Combining Answer Set Programs for Adaptive and Reactive Reasoning

Tony Ribeiro, Katsumi Inoue, Gauvain Bourgne

The Graduate University for Advanced Studies, Tokyo, Japan
National Institute of Informatics, Tokyo, Japan
University Pierre et Marie Curie, Paris, France

ICLP Technical communications, August 27th 2013
Outline

1 Preliminaries
   - Motivation
   - Proposal

2 Design Pattern
   - Module Typology
   - Modelisation

3 Framework
   - Keywords
   - Evaluation

4 Summary
   - Contribution and Outlook
Answer set programming (ASP) is a form of declarative programming introduced by (Gelfond and Lifschitz 1988).

ASP has been successively used in many works:
- Constraint programming (Niemela 1999)
- Knowledge representation and reasoning (Baral 2003/2008)
- Multi-agent systems (Nieuwenborgh et al. 2006)
1 Preliminaries
   • Motivation
   • Proposal

2 Design Pattern
   • Module Typology
   • Modelisation

3 Framework
   • Keywords
   • Evaluation

4 Summary
   • Contribution and Outlook
Our interest is to help the construction of efficient ASP based reasoning systems.

Example (For a Developer: Modeling)
The indivisible nature of ASP programs is causing increasing difficulties as program instances tend to grow in real applications.

Example (For an ASP based System: Reasoning)
Reactivity of an ASP based system is very dependant on the quantity of knowledge, i.e. size of the program, used for reasoning.
Outline

1 Preliminaries
   • Motivation
   • Proposal

2 Design Pattern
   • Module Typology
   • Modelisation

3 Framework
   • Keywords
   • Evaluation

4 Summary
   • Contribution and Outlook
What we propose is a design pattern to model reasoning by multiple ASP programs.

Our idea is quite similar to (Minsky 1991):

*Is a mind composed of smaller and smaller minds, until the pieces become so small that they are no longer mind like?*
What we propose is a design pattern to model reasoning by multiple ASP programs.

Example (Modeling)

It is usually easier to model a system as a set of small interacting components rather than a huge and obscure monolithic system.
Proposal: Framework

A framework to reason by combinations of ASP programs.

Example (Reasoning)

Modular division of ASP programs can allow to reduce the quantity of knowledge used for reasoning.
Outline

1 Preliminaries
   - Motivation
   - Proposal

2 Design Pattern
   - Module Typology
   - Modelisation

3 Framework
   - Keywords
   - Evaluation

4 Summary
   - Contribution and Outlook
We first propose to separate background knowledge and observations.

**Theory Module**
- Background knowledge
- Given
  - Hiding
  - Eat
  - Action
  - Move
  - Discover

**Observations Module**
- Consistent observations
- Acquired
  - Wolf
  - Flower
  - Myself
  - Rabbit
  - Field

Tony Ribeiro, Katsumi Inoue, Gauvain Bourgne
Combining ASP Programs for Adaptive and Reactive Reasoning
Outline

1 Preliminaries
   - Motivation
   - Proposal

2 Design Pattern
   - Module Typology
   - Modelisation

3 Framework
   - Keywords
   - Evaluation

4 Summary
   - Contribution and Outlook
The core idea of our reasoning framework is similar to contextual logic programming (Monteiro and Porto 1989).
The core idea of our reasoning framework is similar to contextual logic programming (Monteiro and Porto 1989).

We define a reasoning system consisting of independent logic programs which can be combined together regarding context needs.

Modules combination

- My moves possibilities
The core idea of our reasoning framework is similar to contextual logic programming (Monteiro and Porto 1989).

We define a reasoning system consisting of independent logic programs which can be combined together regarding context needs.
The core idea of our reasoning framework is similar to contextual logic programming (Monteiro and Porto 1989).

We define a reasoning system consisting of independent logic programs which can be combined together regarding context needs.
The core idea of our reasoning framework is similar to contextual logic programming (Monteiro and Porto 1989).

We define a reasoning system consisting of independent logic programs which can be combined together regarding context needs.
These programs combinations can be known by the agent as meta-knowledge about its own knowledge.

Meta-knowledge

- Movements

In our module typology we use meta-knowledge module to represent knowledge about module combination (dotted circle).
These programs combinations can be known by the agent as meta-knowledge about its own knowledge.

In our module typology we use meta-knowledge module to represent knowledge about module combination (dotted circle).
These programs combinations can be known by the agent as meta-knowledge about its own knowledge.

In our module typology we use meta-knowledge module to represent knowledge about module combination (dotted circle).
These programs combinations can be known by the agent as meta-knowledge about its own knowledge.

Meta-knowledge
- Movements
- My moves
- Threat
- Safe moves
These programs combinations can be known by the agent as meta-knowledge about its own knowledge.

Meta-knowledge
- Movements
- My moves
- Threat
- Safe moves
- Run away
Outline

1 Preliminaries
   - Motivation
   - Proposal

2 Design Pattern
   - Module Typology
   - Modelisation

3 Framework
   - Keywords
   - Evaluation

4 Summary
   - Contribution and Outlook
Our framework parses ASP programs and answer set to interpret respectively two simples keywords:

- `include('module name')`
Our framework parses ASP programs and answer set to interprete respectively two simples keywords:

- `include('module name')`
- `next('module name')`

```
Survive

%#include "Wolf".
%#extern position/2.

wolf :- position(wolf,Position).
next("Hunted") :- wolf.
next("Hunter") :- not wolf.
```

Tony Ribeiro, Katsumi Inoue, Gauvain Bourgne

Combining ASP Programs for Adaptive and Reactive Reasoning
Outline

1 Preliminaries
   • Motivation
   • Proposal

2 Design Pattern
   • Module Typology
   • Modelisation

3 Framework
   • Keywords
   • Evaluation

4 Summary
   • Contribution and Outlook
In this example, we propose a possible division of the reasoning of a rabbit agent into 4 independent parts.

Empty arrows represent inclusions, plain ones represent decisions.
Experimental application based on the survival game example. Arrows represent some movements that agents are considering in order to explore/feed/hide/escape.
Results

Reasoning time evolution regarding number of observations.

Experimental results, for each method it shows rabbit reasoning time average of 1000 runs.
Outline

1 Preliminaries
   • Motivation
   • Proposal

2 Design Pattern
   • Module Typology
   • Modelisation

3 Framework
   • Keywords
   • Evaluation

4 Summary
   • Contribution and Outlook
Contributions

- A design pattern to represent knowledge as multiple ASP programs.
- A framework to use this representation within an ASP based reasoning system.

Outlook

- Dynamic learning of module and combinations
  - Experiment based
- Generic methods to divide a monolithic ASP program
  - Rules dependancy
Knowledge representation, reasoning and declarative problem solving.  
Cambridge university press.

Using answer set programming for knowledge representation and reasoning: Future directions.  
In ICLP. 69–70.

The stable model semantics for logic programming.  
In ICLP/SLP. 1070–1080.

Society of mind: A response to four reviews.  
Contextual logic programming.
In ICLP. 284–299.

Niemelä, I. 1999.
Logic programs with stable model semantics as a constraint programming paradigm.

Hierarchical decision making in multi-agent systems using answer set programming.
In CLIMA. 20–40.