UTF-8

Recorridos de valor mínimo en tableros

Alejandro Moreno Ferreiro, A180413

Table of Contents

0.1 Usage and interface 1
0.2 Documentation on exports 1
efectuar_movimiento/3 (pred) 1
posicion_valida/2 (pred) 1
movimiento_valido/3 (pred) 2
select_cell/4 (pred) 2
select_dir/3 (pred) 2
aplicar_op/3 (pred) 2
generar_recorridoAux/7 (pred) 3
generar_recorrido/6 (pred) 3
generar_recorridos/5 (pred) 3
tablero/5 (pred) 4
minimo_repetido/3 (pred) 4
0.3 Documentation on multifiles 4
^^Fcall_in_module/2 (pred) 4
0.4 Documentation on imports 5

Documentación acerca de los predicados realizados

0.1 Usage and interface

- **Library usage:**
  `:- use_module(/home/davidintm/Descargas/code.pl).`
- **Exports:**
  - *Predicates:*
    `efectuar_movimiento/3`, `posicion_valida/2`, `movimiento_valido/3`, `select_cell/4`, `select_dir/3`, `aplicar_op/3`, `generar_recorridoAux/7`, `generar_recorrido/6`, `generar_recorridos/5`, `tablero/5`, `minimo_repetido/3`.
  - *Multifiles:*
    `Σcall_in_module/2`.

0.2 Documentation on exports

efectuar_movimiento/3: PREDICATE

Usage: `efectuar_movimiento(Pos,D,Pos2)`

Predicado que dada una posición y un dirección, efectua un movimiento en el tablero. El movimiento se efectua sumando o restando 1 a la/s coordenada/s seg.

```
efectuar_movimiento(pos(X,Y),e,pos(X,B)) :-
    B is Y+1.
efectuar_movimiento(pos(X,Y),o,pos(X,B)) :-
    B is Y-1.
efectuar_movimiento(pos(X,Y),s,pos(A,Y)) :-
    A is X+1.
efectuar_movimiento(pos(X,Y),n,pos(A,Y)) :-
    A is X-1.
efectuar_movimiento(pos(X,Y),no,pos(A,B)) :-
    A is X-1,
    B is Y-1.
efectuar_movimiento(pos(X,Y),ne,pos(A,B)) :-
    A is X-1,
    B is Y+1.
efectuar_movimiento(pos(X,Y),so,pos(A,B)) :-
    A is X+1,
    B is Y-1.
efectuar_movimiento(pos(X,Y),se,pos(A,B)) :-
    A is X+1,
    B is Y+1.
```

posicion_valida/2: PREDICATE

Usage: `posicion_valida(N,Pos)`

Predicado que dada una posción y un N que define el tamaño del tablero, indica si esa posición dada forma parte del tablero.

```
posicion_valida(N,pos(X,Y)) :-
    between(1,N,X),
    between(1,N,Y).
```

movimiento_valido/3: PREDICATE

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

Predicado que recibe un N que define el tamaño del tablero, una posción y una dirección, indica que la posición resultante de realizar el movimiento es válida.

```
movimiento_valido(N,Pos,Dir) :-
    efectuar_movimiento(Pos,Dir,Pos2),
    posicion_valida(N,Pos2).
```

select_cell/4: PREDICATE

Usage: `select_cell(Pos,Op,Board,NewBoard)`

Predicado que recibe una posción, una operación, y un tablero. Extrae una Pos de la lista Board obteniendo NewBoard (sin dicha celda) y unificando Op con la operación asociada a la respectiva celda.Utiliza member para sacar la celda que se desea eleminar del tablero, y con el select genera el nuevo tablero sin dicha celda.

```
select_cell(Pos,Op,Board,NewBoard) :-
    member(cell(Pos,Op),Board),
    select(cell(Pos,Op),Board,NewBoard).
```

select_dir/3: PREDICATE

Usage: `select_dir(D,Dirs,NewDirs)`

Predicado que recibe una dirección como argumento y extrae una dirección Dir de las permitidas en Dirs , obteniendo en NewDirs la lista de direcciones permitidas que restan. Utiliza el select, para sacar la dirección dado de la lista de direcciones permitidas y genera una nueva. Se utiliza un condicional que evalua: Si solo se puede mover una vez, la lista de direciones disponibles es la generada por select y sino se resta un movimiento y se actualiza la lista de direcciones permitidas.

```
select_dir(D,Dirs,NewDirs) :-
    select(dir(D,X),Dirs,Resto),
    ( X==1 ->
        NewDirs=Resto
    ; Actualizado is X-1,
      NewDirs=[dir(D,Actualizado)|Resto]
    ).
```

aplicar_op/3: PREDICATE

Usage: `aplicar_op(Op,Valor,ValorResultado)`

Predicado que dada una operación, aplica la operación correspondiente al Valor, generando un nuevo valor.

```
aplicar_op(op(+,Operando),Valor,Valor2) :-
    Valor2 is Valor+Operando.
aplicar_op(op(-,Operando),Valor,Valor2) :-
    Valor2 is Valor-Operando.
aplicar_op(op(*,Operando),Valor,Valor2) :-
    Valor2 is Valor*Operando.
aplicar_op(op(//,Operando),Valor,Valor2) :-
    Valor2 is Valor//Operando.
```

generar_recorridoAux/7: PREDICATE

Usage:
`generar_recorridoAux(Pos,N,Board,DireccionesPermitidas,Recorrido,Valor,V)`

Predicado que utiliza el predicado select_dir para seleccionar una dirección (Dir) de la lista de direcciones permitidas (DireccionesPermitidas) y devuelve la lista de direcciones restantes (Dirs). Verifica si el movimiento (Dir) es válido utilizando el predicado movimiento_valido con los argumentos N, Ipos y Di. Utiliza el predicado efectuar_movimiento para obtener la nueva posición (Npos) a partir de la posición actual (Ipos) y la irección (Dir). Utiliza el predicado select_cell para seleccionar una celda (Op) en el tablero (Board) basándose en la nueva posición (Npos) y devuelve el nuevo tablero (NBoard). Aplica la operacin (Op) a los valores actuales (Valor) y obtiene el nuevo valor (NuevoValor) utilizando el predicado aplicar_op. Llama recursivamente al predicado generar_recorridoAux con la nueva posición (Npos), el mismo tamao del recorrido (N), el nuevo tablero (NBoard), las direcciones restantes (Dirs), la lista de recorrido actualizada [(Npos, NuevoValor)|Recorrido], el nuevo valor (NuevoValor) y una variable de salida (V).

```
generar_recorridoAux(_1,_2,[],_3,[],Valor,Valor).
generar_recorridoAux(Ipos,N,Board,DireccionesPermitidas,[(Npos,NuevoValor)|Reco
    select_dir(Dir,DireccionesPermitidas,Dirs),
    movimiento_valido(N,Ipos,Dir),
    efectuar_movimiento(Ipos,Dir,Npos),
    select_cell(Npos,Op,Board,NBoard),
    aplicar_op(Op,Valor,NuevoValor),
    generar_recorridoAux(Npos,N,NBoard,Dirs,Recorrido,NuevoValor,V).
```

generar_recorrido/6: PREDICATE

Usage: `generar_recorrido(PosN,N,Board,DireccionesPermitidas,Lista,V)`

El predicado selecciona la celda correspondiente a la posición inicial (Ipos) en el tablero (Board) utilizando el predicado select_cell. Luego, se aplica una operación al valor 0 utilizando el predicado aplicar_op y se obtiene un nuevo valor (NuevoValor). A continuación, se llama al predicado auxiliar generar_recorridoAux con los parámetros actualizados (Ipos, N, NBoard, DireccionesPermitidas, Recorrido, NuevoValor, V). Este predicado auxiliar se encarga de realizar la recursión para generar el recorrido completo en el tablero. El resultado final del recorrido se obtiene a través de la variable de salida V.

```
generar_recorrido(Ipos,N,Board,DireccionesPermitidas,[(Ipos,NuevoValor)|Recorri
    select_cell(Ipos,Op,Board,NBoard),
    aplicar_op(Op,0,NuevoValor),
    generar_recorridoAux(Ipos,N,NBoard,DireccionesPermitidas,Recorrido,NuevoVal
```

generar_recorridos/5: PREDICATE

Usage: `generar_recorridos(N,Board,DireccionesPermitidas,Recorrido,Valor)`

Predicado que utiliza el member para seleccionar una de las celdas y una vez seleccionada, se generan los recorridos llamando al predicado generar_recorrido.

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

tablero/5: PREDICATE

Usage:
`tablero(N,Tablero,DireccionesPermitidas,ValorMinimo,NumeroDeRutasConValorMinimo)`

Predicado que utiliza el predicado findall para generar una lista (Valores) de todos los valores obtenidos al llamar al predicado generar_recorridos y un valor no utilizado como salida. Verifica que la lista de valores (Valores) no esté vacia utilizando '=' y llama al predicado minimo_repetido.

```
tablero(N,Tablero,DireccionesPermitidas,ValorMinimo,NumeroDeRutasConValorMinimo)
    findall(Valor,generar_recorridos(N,Tablero,DireccionesPermitidas,_1,Valor),
    Valores\=[],
    minimo_repetido(Valores,ValorMinimo,NumeroDeRutasConValorMinimo).
```

minimo_repetido/3: PREDICATE

Usage: `minimo_repetido(Lista,Minimo,Repeticiones)`

Predicado que dada un lista, selecciona el valor minimo y cuantas veces se repite. Realiza una llamada recursiva al predicado minimo_repetido con el resto de la lista Xs, obteniendo el valor mínimo Min y el número de repeticiones hasta ese punto Repe. Se comparan los valores X y Min para determinar: si X < Min, se actualiza el minimo y las repeticiones, si X = Min, se aumentan las repeticiones y si X > Min se actualizan los datos Minimo y Repeticiones.

```
minimo_repetido([],_1,0).
minimo_repetido([X|Xs],Minimo,Repeticiones) :-
    minimo_repetido(Xs,Min,Repe),
    ( X@<Min ->
        Minimo=X,
        Repeticiones=1
    ; ( X=Min ->
          Repeticiones is Repe+1,
          Minimo=Min
      ; Repeticiones=Repe,
        Minimo=Min
      )
    ).
```

0.3 Documentation on multifiles

Σcall_in_module/2: PREDICATE

No further documentation available for this predicate. The predicate is *multifile*.

0.4 Documentation on imports

This module has the following direct dependencies:

– *Application modules:*

`operators`, `dcg_phrase_rt`, `datafacts_rt`, `dynamic_rt`, `classic_predicates`.

– *Internal (engine) modules:*

`term_basic`, `arithmetic`, `atomic_basic`, `basiccontrol`, `exceptions`, `term_compare`, `term_typing`, `debugger_support`, `hiord_rt`, `stream_basic`, `io_basic`, `runtime_control`, `basic_props`.

– *Packages:*

`prelude`, `initial`, `condcomp`, `classic`, `runtime_ops`, `dcg`, `dcg/dcg_phrase`, `dynamic`, `datafacts`, `assertions`, `assertions/assertions_basic`, `regtypes`.