TP6 - Machine de Turing

Paul Chaignon - Clément Gautrais

November 19, 2013

1 Simulation d'une machine de Turing en Prolog

1.1 Questions

Listing 1: tp6_turing.pro

```
/**
 * Question 1
 * next(+Program, +State0, +Symbol0, -Symbol1, -Dir, -State1)
 */
next(program(Start, Finals, Transitions), State0, Symbol0, Symbol1, Dir, State1):-
  match_next_step(Transitions, State0, Symbol0, Symbol1, Dir, State1).

/**
 * match_next_step(+Transitions, +State0, +Symbol0, -Symbol1, -Dir, -State1)
 */
match_next_step([delta(State0, Symbol0, Symbol1, Dir, State1)|L], State0, Symbol0, Symbol1, Dir, State1).
match_next_step([delta(StateInit, SymbolTete, _, _, _)|L], State0, Symbol0, Symbol1, Dir, State1):-
  match_next_step(L, State0, Symbol0, Symbol1, Dir, State1).


/**
 * Question 2
 * update_tape(+Tape, +Symbol, +Direction, -UpdatedTape)
 */
update_tape(tape(Left, [Head|Right]), Symbol, left, tape(NewLeft, [NewHead,Symbol|Right])):-
  last_elem(Left, NewHead),
  add_blank_left(Left, Left2),
  remove_last(Left2, NewLeft).
update_tape(tape(Left, [Head, NewHead|Right]), Symbol, right, tape(NewLeft, [NewHead|Right])):-
  insert_last(Symbol, Left, NewLeft).
update_tape(tape(Left, [Head]), Symbol, right, tape(NewLeft, [' '])):-
  insert_last(Symbol, Left, NewLeft).

/**
 * last_elem(+List, -NewList)
 */
last_elem([Last], Last).
last_elem([First|List], Last):-
  last_elem(List, Last).

/**
 * remove_last(+List, -NewList)
 */
remove_last([Last], []).
remove_last([First|List], [First|NewList]):-
  remove_last(List, NewList).

/**
 * insert_last(+LastElem, +List, -NewList)
 */
insert_last(Last, [], [Last]).
insert_last(Last, [First|List], [First|NewList]):-
  insert_last(Last, List, NewList).

/**
 * add_blank_left(+List, -NewList)
 * We can't have an empty list.
 */
add_blank_left([Last], [' ', Last]).
add_blank_left([First, Second|List], [First, Second|List]).


/**
 * Question 4
 * run_turing_machine(+Program, +Input, -Output, -FinalState, +Filename)
 */
run_turing_machine(program(Depart, Finals, Trans), BandDroite, Output,
    FinalState, Filename):-
  run_turing_machine_tape(program(Depart, Finals, Trans), Depart,
      tape([' '], BandDroite), tape(Left, Right), FinalState, Dump),
  concat(Left, Right, Output),
  dump_to_mpost(Filename, Dump).

/**
 * run_turing_machine_tape(+Program, +CurrentState, +Input, -Output,
     -FinalState, -Dump)
 */
run_turing_machine_tape(program(Depart, Finals, Trans), CurrentState,
    Input, Input, CurrentState, [(CurrentState, Input)]):-
  membre(CurrentState, Finals).
run_turing_machine_tape(Prog, CurrentState, tape(Left, [Tete|Right]),
    Output, FinalState, [(CurrentState, tape(Left,
    [Tete|Right]))|Dump]):-
  next(Prog, CurrentState, Tete, NewTete, Direct, NextState),
  update_tape(tape(Left, [Tete|Right]), NewTete, Direct, UpdatedTape),
  run_turing_machine_tape(Prog, NextState, UpdatedTape, Output,
      FinalState, Dump).

/**
 * membre(?A, +X)
 */
membre(A, [A|R]).
membre(A, [X|R]):-
  membre(A, R).

```

```
/**
 * concat(+X, +Y, ?T)
 */
concat([], Y, Y).
concat([P|R], Y, [P|T]):-
  concat(R, Y, T).
```

1.2 Tests

Listing 2: tp6_turing_tests.pro

```
copy_prog(Prog), next(Prog, start, 1, X, Y, Z).
  X = ' '
  Y = right
  Z = s2

update_tape(tape([' '], [1, ' ']), ' ', right, UpdatedTape).
  UpdatedTape = tape([' ', ' '], [' ']).

copy_prog(Prog), run_turing_machine(Prog, [1], Output, FinalState).
```