ACM ICPC Manila 2016

Solution Sketches

Problem G: Go Go Go Special Action Force!

- Easy problem
- Check if the input is a valid Sudoku grid
  - Option 1: sets
  - Option 2: bitmasks
- Just be careful!

Problem L: LoL Tournament

- The tournament is a binary tree
- The highest chance is those with fewest battles
  - i.e., minimum depth
- Construct tree and compute depths
- Special case: **p = 0** or **p = 1**
  - All nodes are equiprobable!

Problem F: Frog Pushers

- Given sequence of edge removals, compute **MST** before *each* removal
- *Bad:*
  - E times Prim's/Kruskal's
  - Too slow
  - **$O(E^2 \log E)$**

Problem F: Frog Pushers

- *Good:*
    - Reverse the input; so we're now dealing with edge additions.
    - After each edge addition, compute new MST. Throw away unused edges *permanently*.
    - This keeps edge count at most **V-1** always
    - **O(EV log V)**

Problem F: Frog Pushers

- *Better:*
  - Maintain cost-sorted list of edges.
  - **O(EV α(V))**
- *Betterer:*
  - MST update after an insertion can be done in **O(V)**
    - BFS to find the largest cost in the (new) cycle
  - **O(EV)**

Problem F: Frog Pushers

- Asymptotically faster solutions exist
  - (I think)
  - Link-cut trees?

Problem K: Off the Rails

- Assume single segment / line.
  - Calculus optimization / linear algebra
- Let f(i, j) be this answer for the segment pt[i..j].
- f(i, j) can be computed in O(n).

Problem K: Off the Rails

- Assume single segment / line.
  - Calculus optimization / linear algebra
- Let f(i, j) be this answer for the segment pt[i..j].
- f(i, j) can be computed in **O(1) by precomputing some sums.**

Problem K: Off the Rails

- Let F(j) be the answer for pt[1..j].
  - $F(0) = 0$
  - $F(j) = \min_i F(i) + C + f(i+1, j)$ [$0 \leq i < j$]
- Dynamic programming in **$O(n^2)$**

Problem K: Off the Rails

- Yes, the title doesn't start with a "K"... it's **off the rails**

Problem B: Balloon Distribution

- Simple solution: Priority queue
  - **O(N log M)**. Too slow

Problem B: Balloon Distribution

- Assume **a/b** is the ratio in the last rank. How many balloons?
  - count(a/b) := sum(1 ≤ i ≤ M) floor(P[i]*b/a)
- **Binary search** to find the real last ratio **a/b**.
- Quite tricky. Be careful about precision.

Problem B: Balloon Distribution

- One way to do it:
  - Let M = max(P[i])+1. Find smallest D: count(M/D) ≥ N.
    - *Binary search*
  - Next, let d[i] = floor(P[i]*D/M)
  - Sort the list [P[1]/d[1], P[2]/d[2], ..., P[M]/d[M]]
  - The answer is in this list. *Binary search* again
  - No big ints/floats necessary. Use 64-bits.
- **O(M (log M + log N))**

Problem J: Jack and Jill and Joe

- The dimensions are a^2 and $b(b+a)$, and we want
  - $(a^2 - b(b+a))^2 \leq 900$
  - equivalently, $a^2 - b(b+a) = d$ for some $|d| \leq 30$.
  - equivalently, $(2b+a)^2 - 5a^2 = 4d$ for some $|d| \leq 30$.
- Gives us 61 generalized Pell equations!

Problem J: Jack and Jill and Joe

- Solutions follow a familiar pattern: a 2D recurrence.
  - In fact, Fibonacci recurrence:
    - (a, b) is a solution -> (a + b, a) is a solution
- Pattern-match and precompute all solutions for all d.

Problem J: Jack and Jill and Joe

- There are so few solutions: 443
    - they grow exponentially
- To answer a query **N**, easiest to just linear search.
- Watch out for d’s with multiple “solution families”!

Problem I: Imelda’s Shopping Spree

- Range queries:
  - Range increment.
  - Range reverse.
  - Query: How many contiguous increasing subsequences?

Problem I: Imelda’s Shopping Spree

- Compute the difference array D[i] = P[i+1] - P[i]. Then:
  - Range increment reduces to 2 “point increments”.
  - Range reverse reduces to range reverse + range negate + 2 “point increments”.
  - Query reduces to counting the number of contiguous positive subsequences.

Problem I: Imelda's Shopping Spree

- **Splay tree** or **treap** can handle reverse. Need the following info for every node:
  - Sum of values.
  - Longest pos./neg. streak on left and right.
  - Number of contiguous pos./neg. subsequences.
  - Need to keep track of negatives to account for *range negates*
  - "Reverse" flag for lazy propagation

Problem C: Convex Quadrilaterals

- Only **convex hull** matters.
- **DP**: Choose starting edge and keep track of current edge and number of edges used so far.
- **$O(N^3)$** (**$O(N^2)$** states, **$O(N)$** transition)
- To compute (signed) areas, you can use *shoelace formula*

Problem E: Expression

- Easy/tricky cases
  - If **y = 0**, output `xx-`
  - If **y > 0** and **x = 0**, impossible.
- General case: 28 ops quite tight!
  - Binary, ternary, etc., don't work

Problem E: Expression

- Let `{y}` be a string that evaluates to **xy** for any **x > 0**.
  - Output "`{y}x/`"
- You can write `{y}` in ≤ 27 operations!
  - for **0 < y < 1212**

Problem E: Expression

- Tricks:
  - `{1}` = x
  - `{ab}` ≤ `{a}{b}`x/* for **a, b > 1**
  - `{a+b}` ≤ `{a}{b}`+ for **a, b > 0**
- "≤" means "has the ff. candidate solution"

Problem E: Expression

- Tricky trick:
  - Use subtraction!
  - `{ab-c}` ≤ `{a}{b}x/*{c}-`
    - for **a, b, c < ab-c.**
  - Without it, you might fail 1103.
- With these tricks, generate all.

Problem E: Expression

- Worst case?
  - **27** operations: 823, 1006, 1111, 1198, 1211 ...
  - **28** operations: 1114, 1138, 1166, 1193
- I'd like to see better solutions!

Problem E: Expression

- Be careful with intermediate values!
  - They should be small
  - Always divide first:
    - `{a}{b}*x/ -> {a}{b}x/*`

Problem A: AdoraBalls

- Invert the 4D matrix, and check if the resulting product with the vector is all nonnegative.
- Tricky when non-invertible.

Problem A: AdoraBalls

- Another way to look at it:
  - Given 4D points P_0, P_1, P_2, P_3, P_4, can P_0 be expressed as a *nonnegative* rational linear combination of P_1, P_2, P_3, P_4?
    - Also known as *convex combination*

Problem A: AdoraBalls

- Project all points to a 3D-hyperplane
  - x+y+z+w = 1
- The problem is now: Is the 3D point P_0' inside the *tetrahedron* with vertices P_1', P_2', P_3', P_4'?

Problem A: AdoraBalls

- Is the 3D point P_0' inside the *tetrahedron* with vertices P_1', P_2', P_3', P_4'?
- General case:
  - For each vertex P_i', check if P_0' is in the same side of the plane determined by the remaining three vertices.
    - Cross products + Dot products

Problem A: AdoraBalls

- Watch out for special/degenerate cases! E.g.
  - The tetrahedron has zero area, e.g. it's 2D, 1D, 0D, etc.
  - One (or more) of the original 4D points is the origin (0,0,0,0).
    - Can't be projected; needs special handling.
- Be careful with precision!
  - Add fractions with lcm.

Problem D: Disco Dance Debacle

- A disco dance always alternates row and column and ends up at the starting point.
- Thus, number of lit cells it unlits per row/column is even.
- Thus, *necessary* condition is: each row/column must have even no. of lit cells.

Problem D: Disco Dance Debacle

- It's also *sufficient*!
  - Draw a bipartite graph with **m+n** nodes, and edge **(i, j)** if cell **(i, j)** is lit.
  - Then a disco dance is just a simple cycle!
  - Each connected component can be decomposed into a cycle
    - **Eulerian cycle**

Problem D: Disco Dance Debacle

- So the goal is: make the no. of lit cells even, per row/column.
- If there are **R** rows and **C** columns with odd no. of lit cells, then **max(R,C)** moves are *necessary* to make them all even.
- It's also *sufficient*!
  - Why?

Problem D: Disco Dance Debacle

- So we have to compute **R** and **C**. (Symmetric)
- Sweep line + Range queries to process events:
  - Range increment/decrement.
  - Range “how many are equal to 0”.
- **Segment tree**:
  - Insight: “0” only ever appears as the min.
  - (min value, frequency of min value)

Problem H: Handbags

- BFS/DFS too slow; large dimensions
- Be careful; prices can’t “pass through” sources.
- Sometimes sources even block off some sections of the map.

Problem H: Handbags

- Solution: **Coordinate compression**.
  - Let X = {0, a} ∪ {x-1, x, x+1 for all sources (x,y)}
  - Let Y = {0, b} ∪ {y-1, y, y+1 for all sources (x,y)}
  - Dijkstra on the coordinates X and Y.
    - Coordinate compression
  - "Fill in" each (big) cell in O(1) using tricky arithmetic

Thank you!

- Credits
    - **AdoraBalls** - Atienza
    - **Balloon Distribution** - Muga, Atienza
    - **Convex Quadrilateral** - Manalastas
    - **Disco Dance Debacle** - Atienza
    - **Expression** - Atienza
    - **Frog Pushers** - Atienza
    - **Go Go Go Special Action Force!** - Chua
    - **Handbags** - Muga, Atienza
    - **Imelda's Shopping Spree** - Atienza
    - **Jack and Jill and Joe** - Sioson, Atienza
    - **Off the Rails** - Zuniga
    - **LoL Tournament** - Atienza
- Judges
    - Dr. Allan Sioson
        - Also chief judge
    - Dr. Pablo Manalastas
    - Mr. Kevin Atienza
        - Also testing
        - Also additional test cases
        - Also problem extensions
    - Dr. Philip Zuniga
    - Dr. Felix Muga
    - Dr. Caslon Chua