NCPC 2015
Presentation of solutions

Heads of Jury: Markus Dregi and Lukáš Poláček

2015-10-10

NCPC Jury

Problem authors

- Oskar Werkelin Ahlin (Spotify)
- Pål Grønås Drange (UiB)
- Markus Dregi (UiB)
- Björn Hegerfors (Spotify)
- Andreas Lundblad (Oracle)
- Lukáš Poláček (Spotify)
- Pehr Söderman (KTH)
- Marc Vinyals (KTH)

A – Adjoin the Networks

Problem

Given a forest with n vertices, add edges to make it into a tree with minimal diameter.

Insight

- Add an edge between two trees only between the two centers (a vertex minimizing the maximum distance in the tree).
- Every tree connects to the one of largest diameter.

Solution

Solution is the maximum of:

- Largest diameter of a tree
- The sum of the largest and second largest radii $+1$
- The sum of the second and third largest radii $+2$

Standard algorithms compute the diameters and radii in $O(n)$ time.

B – Bell Ringing

Problem

Generate sequence of $n!$ unique permutations of size n, such that consecutive permutations are only "one step away" from each other.

Solution

- Recursion: generate a solution for $n - 1$.
- For example, the solution for 2 is $(1, 2), (2, 1)$.
- Insert a 3:

  1 2 3
  1 3 2
  3 1 2
  3 2 1
  2 3 1
  2 1 3

C – Cryptographer's Conundrum

Problem

Count the number of characters needed to change a string to "PERPERPER...".

Solution

Compare the original string with "PERPERPER...":

PERPERPERPER
PERTEHRAPPER

D – Disastrous Downtime

Problem

Given n timestamps of request arrivals, how many machines are needed if a machine can serve at most k requests at once?

Insight

We treat a request at t_i as an interval $(t_i, t_i + 1000)$. Find the maximum number of intervals that overlap.

Solution

- Put incoming requests in a FIFO queue according to arrival.
- Before requests are added, pop all the requests that are at least 1000 ms old.
- Keep track of the maximum size S of the queue.
- Output $\lceil \frac{S}{k} \rceil$.

Time and memory complexity $\mathcal{O}(n)$.

E – Entertainment Box

Problem

Given n TV shows and recording machine that can record k shows at once, how many shows can you record in full length?

Insight

We can greedily add or discard the interval that ends the earliest, depending on whether it is in conflict with the already added intervals.

Solution

- Consider the k "tracks" available and the latest time t_i when track i was occupied by a TV show.
- Greedily add the new show from a to b to the track with the highest t_i for which $t_i \leq a$. Update the value of t_i to b.

Can be done in $O(n \log k)$ time by a multiset in C++ or a TreeSet in Java (be careful about equal t_i's).

F – Floppy Music

Problem

Given a sequence of integers $L_1, \ldots, L_n$, multiply each number by -1 or 1, so that each prefix sum never turns negative or exceeds a given number T.

Insight

If the prefix sum can be S_i after the first $i-1$ elements, then it can also be both $S_i - L_i$ and $S_i + L_i$ after processing the i-th element, provided that these do not fall outside the allowed range.

Solution

For each frequency:

- Keep a boolean table for each floppysecond, from 0 to T inclusive. Initialise with all `true`'s.
- Update iteratively as above.

For each frequency, the solution is in $O(nt)$.

G – Goblin Garden Guards

Problem

Given n positions of golins and c sprinklers with various radii, find out which goblins will become wet from the sprinklers.

Insight

Group all goblins in an array of sets, where goblins with the same x-coordinate are in the same set. For a sprinkler with radius r and position (x, y), check goblins at positions between $x - r$ and $x + r$.

Solution

- Store every group of goblins in an ordered set by y-value.
- Remove goblins soaked by a sprinkler in $O(a \cdot \log n)$ time, where a is the number of soaked goblins.
- Use `TreeSet.ceiling`, `multiset<int>.upper_bound` etc.

This yields an $\mathcal{O}((n + c) \log n)$ algorithm.

G – GoblinGardenGuards

Alternative solution

A heavily optimized naive solution might be able to pass with some added constant time optimizations, for example utilizing bounding squares around the circles to avoid costly operations as much as possible.

H – HeroPower

Problem

Given n notes and p Star Power (SP) phrases, play a simplified version of Guitar Hero. SP can be charged during SP phrases. Activate at any time, draining all SP that has been charged, but any SP phrase overlapped by an activation is unusable. Double points are awarded for every note hit during activation.

Insight

Each activation eventually leads to a situation where you're at the start of an SP phrase with 0 SP charged. From there, it's just like you're at the beginning of a new, smaller song.

H – HeroPower

Solution

- Find the maximum score for each sub-song.
- Reuse the results of later sub-songs when computing for earlier sub-songs – dynamic programming.
- Starting at phrase x, find the optimal activation ending no later than phrase y. Combine that with the optimal score starting at y.
- If implemented cleverly, computing the maximum score for a sub-song takes $O(n)$ time.

This yields an $O(np)$ algorithm.

I – iCar

Problem

Given a plane with vertical segments (red lights), find the shortest route using parabolas that does not cross any segment.

Insight

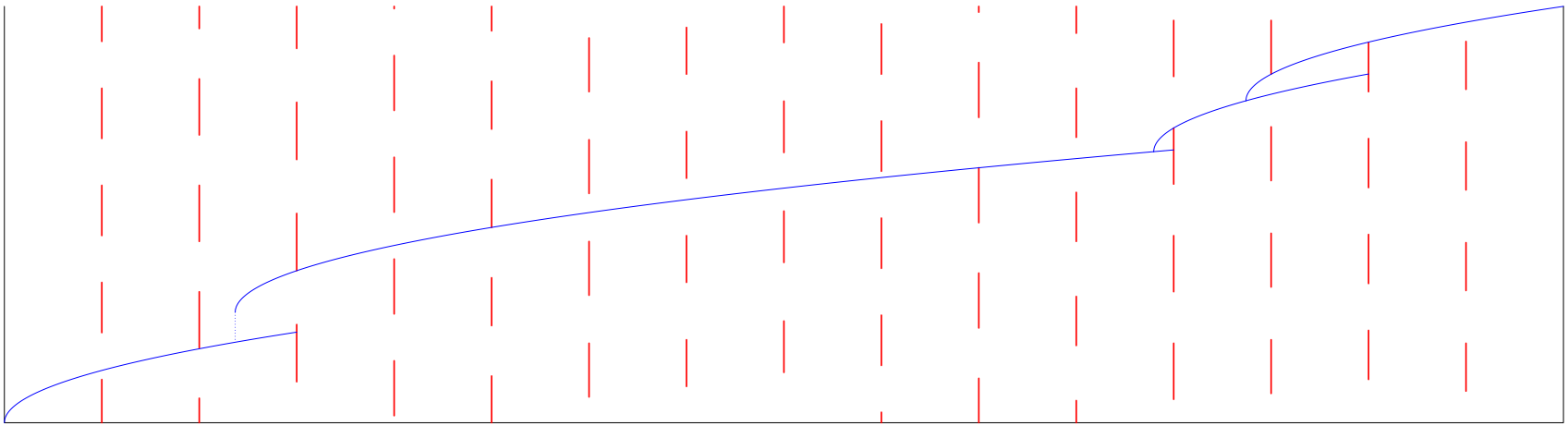
Only consider parabolas that satisfy two out of:

- Touch the beginning of a segment (red light)
- Touch the end of a segment (green light)
- Start at another parabola

Solution

- Compute parabolas that touch a green and a red light
- Add parabolas that touch a parabola and a light
- DFS on reachability graph

I – iCar



Example

- First parabola touches start (green) and a red light.
- Second waits and touches a red and a green light.
- Third starts from second and touches green light.
- Fourth starts from third and touches red light.

J – Just a Quiz

Problem

Find a strategy to correctly answer to as many questions as possible in limited time; interrupting is allowed.

Insight

Recurrence for time t and prefix q:

$\text{score}(t, q) = \max(\text{answer}, \text{pass})$

$\text{answer} = \Pr(\text{ok}) + \text{score}(t + 1, \emptyset)$

$\text{pass} = \mathbb{E}_w(\text{score}(t + 1, q \cdot w))$

Solution

- Store questions in a *trie* data structure (node $\rightarrow$ prefix)
- Compute probability of answering correctly at each node.
- Compute expected score for each node and time (DP).

Fun facts from the competition

- 307 teams competing at 19 different sites, with about 800 contestants in total
- First accepted submission after 00:03:33.
- Lukáš had 657 and Fredrik 634 e-mails in their inboxes regarding the contest
- 546 lines were enough to solve the whole contest and 227 out of that were solely for iCar – about 42% of the total
- iCar had 108 test cases, mostly randomly generated
- Slowest submission for A ran in 4.988 seconds, while the time limit was 5.