Solution outlines

BAPC Finals 2011

October 15, 2011

J – Treasure Map

Problem

Find smallest x and y such that $N = y^2 - x^2$

Observation

$y^2 - x^2 = (y - x)(y + x) = N$

Solution

- Not possible if and only if $N \bmod 4 = 2$
- For all dividers d of N, try to solve:
  - $d = (y - x)$ → $2x = N/d - d$
  - $N/d = (y + x)$ → $2y = N/d + d$
- Keep smallest x and y such that $x, y \in \mathbb{N}$

Benelux Algorithm Programming Contest 2011

H – Walking the Plank

- Basic simulation problem
- Keep track of the queues on both sides
  - Use priority queue for efficiency
  - Make sure order of pirates is correct!
- Then simply handle all events correctly

B – Quick out of the Harbour

- Basically a shortest path problem
- Solve using Dijkstra
- Also possible using BFS
  - But not in a standard way!
  - Somehow need to expand single step into d+1 steps

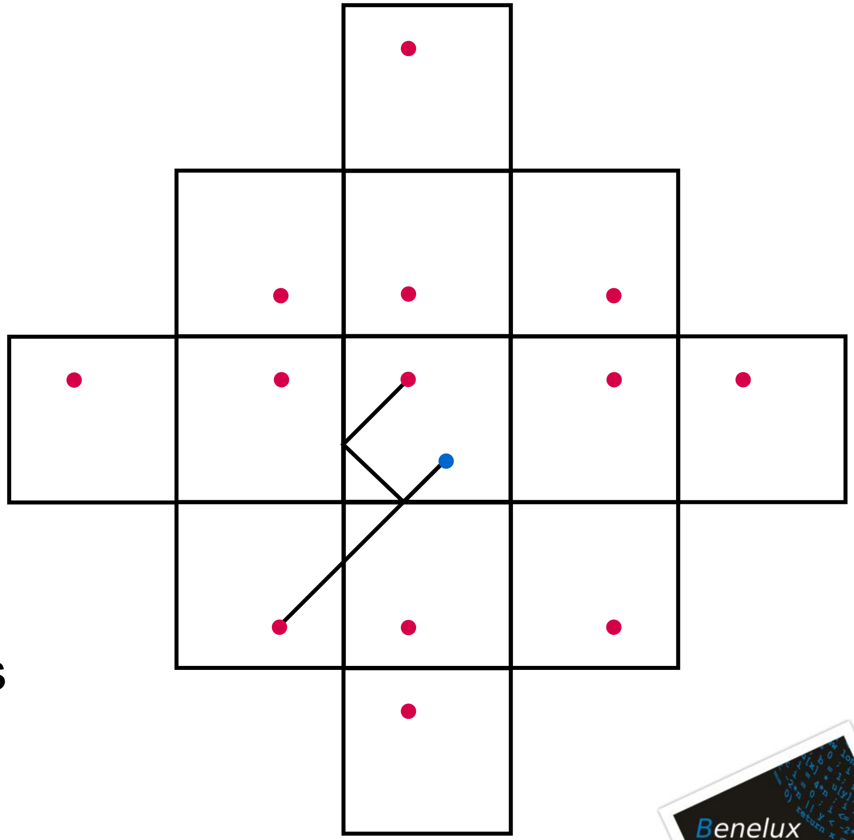
Benelux Algorithm Programming Contest 2011

F – Ultimate Finishing Strike

Idea

Copy and mirror room to simulate reflection (bouncing)

- Check “rooms” with k bounces
  - Simply compute closest
  - Compute type of bounces
  - Runs in O(k) time



- Finally sort and remove duplicates
- Watch out for overflow!
- Also possible in O(1) time

I – Parking Ships

Observation

Clearly ships must follow order of centers

DP Solution

- F[i][k] = minimum coordinate of right side of rightmost ship placing k ships of ships 1 .. i (∞ if not feasible)
- For every ship, decide to place it (if possible) or not
- Place a ship as far to the left as possible
- Use special case for captain (or solve two problems)

Greedy Solution

- Choose as next ship the one for which the right side is leftmost
- Break ties by order of centers
- Can run in $O(n \log n)$ time, but $O(n^2)$ (DP) is fine

Benelux Algorithm Programming Contest 2011

D – Bad Wiring

Observations

- Order does not matter
- Flipping a switch twice does nothing
- Solution is essentially a bitstring

Simple backtracking from left to right

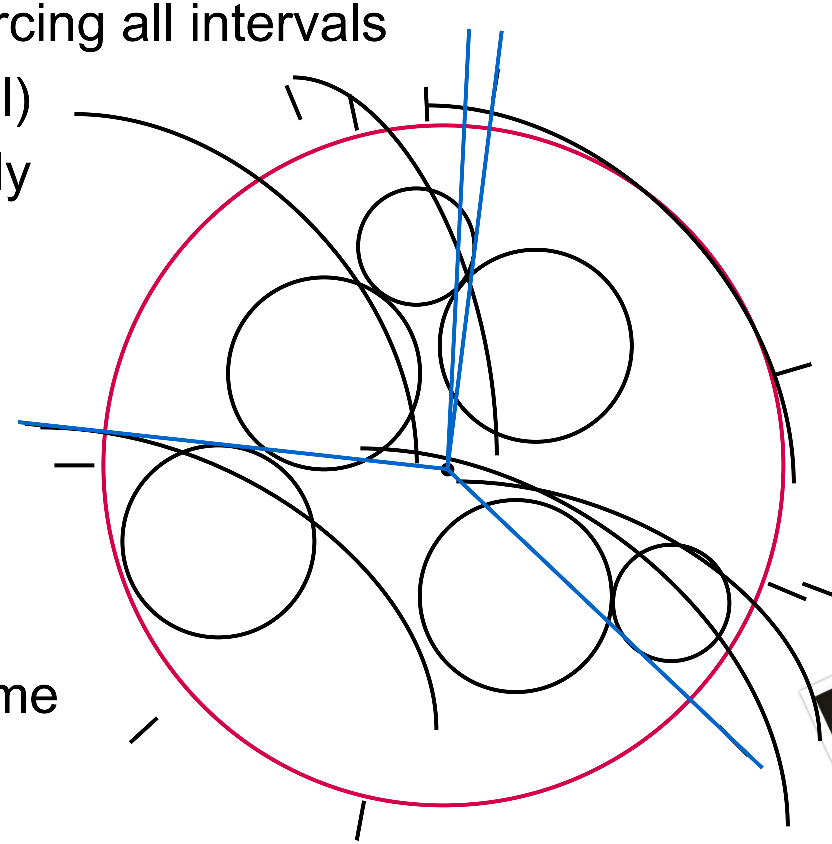
- Flip switch or not
- At some point light moves out of “window”
- At that point choice is fixed
- Running time is $O(nD\ 2^D)$

Alternative solution

- Solve linear system in Z_2
- Solution not unique!
- Also compute null-space

A – Popping Balloons

- See all balloons as circular intervals
  - Requires some geometric computations
- Consider canonical solutions
  - Each line passes through endpoint interval
- Find smallest set of lines piercing all intervals
  - Pick a starting line (try all)
  - Compute the rest greedily



- Also possible in O(n log n) time

Benelux Algorithm Programming Contest 2011

G – Doubloon game

Another NIM-variant

- Like Shuriken Game from Preliminaries
- This time DP does not work

Nice Solution

- Use “nimbers” from impartial game theory
- F(0) = 0,   0 means you lose
- $F(n) = \text{mex}(\ \{F(x) \mid n - x \text{ is power of } K\}\ )$
- K odd □ $F(n) = n \bmod 2$
- K even □ $F(n) = \begin{cases} 2 & \text{if } n \bmod (K+1) = K \\ (n \bmod (K+1)) \bmod 2 & \text{otherwise} \end{cases}$
- Optimal solution is 1 or K (or 0)

Simple Solution: Simply recognize pattern and make formula

E – Undercover Pirate

Notation

- Category A: “Ninjas” that can weigh W, $\geq W$, or $\leq W$
- Category B: “Ninjas” that can weigh W or $\geq W$
- Category C: “Ninjas” that can weigh W or $\leq W$
- Category D: Ninjas that weigh W
- k: #times to use the scale

Necessary invariant: $2\,|A| + |B| + |C| \leq 3^k$

Case 1 (start case):

- $x = \min(\, |A|/2 \,,\, (3^{k-1} - 1)/2 \,)$
- x of A vs. x of A

Benelux Algorithm Programming Contest 2011

E – Undercover Pirate

Necessary invariant: $2\,|A| + |B| + |C| \leq 3^k$

Case 2 ($|A| \leq |D|$, $|B| = |C| = 0$):

$x = \min(\,|A|\,,\, 3^{k-1}\,)$

x of A vs. x of D

Case 3 ($|A| = 0$, $|B| + |C| \leq 3^k$, assume $|B| \geq |C|$):

Case 3a ($|B|/2 < 3^{k-1}$):

$x = |B|/2$, $y = \min(\,|C|/2\,,\, 3^{k-1} - x\,)$

x of B and y of C vs. x of B and y of C

Case 3b ($|B|/2 \geq 3^{k-1}$): 3^{k-1} of B vs. 3^{k-1} of B

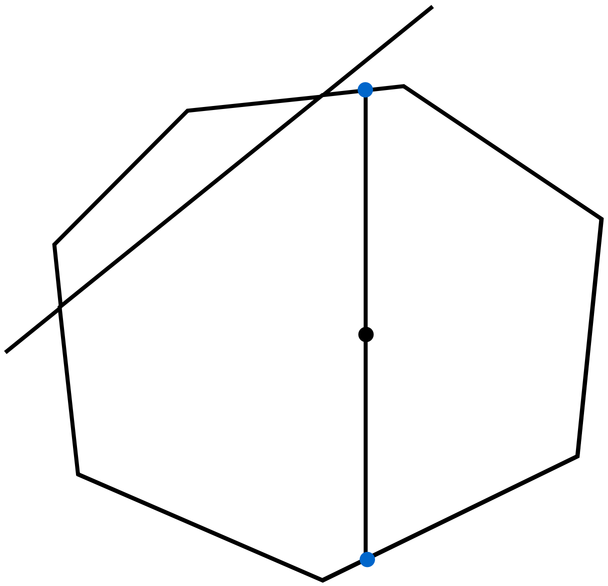
- Also a base case for $k = 1$
- Tricky to keep track of (ranges) of ninjas

C – Find the Treasure

- Every view defines a line
  - Island with treasure must be above or below line

Basic solution

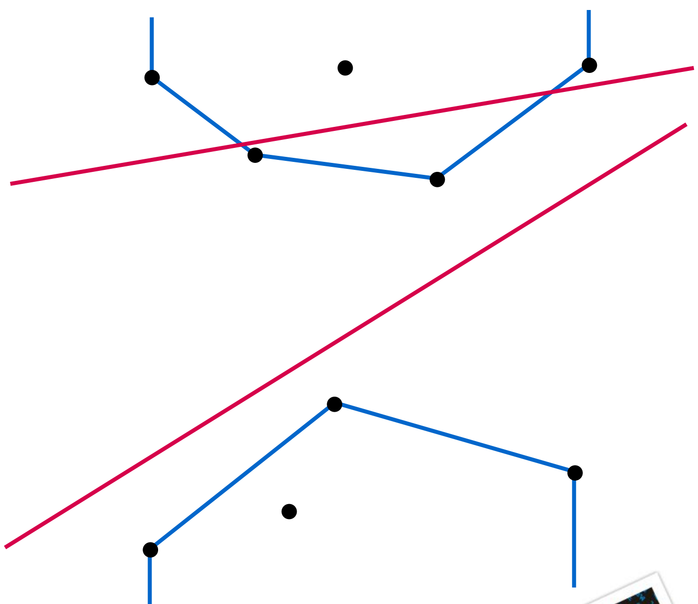
- Construct convex polygon from lines
  - Use amortized O(log m) per line
- Check every island with convex polygon
  - Use O(log m) time per island



C – Find the Treasure

Alternative solution

- Duality!
  - A point $p=(x_p, y_p)$ becomes a line p^*: $x_p x - y_p$
  - A line L: $Ax + B$ becomes a point $L^* = (A, -B)$
  - Aboveness relation is preserved
- Every line (view) is now a point
- Compute two convex chains
  - Use Graham scan or …
- Every island is now a line
- Island is valid if it doesn’t cross a chain
  - Determine using binary search



Benelux Algorithm Programming Contest 2011