Administrivia

• mailing list
  – ling364@listserv.arizona.edu
  – you should have received a welcome email
Today’s Topic

• Getting familiar with SWI-Prolog

• Do exercises in class and as part of Homework 1
How to start

• Windows Start Menu
Introduction

• Prolog is a logic programming language
  – allows you to express
    • facts
    • inference rules
  – can hold a database of these two things
    • the database represents a scenario or (possible) world
    • initially, the world is empty
    • you can add facts or inference rules to this database
  – finally, you can ask questions about this world
    • questions involving facts or facts inferred by inference rules
Facts

• Example:
  – Mary is a baseball fan.
  – Pete is a baseball fan.
  – John is a baseball fan.
Facts

• Example:
  – baseball_fan(mary).
  – baseball_fan(pete).
  – baseball_fan(john).

**underscore:** _
can be part of a word, use it to make predicates easier to read, cf. baseballfan

**words begin with a lower case letter**
e.g. mary not Mary
(variables begin with an initial upper case letter)
Facts

• How to add facts to the database (world):
  – `?- assert(<Fact>).`
  
• Means:
  – assert <Fact> is true in this world

• Example:
  – `?- assert(baseball_fan(mary)).`
  – asserts baseball_fan(mary) is true in this world
Facts

- How to get a list of what’s in the world:
  - `?- listing`.

- How to “unadd” or retract a fact from the database:
  - `?- retract(` Fact `)`. 
Facts

- Asking questions:
  - `?- baseball_fan(mary).`
  - Yes
  - `?- baseball_fan(jill).`
  - No

- Assuming our world contains:
  - `baseball_fan(mary).`
  - `baseball_fan(pete).`
  - `baseball_fan(john).`

Prolog uses the **Closed World Assumption**
the world is defined by `?-` listing.
i.e. if a fact isn’t in or inferable from the database,
it isn’t true in our world
Facts

• Questions with logical variables
  – Logic variables are words that begin with an upper case letter
    • e.g. X, Mary, MARY, M33 are all (distinct) variables
    • x, mARY, m33 are all individuals (non-variables)
  – Example:
    • ?- baseball_fan(X).
    • asks what is the value of X such that the proposition baseball_fan(X). is true in this world
    • X = mary ;
    • X = pete ;
    • X = john ;
    • No

  semicolon ;
  indicates disjunction (or)
  used to ask Prolog for more answers
Facts

- Questions with logical variables
  - Example:
    - `?- baseball_fan(x).`
    - No
    - asks if individual x is a baseball fan in this world
  - Example:
    - `?- baseball_fan(X), baseball_fan(Y).`
    - `X = mary, Y = mary ;`
    - `X = mary, Y = pete ;`
    - .... a total of 9 possible answers
  - Example:
    - `?- baseball_fan(X), baseball_fan(X).`
    - has only 3 possible answers
Facts

• Questions with logical variables
  – Example:
    • `?- baseball_fan(X), baseball_fan(Y), \+ X = Y.`
    • asks for what value of X and for what value of Y such that `baseball_fan(X).` is true and `baseball_fan(Y).` is true in this world
    • and it is not (\+) the case that X = Y (equals)
    • How many answers should I get?

Prolog negation \+ a limited form of logical negation
Useful things to know...

• Data entry:
  – you can either enter facts at the Prolog prompt `-`
  – or edit your facts in a text file, give it a name, and load in or “consult” that file `- [<filename>].
Useful things to know...

• The up arrow and down arrow keys can be used at the Prolog prompt to retrieve previous queries
  – you can edit them and resubmit
  – saves typing...
Useful things to know...

• **Getting stuck?**
  – Type `<control>-C`
  – Then type `a` (for abort)
  – gets you back to the Prolog interpreter prompt (`?-`)

• **how to see what the current working directory is?**
  – (the working directory is where your files are stored)
  – *important*: *every machine in the lab is different*
  – `?- working_directory(X,Y).`
    – `X`: current working directory, `Y`: new working directory

• **How to change to a new working directory?**
  – `?- working_directory(X, NEW).`
Homework 1

• Do the following exercises during this lab session and after class as your homework
  – Submit your answers by email
  – Submit all relevant output and databases
    • you can copy and paste from the Prolog window

• Homework Policy (Revisited):
  – due one week from today
  – in my inbox by midnight
Exercise 1a

(4pts)

• Enter Prolog facts corresponding to:
  – Mary is a student
  – Pete is a student
  – Mary is a baseball fan
  – Pete is a baseball fan
  – John is a baseball fan

• Construct the Prolog query corresponding to:
  – who is both a student and a baseball fan?

• Run the query
Exercise 1b

• (2pts)
• Construct the Prolog query corresponding to:
  – who is a baseball fan and not a student?
• Run the query
Relations as Facts

- So far we have just been using predicates with a single argument
- It is useful to have predicates with multiple arguments (separated by a comma) to express relations

• Example:
  - the square is bigger than the circle
  - bigger_than(square,circle).

• Queries:
  - ?- bigger_than(square,X).
  - ?- bigger_than(X,circle).
Rules

• We can write inference rules in Prolog and put them in the database
• Prolog will use them to make inferences when referenced
• Example (adapted from quiz 1):
  – Mary is sleeping
  – John is snoring
  – snoring presupposes sleeping
Rules

• English:
  – Mary is sleeping
  – John is snoring
  – (R1) snoring presupposes sleeping

• Prolog:
  – sleeping(mary).
  – snoring(john).
  – sleeping(X) :- snoring(X).
  – means X is sleeping if X is snoring

Prolog limitations:
- **head** must contain only a single fact
- **body** may contain facts connected by ; and , or negated \+
Rules

- **Prolog:**
  - sleeping(mary).
  - snoring(john).
  - sleeping(X) :- snoring(X).

- **Query:**
  - ?- sleeping(john).

  - notice that there is no fact sleeping(john). in the database, so we cannot immediately conclude it is true.

- but we can use the inference rule for (R1) since the query matches the head of the rule
  - i.e. from:
    - ?- sleeping(john).
    - sleeping(X) :- snoring(X).
  - we can reduce the query to:
    - ?- snoring(john).
    - which matches
      - snoring(john).
  - in the database
- we can conclude then that sleeping(john). is true in this world
Exercise 2

• (4pts)
  – Two sentences are **synonymous** if they have the same meaning, i.e. they have the same truth conditions:
  – (5) The square is bigger than the circle
  – (6) The circle is smaller than the square
  – (chapter 1: page 18)
  – we know
  – (R2) If X is bigger than Y, then Y is smaller than X

• Write the Prolog fact and rule corresponding to (5) and (R2)
• Demonstrate you can conclude (6)
Exercise 3a

• (2pts)
  – Two sentences are contrary if both can’t be true:
  – (7) The square is bigger than the circle
  – (8) The square is smaller than the circle  
    – (chapter 1: page 19)

• Enter the Prolog fact corresponding to (7) and use (R2) from exercise 2

• Construct the Prolog query corresponding to the conjunction of (7) and (8).

• Show the result of the query.
Exercise 3b

• (3pts)
  – Two sentences are contrary if both can’t be true:
  – (7) The square is bigger than the circle
  – (8) The square is smaller than the circle
  – (chapter 1: page 19)

• Enter the Prolog fact corresponding to (8) and (R3)
  – (R3) If X is smaller than Y, then Y is bigger than X

• Construct the Prolog query corresponding to the conjunction of (7) and (8).

• Show the result of the query.
Negation and Prolog

• Prolog has some limitations with respect to \+(negation). We have already mentioned this before:

\[
\text{<Fact}_1 \text{ :-<Fact}_2>. \\
\text{:- means “if”}
\]

Prolog limitations:
- **head** must contain only a single fact
- **body** may contain facts connected by ; and , or negated \+

• Doesn’t allow:
  - \+ baseball_fan(lester).
  - \+ baseball_fan(X) :- never_heard_of_baseball(X).
Negation and Prolog

• Can’t have:
  – baseball_fan(mary).
  – \+ baseball_fan(john).

• 2nd fact is by default true given the Closed World Assumption with database:
  – baseball_fan(mary).

• Also can’t have:
  – baseball_fan(john).
  – \+ baseball_fan(john).
Negation and Prolog

• Also, technically:
  – football_fan(mary).
  is false given the same Closed World Assumption.

• Prolog assumes unknown predicate/arguments are errors
  – Well, actually, Prolog calls them “errors”
  – Example:
    • ?- a(X).
    • ERROR: Undefined procedure: a/1

• To change Prolog’s behavior to the pure Closed World Assumption behavior for predicate a/1:
  – ?- dynamic a/1.
Exercise 4

(4pts) Extra Credit

• From Quiz 1:
  – 3. Given the statement “All crows are black”, give an example of a sentence expressing a tautology involving this statement?
• Possible answer:
  – All crows are black or not all crows are black
• Let Prolog predicate p/0 denote the proposition “All crows are black”
  – ?- assert(p). “All crows are black is true in this world”
• Construct the Prolog version of the tautology
• Show that it is true no matter what the scenario
• Construct a contradictory statement involving p
• Show that it is false no matter what the scenario
Homework Summary

• Homework 1
  – 15 points on offer
  – 4 points extra credit

  – (cf. Quiz 1: 3 pts)