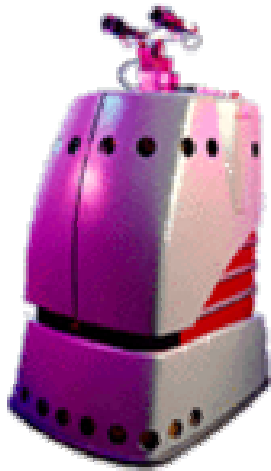


# Prolog

## Artificial Intelligence

### *Lecture 2*

Karim Bouzoubaa



# Content



- Introduction
- Declarative and logic programming
- Example
- Computational model
- Prolog reasoning
- Structure of prolog programs
- Prolog concepts
- Installing Prolog+CG
- Executing a Prolog+CG program

# Introduction



- Prolog is an AI tool
- PROLOG = PROgrammation LOGique
- In 1972 by Alain Colmerauer, Université de Marseille
- Basic, Modular & OO programming
- Prolog is a declarative programming language
  - Subset of logic (1<sup>st</sup> order logic of predicates logic)
- Idea 1: symbolic language
  - Declarative method (represent objects and their relations)
  - Procedural method (automate the logical reasoning)
- Idea 2: the programmer declares what (s)he thinks about the solution using the language of logic
- 
- Consequence: The programmer does not need to code explicitly all instructions

# Pure Prolog



- A Prolog program =
  - $\{P \leftarrow A1 \ \& \ A2 \ \& \ A3 \ \dots \ \& \ An\}$
  - P is true if all  $A_i$  are true
- Prolog Language =
  - Pure prolog (logic programming) +
  - specific instructions (non logic)
    - Flow control
    - Input/output instructions
    - Arithmetic instructions
    - etc.

# Example



- Computational model of classical programming
  - Data + Processing (describes step by step what the machine should do)
- Computational model of Prolog
  - Knowledge + Question → Answer
- Example:
  - K:           saturday, it was raining  
                  it is cold if it rains
  - Q:           was it cold saturday?
  - A:           it was cold saturday
- Prolog
  - K:           rains(saturday).  
                  cold(X) :- rains(X).
  - Q:           cold(saturday)?
  - A:           cold(saturday)

# Computational model



○ K:  
rains(saturday).  
cold(X) :- rains(X).



## Prolog program

← facts  
← rules (clauses)

○ Q:  
cold(saturday)?



## Inference mechanism

- Try to match the question to each rule: unification
- Substitute of variables if necessary: instantiation

○ A:  
cold(saturday)

# Computational model



- Prolog Language =
  - Specific instructions (non logic) +
  - Pure prolog (logic programming)
    - Knowledge + Question → Answer
- Set of Knowledge = Knowledge Base (KB)
  - KB = Fact Base (FB) + Rules
  - Fact: represents a true assertion
  - Rule: conditions to verify a goal (answer a question)
- Question/Answer = Inference Mechanism
  - Consult the KB
  - Generate/Delete Facts/Rules
  - Verifies and satisfies goals

- K:

```
rains(saturday).           Fact1
cold(X) :- rains(X).       Rule1
```
- Q:

```
cold(saturday)?
```
- A:

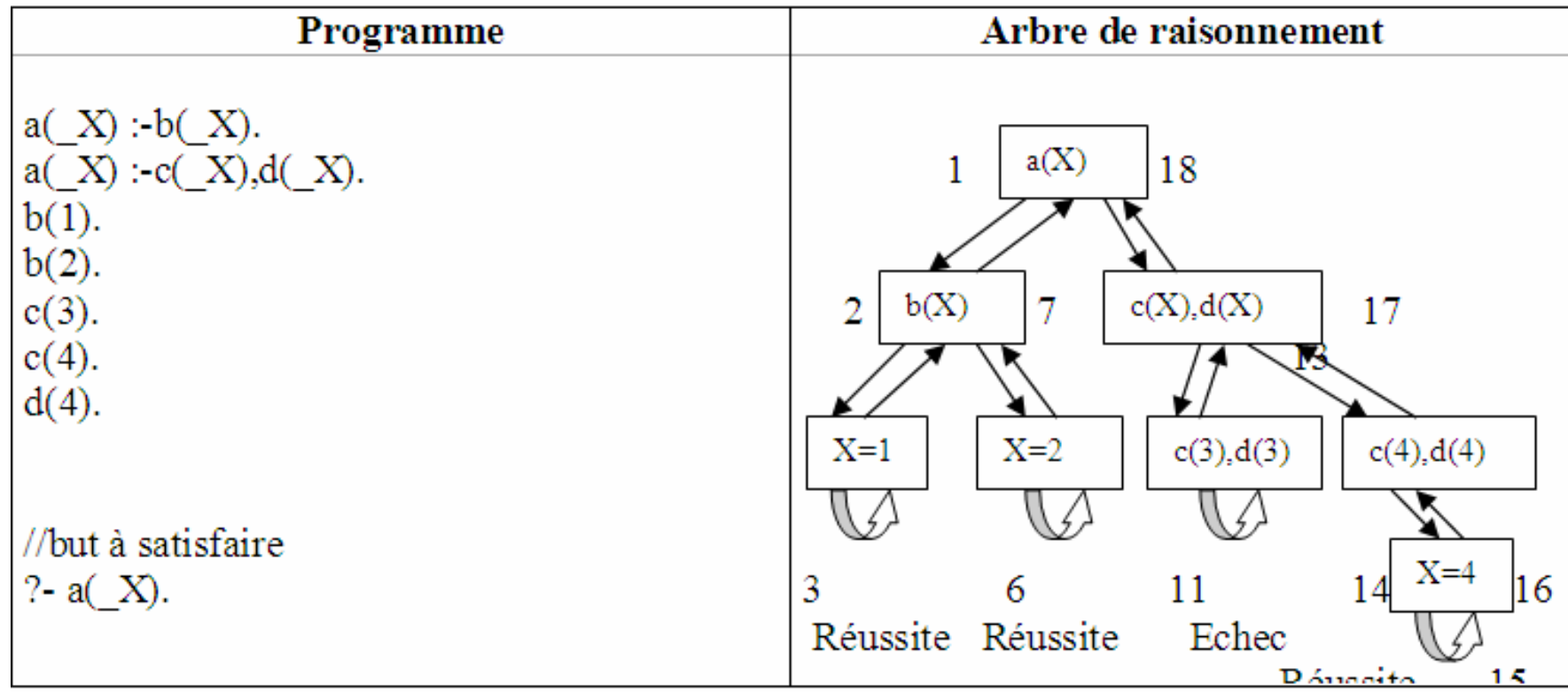
```
cold(saturday)
```
- IM
  - Step 1: cold(saturday)? Goal
  - Step 2: Try to match the question to each rule: **unification**
  - Step 3: Q unifies with Rule1  
→ X = saturday
  - Step 4: new goal rains(saturday)?
  - Step 5: rains(saturday) is a fact  
→ rains(saturday) is true  
→ cold(saturday) is true

# Prolog Reasoning



- Every inference
  - Step 1 (*unification*): unify the question with rules of the KB
  - Step 2 (*reduction*): replaces the original question with other questions derived from the chosen rule
- Exercise:
  - add of `rains(monday)` .
  - Question: `cold(Which_Day)?`
  - a question may have multiple different answers
- Prolog reasoning
  - Recherche dans la liste des clauses (faits et règles) de haut en bas
  - Traitement des buts dans l'ordre de gauche vers la droite
  - Si un but coïncide avec la tête d'une règle, il est remplacé par son corps
  - Suite à un succès ou un échec, Prolog revient *en arrière* à la plus récente liste où une règle peut encore être satisfaite

# Prolog Reasoning



- Prolog Feature *backtracking*: if a track fails, Prolog backs to the last choice and tries another track

# Structure of a Prolog Program



- Prolog+CG Program = Rule\* .
- Rule = Goal [":-" Goal {"," Goal}\*] "." .
- Goal = Java\_Message | SimpleGoal .
- SimpleGoal = Identifier | String | Variable | Term .
- Java\_Message =  
(Identifier | String | Variable) ":" (Term | Variable)

# Prolog Concepts



Concepts	Commentaire	Exemples
<i>Type classique</i>	Entier, réel, booléen, chaîne de caractères	4 8.65 true "chaine"
<i>Identificateur (symbole)</i>	Un identificateur commence avec au minimum deux lettres et peut être suivi par 0 ou plusieurs caractères	ali rabat
<i>Liste</i>	Commence par le caractère '[' et finit par le caractère ']'	[ ali, 3, "chaine" ]
<i>Variable</i>	Une variable commence par un underscore suivi par 0 ou plusieurs caractères. Une variable peut également commencer par une lettre suivi nécessairement par un underscore ou un digit suivi par 0 ou plusieurs caractères.	_X _XY _r_x r2
<i>Terme (prédicat)</i>	Un terme est une structure composée constituant une relation entre plusieurs objets (variable, identificateur, type classique, liste, terme, etc.)	parents_de (_enfant, _pere, _mere) pere_de(_enfant, _pere)
<i>Fait</i>	Un fait est un terme qui se termine par un point.	// les faits (commentaire) parents_de(rachid, ali, aicha). parents_de(imane, ali, aicha). parents_de(jamal, ali, aicha). parents_de(Hicham, khalid, fatima).
<i>Règle</i>	Une règle de type P :-c1,c2,c3. est traduite par : le prédicat P est vrai si les conditions (prédicats) c1, c2 et c3 le sont. Une règle se termine par un point et les conditions sont séparées par des virgules.	// les règles pere_de(_enfant, pere) :- parents_de(_enfant, _pere, _). /* Lorsqu'on remplace un argument par le caractère "_", cela indique que la valeur de ce dernier n'a pas d'importance et peut se traduire par quelle que soit cette valeur */
<i>Prédicat prédéfini</i>	Certains prédicats sont déjà prédéfinis dans Prolog	write, writeln, read, readln, fail, !
<i>Commentaire</i>	Un commentaire en Prolog+CG est similaire au commentaire de Java	// ceci est un commentaire

# Installing and executing Prolog+CG



- Install jdk
- Copy Amine in your HD
- Copy the file runAmine.bat
- Make a shortcut on your desktop
- Double click
- Click Prolog+CG icon

# Executing Prolog+CG



- To execute a Prolog+CG program
  - Launch Prolog+CG
  - Edit the program / open it
  - Create a new console (if not already)
    - `menu console/new`
  - Load the program in memory and compile it
    - `menu console/consult`
  - Type the question/goal in the Console menu
    - Verify the validity of given arguments
      - `parents_de(rachid, ali, aicha).`
    - Search possible instances for variables
      - `pere_de(_X, ali).`
    - To obtain all possible solutions, type ";" then "enter"
- To debug a program (question), activate debug mode
  - `menu console/Debug`
- Example programs:
  - `cold1.pcg / cold2.pcg / familyTree.pcg / meal.pcg`