GetOut Game In Spin

김대희, 한기홍

목 차

1. Properties

properties of the system to verify

1. 1번 블록 부터 상하좌우 순으로 움직일 수 있는지 검사.
2. 움직일 수 있는 경우, 한 칸 이동 후 다음 블록을 1번 규칙으로 적용.
3. 5번 블럭이 34 35 36 에 위치해 있거나, 31 라인에 위치해있고 장애물 없을 시 종료.
4. 더 이상 모든 블럭이 움직일 수 없을 때 종료.

2. GetOut Game

```
1   bit map[36] = { 0 };
2
3
4   int blocks[25] = { 4, 5,
5                      10, 11,
6                      12, 13,
7                      18, 19,
8                      25, 26, 27,
9                      32, 33,
10                     24, 30,
11                     8, 14,
12                     3, 9,
13                     15, 21,
14                     16, 22,
15                     28, 34 };
16
17
18  // 초기 셋팅
19  byte init_i;
20  proctype initGame()
21  {
22          init_i = 0;
23  again:
24          do
25          :: timeout  -> goto again
26          :: if
27             :: ( init_i >= 25 ) -> goto done
28             :: else              -> map[ blocks[init_i] ] = 1 ; init_i = init_i + 1
29             fi
30          od
31
32  done:
33          run getOut()
34  }
```

Index 0, 1 이 블록 1.
Index 2, 3 이 블록 2.
…

Timeout?
Exception

2. GetOut Game

```
// 전체 map 검사.
int block_i;
proctype getOut()
{
again:
        do
        :: timeout   -> goto again
        :: if
          :: ( blocks[8]==15  && blocks[9]==16 && blocks[10]==17 ) -> break ; printf("end\n")
          fi
        :: if
          :: ( blocks[8]==14  && blocks[9]==15 && blocks[10]==16 && map[17]==0 ) -> break ; printf("end\n")
          fi
        :: if
          :: ( blocks[8]==13  && blocks[9]==14 && blocks[10]==15 && map[16]==0 && map[17]==0 ) -> break ; printf("end\n")
          fi
        :: if
          :: ( blocks[8]==12  && blocks[9]==13 && blocks[10]==14 && map[15]==0 && map[16]==0 && map[17]==0 ) -> break ; printf("end\n")
          fi
again2:
        :: block_i = 0

        :: do
          :: timeout   -> goto again2
          :: if
            :: ( block_i >= 25 ) -> break
            fi
          :: if
            :: ( block_i == 4 ) -> atomic { run checkThreeBlock(blocks[8], blocks[9], blocks[10], block_i)  ; block_i = block_i+3 }
            :: else             -> atomic { run checkTwoBlock(blocks[block_i], blocks[block_i+1], block_i) ; block_i = block_i+2 }
            fi
          od
        od
}
```

탈출 조건

3 블록, 2 블록 선택.

2. GetOut Game

```
// 2개짜리 블록, 상하좌우 확인 후 이동.
proctype checkTwoBlock(int index1; int index2; int blockNum)
{
        int checkWidthHeight = index2 - index1;
        if
        :: ( checkWidthHeight == 1 ) -> atomic { run check2Width(index1, index2, blockNum) }
        :: else                      -> atomic { run check2Height(index1, index2, blockNum) }
        fi
}
```

가로블록, 세로블록 선택

2. GetOut Game

음수 값으로 map 영역 벗어나는 예외처리, 이미 블록 존재 등 검사.

```
int tmp1;
int tmp2;
proctype check2Width(int index1; int index2; int blockNum)
{
        tmp1 = 0 ; tmp2 = 0;

        // 상
        tmp1 = index1 - 6 ; tmp2 = index2 - 6
        if
        :: ( tmp1<0 || tmp2<0 || map[tmp1]==1 || map[tmp2]==1 )
          ->

        // 하
          tmp1 = index1 + 6 ; tmp2 = index2 + 6 ;
          if
          :: ( tmp1>35 || tmp2>35 || map[tmp1]==1 || map[tmp2]==1 || tmp1<0 || tmp2<0)
            ->

        // 좌
            tmp1 = index1 - 1 ; tmp2 = index2 - 1 ;
            if
            :: (tmp1==-1 || tmp1==5 || tmp1==11 || tmp1==17 || tmp1==23 || tmp1==29 || map[tmp1]==1 || tmp1<0 || tmp2<0)
                ->

        // 우
                tmp1 = index1 + 1 ; tmp2 = index2 + 1 ;
                if
                :: (tmp2==6 || tmp2==12 || tmp2==18 || tmp2==24 || tmp2==30 || tmp2==36 || map[tmp2]==1 || tmp1<0 || tmp2<0 )
                  -> goto done
                :: else
                ->
                map[index1]=0 ; map[index2]=0 ; blocks[blockNum]=tmp1 ; blocks[blockNum+1]=tmp2 ; map[blockNum]=1 ; map[blockNum+1]=1 // 우, 이동
                fi
            : else
            ->
            map[index1]=0 ; map[index2]=0 ; blocks[blockNum]=tmp1 ; blocks[blockNum+1]=tmp2 ; map[blockNum]=1 ; map[blockNum+1]=1 // 좌, 이동
            fi
        :: else
        ->
        map[index1]=0 ; map[index2]=0 ; blocks[blockNum]=tmp1 ; blocks[blockNum+1]=tmp2 ; map[blockNum]=1 ; map[blockNum+1]=1 // 하, 이동
        fi
        :: else
        ->
        map[index1]=0 ; map[index2]=0 ; blocks[blockNum]=tmp1 ; blocks[blockNum+1]=tmp2 ; map[blockNum]=1 ; map[blockNum+1]=1  // 상, 이동
        fi
done:
}
```

이동가능 시 한 칸 이동. Map도 1로 표시.

2. GetOut Game

```
proctype check2Height(int index1; int index2; int blockNum)
{
        tmp1 = 0 ; tmp2 = 0;

        // 상
        tmp1 = index1 - 6 ; tmp2 = index2 - 6
        if
        :: ( tmp1<0 || map[tmp1]==1 )
          ->

        // 하
          tmp1 = index1 + 6 ; tmp2 = index2 + 6 ;
          if
          :: ( tmp2>35 || map[tmp2]==1 || tmp1<0 || tmp2<0 )
            ->

        // 좌
            tmp1 = index1 - 1 ; tmp2 = index2 - 1 ;
            if
            :: (tmp1==-1 || tmp1==5 || tmp1==11 || tmp1==17 || tmp1==23 || map[tmp1]==1 ||
                  tmp2==5 || tmp2==11 || tmp2==17 || tmp2==23 || tmp2==29 || map[tmp2]==1 || tmp1<0 || tmp2<0 )
                 ->

         // 우
                  tmp1 = index1 + 1 ; tmp2 = index2 + 1 ;
                  if
                  :: (tmp1==6 || tmp1==12 || tmp1==18 || tmp1==24 || tmp1==30 || map[tmp1]==1 ||
                     tmp2==12 || tmp2==18 || tmp2==24 || tmp2==30 || tmp2==36 || map[tmp2]==1 || tmp1<0 || tmp2<0 )
                     -> goto done
                  :: else
                  ->
                  map[index1]=0 ; map[index2]=0 ; blocks[blockNum]=tmp1 ; blocks[blockNum+1]=tmp2 ; map[blockNum]=1 ; map[blockNum+1]=1 // 우, 이동
                  fi

            :: else
            ->
            map[index1]=0 ; map[index2]=0 ; blocks[blockNum]=tmp1 ; blocks[blockNum+1]=tmp2 ; map[blockNum]=1 ; map[blockNum+1]=1 // 좌, 이동
            fi

          :: else
          ->
          map[index1]=0 ; map[index2]=0 ; blocks[blockNum]=tmp1 ; blocks[blockNum+1]=tmp2 ; map[blockNum]=1 ; map[blockNum+1]=1 // 하, 이동
          fi

        :: else
        ->
        map[index1]=0 ; map[index2]=0 ; blocks[blockNum]=tmp1 ; blocks[blockNum+1]=tmp2 ; map[blockNum]=1 ; map[blockNum+1]=1  // 상, 이동
        fi
done:
}
```

가로 블록 때 와의 차이점.

2. GetOut Game

```
// 3개짜리 블록, 상하좌우 확인 후 이동.
int tmp3;
proctype checkThreeBlock(int index1; int index2; int index3; int blockNum)
{
        tmp1 = 0 ; tmp2 = 0 ; tmp3 = 0 ;

        // 상
        tmp1 = index1 - 6 ; tmp2 = index2 - 6 ; tmp3 = index3 - 6 ;
        if
        :: ( tmp1<0 || tmp2<0 || tmp3<0 || map[tmp1]==1 || map[tmp2]==1 || map[tmp3]==1 )
          ->

        // 하
          tmp1 = index1 + 6 ; tmp2 = index2 + 6 ; tmp3 = index3 + 6 ;
          if
          :: ( tmp1>35 || tmp2>35 || tmp3>35 || map[tmp1]==1 || map[tmp2]==1 || map[tmp3]==1 || tmp1<0 || tmp2<0 || tmp3<0)
            ->

        // 좌
            tmp1 = index1 - 1 ; tmp2 = index2 - 1 ; tmp3 = index3 - 1 ;
            if
            :: (tmp1==-1 || tmp1==5 || tmp1==11 || tmp1==17 || tmp1==23 || tmp1==29 || map[tmp1]==1 || tmp1<0 || tmp2<0 || tmp3<0)
                    ->

         // 우
                    tmp1 = index1 + 1 ; tmp2 = index2 + 1 ; tmp3 = index3 + 1 ;
                    if
                    :: (tmp3==6 || tmp3==12 || tmp3==18 || tmp3==24 || tmp3==30 || tmp3==36 || map[tmp3]==1 || tmp1<0 || tmp2<0 || tmp3<0)
                      -> goto done
                    :: else
                    ->
                    map[index1]=0 ; map[index2]=0 ; map[index3]=0 ; blocks[blockNum]=tmp1 ; blocks[blockNum+1]=tmp2 ; blocks[blockNum+2]=tmp3 ; map[blockNum]=1 ; map[blockNum+1]=1 ; map[blockNum+2]=1 // 우, 이동
                    fi

              :: else
              ->
              map[index1]=0 ; map[index2]=0 ; map[index3]=0 ; blocks[blockNum]=tmp1 ; blocks[blockNum+1]=tmp2 ; blocks[blockNum+2]=tmp3 ; map[blockNum]=1 ; map[blockNum+1]=1 ; map[blockNum+2]=1 // 좌, 이동
              fi

            :: else
            ->
           map[index1]=0 ; map[index2]=0 ; map[index3]=0 ; blocks[blockNum]=tmp1 ; blocks[blockNum+1]=tmp2 ; blocks[blockNum+2]=tmp3 ; map[blockNum]=1 ; map[blockNum+1]=1 ; map[blockNum+2]=1  // 하, 이동
            fi

        :: else
        ->
        map[index1]=0 ; map[index2]=0 ; map[index3]=0 ; blocks[blockNum]=tmp1 ; blocks[blockNum+1]=tmp2 ; blocks[blockNum+2]=tmp3 ; map[blockNum]=1 ; map[blockNum+1]=1 ; map[blockNum+2]=1  // 상, 이동
        fi
done:
}
```

인덱스 3개, 각 예외처리 조건 하나 씩 증가.

2. GetOut Game

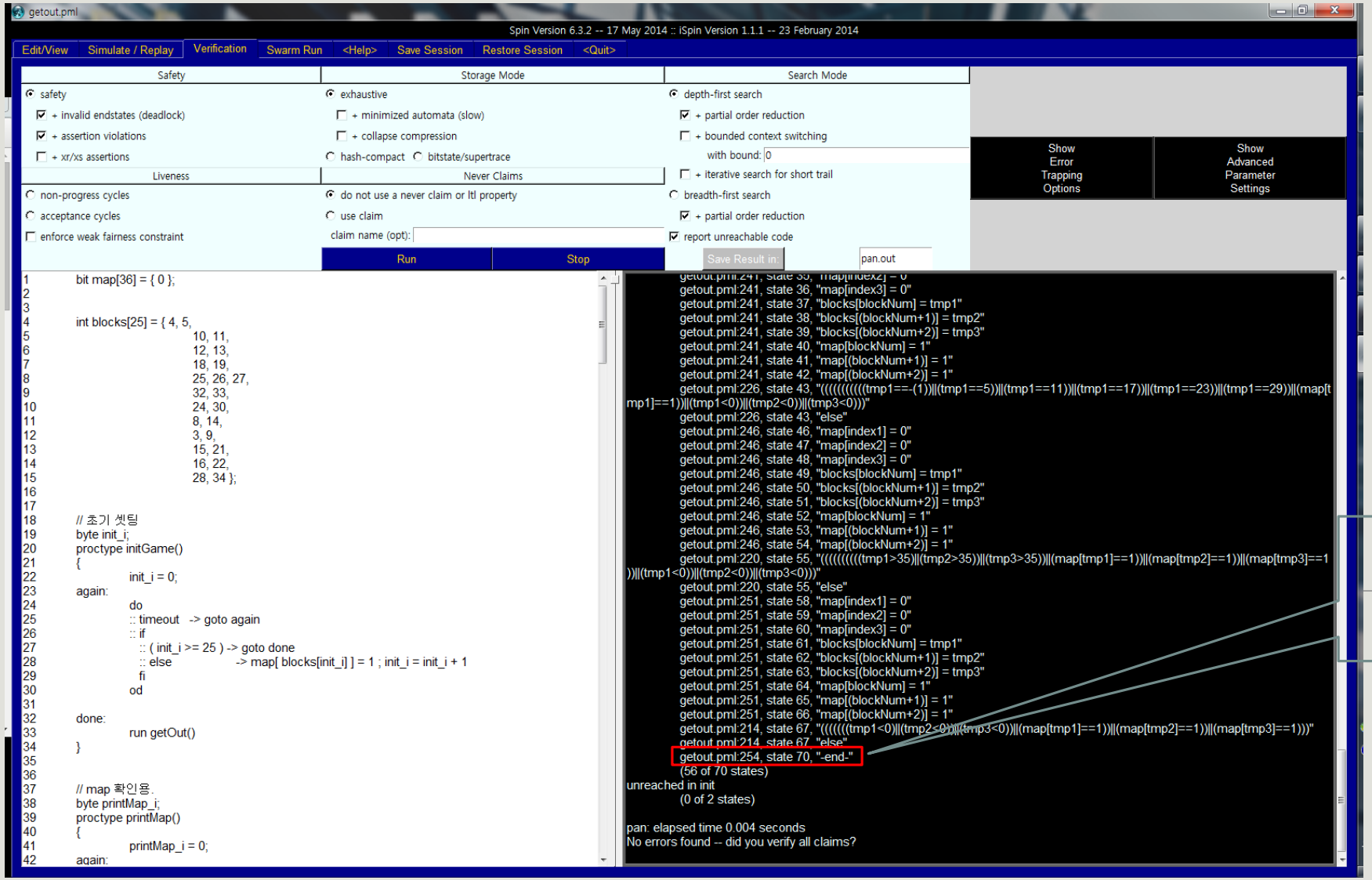
```
[variable values, step 3735]
blocks[4]  =  0
blocks[5]  =  0
blocks[6]  =  0
blocks[7]  =  6
blocks[8]  =  0
blocks[9]  =  8
init_i  =  25
map[0]  =  0
map[10]  =  1
map[11]  =  1
map[12]  =  1
map[13]  =  1
map[14]  =  1
map[15]  =  1
map[16]  =  1
map[17]  =  1
map[18]  =  1
map[19]  =  1
map[1]  =  0
map[20]  =  1
map[21]  =  1
map[22]  =  1
map[23]  =  1
map[24]  =  1
map[25]  =  1
map[26]  =  1
map[27]  =  0
map[28]  =  0
map[29]  =  0
map[2]  =  0
map[30]  =  0
map[31]  =  0
map[32]  =  0
map[33]  =  0
map[34]  =  0
map[35]  =  0
map[3]  =  1
map[4]  =  1
map[5]  =  1
map[6]  =  0
map[7]  =  1
map[8]  =  1
map[9]  =  1
printMap_i  =  0
tmp1  =  -7
```

```
3735:    proc 31 (checkTwoBlock:1) getout.pml:100 (state 9)
3735:    proc 30 (checkThreeBlock:1) getout.pml:254 (state 70)
3735:    proc 29 (check2Height:1) getout.pml:202 (state 53)
3735:    proc 28 (checkTwoBlock:1) getout.pml:100 (state 9)
3735:    proc 27 (check2Height:1) getout.pml:202 (state 53)
3735:    proc 26 (checkTwoBlock:1) getout.pml:100 (state 9)
3735:    proc 25 (check2Height:1) getout.pml:202 (state 53)
3735:    proc 24 (check2Height:1) getout.pml:202 (state 53)
3735:    proc 23 (checkTwoBlock:1) getout.pml:100 (state 9)
3735:    proc 22 (check2Height:1) getout.pml:202 (state 53)
3735:    proc 21 (checkTwoBlock:1) getout.pml:100 (state 9)
3735:    proc 20 (checkTwoBlock:1) getout.pml:100 (state 9)
3735:    proc 19 (check2Height:1) getout.pml:202 (state 53)
3735:    proc 18 (checkTwoBlock:1) getout.pml:100 (state 9)
3735:    proc 17 (check2Height:1) getout.pml:202 (state 53)
3735:    proc 16 (checkTwoBlock:1) getout.pml:100 (state 9)
3735:    proc 15 (check2Width:1) getout.pml:151 (state 53)
3735:    proc 14 (check2Height:1) getout.pml:202 (state 53)
3735:    proc 13 (checkTwoBlock:1) getout.pml:100 (state 9)
3735:    proc 12 (check2Width:1) getout.pml:151 (state 53)
3735:    proc 11 (checkTwoBlock:1) getout.pml:100 (state 9)
3735:    proc 10 (checkTwoBlock:1) getout.pml:100 (state 9)
3735:    proc  9 (check2Height:1) getout.pml:202 (state 53)
3735:    proc  8 (checkTwoBlock:1) getout.pml:100 (state 9)
3735:    proc  7 (checkThreeBlock:1) getout.pml:254 (state 70)
3735:    proc  6 (check2Width:1) getout.pml:151 (state 53)
3735:    proc  5 (checkTwoBlock:1) getout.pml:100 (state 9)
3735:    proc  4 (check2Width:1) getout.pml:151 (state 53)
3735:    proc  3 (checkTwoBlock:1) getout.pml:100 (state 9)
3735:    proc  2 (getOut:1) getout.pml:82 (state 34)
3735:    proc  1 (initGame:1) getout.pml:34 (state 15)
3735:    proc  0 (:init::1) getout.pml:261 (state 2)
271 processes created
```

```
258    init
259    {
260            run initGame()
261    }
262
```

끝난 위치.

2. GetOut Game



End 확인.

Q & A