The DLV Project: A Tour from Theory and Research to Applications and Market

Nicola Leone
Department of Mathematics
University of Calabria
leone@unical.it

Joint work with
W. Faber, G. Pfeifer, T. Eiter, ...
and about 20 other contributors

Roadmap

- DLP with Stable Model Semantics
- The DLP System DLV (main features)
- A Flavour of the DLV Language
- DLV History
- Some Lessons Learned while Developing DLV
- Industry-level Applications
- Market Perspectives
- Conclusions

Disjunctive Logic Programming with Stable Model Semantics (DLP)

DLP Programs

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

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

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

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

mother(P,S) v father(P,S) :- parent(P,S).

A program with variables is a shorthand for its ground instantiation

Informal Semantics

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

If all the $b_1 \ldots b_k$ are true and none of $b_{k+1} \ldots b_m$ is true,
then at least one among $a_1 \ldots a_n$ is true.

isInterestedinDLP(john) v isCurious(john) :- attendsDLP(john).
attendsDLP(john).

Two (minimal) models, encoding two plausible scenarios:

M1: {attendsDLP(john), isInterestedinDLP(john) }

M2: {attendsDLP(john), isCurious(john) }

Integrity Constraints

Discard interpretations which verify the condition

Rules usually construct the models.

Integrity constraints can be used to eliminate models

:- L_1, … , L_n.

Meaning: discard interpretations in which L_1, … , L_n are simultaneously true.

Stable Model Semantics

Positive Programs:

M is a stable model if it is a minimal model

General Programs (with negation):

M is a stable model if it is a minimal model of the GL-reduct

3-colorability

Input: a Map represented by state(_) and border(_,_).

Problem: assign one color out of 3 colors to each state such that two neighbouring states have always different colors.

Solution:

col(X,red) ∨ col(X,green) ∨ col(X,blue) :-state(X). } Guess

:- border(X,Y), col(X,C), col(Y,C). } Check

The DLP System DLV

Main Features

Advanced Knowledge Modeling Capabilities (1)

- Language:
  - Disjunctive Logic Programs with Stable Model Semantics
  - Extension with aggregates, weak constraints, functions, lists, sets...

- High Expressiveness:
  - Captures Σ^P_2 (NP^{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

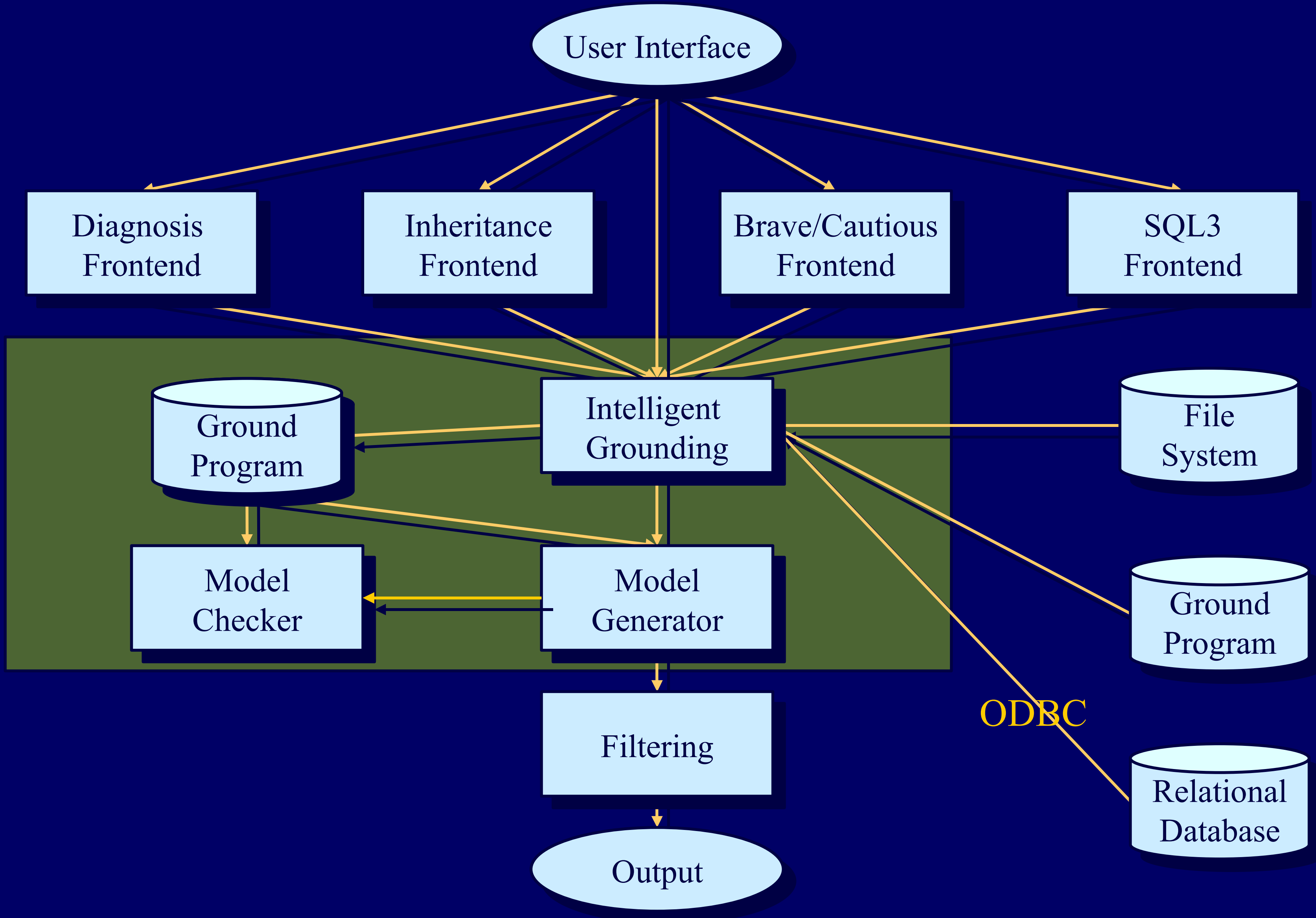
- Database Optimization Techniques
  - Join Ordering Methods
  - Magic Sets
  - Indexing

- Search Optimization Methods:
  - Heuristics
  - Backjumping
  - Pruning Operators

Interoperability

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

System Architecture



A Flavour of DLV Language

Weak Constraints: *Exams Scheduling*

1. 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.

1. 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 Functions: *Team Building*

p_1 The team consists of a certain number of employees

p_2 At least a given number of different skills must be present in the team

p_3 The sum of the salaries of the employees working in the team must not exceed the given budget

p_4 The salary of each individual employee is within a specified limit

```
in(I) v out(I) :- emp(I,Sx,Sk,Sa).
:- nEmp(N), not #count{ I : in(I) } = N.
:- nSkill(M), not #count{ Sk : emp(I,Sx,Sk,Sa), in(I) }  ≥ M.
:- budget(B), not #sum{ Sa, I : emp(I,Sx,Sk,Sa), in(I) } ≤ B.
:- maxSal(M), not #max{ Sa : emp(I,Sx,Sk,Sa), in(I) } ≤ M.
```

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

alternative encoding:

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

Infinitely large Herbrand models, but stable models are finite

DLV computations are sound and complete here

Don’t miss Ianni’s presentation on Thursday!

DLV History

DLV History (1)

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

- [_, Rullo, Information Systems, 1992]

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

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

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

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

1996 (november) DLV project starts in Vienna

DLV History (2)

1997 (july, lpnmr) 1st Release of DLV

- First implementation of a Stable Model System for (standard) DLP
- 3COL: Smodels 0.27 secs, DLV killed after 2 hours!

1998 (june, KR) 5th Release of DLV

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

1998 --> 2007 29 Releases

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

2007 (lpnmr) DLV wins in the ASP System Competition

- 1st in the DLP Category
- 1st in the MGS ("Royal") Category

2008 DLV widely used in academy, interest in industry

Some Lessons Learned while Developing DLV

Some Lessons Learned (1)

LANGUAGE

- As standard as possible
  - Do not re-invent the wheel!
- Make an extension only if it is strongly motivated
- Define a clear and intuitive semantics
- Study language properties in depth

Some Lessons Learned (2)

EFFICIENCY

- In-depth complexity analysis
  - Single out tractable cases
  - Handle them suitably
- Data Structures
- Heuristics and Optimizations techniques
- Benchmarking, a lot of benchmarking

Some Lessons Learned (3)

For wide dissemination and usage

- User friendly interfaces
- Interoperability mechanisms
- Tools for development
  - Error Messaging
  - Debugger

Industry-level Applications

Industry-level Applications

Application Area: Knowledge Management

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

DLV-based systems/applications for:

- Ontology Representation and Reasoning
  - OntoDLV System
- Semantic Information-Extraction
  - HiLex System
- Text Classification
  - Olex System
- Data Integration

Data Integration System

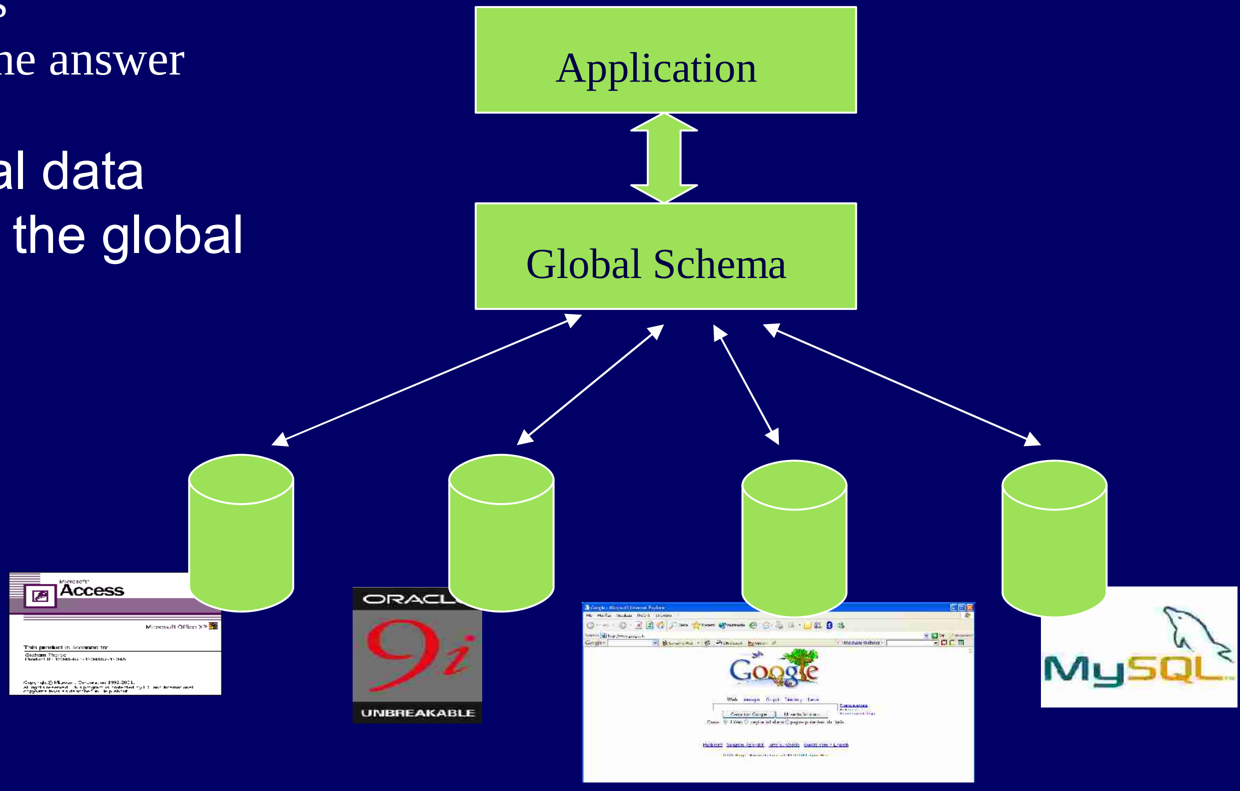
- Offers a global view and a uniform access to a set of autonomous and heterogeneous sources

When the user issues a query over the global schema, the system:

- determines which sources to query and how
- issues suitable queries to the sources
- assembles the results and provides the answer

PROBLEM: The integration of local data from autonomous DBs can violate the global constraints

Repair techniques are needed to provide consistent answers!



Logic Programming for DIS

Basic Idea:

- Map Data Integration System specifications into DLP programs
- Consistent query answering coincides with cautious reasoning on DLP programs
- DLV becomes the core for the computation

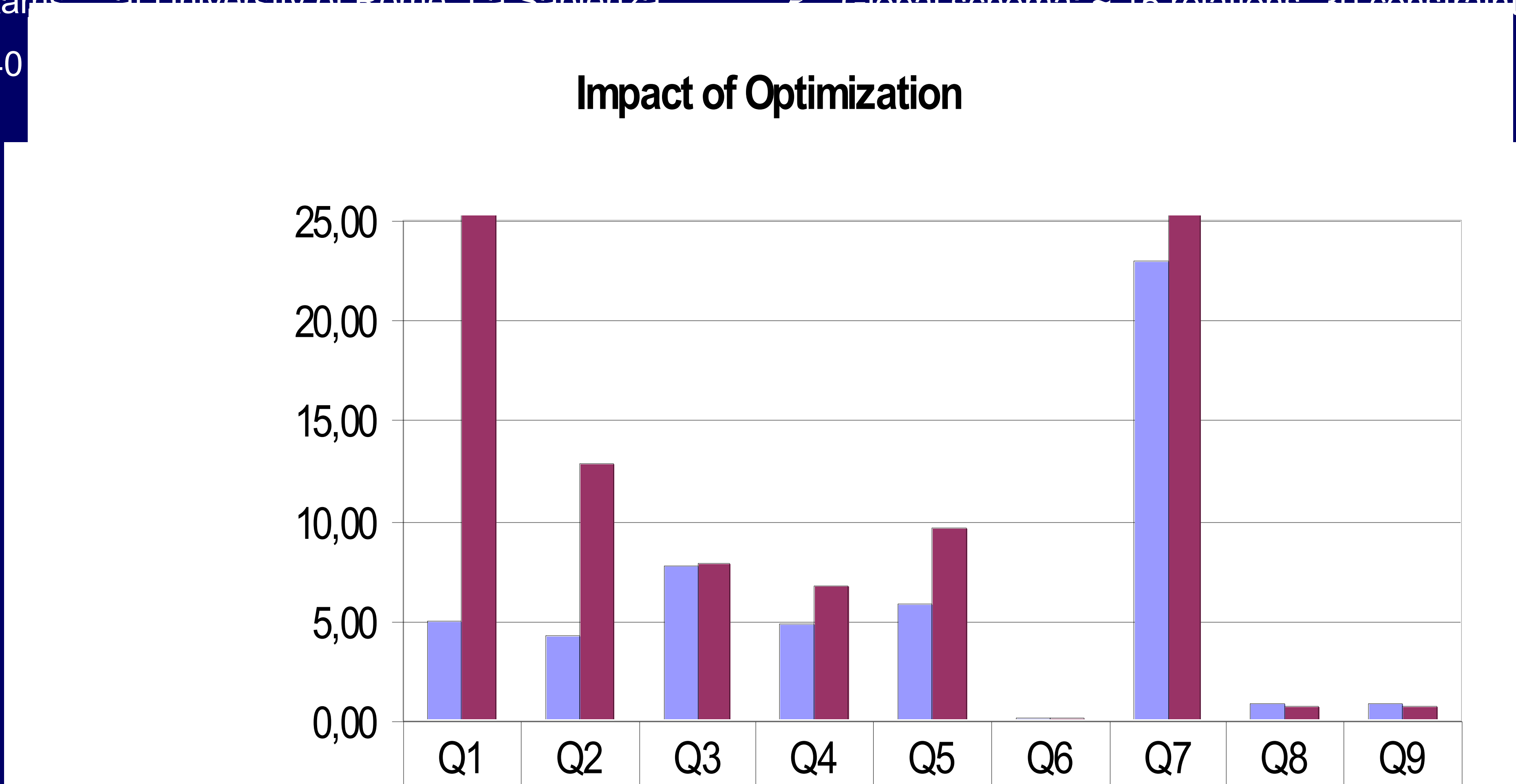
Advantages of the Approach

- executable specification of semantics (easy to change)
- obtain the needed computational power
  - Query answering is co-NP for Keys alone!

Performances (1/2)

DEMO scenario:

- information about students, courses, professors, exams, ... at University of Rome "La Sapienza"
- ~40
- 3 legacy databases (MySQL), lots of web pages
- Global schema: ~ 15 relations, 30 constraints



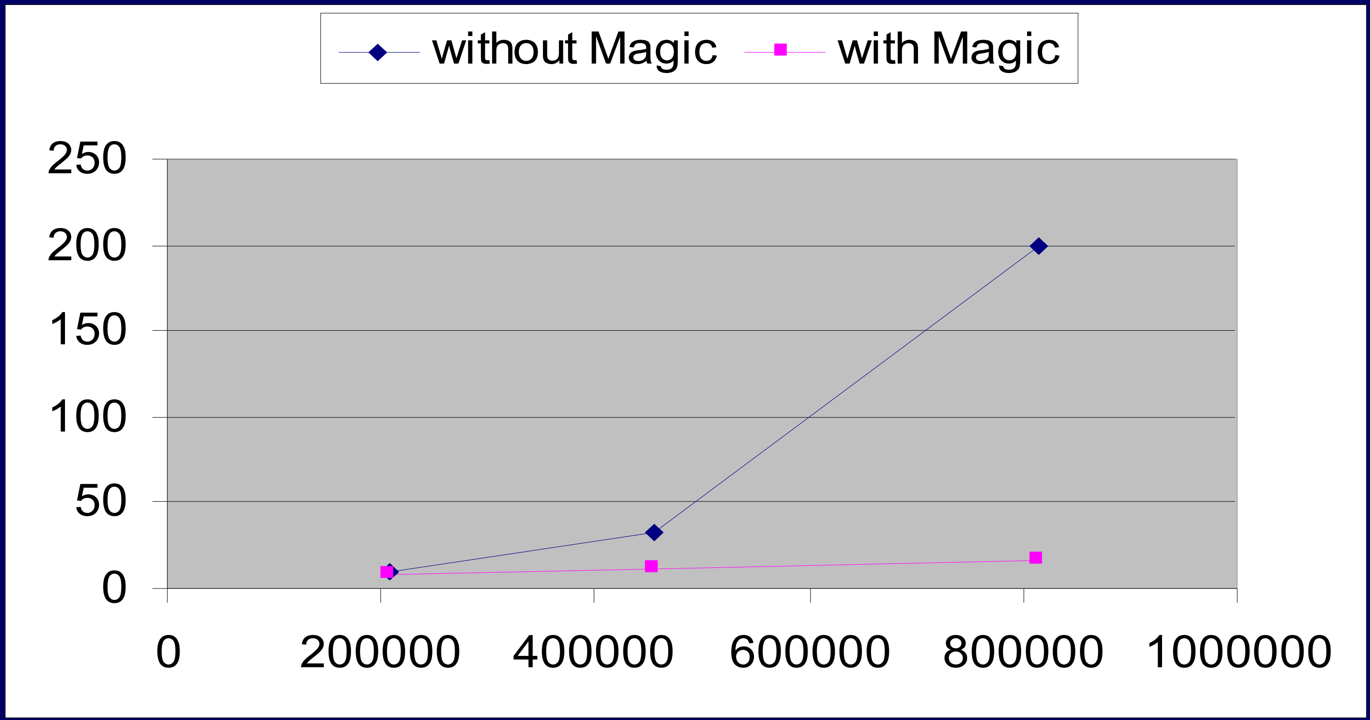
	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9
Magic Sets	4,90	4,20	7,70	4,80	5,80	0,08	23,00	0,81	0,81
No Optimization	129,00	12,93	7,90	6,68	9,70	0,08	197,00	0,78	0,78

Performances (2/2)

DEMO scenario:

- information about students, courses, professors, exams ... at University of Rome “La Sapienza”
- ~40 Wrappers (query wrapper, visual wrappers)
- 3 legacy databases (MySQL), lots of web pages
- Global schema: ~ 15 relations, 30 constraints (KDs,IDs,EDs)

Scaling



Market Perspectives

First results are promising: a lot of interest in the area of Knowledge Management (KM)

- EXEURA (spin off University of Calabria)
  - Consulting on “Semantic Technologies”
    - Using OntoDLV, HiLex, Olex
  - Already 30 permanent employees
  - Branch in Chicago
- DLVSYSTEM
  - Founded by the DLV team
  - DLV engineering and maintenance
  - “A door where to knock on” if a problem surfaces
  - Fast bug fixing
  - Implementation of required extensions

Conclusions

- A lot of work in 12 years
  - For building the theoretical foundations of DLV
  - For implementing and maintaining the system

- But the gratification is even higher
  - Appreciation in the research community
  - More than 30 journal papers and 100 conference papers related to DLV
  - DLV widely disseminated throughout the world
  - We are contacted nearly every week by DLV users
  - The first tries of DLP exploitation for Knowledge Management stimulated some interest in industry

- The feedback "from the field" suggests challenging issues for both the theory and the system: we'll do our best to improve

  DON'T MISS THE NEXT RELEASE OF DLV!

Ontology Representation and Reasoning:
the OntoDLV System

Ontology Representation and Reasoning

- The strong need of knowledge-based technologies is perceived by industries today
- The Description Logic (DL) community did a very good job in divulgating and advertising OWL as <u>the</u> language for Ontologies
- OWL is a W3C standard
- Many companies invest in OWL and try to use it also for enterprise/corporate ontologies

Ontology Representation and Reasoning

- OWL semantic assumptions are fine for the Semantic Web
- Some semantic features of OWL (especially the Open-World assumption) make it unsuited for enterprise/corporate ontologies
  - enterprise ontologies often extend the enterprise databases

supplier	branch city	branch street
barilla	rome	veneto
barilla	naples	plebiscito
voiello	naples	cavour

List the suppliers whose branches are only in Naples

- In ASP (according with the Closed World Assumption)
  - answer = {voiello}
- In OWL (according with the Open World Assumption)
  - you cannot entail that voiello has only a branch in Naples,
  - since you assume that, e.g., supplier(voiello,rome,veneto) may hold but it is presently unknown
  - answer = {}

OntoDLV

An ASP-based Ontology Management System

What is it?

- Advanced platform for ontology management
  - Specification
  - Browsing
  - **Querying**
  - **Reasoning**

- Powerful and user-friendly visual environment
  - Advanced visual querying interface (à la QBE)
  - Automatic Error Detection (advanced type checking)
  - Application Programming Interface (API)

- Able to deal with data-intensive applications
  - Persistency on DBMS
  - Can work in mass-memory (via DLV^{DB})

- Based on a DLP extension named OntoDLP

OntoDLV is our computational core

- Olex - the text classification system, and
- Hilex - the semantic information-extraction system

are built on top of OntoDLV

OntoDLV is well suited also for direct application-development

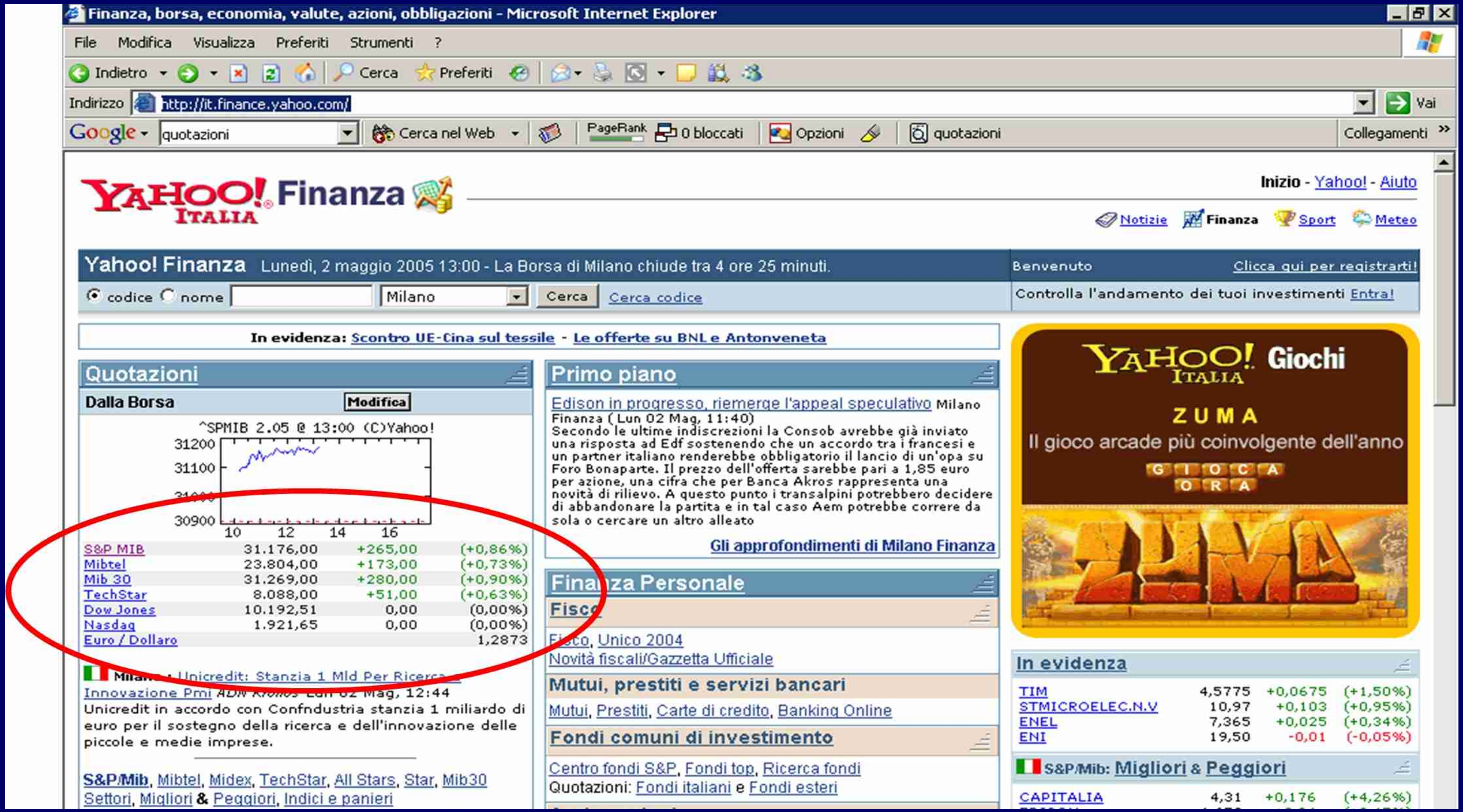
- The RAP platform (Orangee):
  - **Governance of the** distribution process of antiblastic medicines in hospitals
  - Agent-Based (JADE <--> OntoDLV API)
  - “The agent’s brain” is an OntoDLP program

Semantic Information Extraction: the HiLex System

What is HiLex?

- An Advanced Tool for **Semantic** Information-Extraction
- Ontology Driven
  - Exploits an OntoDLP ontology of the domain
- Recognition of “Semantic” Regular Expressions
  - representing ways of writing a concept in a document
  - elements of the expression can be concepts specified through queries over the ontology (e.g., any member of a class)
  - the concepts which are recognized in the document are stored in the OntoDLV ontology
- Rewriting to DLP
  - Information Extraction amounts to Stable Model Computation

Example



Finanza, borsa, economia, valute, azioni, obbligazioni - Microsoft Internet Explorer
http://it.finance.yahoo.com/
YAHOO! ITALIA Finanza
Inizio - Yahoo! - Aiuto
Yahoo! Finanza Lunedì, 2 maggio 2005 13:00 - La Borsa di Milano chiude tra 4 ore 25 minuti.
In evidenza: Scontro UE-Cina sul tessile - Le offerte su BNL e Antonveneta
Quotazioni
Dalla Borsa
^SPMIB 2.05 @ 13:00 (C)Yahoo!
S&P MIB 31.176,00 +265,00 (+0,86%)
Mibtel 23.804,00 +173,00 (+0,73%)
Mib 30 31.269,00 +280,00 (+0,90%)
TechStar 8.088,00 +51,00 (+0,63%)
Dow Jones 10.192,51 0,00 (0,00%)
Nasdaq 1.921,65 0,00 (0,00%)
Euro / Dollaro 1,2873
Primo piano
Edison in progresso, riemerge l'appeal speculativo
Finanza Personale
YAHOO! ITALIA Giochi
ZUMA
Il gioco arcade più coinvolgente dell'anno

Example

```
stock_index_table: stock_index_structure (
    type: hilex_type,
    expression: "tableOf (arg: [@stock_index, unsigned_float,
                 signed_float, percentage], range: {3, },
                 dir: vertical, sep: blank_char),
    label: "table of stock_index_variation_row").
```

TechStar techStar	8.088,00 unsigned_float	+51,00 signed_float	+0,63% percentage
Dow Jones dowJones	10.192,51 unsigned_float	0,00 signed_float	0,00% percentage
Nasdaq dasdaq	1.921,65 unsigned_float	0,00 signed_float	0,00% percentage

Example

(a)

S&P MIB	31.176,00	+265,00	(+0,86%)
Mibtel	23.804,00	+173,00	(+0,73%)
Mib 30	31.269,00	+280,00	(+0,90%)
TechStar	8.088,00	+51,00	(+0,63%)
Dow Jones	10.192,51	0,00	(0,00%)
Nasdaq	1.921,65	0,00	(0,00%)

(b)

S&P MIB	31.176,00	+265,00	(+0,86%)
Mibtel	23.804,00	+173,00	(+0,73%)
Mib 30	31.269,00	+280,00	(+0,90%)
TechStar	8.088,00	+51,00	(+0,63%)
Dow Jones	10.192,51	0,00	(0,00%)
Nasdaq	1.921,65	0,00	(0,00%)

(c)

S&P MIB	31.176,00	+265,00	(+0,86%)
Mibtel	23.804,00	+173,00	(+0,73%)
Mib 30	31.269,00	+280,00	(+0,90%)
TechStar	8.088,00	+51,00	(+0,63%)
Dow Jones	10.192,51	0,00	(0,00%)
Nasdaq	1.921,65	0,00	(0,00%)

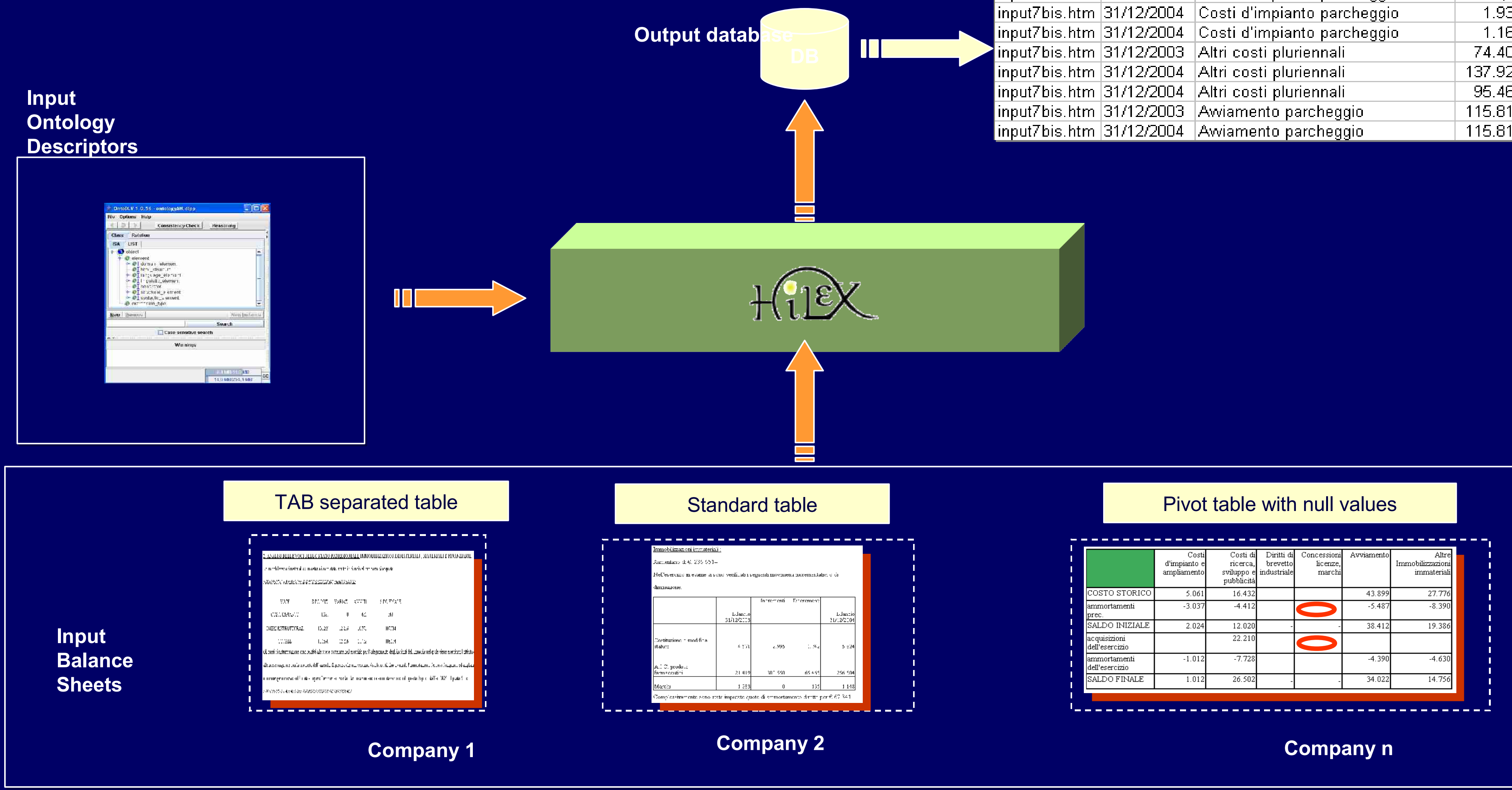
An HiLex Application: automatic extraction of information from balance notes

- A company provides financial information to banks
- The balances of all italian companies are analyzed (~800,000 per year in PDF format)
- Balance sheets contain important information also in attached notes written in natural language

HiLex application

- input PDF balances are mapped to HTML
- Hilex extracts the information from the balance notes
- and stores them in a relational database
- about 90% successful extractions

Automatic extraction of information from balance notes: the process



inputName	vallstInt	vallstVoce	val (€)
input7bis.htm	31/12/2003	Costi d'impianto parcheggio	1.937,00
input7bis.htm	31/12/2004	Costi d'impianto parcheggio	1.937
input7bis.htm	31/12/2004	Costi d'impianto parcheggio	1.162
input7bis.htm	31/12/2003	Altri costi pluriennali	74.406
input7bis.htm	31/12/2004	Altri costi pluriennali	137.928
input7bis.htm	31/12/2004	Altri costi pluriennali	95.461
input7bis.htm	31/12/2003	Awiamento parcheggio	115.815
input7bis.htm	31/12/2004	Awiamento parcheggio	115.815

Text Classification: The Olex system

The Classification Problem

Given a set $C = \{c_1, . . . , c_m\}$ of categories,

and a set $D = \{d_1, . . . , d_n\}$ of documents,

assign to each document its category(/ies)

Typical example:

- Classify Reuters' news according to their contents

Olex Classifier

- Exploitation of Default Negation

- Classifier: set of classification rules

  $C \leftarrow t_0, \text{ not } t_1, \ldots, \text{ not } t_n$

- One positive literal and (zero or) more **negative** literals

Intuitively:

- Positive atoms allow us to catch most of the right documents (thus, providing high "recall")

- Negative atoms help us in avoiding "too many" mistakes (thus, providing high "precision").

Learning and Classification

LEARNING PHASE

- On a training set of documents
  - Transform the document in logical facts
  - Select the discriminating terms
  - generate the classifier (an OntoDLP program)
  - Validate the classifier (OntoDLV run)

CLASSIFICATION PHASE

- Run the classifier on the (OntoDLP version of the) input documents
- Classify them according to the resulting answer sets

Experimental Results

Olex has been tested on two well-known corpora:

- **Reuters** (ModApte split):
  - 7,063 articles in the training set,
  - 2,742 articles in the test set,
  - 118 categories
- **OHSUMED**:
  - 20,000 documents,
  - 23 categories

Olex works efficiently and with a very good precision

Is there a market for these ASP applications?

- First results are very promising; a lot of interest in the area of Knowldge Management (KM)

- Exeura (spin off University of Calabria)
  - Consulting on exploiting ASP (/DLV) for KM
  - Already 30 permanent employees

- FourthCodex: Joint venture between Exeura (Calabria) and Herzum (Chicago)
  - Industrial development of the 4 KM products
  - Distribution in the US market
  - Some license already given, interest from big companies

Is there a market for these ASP applications?

- In order to employ DLV in industrial applications, companies wanted the warranty that DLV is maintained
  - "A door where to knock on" if a problem surfaces
  - Fast bug fixing
  - Implementation of required extensions

- Creation of company DLVSYSTEM to play this role
  - DLV engineering and maintenance

Future Directions

- What lessons have we learned on the field?
- What are the main challenges?

Lessons Learned (1)

- Engineers often are unable to write (correct and efficient) ASP Programs
- Application programs are frequently "easy" (stratified or nearly such); but have to deal with HUGE amount of data
- Input data often in databases or on the web

Lessons Learned (2)

- Language expressiveness of ASP is more than enough for many applications; but some "practical" features are missing
  - Application specific functions
  - Data Types (data and methods)
    - floating-point numbers
    - strings
    - date and currency
  - Other aggregate functions (e.g., Average)
  - Lists and Sets

- ASP systems are mostly used as the "intelligent" engine of an ample software architecture

Challenges (1)

1. *Engineers often are unable to write (correct and efficient) ASP Programs*

- Develop tools for programmers
  - Programming Environments
  - Debuggers (tools and techniques)
  - Friendly Interfaces
- Design ASP programming methodologies

Challenges (2)

2. *Application programs are frequently "easy" (stratified or nearly such); but have to deal with HUGE amount of data*

- Improve the ASP instantiators
- Database technologies
- Mass memory computation
- Partial evaluation techniques
- More on Magic Sets

Challenges (3)

3. *Input data often reside on databases or on the web*

- Interoperability with DBMSs
- Interoperability with OWL/RDF
- Technological and theoretical issues

Challenges (4)

4. *Language expressiveness of ASP is even more than enough; but some "practical" features are missing*
    - *Application specific functions*
    - *Data Types (data and methods)*
        - *floating-point number, strings, dates,...*
    - *Other aggregates functions (e.g., Average)*
    - *Lists and Sets*

- Not easy at all, semantic and implementation issues
- Dealing with infinite domains
- Inside ASP systems or on top?

Challenges (5)

5. *ASP systems are mostly used as the "intelligent" engine of an ample software architecture*

- Application Programming Interfaces (API)
- Mechanisms and tools for interoperability

Conclusion

- After more than 20 years of theoretical research, serious efforts on implementation have been done, and efficient ASP systems are available, making ASP viable for applications
- Our first tries of ASP exploitation in Knowledge Management stimulated much interest in industry
- We are having feedback “from the field”, suggesting challenging issues for both theory and systems
- I encourage other groups to “attack” real applications:
  - quite some work;
  - but also big fun!!

A sample case:
The development of aggregates

Aggregate functions

emp(EmpId, Salary)

Compute the sum of the salaries of the employees

- Easily expressed in SQL
- Representation in logic is rather unnatural
  - recursion needed to express Sum
  - quadratic space

Sum (DLP vs DLPA)

```
% Order employees by id
    precedes(X,Y) :- emp(X,_), emp(Y,_), X<Y.

% Define successor, first and last
    succ(X,Y) :- precedes(X,Y), not elementInMiddle(X,Y).
    elementInMiddle(X,Y) :- precedes(X,Z), precedes(Z,Y).
    first(X) :- emp(X,_), not hasPredecessor(X).
    last(X) :- emp(X,_), not hasSuccessor(X).
    hasPredecessor(X) :- succ(Y,X).
    hasSuccessor(Y) :- succ(Y,X).

% sum salaries recursively
    partialSum(X,Sx) :-  first(X), emp(X,Sx).
    partialSum(Y,S) :-  succ(X,Y), partialSum(X,PSx), emp(Y,Sy), S=PSx+Sy.

% select the total
    sum(S) :- last(L), partialSum(L,S).
```

Aggregate atoms

f{S} <* X

S : symbolic set

f : function name among { #count, #sum, #times, #min, #max }

<*: comparison operator in { <, ≤, >, ≥ }

#count { EmpId : emp(EmpId, male, Skill, Salary) } ≤ 10

The atom is true if the number of male employees does not exceed 10.

Formal semantics: extension of the notion of answer set; quite difficult if aggregates are recursive (unstratified).

Problems with the Semantics of Recursive Aggregates

- Gelfond-Lifschitz reduct differentiates between positive and negative literals
- Aggregate atoms can be similar to positive AND negative literals
    - #count{a} > 1 is like a positive literal
    - #count{a} < 1 is like a negative literal
    - #sum over (signed) integers is different from both
- How to deal with aggregate atoms in the reduct?

Novel Definition of Reduct

[Faber, Leone, Pfeifer, JELIA'04]

Reduct P^I of P w.r.t. I:

- Delete the rules where a body literal is false

That's it!

An answer set is an interpretation which is a minimal model of the reduct.

Novel Definition of Reduct

Theorem: On standard DLP programs the definition of [Faber,Leone,Pfeifer ‘04] yields precisely the same answer sets as [Gelfond&Lifschitz ‘91]

- Simpler: Only one condition, rules not altered
- Uniform treatment of positive and negative literals
- Uniform treatment of standard and aggregate literals
- **Useful also for standard programs (proofs)!**

The case of Aggregates: Complexity Analysis

Main Decision Problem

[Cautious Reasoning]

Given a DLP program P, and a ground literal A,

is A true in ALL answer sets of P?

Complexity of aggregates

Theorem

Cautious Reasoning on ground DLP programs with aggregates is Π^P_2 - complete

It seems that the addition of aggregates to DLP comes for free

Are we sure that there is never a computational overhead?
Let's analyze some language fragments.

Monotonicity of Literals

- Monotone Literals:
  truth for interpretation I implies truth for all J extending I

- Antimonotone Literals:
  truth for interpretation I implies truth for all J included in I

- Nonmonotone Literals:
  neither monotone nor antimonotone

- Positive standard literals are monotone
- Negative standard literals are antimonotone
- Aggregate SUM over (possibly negative) integers is nonmonotone

Restrictions on Aggregates

Ms = stratified monotone aggregates

M = full (possibly recursive) monotone aggregates

As = stratified antimonotone aggregates

A = full antimonotone aggregates

Ns = stratified nonmonotone aggregates

N = full nonmonotone aggregates

Complexity of Cautious Reasoning

for programs with restricted aggregates

	{}	not	V	not, V
{}	P	coNP	Π^P_2	Π^P_2
=> M, As, Ns	P	coNP	Π^P_2	Π^P_2
M, A, Ns	coNP	coNP	Π^P_2	Π^P_2
N	Π^P_2	Π^P_2	Π^P_2	Π^P_2
M, A, N	Π^P_2	Π^P_2	Π^P_2	Π^P_2

[W.Faber, N.Leone, G. Pfeifer 2004 & 2005]

Roadmap

- Theoretical Foundations (selected contributions)
  - Theoretical Properties of DLP
    - Stable Models and Unfounded Sets
  - ?? Optimization Techniques
  - Language Extensions
    - Aggregate Functions
    - Weak Constraints
  - Computational Complexity
    - DLP
    - Aggregates
  - Spin-off Companies
    - DLVSYSTEM
    - EXEURA
  - Industry-level Applications
    - Ontology Representation and Reasoning
    - Data Integration
    - Information Extraction
    - Text Classification
- Conclusions

Semantics for Positive Programs

Assume now that 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)

[Gelfond & Lifschitz 1988, 1991]

Reduct P^I of P w.r.t. I:

- Delete all rules with a negative false literal (w.r.t. I);
- Delete the negative literals from the bodies of the remaining rules.

A Stable Model of a program P is an interpretation I such that I is a stable model of P^I.