PPS2017 - lab13

Java-Scala Prolog integration

a.a. 2017/2018

Prof. Mirko Viroli

Case 1: Java-Prolog integration

- An interesting application of Prolog is to build the part of an application dealing with complex algorithms and symbolic reasoning
- Coding a data structure and its related algorithms using Prolog facilities
  - providing ways of accessing them through Java
  - handling Input/Output aspects in Java
- An example, execute:
  - java -cp 2p.jar alice.tuprologx.ide.CUIConsole
  - consult('ttt.pl').

Step 1.1: Tic-Tac-Toe in Prolog

- It is the most simple two-player game, known as Tris
- It is solved in Prolog in the code ttt.pl
  - start it by goal   :- ttt.
  - make a move by digiting
    - number + <ENTER> + <CTRL-Z> + <ENTER> (Windows)

      number + <ENTER> + <CTRL-D> (Linux/Mac)

- Independently of the quality of reasoning, Input/Output is sacrificed by Prolog inadequacies
  - may want to realise a Java front-end for handling I/O

Step 1.2: Analyse prolog code

Main predicates in the ttt.pl Prolog code:

- newboard(-B)

- creates a new board and stores it in B

initially [['1', '2', '3'], ['4', '5', '6'], ['7', '8', '9']]

- printboard(+B)

– prints the board on screen (will no more be used in Java!)

- getvalidmove(-Ans, +Board)

– gets a valid move Ans from the human user (will no more be used in Java!)

Step 1.3: Analyse prolog code

- filledsquare(+Board, +Pos)
  - is the position already filled?
- response(+Board,+Type ,-Ans)
  - calculates a new PC move in Ans, Type is either ‘X’ or ‘O’
  - this is really the important part we require from Prolog!
- gameover(+B)
  - is the game finished? It calls boardfilled and threeinarrow
- setsquare (+OBoard, +Pos, +Type, -TBoard)
  - adds ‘X’ or ‘O’ to OBoard, output in TBoard

Step 2: An App for TicTacToe

- In the lab repo, you can already find an implementation of a Java/Scala/Prolog app for ttt
  - https://bitbucket.org/metaphori/pps-lab-scala
  - package u13lab.code (launch TicTacToeApp.java)
  - should possibly need to add library 2p.jar in IntelliJ
    - Project settings → Libraries → add 2p.jar
    - .. or fix dependency in sbt (see slides)
- Architecture
  - TicTacToe is a Java interface for the "model"
  - TicTacToeApp is a Java GUI that implements "view and control" of the game
  - TicTacToeImpl is a Scala implementation that uses Prolog, to be completed (now, it plays "badly", because of too simple implementation of setComputerCell method)
- Task
  - fix setComputerCell: make it call predicate 'response' in the prolog theory!

Step 3: monadic permutations

- In slides we saw how creating permutations of a list in Scala can be done by Java integration
- We now ask you to solve the same problem in pure Scala by a monadic approach
- Simple approach:
  - use for comprehension (recall n-queens problem)
  - use scala collections
- Suggestion:
  - start with the code proposed in Permutation.scala