence**, 2010 ]

# Functions and Lists: *Simple Paths*

A *simple path* of a graph is a path without any node repetition.

A DLV program computing simple paths.

```
simplePath([X,Y]) :- edge(X,Y), X≠Y.
```

```
simplePath([X|W]) :- simplePath(W), #head(W,Y),  
                    edge(X,Y), not #member(X,W).
```

alternative encoding:

```
simplePath([X|[Y|T]]) :- edge(X,Y), simplePath([Y|T]),  
                        not #member(X,[Y|T]).
```

# Functions and Lists: *Computational Properties*

DLV treats the entire class of Finitely-Ground (FG) programs:

- Express every computable function
- Infinitely large Herbrand models, but stable models are finite
- Brave and cautious reasoning are decidable

DLV computations are sound and complete on every Finitely-Ground program [Calimeri,Cozza,Ianni,Leone, ICLP 2008]

On demand, DLV can “a priori” guarantee termination (finite-domain check)

$DLV^{DB}$  for the execution “on DBMS”

# DLV<sup>DB</sup>

- Powerful disjunctive datalog reasoning on top of data distributed in a number of DBMS
- Push, as much as possible, the computation to the DBMSs where input data reside
- Control the execution flow in main memory, delegating the evaluation of rule bodies to DBMSs, via SQL rewriting
- Full support to disjunctive datalog with unstratified negation, and aggregates
- Normal stratified programs are evaluated “on DBMS” only
- For disjunctive or unstratified programs, only the residual “hard” part is carried out in main-memory
- Exploit logic-based optimization techniques to speed-up executions

# Spin-Off Companies

## DLVSYSTEM

- Spin-Off of University of Calabria
- Founded by the DLV team
- Mission:
  - Enhancement and commercial distribution of DLV
  - System extension and engineering
  - Licensing and maintenance

## EXEURA

- Spin-Off of University of Calabria
- Consulting on “Semantic Technologies”
- Innovative Knowledge Management solutions
- Development of DLV-based products and applications
- Already 30 permanent employees

# Industry-level Applications

Main Application Area: Knowledge Management (KM)

*In the “Society of Knowledge” there is an increasing demand of methods and powerful tools for Knowledge Representation and Management*

Three industrial KM products of Exeura are strongly based on DLV:

➤ OntoDLV

- Advanced platform for Ontology Representation and Reasoning

➤ HiLex

- Ontology-based Information-Extraction
- Ontologies + Semantic Enhanced Regular Expressions

➤ Olex

- A corporate classification system
- Text Classification, Document Storage and Organization

**Exeura Presentation + Demo in the industrial session on Thursday!**

# Industry-level Applications

- Team Building in the Gioia-Tauro Seaport
- E-Tourism: The IDUM system
- Automatic Itinerary Search (for the Transportation Department)
- Data Integration
- Census Data Repair
- Detection of price manipulations and customer privacy
- e-Government
- e-Medicine

Demo of DLV and DLV-based Applications  
during the coffee-break!

# Conclusions

## The Disjunctive Datalog System DLV

- Based on solid theoretical foundations
- Optimized with techniques from AI and Databases
- Enriched with “practical” features to ease application development
- DLV widely disseminated throughout the world in academia
- The first tries of DLV exploitation for Knowledge Management stimulated quite some interest in industry

### Main reference:

N. Leone, G. Pfeifer, W. Faber, T. Eiter, G. Gottlob, S. Perri, F. Scarcello  
The DLV System for Knowledge Representation and Reasoning  
**ACM Transactions on Computational Logic**. 7(3), 499-562, 2006

Check [www.dlvsystem.com](http://www.dlvsystem.com) for news and updates

**DON'T MISS THE NEXT RELEASE OF DLV!**

# DLV Demo during the coffee-break

- DLV with functions, lists, and sets
- Data Integration with DLV<sup>DB</sup>
  - Easy integration of multiple database sources
  - Consistent query answering
- Team Building in the Gioia-Tauro Seaport
  - Intelligent resource allocation
  - Constraints satisfaction
- E-Tourism: the IDUM system
  - Semantic Information Extraction and
  - Text Classification
  - Intelligent Querying
- Automatic reasoning on top of GOOGLE Calendars
  - Rule-based reasoning on RDF ontologies

**COME AND SEE DLV AT WORK !**