UTF-8

Game Board Operations Module

Alvaro Cabo Ciudad

Table of Contents

Module for Game Board Operations in Prolog

0.1 Predicates

0.1.1 Board definition

Defines a fixed 4x4 game board with pre-set operations.

0.1.2 Game logic predicates

Defines predicates for checking valid movements, selecting cells and directions, applying operations to cells and generating a game path.

Predicate that calculates the resulting position `Pos2` when moving from the given position `Pos` in the specified direction `Dir`.

```
efectuar_movimiento(pos(Row,Col),n,pos(NewRow,Col)) :-
    NewRow is Row-1.
efectuar_movimiento(pos(Row,Col),s,pos(NewRow,Col)) :-
    NewRow is Row+1.
efectuar_movimiento(pos(Row,Col),e,pos(Row,NewCol)) :-
    NewCol is Col+1.
efectuar_movimiento(pos(Row,Col),o,pos(Row,NewCol)) :-
    NewCol is Col-1.
efectuar_movimiento(pos(Row,Col),ne,pos(NewRow,NewCol)) :-
    NewRow is Row-1,
    NewCol is Col+1.
efectuar_movimiento(pos(Row,Col),no,pos(NewRow,NewCol)) :-
    NewRow is Row-1,
    NewCol is Col-1.
efectuar_movimiento(pos(Row,Col),se,pos(NewRow,NewCol)) :-
    NewRow is Row+1,
    NewCol is Col+1.
efectuar_movimiento(pos(Row,Col),so,pos(NewRow,NewCol)) :-
    NewRow is Row+1,
    NewCol is Col-1.
```

Predicate that checks if a move is valid. A move is valid if the resulting position from moving from `Pos` in direction `Dir` results in a valid position on an `N x N` board.

```
movimiento_valido(N,pos(Row,Col),_1) :-
    distance(1,N,Row),
    distance(1,N,Col).
```

Predicate that extracts a cell with position `IPos` from the `Board` list to obtain a `NewBoard` list (without the extracted cell). The operation associated with the selected cell is unified with `Op`.

```
select_cell(IPos,Op,Board,NewBoard) :-
    select_aux(cell(IPos,Op),Board,NewBoard).
```

Predicate that extracts a direction `Dir` from the list of alLed directions `Dirs`, obtaining in `NewDirs` the list of remaining alLed directions. `NewDirs` may be the same list as `Dirs` but with the number of applications of the selected direction decreased by one, or, if this was the last alLed application, without that element.

```
select_dir(Dir,Dirs,NewDirs) :-
    select_aux(dir(Dir,Count),Dirs,TempDirs),
    NewCount is Count-1,
    ( NewCount>0 ->
        NewDirs=[dir(Dir,NewCount)|TempDirs]
    ; NewDirs=TempDirs
    ).
```

Predicate that applies the operation indicated by `Op` to `Valor` to obtain `Valor2`. Given `Op` as `op(Operator,Operand)`, it applies the operation on `Valor`.

```
aplicar_op(op(Op,Val),Valor,Valor2) :-
    Expr=..[Op,Valor,Val],
    Valor2 is Expr.
```

This predicate generates a game path. Its exact functionality needs to be defined in the context of your program.

```
generar_recorrido(Ipos,N,Board,Dirs,Recorrido,FinalValue) :-
    generar_recorrido_aux(Ipos,N,Board,Dirs,Recorrido,0,FinalValue).
```

This predicate generates multiple game paths. Its exact functionality needs to be defined in the context of your program.

```
generar_recorridos(N,Board,DireccionesPermitidas,Recorrido,Valor) :-
    member(cell(Ipos,_1),Board),
    generar_recorrido(Ipos,N,Board,DireccionesPermitidas,Recorrido,Valor).
```

This predicate relates to the game board. Its exact functionality needs to be defined in the context of your program.

```
tablero(N,Tablero,DireccionesPermitidas,ValorMinimo,NumeroDeRutasConValorMinimo).
```

Included at the end of the document all the `:- test` assertions.

0.1.3 Running the tests

Testing can be run using the ciao console or using the integrated Ciao debugger on Emacs

```
?- use_module(library(unittest)).

yes
?- run_tests_in_module('/home/varo/UPM/3ero/ProDec/Pr_2/code.pl').
{Reading /home/varo/UPM/3ero/ProDec/Pr_2/code.pl
WARNING: (lns 323-326) [DireccionesPermitidas,N,NumeroDeRutasConValorMinimo,Table
}
{Reading /home/varo/UPM/3ero/ProDec/Pr_2/code.pl
WARNING: (lns 323-326) [DireccionesPermitidas,N,NumeroDeRutasConValorMinimo,Table
}
{In /home/varo/UPM/3ero/ProDec/Pr_2/code.pl
PASSED: (lns 327-334) efectuar_movimiento/3.
PASSED: (lns 335-337) efectuar_movimiento/3.
FAILED: (lns 338-341) distance/3.
(lns 338-341) distance(_,_,_1) run-time check failure.
Requires in *success*:
   _1=3.
But instead:
   _1=1
```

```
PASSED: (lns 342-344) distance/3.
PASSED: (lns 345-348) select_aux/3.
PASSED: (lns 349-352) movimiento_valido/3.
WARNING: (lns 353-356) select_cell/4. Goal tested failed, but test does not speci
PASSED: (lns 357-360) select_dir/3.
PASSED: (lns 361-364) aplicar_op/3.
PASSED: (lns 365-367) aplicar_op/3.
PASSED: (lns 368-370) generar_recorrido/6.
WARNING: (lns 371-374) generar_recorrido/6. Goal tested failed, but test does not
WARNING: (lns 375-378) generar_recorridos/5. Goal tested failed, but test does no
}

Note: {Total:
Passed: 12 (92.31%) Failed: 1 (7.69%) Precond Failed: 0 (0.00%) Aborted: 0 (0.00%
}


yes
?-
```

0.1.4 efectuar_movimiento/3 tests

Basic Test 1

```
:- test efectuar_movimiento(Pos, Dir, NewPos) :
   (Pos = pos(1,1), Dir = n) => (NewPos = pos(0,1)).
```

Basic Test 2

```
:- test efectuar_movimiento(Pos, Dir, NewPos) :
   (Pos = pos(1,1), Dir = e) => (NewPos = pos(1,2)).
```

0.1.5 distance/3 tests

Basic Test

```
:- test distance(L, H, Val) :
(L = 1, H = 5) => (Val = 3).
```

Advanced Test

```
:- test distance(L, H, Val) :
(L = 0, H = 0) => (Val = 0).
```

0.1.6 select_aux/3 tests

Basic Test

```
:- test select_aux(X, L, NewL) :
   (X = a, L = [a,b,c]) => (NewL = [b,c]).
```

0.1.7 movimiento_valido/3 tests

Basic Test

```
:- test movimiento_valido(N, Pos, _) :
(N = 3, Pos = pos(2,2)) => true.
```

0.1.8 select_cell/4 tests

Basic Test

```
:- test select_cell(IPos, Op, Board, NewBoard) :
(IPos = pos(1,1), Op = op(*,-3), Board = board1(Board)) => (NewBoard is _).
```

0.1.9 select_dir/4 tests

Basic Test

```
:- test select_dir(Dir, Dirs, NewDirs) :
(Dir = n, Dirs = [dir(n, 2), dir(s, 1)], NewDirs = [dir(n, 1), dir(s, 1)]).
```

0.1.10 aplicar_op/3 tests

Basic Test

```
:- test aplicar_op(Op, Valor, Valor2) :
(Op = op(*,3), Valor = 2) => (Valor2 = 6).
```

Basic Test 2

```
:- test aplicar_op(Op, Valor, Valor2) :
(Op = op(-,1), Valor = 5) => (Valor2 = 4).
```

0.2 Usage and interface

- **Library usage:**
  `:- use_module(/home/varo/UPM/3ero/ProDec/Pr_2/code.pl).`
- **Exports:**
  - *Predicates:*
    `author_data/4`, `board1/1`, `efectuar_movimiento/3`, `distance/3`, `movimiento_valido/3`, `select_cell/4`, `select_dir/3`, `select_aux/3`, `aplicar_op/3`, `generar_recorrido/6`, `generar_recorrido_aux/7`, `generar_recorridos/5`, `tablero/5`.

0.3 Documentation on exports

author_data/4: PREDICATE

No further documentation available for this predicate.

board1/1: PREDICATE

Usage: `board1(-Board)`

Given the `Board` returns the initial state of the game board.

efectuar_movimiento/3: PREDICATE

Usage: `efectuar_movimiento(+Pos,+Dir,-Pos2)`

Calculates the resulting position `Pos2` when moving from the given position `Pos` in the specified direction `Dir`.

distance/3: PREDICATE

No further documentation available for this predicate.

movimiento_valido/3: PREDICATE

Usage: `movimiento_valido(+N,+Pos,+Dir)`

Checks if a move is valid. A move is valid if the resulting position from moving from `Pos` in direction `Dir` results in a valid position on an `N` x `N` board.

select_cell/4: PREDICATE

Usage: `select_cell(+IPos,-Op,+Board,-NewBoard)`

Extracts a cell with position `IPos` from the `Board` list to obtain a `NewBoard` list (without the extracted cell). The operation associated with the selected cell is unified with `Op`.

select_dir/3: PREDICATE

Usage: `select_dir(+Dir,+Dirs,-NewDirs)`

Extracts a direction `Dir` from the list of alLed directions `Dirs`, obtaining in `NewDirs` the list of remaining alLed directions. `NewDirs` may be the same list as `Dirs` but with the number of applications of the selected direction decreased by one, or, if this was the last alLed application, without that element.

select_aux/3: PREDICATE

No further documentation available for this predicate.

aplicar_op/3: PREDICATE

Usage: `aplicar_op(+Op,+Valor,-Valor2)`

Applies the operation indicated by `Op` to `Valor` to obtain `Valor2`. Given `Op` as `op(Operator,Operand)`, it applies the operation on `Valor`.

generar_recorrido/6: PREDICATE

Usage: `generar_recorrido(+Ipos,+N,+Board,+Dirs,-Recorrido,-FinalValue)`

Generates a game path `Recorrido` with its final value `FinalValue`, starting from initial position `Ipos` in a `N` x `N` `Board` with possible directions `Dirs`.

generar_recorrido_aux/7: PREDICATE

No further documentation available for this predicate.

generar_recorridos/5: PREDICATE

No further documentation available for this predicate.

tablero/5: PREDICATE

No further documentation available for this predicate.

0.4 Documentation on imports

This module has the following direct dependencies:

- *Internal (engine) modules:*
  `term_basic`, `arithmetic`, `atomic_basic`, `basiccontrol`, `exceptions`, `term_compare`, `term_typing`, `debugger_support`, `basic_props`.
- *Packages:*
  `prelude`, `initial`, `condcomp`, `assertions`, `assertions/assertions_basic`.