Tema IA Pîslari Vadim 342C3

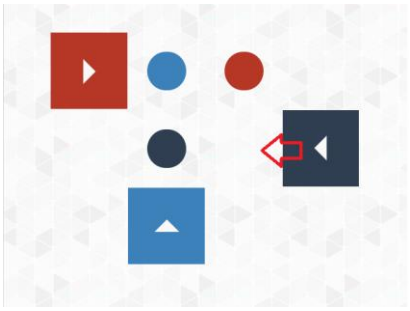
1. Reprezentarea Strips:

Predicate:

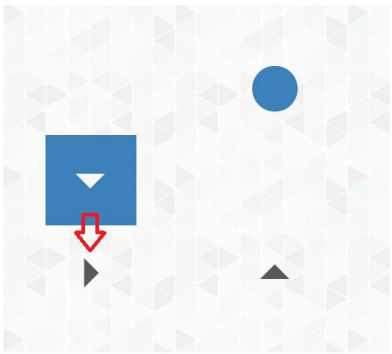
- *pos (Pătrat, X, Y)*
- *dir (Pătrat, Direcție)*
- *delta (Direcție, X, Y, X1, Y1)*
- *empty (X, Y)*
- *schimbator (X, Y, Direcție)*
- *fara schimbator (X, Y)*
- *between (Jos, X, Sus)*

Operatori:

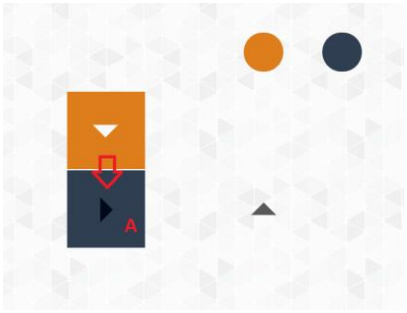
1. miscareSimpla(x, y) {se misca patratul si nu e nimic pe noua pozitie}
   LP(Precond): pos (patrat, x, y) ^ dir(patrat, dir) ^ delta(dir, x, y, x1, y1) ^ empty(x1, y1) ^ fara_schimbator(x1, y1)
   LE(Delete): pos (patrat, x, y) ^ dir(patrat, dir) ^ delta(dir, x, y, x1, y1) ^ empty(x1,y1)
   LA(Add): pos (patrat, x1, y1) ^ dir(patrat, dir) ^ empty(x, y)



2. miscareSch(x, y) { se misca patratul si e un schimbator pe noua pozitie dar nu e un alt patrat}
   LP: pos (patrat, x, y) ^ dir(patrat, dir) ^ delta(dir, x, y, x1, y1) ^ empty(x1, y1) ^ schimbator(x1, y1, dir1)
   LE: pos (patrat, x, y) ^ dir(patrat, dir) ^ delta(dir, x, y, x1, y1) ^ empty(x1, y1)
   LA: pos (patrat, x1, y1) ^ dir(patrat, dir1) ^ empty(x, y)

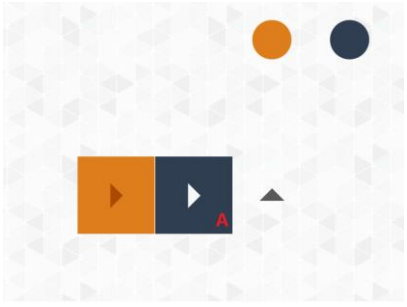


3. miscarePat(x,y) {se misca patratul si e un patrat pe noua pozitie}
   LP: pos (patrat, x, y) ^ dir(patrat, dir) ^ delta(dir, x, y, x1, y1) ^ pos(patrat1,x1, y1) ^ delta(dir, x1, y1, x2, y2) ^ fara_schimbator(x1, y1) ^ fara_schimbator(x2, y2)
   LE: pos (patrat, x, y) ^ dir(patrat, dir) ^ delta(dir, x, y, x1, y1) ^ pos(patrat1,x1, y1) ^ delta(dir, x1, y1, x2, y2)
   LA: pos (patrat, x1, y1) ^ dir(patrat, dir) ^ delta(dir, x, y ,x1, y1) ^ pos(patrat1, x2,y2) ^ empty(x, y)



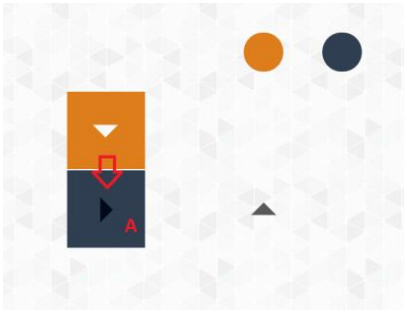
(Pe poziția unde se află pătratul A nu este schimbator)

4. miscarePat_nextSch(x,y) {se misca patratul si e un patrat pe noua pozitie iar odata impins acel patrat va ajunge pe un schimbator}
   LP: pos (patrat, x, y) ^ dir(patrat, dir) ^ delta(dir, x, y, x1, y1) ^ pos(patrat1,x1, y1) ^ delta(dir, x1, y1, x2, y2) ^ fara_schimbator(dir, x1, y1) ^ dir(patrat1, dir1) ^ schimbator(dir2, x2, y2)
   LE: pos (patrat, x, y) ^ dir(patrat, dir) ^ delta(dir, x, y, x1, y1) ^ pos(patrat1,x1, y1) ^ delta(dir, x1, y1, x2, y2) ^ dir(patrat1, dir1)
   LA: pos (patrat, x1, y1) ^ dir(patrat, dir) ^ pos(patrat1, x2,y2) ^ dir(patrat1, dir2) ^ empty(x, y)



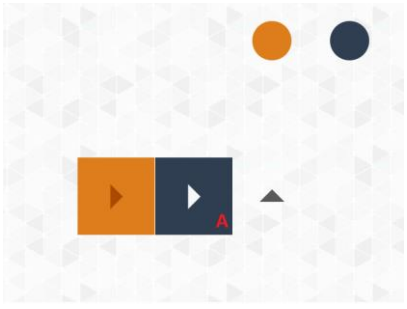
(Pe poziția unde se află pătratul A nu este schimbator)

5. miscarePatSch(x,y) {se misca patratul si e si un patrat si un schimbator}
   LP: pos (patrat, x, y) ^ dir(patrat, dir) ^ delta(dir, x, y, x1, y1) ^ pos(patrat1,x1, y1) ^ delta(dir, x1, y1, x2, y2) ^ fara_schimbator(x1, y1) ^ schimbator(x2, y2, dir1)
   LE: pos (patrat, x, y) ^ dir(patrat, dir) ^ delta(dir, x, y, x1, y1) ^ pos(patrat1,x1, y1) ^ delta(dir, x1, y1, x2, y2)
   LA: pos (patrat, x1, y1) ^ dir(patrat, dir1) ^ pos(patrat1, x2,y2) ^ empty(x, y)



(Pe poziția unde se află pătratul A este schimbator spre dreapta)

6. miscarePatSch_nextSch(x,y) {se misca patratul si e si un patrat si un schimbator iar odata impins acel patrat va ajunge pe un alt schimbator }
   LP: pos (patrat, x, y) ^ dir(patrat, dir) ^ delta(dir, x, y, x1, y1) ^ pos(patrat1,x1, y1) ^ delta(dir, x1, y1, x2, y2) ^ schimbator(dir3, x1, y1) ^ dir(patrat1, dir1) ^ schimbator(dir2, x2, y2)
   LE: pos (patrat, x, y) ^ dir(patrat, dir) ^ delta(dir, x, y, x1, y1) ^ pos(patrat1,x1, y1) ^ delta(dir, x1, y1, x2, y2) ^ dir(patrat1, dir1)
   LA: pos (patrat, x1, y1) ^ dir(patrat, dir3) ^ pos(patrat1, x2,y2) ^ dir(patrat1, dir2) ^ empty(x, y)



(Pe poziția unde se află pătratul A este schimbator)

2. Definitie de tip ADL

4 operatori in functie de directia miscarii

1. (:action moveEast

```
 :parameters
        square: Patratul, x: CoordonataX, y:CoordonataY, x1: CoordonataX1,
y1:CoordonataY1
        dir: directia, east: Directia Est, _x: CoordonataX2, _y:CoordonataY2, _x2:
CoordonataX3, _y2:CoordonataY3
        _x3: CoordonataX4, _y3:CoordonataY4
 :precondition
        exists square and(
                pos (square, x, y)
                dir (square, dir)
        )
 :effect
        forall square
            when (
                and(
                   pos(square, x, y),
                   dir(square, east),
                   delta(east, x, y, x1, y1)
                )
            and(
                pos(square, x1, y1),
                forall square
                    when (
                        and(
                            pos(square, _x1,_y1)
                            not exists square
                                and (
                                    not pos(square, _x2,_y1)
                                    between (x, _x2, _x1)
                                )
                            delta(east, _x1, _y1, _x3, _y3)
                        )
                pos(square,_x3, y3)
                )
)
```

2. (:action moveWest

```
:parameters
        square: Patratul, x: CoordonataX, y:CoordonataY, x1: CoordonataX1,
y1:CoordonataY1
        dir: directia, west: Directia Vest, _x: CoordonataX2, _y:CoordonataY2, _x2:
CoordonataX3, _y2:CoordonataY3
        _x3: CoordonataX4, _y3:CoordonataY4
:precondition
        exists square and(
                pos (square, x, y)
                dir (square, dir)
        )
:effect
        forall square
                when (
                        and(
                           pos(square, x, y),
                           dir(square, west),
                           delta(west, x, y, x1, y1)
                        )
                 and(
                        pos(square, x1, y1),
                        forall square
                                when (
                                        and(
                                                pos(square, _x1,_y1)
                                                not exists square
                                                    and (
                                                            not pos(square, _x2,_y1)
                                                            between (_x1, _x2, x)
                                                    )
                                               delta(west, _x1, _y1, _x3, _y3)
                                        )
                        pos(square,_x3, y3)
                        )
)
```

3. (:action moveNorth

```
:parameters
        square: Patratul, x: CoordonataX, y:CoordonataY, x1: CoordonataX1,
y1:CoordonataY1
        dir: directia, north: Directia Nord, _x: CoordonataX2, _y:CoordonataY2, _x2:
CoordonataX3, _y2:CoordonataY3
        _x3: CoordonataX4, _y3:CoordonataY4
 :precondition
        exists square and(
                pos (square, x, y)
                dir (square, dir)
        )
 :effect
        forall square
            when (
                and (
                   pos(square, x, y),
                   dir(square, north),
                   delta(north, x, y, x1, y1)
                )
             and (
                pos(square, x1, y1),
                forall square
                    when(
                        and(
                            pos(square, _x1,_y1)
                            not exists square
                                and (
                                    not pos(square, _x1,_y2)
                                    between (y, _y2, _y1)
                                )
                           delta(north, _x1, _y1, _x3, _y3)
                        )
                pos(square,_x3, y3)
                )
)
```

4. (:action moveSouth

```
:parameters
        square: Patratul, x: CoordonataX, y:CoordonataY, x1: CoordonataX1,
y1:CoordonataY1
        dir: directia, south: Directia South, _x: CoordonataX2, _y:CoordonataY2, _x2:
CoordonataX3, _y2:CoordonataY3
        _x3: CoordonataX4, _y3:CoordonataY4
 :precondition
        exists square and(
                pos (square, x, y)
                dir (square, dir)
        )
 :effect
        forall square
            when (
                and (
                   pos(square, x, y),
                   dir(square, south),
                   delta(south, x, y, x1, y1)
                )
             and (
                pos(square, x1, y1),
                forall square
                    when(
                        and(
                            pos(square, _x1,_y1)
                            not exists square
                                and (
                                    not pos(square, _x1,_y2)
                                    between (_y1, _y2, y)
                                )
                           delta(south, _x1, _y1, _x3, _y3)
                        )
                pos(square,_x3, y3)
                )
)
```

Ex 4/Bonus

Euristica aleasa este suma euclidiana intre pozitia fiecarui patrat si pozitia finala (Goal). Iar algoritmul ales este IDA*.

Rezultatele checkerului:

```
Summary
22 / 22 operation tests successful
20 / 20 correct plans
16 / 20 efficient searches,
3 / 20 less efficient searches 51% / 177% / 34% over the limit, median 51%
1 / 20 out of bounds searches

Process finished with exit code 0
```