Logic Programming, 20-21

Lecturer: Vítor Santos Costa
DCC-FCUP
vsc@dcc.fc.up.pt (room: 1.45)

These slides are largely based on Prof. Inês Dutra's and Prof. Alípio Jorge

17 de março de 2022

Contents

- Introduction to Prolog and Logic Programming
- The expressive power of Prolog
- How Prolog computes
- Lists and structures in Prolog
- Not so logical Prolog
- The power of search
- Difference lists
- Manipulating Prolog in Prolog

Web site, Moodle etc

- Classes are recorded in Moodle
- Please use the email: `vscosta@dfc.up.pt`

Bibliography

- Leon Sterling and Ehud Shapiro, “The Art of Prolog”, MIT Press.
- Ivan Bratko, “Prolog Programming for Artificial Intelligence”, Addison-Wesley.
- William Clocksin and Chris Mellish, “Programming in Prolog”, Springer-Verlag.

Evaluation

- 40% **Assignment** (submit in December - but with regular checkpoints starting October)
- 60% **Exam** (programming exercises in the computer)

Why Prolog?

- (Very) high level language.
- Great expressiveness.
- Declare, do not give instructions.
- Easy manipulation of symbols.
- Meta-level (parts of programs can be treated as data)
- Theorem proving and natural language processing.
- Aesthetics (Prolog can be beautiful, but it is also easily ruined)

Theorem proving or Automatic Deduction

- *Declarative* knowledge representation, using mathematical logic.
- Inference mechanism (theorem prover) *infers* solutions for problems.
- Except for classes of restricted problems, the search space can grow exponentially.
- Requires intelligent search strategies.
- Not much success after lots of work in the 60s.

What makes logic programming different from theorem proving?

- Logic Programming is *programming*!
- Programmer cares about efficiency and feasibility (does the program run and finishes in a reasonable amount of time?)
- Utilization of a restricted form of logic (Horn clauses).
- Simple and efficient inference mchanism.
- Algorithm = Logic + Control.
- Control can be ignored when we read the program (declarative semantics).

Where is Prolog used? (taken from stackoverflow)

- The first Erlang interpreter was developed in Prolog by Joe Armstrong
- Erlang is a (functional) programming language used to build massively scalable soft real-time systems with requirements on high availability. Some of its uses are in telecoms, banking, e-commerce, computer telephony and instant messaging. Erlang's runtime system has built-in support for concurrency, distribution and fault tolerance

Where is Prolog used? (taken from stackoverflow)

Companies using Erlang:

- **Amazon** uses Erlang to implement SimpleDB, providing database services as a part of the Amazon Elastic Compute Cloud (EC2).
- **Yahoo!** uses it in its social bookmarking service, Delicious, which has more than 5 million users and 150 million bookmarked URLs.
- **Facebook** uses Erlang to power the backend of its chat service, handling more than 100 million active users.
- **WhatsApp** uses Erlang to run messaging servers, achieving up to 2 million connected users per server.
- **T-Mobile** uses Erlang in its SMS and authentication systems.
- **Motorola** is using Erlang in call processing products in the public-safety industry.
- **Ericsson** uses Erlang in its support nodes, used in GPRS and 3G mobile networks worldwide.

Where is Prolog used? (taken from stackoverflow)

- IBM used Prolog to parse natural-language questions into new facts that could be used in the IBM Watson pipeline. In 2011, the system competed in the game Jeopardy! and defeated former winners of the game.
- Prolog was used by NASA to build a software named "clarissa", for the Intl Space Station. Clarissa is a voice user interface for browsing space station procedures.
- With PrologBeans, you can build even a web app (integrated with other languages)
- SICStus Prolog (http://sicstus.sics.se) has been running systems that handle a third of all airline tickets, and helping railways to operate their trains.
- Arezzo - "Clinical decision support"
- InFlow - Social network analysis (looking for terrorists)

Basic concepts in logic

- Computation: formal “reasoning” method.
- Reasoning objects: sentences about the world (*facts* or *rules*).
- “Reasoning”: use the set of sentences to prove a new fact is false or true.

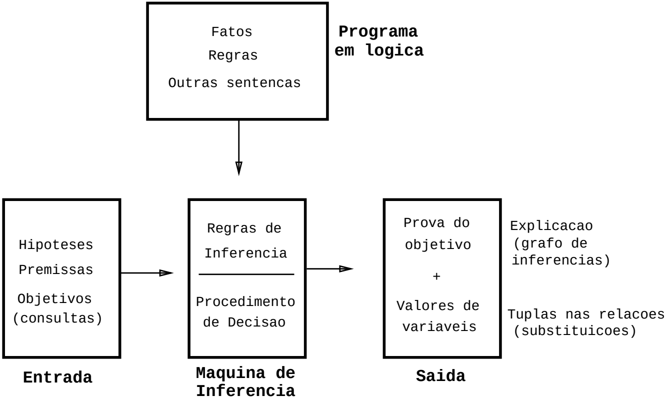
Basic Concepts in Logic

- Basic computational model: *inference machine*.
- *Facts*: basic entities in logic, they are always true (axioms).
- E.g.: the price of this book is 49 euros, In 13/03/98 it was raining at 5pm, the factorial of 3 is 6.
- Form of expressing facts: *relations*.
- Relations: sets of tuples.
- Each tuple: set of objects that share the same features or have the same property.
- E.g.: relation `hair_color(ines,grey)`

Basic Concepts in Logic

- Other forms of defining a relation: “If A is a true, then B is also true”
- Contradictions are not allowed: A is true and A is not true
- *Inference*: determines if a sentence is true through the verification of the truth values of other sentences

Inference Machine



Formal Logic Systems

- Syntax + Semantics.
- *Well formed formula*: expressions syntactically correct.
- Set of allowed symbols:
  - constants;
  - functions;
  - predicates;
  - logical variables;
  - logical connectives: implication, conjunction, disjunction, negation, relational;
  - quantifiers.

Inference Rule

- Generalized Resolution with variable substitution.
- Same rules of propositional logic can be applied.
- Need more rules to deal with the variables.
- Replacing variables with individual constants: $SUBST(\theta, \alpha)$.
- Ex:
  $SUBST(\{x/Sam, y/Pam\}, Likes(x, y)) = Likes(Sam, Pam)$
- 3 new rules:

  Universal Elimin $\frac{\forall v \;\; \alpha}{SUBST(\{v/g\},\alpha)}$

  Existencial Elimin $\frac{\exists v \;\; \alpha}{SUBST(\{v/k\},\alpha)}$

  Introd Existencial $\frac{\alpha}{\exists v \;\; SUBST(\{g/v\},\alpha)}$
- Important: Existencial Elimination must replace variables with constants that have **not** yet appeared in the knowledge base.

Genesis Example

```
homem(adao).
homem(abel).
homem(caim).

mulher(eva).

numa_relacao(adao,eva).
numa_relacao(X,Y):- numa_relacao(Y,X), !.

pai(adao,abel).
pai(adao,caim).
mae(eva, adao).
mae(eva, caim).
```

Diagnosis

```
febre(maria).
espirros(maria).
tosse(maria).

gripe(X) :- febre(X), espirros(X), tosse(X).
alergia(X):- espirros(X), not(febre(X)).

gripe(X) :- gripe(Y), contactou(X,Y), not(imune(X)).
```

Facebook

```
friend(X,Y).
```

More examples

1. Molecule
2. Choosing a car
3. Deck of cards