Teaching Answer Set Prolog and its Extensions

Michael Gelfond

Computer Science Department

Texas Tech University

December 17, 2021

History

At the university level I taught *programming in Answer Set Prolog and its extensions (ASP)* **in "Design of Intelligent Systems", "Logic Programming", and "Programming Languages".**

Classes were taught at graduate and undergraduate levels, with number of students ranging from 10 to 50.

Course "Introduction to Computer Programming for AI" was taught for high school students.

Last 15 years I used my book with Yulia.

Some Teaching Goals

In addition to helping students to master the material I aim to:

• awaken/increase students' sense of wonder and awe, and familiarize them with process of discovery;

• demonstrate importance of rigorous analytical thinking;

• improve students' taste and help to develop personal scientific and programming styles;

• show that computer science is an integral part of human culture.

• relate all these closely with practical needs of their trade and other aspects of their life.

Design of Intelligent Systems

Goal: learn how to build software components of intelligent agents capable of reasoning and acting in changing environment.

Assumption: to exhibit intelligent behavior an agent should have a mathematical model of its environment and its own capabilities and goals, as well as algorithms for achieving these goals.

We will study such models and algorithms.

Modeling an Agent

Mathematical model of an intelligent agent normally consists of

- **A formal language(s) for representing knowledge of the agent.**
- **Reasoning algorithms using this knowledge to perform planning, diagnostics, learning, etc.**
- **Agent architecture - a structure combining sub-models of an agent (normally related to different reasoning tasks) in one coherent whole.**

This determines emphasis on the use of ASP for KR, agent algorithms, etc.

Typical exam questions

(a) Represent the following information in Answer Set Prolog: As a rule, shooting a person is prohibited. But shooting in self-defense is allowed. Use relations $can\_shoot(X, Y)$ and $attacked(X, Y)$. Let T be your representation together with two facts: $person(bob)$ and $person(jim)$.

(i) How does your representation T answer query $can\_shoot(bob, jim)$?

(ii) Will the answer change if T is expanded by $attacked(jim, bob)$?

(b) Describe the Frame Problem, its importance and its ASP solution.

(c) Write an ASP planning module.

Teaching in high school

In my view schools fail in teaching students how to read, write, and listen.

Even at the university few students understand the difference between meaningful statement and nonsensical one, formal definition and an intuitive description of a notion, etc.

Lack of basic skills and understanding of the importance of rigor severely limits students ability to learn and to have a good life.

So, in addition to teaching ASP programming, I wanted to alleviate these problems, which influenced selection of the class material.

Declarative Programming: Facts

Problem: Teach a computer basic facts about family relations. Objects of this domain are people. (As an example we consider a small family consisting of John, Mary, their son Sam and daughter Cora.)

The domain will be structured in terms of relations "father", "mother", and "gender".

Check the program by asking queries, e.g.,

$$? \ father(X, sam)$$

Declarative Programming: Rules

Introduce your program to the notion of "parent".

To do that formulate definition of "parent" in English (use variables if needed) and write it in the formal language.

Check if the program understood the new term by asking question (the same way we do with people).

Defining *sister*

Next I ask to teach the program relation "X is a sister of Y". This opens possibilities for useful discussions.

(a) Usually there are two different solutions. Some require X and Y to have two common parents, others believe that one is sufficient.

Often the difference is determined by culture.

They encounter the Tower of Babel phenomena: people use the same common word but talk about different things.

Not all of them are aware of this phenomena and some are genuinely surprised.

Defining *sister* (continued)

(b) Almost always the students' definitions imply that X is her own sister.

This opens a possibility to talk about hidden assumptions and importance of making these assumptions explicit.

(c) So far our definition tells us when X is a sister of Y. But when is X not a sister of Y?

Assume completeness of the family database and introduce classical and default negations to express: "No one else is a sister of Y".

Talk a little about what to do if the database is incomplete.

Other topics

Continue by using "ancestor" to teach recursive definitions.

Talk about hierarchies, defaults and their exceptions, etc.

Use simplest possible examples.

I believe that most people who tried to understand the material, did so. And it was fun to teach.

THANKS!

Related Talks

If someone is interested two general talks on teaching entitled

"Some Thoughts on Teaching Computer Science"

and

"On Learning History"

can be found on my WEB page.