Project 2 - Free Flow to SAT Reduction
CS494 - Lillis

Authors
Hanna Thayyil
Jason Batson

Instructions for Use

1. Run Code from inside 'Python' folder
2. Instantiate puzzle inside the 'input' file

Variables

- r = number of rows
- c = number columns
- n = number of nodes (r*c)
- p = the 2D grid
- p[y][x] = the node at coordinates x,y
- k = number of paths/colors + 1

Constraints

1. Terminal nodes must be exactly one color and are given in the problem instance

   k-1 clauses of 2 literals each + 1 clause of 1 literal for each terminal node

   Total Clauses: O(k^2) Total Literals: O(k^2)

   Example: two colors { green, blue }, the following clauses are added for one green terminal node where p[y][x] is the node

   p[y][x]['g']
   (!p[y][x]['g'] v !p[y][x]['b'])
   (!p[y][x]['g'] v !p[y][x]['e'])

2. Terminal nodes must have exactly one matching neighbor. This can be expressed as….

Note: Make sure to mark edge/corner cases as FALSE where the index would fall out of bounds

Example for a terminal node for the color green

p[y][x][‘g’] ->

p[y][x+1][‘g’] v p[y][x-1][‘g’] v p[y+1][x][‘g’] v p[y-1][x][‘g’]

Exactly 1 of these…

- p[y][x+1][‘g’]
- p[y][x-1][‘g’]
- p[y+1][x][‘g’]
- p[y-1][x][‘g’]

Which expands into the following clauses for CNF:

!p[y][x][‘g’] v p[y][x+1][‘g’] v p[y][x-1][‘g’] v p[y+1][x][‘g’] v p[y-1][x][‘g’]
!p[y][x][‘g’] v !p[y][x+1][‘g’] v !p[y][x-1][‘g’]
!p[y][x][‘g’] v !p[y][x+1][‘g’] v !p[y+1][x][‘g’]
!p[y][x][‘g’] v !p[y][x+1][‘g’] v !p[y-1][x][‘g’]
!p[y][x][‘g’] v !p[y][x-1][‘g’] v !p[y+1][x][‘g’]
!p[y][x][‘g’] v !p[y][x-1][‘g’] v !p[y-1][x][‘g’]
!p[y][x][‘g’] v !p[y+1][x][‘g’] v !p[y-1][x][‘g’]

These clauses are capped at 7 for a single color on nodes with 4 adjacent nodes.
Total clauses: O(k) Total Literals: O(k)

3. A non-terminal node must be exactly one of the colors specified in the input or blank (“white”)

(k^2-k)/2 clauses of 2 literals + 1 clause of k literals for each for each non-terminal node

Total Clauses: O(n*k^2) Total Literals: O(n*k^2)

Example: two colors { green: g, blue: b }, the following clauses are added for one non-terminal node where p[y][x] is the node

p[y][x][‘g’] v p[y][x][‘b’] v p[y][x][‘e’]
(!p[y][x][‘g’] v !p[y][x][‘b’]) these are derived from the given python function
(!p[y][x][‘g’] v !p[y][x][‘e’]) exactly_one()
(!p[y][x][b] v !p[y][x][‘e’])

4. If a non-terminal node is not white/blank, it must have exactly 2 adjacent nodes that are the same color

p[y][x][‘g’] ->
p[y][x+1][‘g’] v p[y][x-1][‘g’] v p[y+1][x][‘g’] v p[y-1][x][‘g’]
Exactly 1 of these…

- p[y][x+1][‘g’] ->
  Exactly 1 of these…
  - p[y][x-1][‘g’]
  - p[y+1][x][‘g’]
  - p[y-1][x][‘g’]
- p[y][x-1][‘g’] ->
  Exactly 1 of these…
  - p[y][x+1][‘g’]
  - p[y+1][x][‘g’]
  - p[y-1][x][‘g’]
- p[y+1][x][‘g’] ->
  Exactly 1 of these…
  - p[y][x-1][‘g’]
  - p[y][x+1][‘g’]
  - p[y-1][x][‘g’]
- p[y-1][x][‘g’] ->
  Exactly 1 of these…
  - p[y][x-1][‘g’]
  - p[y][x+1][‘g’]
  - p[y+1][x][‘g’]

Simplifies to…..

(p[y][x][‘g’] v p[y][x+1][‘g’] v p[y][x-1][‘g’] v p[y-1][x][‘g’]) ^
(!p[y][x+1][‘g’] v ![y][x-1][‘g’] v !p[y+1][x][‘g’] v !p[y-1][x][‘g’]) ^
(p[y][x+1][‘g’] v ![y][x-1][‘g’] v p[y+1][x][‘g’] v p[y-1][x][‘g’]) ^
(p[y][x+1][‘g’] v [y][x-1][‘g’] v !p[y+1][x][‘g’] v p[y-1][x][‘g’])

Total clauses: O(k*n) Total Literals: O(k*n)

```
[1white, 1a, 1b]
[Not(1white), Not(1a)]
[Not(1white), Not(1b)]
[Not(1a), Not(1b)]
[0a, 0b]
[Not(0a), Not(0b)]
[3white, 3a, 3b]
[Not(3white), Not(3a)]
```

```
[Not(3white), Not(3b)]
[Not(3a), Not(3b)]
[2a, 2b]
[Not(2a), Not(2b)]
[5white, 5a, 5b]
[Not(5white), Not(5a)]
[Not(5white), Not(5b)]
[Not(5a), Not(5b)]
[4white, 4a, 4b]
[Not(4white), Not(4a)]
[Not(4white), Not(4b)]
[Not(4a), Not(4b)]
[7white, 7a, 7b]
[Not(7white), Not(7a)]
[Not(7white), Not(7b)]
[Not(7a), Not(7b)]
[6a, 6b]
[Not(6a), Not(6b)]
[8a, 8b]
[Not(8a), Not(8b)]
[Or(1white, 1a, 1b),  Or(Not(1white), Not(1a)),  Or(Not(1white), Not(1b)),
Or(Not(1a), Not(1b)),  Or(0a, 0b),
 Or(Not(0a), Not(0b)),  Or(3white, 3a, 3b),
 Or(Not(3white), Not(3a)),  Or(Not(3white),
Not(3b)),  Or(Not(3a), Not(3b)),
Or(2a, 2b),  Or(Not(2a), Not(2b)),
Or(5white, 5a, 5b),  Or(Not(5white),
Not(5a)),  Or(Not(5white), Not(5b)),
Or(Not(5a), Not(5b)),  Or(4white, 4a, 4b),
 Or(Not(4white), Not(4a)),  Or(Not(4white),
 Not(4b)),  Or(Not(4a), Not(4b)),  Or(7white, 7a, 7b),
 Or(Not(7white), Not(7a)),  Or(Not(7white), Not(7b)),
 Or(Not(7a), Not(7b)),  Or(6a, 6b),  Or(Not(6a),
 Not(6b)),  Or(8a, 8b),  Or(Not(8a), Not(8b)),
 6b,  Or(7b, 3b),  Or(Not(7b), Not(3b)),
0a,  Or(3a, 1a),  Or(Not(3a), Not(1a)),  2b,
Or(1b, 5b),  Or(Not(1b), Not(5b)),  8a,  Or(5a, 7a),  Or(Not(5a), Not(7a)),
Or(And(1a,
Or(And(Or(And(2a, 4a), And(2a, 0a), And(4a, 0a)),
And(Or(Not(And(2a, 4a)), Not(And(2a, 0a))),
Or(Not(And(2a, 4a)), Not(And(4a, 0a))),
Or(Not(And(2a, 0a)), Not(And(4a, 0a)))))),
Not(1a)),  Or(And(1b, Or(And(Or(And(2b, 4b), And(2b, 0b), And(4b, 0b)),
 And(Or(Not(And(2b, 4b)), Not(And(2b, 0b))),
Or(Not(And(2b, 4b)), Not(And(4b, 0b))),
Or(Not(And(2b, 0b)), Not(And(4b, 0b)))))),     Not(1b)),
```

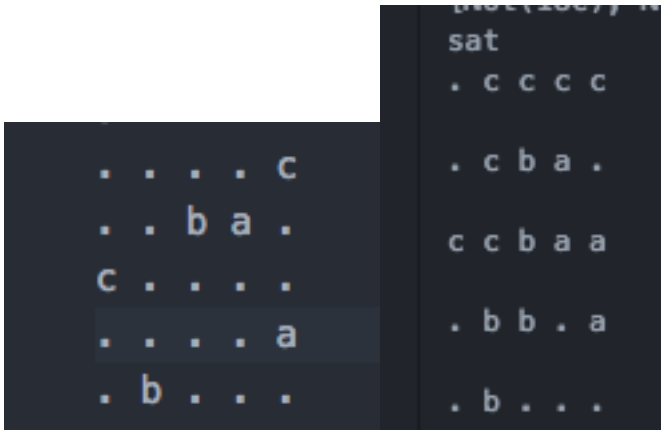
```
Or(And(3a, Or(And(Or(And(0a, 6a), And(0a, 4a), And(6a, 4a)),
And(Or(Not(And(0a, 6a)), Not(And(0a, 4a))), Or(Not(And(0a, 6a)), Not(And(6a,
4a))),
 Or(Not(And(0a, 4a)), Not(And(6a, 4a)))))),
Not(3a)),  Or(And(3b, Or(And(Or(And(0b, 6b), And(0b, 4b), And(6b, 4b)),
And(Or(Not(And(0b, 6b)), Not(And(0b, 4b))),
Or(Not(And(0b, 6b)), Not(And(6b, 4b))),
Or(Not(And(0b, 4b)), Not(And(6b, 4b)))))),     Not(3b)),
Or(And(5a, Or(And(Or(And(8a, 4a), And(8a, 2a), And(4a, 2a)),
And(Or(Not(And(8a, 4a)), Not(And(8a, 2a))),
Or(Not(And(8a, 4a)), Not(And(4a, 2a))),                    Or(Not(And(8a,
2a)), Not(And(4a, 2a)))))),     Not(5a)),  Or(And(5b, Or(And(Or(And(8b,
4b), And(8b, 2b), And(4b, 2b)),                And(Or(Not(And(8b, 4b)),
Not(And(8b, 2b))),                    Or(Not(And(8b, 4b)), Not(And(4b,
2b))),                    Or(Not(And(8b, 2b)), Not(And(4b, 2b)))))),
Not(5b)),  Or(And(4a, Or(And(Or(And(7a, 1a), And(7a, 5a), And(7a,
3a),And(1a, 5a), And(1a, 3a),
And(5a, 3a)),
And(Or(Not(And(7a, 1a)), Not(And(7a, 5a))),
 Or(Not(And(7a, 1a)), Not(And(7a, 3a))),
Or(Not(And(7a, 1a)), Not(And(1a, 5a))),
Or(Not(And(7a, 1a)), Not(And(1a, 3a))),
 Or(Not(And(7a, 1a)), Not(And(5a, 3a))),
Or(Not(And(7a, 5a)), Not(And(7a, 3a))),
Or(Not(And(7a, 5a)), Not(And(1a, 5a))),
 Or(Not(And(7a, 5a)), Not(And(1a, 3a))),
Or(Not(And(7a, 5a)), Not(And(5a, 3a))),
Or(Not(And(7a, 3a)), Not(And(1a, 5a))),
Or(Not(And(7a, 3a)), Not(And(1a, 3a))),
Or(Not(And(7a, 3a)), Not(And(5a, 3a))),
Or(Not(And(1a, 5a)), Not(And(1a, 3a))),
Or(Not(And(1a, 5a)), Not(And(5a, 3a))),
Or(Not(And(1a, 3a)), Not(And(5a, 3a)))))),
Not(4a)),  Or(And(4b, Or(And(Or(And(7b, 1b),
 And(7b, 5b),
And(7b, 3b),
And(1b, 5b),
 And(1b, 3b),
And(5b, 3b)),
And(Or(Not(And(7b, 1b)), Not(And(7b, 5b))),
Or(Not(And(7b, 1b)), Not(And(7b, 3b))),
 Or(Not(And(7b, 1b)), Not(And(1b, 5b))),
  Or(Not(And(7b, 1b)), Not(And(1b, 3b))),
   Or(Not(And(7b, 1b)), Not(And(5b, 3b))),
 Or(Not(And(7b, 5b)), Not(And(7b, 3b))),
 Or(Not(And(7b, 5b)), Not(And(1b, 5b))),
```

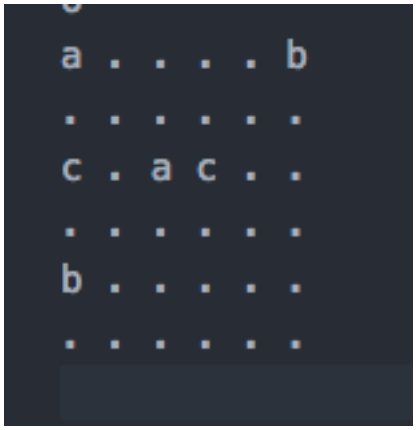
```
 Or(Not(And(7b, 5b)), Not(And(1b, 3b))),
 Or(Not(And(7b, 5b)), Not(And(5b, 3b))),
Or(Not(And(7b, 3b)), Not(And(1b, 5b))),
Or(Not(And(7b, 3b)), Not(And(1b, 3b))),
 Or(Not(And(7b, 3b)), Not(And(5b, 3b))),
  Or(Not(And(1b, 5b)), Not(And(1b, 3b))),
  Or(Not(And(1b, 5b)), Not(And(5b, 3b))),
 Or(Not(And(1b, 3b)), Not(And(5b, 3b)))))),
Not(4b)),  Or(And(7a,
 Or(And(Or(And(6a, 4a), And(6a, 8a), And(4a, 8a)),
And(Or(Not(And(6a, 4a)), Not(And(6a, 8a))),
Or(Not(And(6a, 4a)), Not(And(4a, 8a))),
 Or(Not(And(6a, 8a)), Not(And(4a, 8a)))))),     Not(7a)),
Or(And(7b, Or(And(Or(And(6b, 4b), And(6b, 8b), And(4b, 8b)),
And(Or(Not(And(6b, 4b)), Not(And(6b, 8b))), Or(Not(And(6b, 4b)), Not(And(4b,
8b))),
Or(Not(And(6b, 8b)), Not(And(4b, 8b)))))), Not(7b))]
```

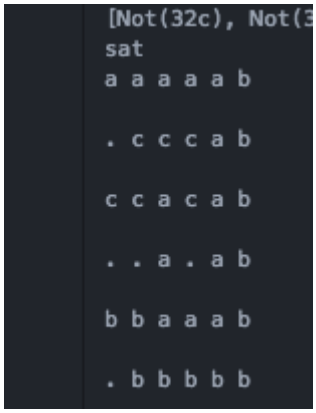
I used z3 in Python as my solver.

Experiments

Size	# of Colors	Time
2	1	0.294a
2	2	0.257
3	2	0.308
3	3	0.332
4	1	0.31
4	2	0.419
4	3	0.509
6	2	0.941
6	3	1.146
6	4	1.539


sat
. c c c c
. c b a .
c c b a a
. b b . a
. b . . .
. . . . c
. . b a .
c . . . .
. . . . a
. b . . .


a . . . . b
. . . . . .
c . a c . .
. . . . . .
b . . . . .
. . . . . .


[Not(32c), Not(3
sat
a a a a a b
. c c c a b
c c a c a b
. . a . a b
b b a a a b
. b b b b b