# *Logic Programming, 16-17*

Inês Dutra
DCC-FCUP
ines@dcc.fc.up.pt (room: 1.31)

September 21, 2016

# *Contents*

- Introduction to logic languages.
- Differences between logic languages and other languages.
- Introduction to Prolog.
- Relation between Prolog and logic.
- Relation between Prolog and search.
- Relation between Prolog and databases (Datalog).
- Prolog: syntax and semantics.
- Basic data types.
- List manipulation. Recursion.
- Difference lists.
- Good practices in Prolog programming.
- Controlling the search space. Cut operator.
- Debugging Prolog programs.
- Constraint Logic Programming.
- Implementation of de Prolog (WAM).
- Advanced programming with Prolog.

# *Bibliography*


- Leon Sterling and Ehud Shapiro, "The Art of Prolog", MIT Press.
- Ivan Bratko, "Prolog Programming for Artificial Intelligence", Addison-Wesley.
- Peter M. Kogge, "The Architecture of Symbolic Computers", McGraw-Hill.
- William Clocksin and Chris Mellish, "Programming in Prolog", Springer-Verlag.
- Hassan Aït-Kaci, "Warren's Abstract Machine – A Tutorial Reconstruction", MIT Press.
- Pascal van Hentenrick, "Constraint Satisfaction in Logic Programming", MIT Press.

# *Bibliography*


- Peter van Roy, "1983-1993–The Wonder Years of Sequential Prolog Implementation", JLP, 1994, V. 20, pp. 385–441.

# *Web site, Moodle etc*

- `http://www.dcc.fc.up.pt/~ines/aulas/1617/PL/PL.html`
- Please, use the email address: `ines@dcc.fc.up.pt`
- Lists of exercises and mini-tests will be put in Moodle.
- Getting in touch with me: send me an email message to arrange day/time.

# *Motivation*

- High level language.
- Great expressiveness.
- Formal.
- Easy manipulation f symbols.
- Theorem proving and Natural language processing.

# *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 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's Watson uses Prolog for the NLP-part.
- Prolog was also 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.
- There's also PrologBeans, which you can use to 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 better.
- DealBuilder - automatic construction of legal documents
- 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*).
- "logic computation": use the set of sentences to predict or prove if another sentence is false or true.

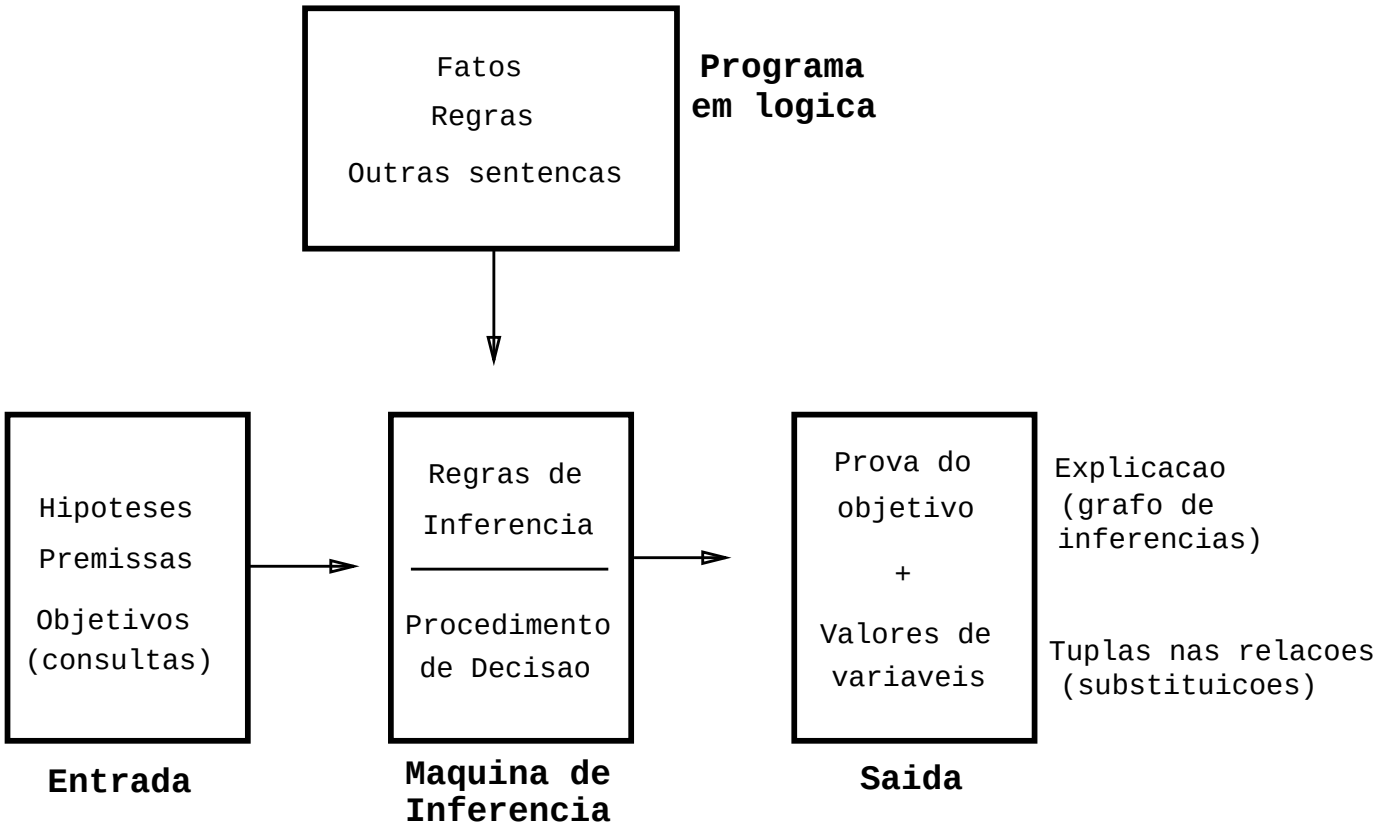
# *Basic Concepts in Logic*

- 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 relation: "If A is a true, then B is also true".
- Contradictions are not allowed: A is true and A is not true.
- *Inference*: concludes if sentence is true through the verification of the truth values of other sentences, without having to search exhaustively the whole set of sentences.
  Ex: If x has a child, then x is a father.
  Vítor is the father of Mariana.
- It is not necessary to look for all tuples father in order to prove that Vítor is a father.

# *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.