Domino Magic
Threefold Problem Set

Xun Zou, slides adapted from Chih-Hung Liu, Daniel Wolleb, and Andreas Bärtschi

November 25, 2020

ETH Zürich

Threefold Problem Set

Goal: simulate the thinking steps of a six hour exam in one hour.

First Part: (16:15-17:00)

- think about the problems
- sketch solutions on paper
- no coding required

Break: (17:00-17:10)

Second Part: (17:10-)

- solution discussion
- Q & A

The problem set is available on moodle. Please ask questions in Code Expert.

Note: we need a new technique to solve the second problem.

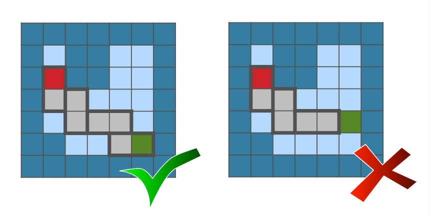
Domino Snake – The problem

Given:

- $h \times w$ grid with obstacles
- p queries of point pairs $((q, r), (s, t))$

Wanted:

- `y` or `n` per query:
  Does a domino snake between the two points exist?



What corresponds to a *domino snake*?

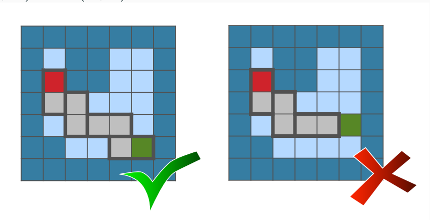
- a (q, r)-(s, t)-path on the 4-neighborhood grid graph with holes
- this path needs to have even length (i.e. an even number of vertices)

Is even length necessary and sufficient?

- *necessary*: odd length paths can not be tiled into dominos of area 2.
- *sufficient*: a path $P = (p_1, p_2, \ldots, p_l)$ of even length $l = 2k$ can always be tiled into dominoes of the form (p_1, p_2), (p_3, p_4), $\ldots$, (p_{l-1}, p_l).

Domino Snake – Handling a single query

BFS/DFS can answer if any path from (q, r) to (s, t) exists.

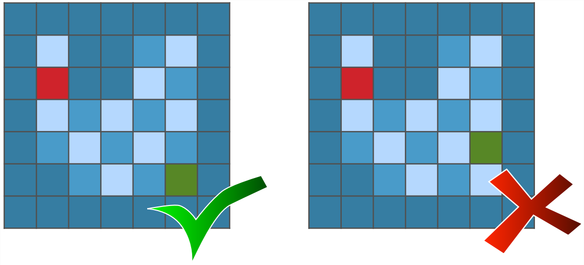


What if this path is of odd length?

Can we add a detour to find a path of the right parity?

No! Chessboard coloring argument: bipartite graph

We require: $(q + r) \not\equiv_2 (s + t)$



Domino Snake – Handling multiple queries

$p = 1$: check that BFS/DFS from (q, r) visits (s, t) and $(q + r) \not\equiv_2 (s + t)$.

$p > 1$:

- Running BFS/DFS over and over is expensive $\to \mathcal{O}(hwp)$
- Are (q, r) and (s, t) in the same connected component?
- Precompute the connected components in $\mathcal{O}(hw)$.
- Each of the p queries can then be answered in $\mathcal{O}(1)$ by checking
  - component$((q, r))$=component$((s, t))$
  - $(q + r) \not\equiv_2 (s + t)$.

Overall runtime: $\mathcal{O}(hw + p) = \mathcal{O}(hw)$

New Tiles – The problem

Problem

Given a $h \times w$ matrix of 0's and 1's.

Find the maximum number of non-overlapping 2×2 matrices of the form: $\begin{matrix} 1 & 1 \\ 1 & 1 \end{matrix}$

Greedily take every 2x2 free space.

Counterexample (ignoring zeros on the boundary):

0110

1111

1111

Greedy gives answer 1, while the maximum is 2.

Maximum Independent Set

Take all 2x2 all-ones matrices as vertices.

Edges indicate overlapping.

We do not have a bipartite graph.

0110

1110

1110

Note that $h \leq 100$, $w \leq 17$ are relatively small.

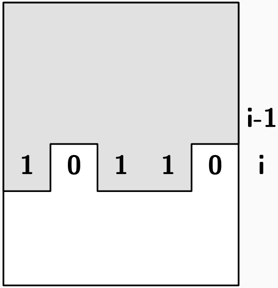
New Tiles – Dynamic programming

State of the subproblem: $[i, b]$, $1 \leq i \leq h$, $b \in \{0,1\}^w$ is a bitmask

$DP[i][b] :=$ maximum number of 2x2 matrices we can take from the first i rows where the i-th row is constrained to the $bitmask$.

$b_j = 1$ iff. j-th column has been used in calculating $DP[i][b]$ (regardless of the input)

Wanted: $DP[h][\{1\}^w]$



New Tiles – Dynamic programming

State of the subproblem: $[i, b]$, $1 \leq i \leq h$, $b \in \{0, 1\}^w$ is a bitmask

$DP[i][b] :=$ maximum number of 2x2 matrices we can take from the first i rows where the i-th row is constrained to the $bitmask$.

$b_j = 1$ iff. j-th column has been used in calculating $DP[i][b]$ (regardless of the input)

Wanted: $DP[h][\{1\}^w]$

Initialization:

- For the first row, no tiles fit: $DP[1][b] = 0$, for all $b \in \{0, 1\}^w$

New Tiles – Recurrent Formula

Assume $DP[i-1][b']$ is available from all b', compute $DP[i][b]$ for all b

Now we want to take matrices that occupy only row $i-1$ and row i

If we do not use the i-th row at all: $DP[i][\{0\}^w] = \max_b(DP[i-1][b])$

1	1	1	1	0	0	1	0	0	0	0	1	1	0	0	lookup in row i-1
0	0	0	0	1	1	0	1	1	1	1	0	0	1	1	state in row i

0	1	1	0	1	1	1	1	1	1	1	1	0	1	1	row i-1 of the input
0	1	1	1	1	1	0	1	1	1	1	0	1	1	1	row i of the input

Enumerate b (red) that have even number of consecutive 1s

Check whether b is compatible with the input (yellow)

$DP[i][b] = \max(DP[i][b], DP[i-1][\text{negated } b] + \text{bitcount}(b)/2)$

New Tiles – Dynamic programming

We take some 2×2 matrices (specified by red and yellow) in the $i-1, i$ strip

1	1	1	1	0	0	1	0	0	0	0	1	1	0	0	lookup in row i-1
0	0	0	0	1	1	0	1	1	1	1	0	0	1	1	state in row i

0	1	1	0	1	1	1	1	1	1	1	1	0	1	1	row i-1 of the input
0	1	1	1	1	1	0	1	1	1	1	0	1	1	1	row i of the input

For the example: $DP[i][000011011110011] \leftarrow DP[i-1][111100100001100] + 4$

How about invalid bitmasks? Drop some one-bits.

- $DP[i][b] = \max_{b' \subset b} DP[i][b']$, $\mathcal{O}(2^w)$
- $DP[i][b] = \max_{j \in [w]} DP[i][b$ with j-th bit set to 0], $\mathcal{O}(w)$
- Compute $DP[i][b]$ in lexicographical order of b

New Tiles – Dynamic programming

Formula:

$$DP[i][b] = \max\Big(\max_{j \in [w]} DP[i][b \text{ with } j\text{-th bit unset}],$$
$$DP[i-1][\text{negated } b] + \text{bitcount}(b)/2\Big)$$

Runtime:

Size of the table: $2^w \cdot h$.

Update step per entry: $\mathcal{O}(2^w)$ or $\mathcal{O}(w)$.

Runtime: $\mathcal{O}(4^w \cdot h)$ or $\mathcal{O}(2^w \cdot h \cdot w)$.

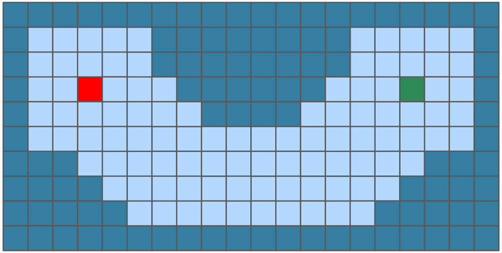
Snakes strike back – The problem

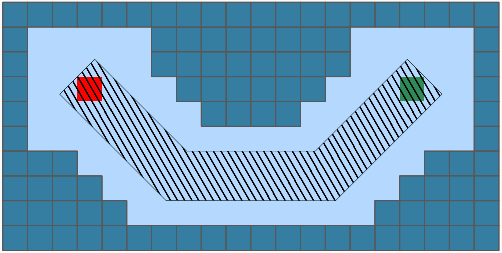
Given:

- $h \times w$ grid with obstacles (snake cages) and entrance/exit pair.
- Path width p, safety distance $p/2$. (Cases: $p = 1,\ p = 2,\ p \leq 30$)

Wanted:

- `yes` or `no`: Is there a path between the entrance and the exit with minimum width p which keeps clear from obstacles by $p/2$?

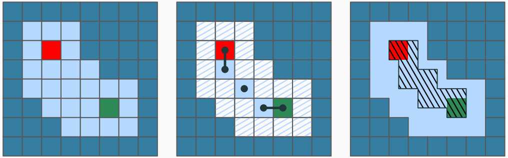




Snakes strike back – Case $p = 1$

Rough Idea:

Use the graph from Domino Snake, but delete squares which are adjacent to snake cages.



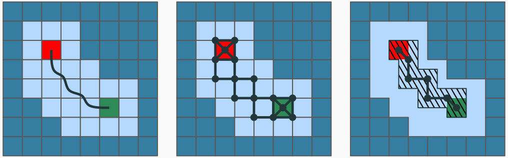
Problem: Fails, graph is disconnected!

Reason: Deleting too much.

Equivalent: **Curve** (has 0 width!) with a safety distance: $\boldsymbol{p}$.

Solution Approach:

Go along the boundary of cells which are not adjacent to snake cages.



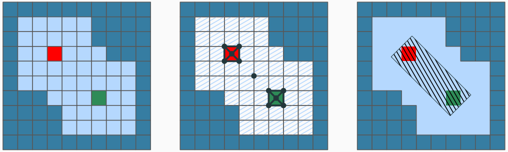
Solution $(\boldsymbol{p = 1})$**:**

BFS on grid graph given by vertices and by boundary segments with distance ≥ 1 to snake cage boundarys.

Can we adapt for larger p?

Snakes strike back – Case $p = 2$

Rough Idea (first case solution adapted): BFS on grid graph given by vertices and by boundary segments with distance ≥ 2 to snake cage boundarys.



Problem: the grid graph becomes disconnected.

Look at the problem statement again:

Originally: Is there a path of width p which keeps clear from obstacles by $p/2$?

Redefined: Is there a path of width 0 which keeps clear from obstacles by p?

Redefinition II: Is there a path of width $2p$ which keeps clear from obstacles by 0?

Snakes strike back – Arbitrary p

Draw a path with a brush of diameter $2p$.

This sounds familiar:

H1N1 – How to move a disk D without colliding with a given point set P?

Move disk along Voronoi Diagram edges / in the Delaunay triangulation.

Problem: Here our obstacles are cells, not points.

Solution: Replace every snake cage square by its 4 vertices.

Careful:

- We need vertices of each square! (Only considering the convex & concave vertices of the boundary of the full obstacle is not enough.)