Answer Set Programming

Introduction

Martin Baláž

Department of Applied Informatics
Comenius University in Bratislava

2009/2010

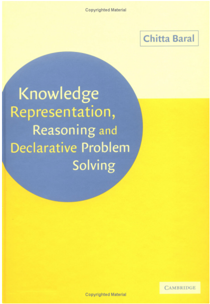
Contacts

Martin Baláž
KAI - I 7
balaz@ii.fmph.uniba.sk

Jozef Šiška
KAI - I 7
siska@ii.fmph.uniba.sk

Literature

Chitta Baral. Knowledge Representation, Reasoning and Declarative Problem Solving. Cambridge University Press, Cambridge, 2003



Outline

1. Motivation

2. Introduction
   - Prolog
   - Logic Programming
   - Answer Set Programming

Motivation

Example

Six people sit at round table.
Each drinks a different kind of soda.
Each plans to visit a different French-speaking country.
The person who is planning a trip to Quebec, who drank either blueberry or lemon soda, didn't sit in seat number one.
Jeanne didn't sit next to the person who enjoyed the kiwi soda.
The person who has a plane ticket to Belgium, who sat in seat four or seat five, didn't order the tangelo soda.
...
What is each of then drinking and where is each of then going?

Motivation

- Knowledge representation
- Nonmonotonic reasoning
- Declarative language
- Incomplete information
- Inconsistent information

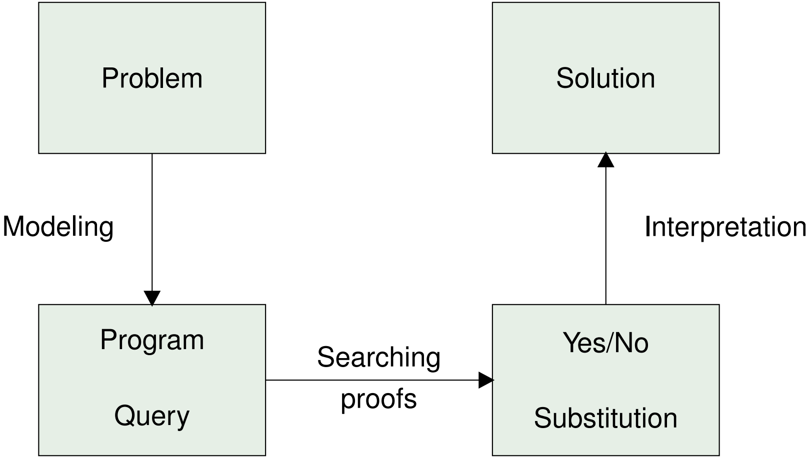
Prolog

Example

```
reverse([], []).
reverse([X|Xs], Zs) :- reverse(Xs, Ys),
                       append(Ys, [X], Zs).
```

Query: `? reverse([1, 2, 3], Zs).`
Answer: `Zs = [3, 2, 1]`

Solving Problems with Prolog



Query Processing

Top-down query processing (SLD resolution) - from query to facts.

Example

```
man(dilbert).
single(X) :- man(X), not(husband(X)).
husband(X) :- man(X), not(single(X)).

? single(X).
...
```

Ordering Rules

Ordering of rules matters.

Example

```
reverse([X|Xs], Zs) :- reverse(Xs, Ys),
                       append(Ys, [X], Zs).
reverse([], []).

? reverse(Xs, [3,2,1]).
Xs = [1,2,3]
...
```

Ordering Literals

Ordering of literals matters.

Example

```
reverse([], []).
reverse([X|Xs], Zs) :- append(Ys, [X], Zs),
                       reverse(Xs, Ys).

? reverse([1,2,3], Zs).
Zs = [3,2,1]
...
```

Floundering

Negation as failure

Example

```
man(dilbert).
husband(bill).
single(X) :- man(X), not(husband(X)).

? single(X).
X = dilbert
```

Floundering

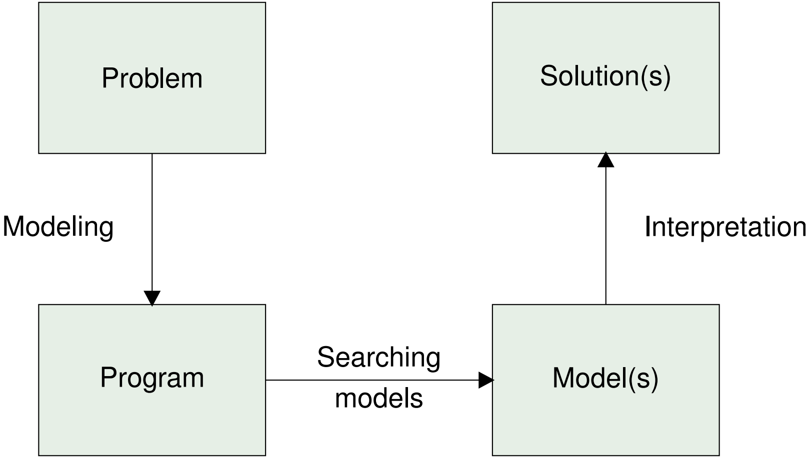
Negation as failure

Example

```
man(dilbert).
husband(bill).
single(X) :- not(husband(X)), man(X).

? single(X).
No
```

Solving problems with Logic Programming



Semantics of Logic Programs with Negation

Example (Multiple models)

```
man(dilbert).
single(X) :- man(X), not husband(X).
husband(X) :- man(X), not single(X).
```

M_1 = `{man(dilbert), single(dilbert)}`
M_2 = `{man(dilbert), husband(dilbert)}`

- Single Intended Model Approach
  - Select a single model of all classical models
  - Well-founded semantics, Perfect model semantics
- Multiple Preferred Model Approach
  - Select a subset of all classical models
  - Stable model semantics

Answer Set Programming

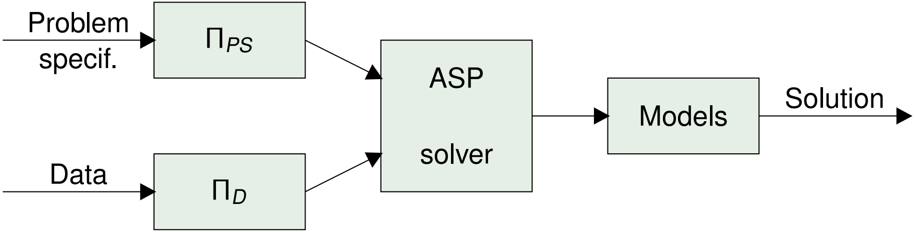
ASP = disjunctive (extended) logic programs with stable model semantics

ASP is more expressive then classical logic

Example (Transitive closure)

```
path(X, Y) :- edge(X, Y).
path(X, Z) :- edge(X, Y), path(Y, Z).
```

ASP in Practice



Example (Problem specification)

```
color(X, r) v color(X, g) v color(X, b) :- node(X).
:- node(X), color(X, C), color(X, D), C <> D.
:- edge(X, Y), color(X, C), color(Y, C).
```

Usage of ASP

- constraint satisfaction
- planing
- diagnosis
- security engineering
- Semantic Web
- configuration
- data integration
- ...