BAPC 2020 Preliminaries

Solutions presentation

November 14, 2020

A: Adversarial Memory

Problem Author: Mike de Vries and Ragnar Groot Koerkamp

- Problem: given that you can choose the cards, make Charlie's game of memory last as long as possible: at least $2n - 1$ turns.
- Keep track of which places already have an assigned card (in case the same card is picked), which cards you have shown once so far, and which cards you have not shown at all.
- Postpone Charlie making pairs as long as possible.
- For the first card that Charlie chooses:
  - Show him a card he has never seen before (unless the card is already fixed).
  - Once all card values have been shown at least once, choose them a second time.
- For the second card that Charlie chooses:
  - Show him a card that he has already seen earlier, if this doesn't give him a pair.
  - Else, show him a card he's never seen before, if possible.
  - Else, just give him the pair.
- Googling *Adversarial Memory*, the first hit gives the solution.

B: Binary Seating

Problem Author: Pim Spelier

- Problem: find the expected length of the exam, given the time it takes for each student to finish the exam
- Compute the average of $\max(S)$ over all subsets $S \subseteq \{t_1, \ldots, t_n\}$.
- Too slow: try all subsets: runs in $\mathcal{O}(n2^n)$.
- First sort the input: $t_1 \geq t_2 \ldots \geq t_n$.
- With 50% chance student 1 is in your room, and you have to wait t_1 time no matter what.
- If not, then with 50% chance you have student 2, and you have to wait t_2 time.
- Etc...

$$\text{Solution} = \frac{1}{2}t_1 + \frac{1}{4}t_2 + \cdots + \frac{1}{2^n}t_n.$$

- Runtime: $\mathcal{O}(n\log(n))$ for sorting.

C: Cutting Corners

Problem Author: Mike de Vries

- Problem: given a w by h rectangle, how much shorter is the diagonal than $w + h$?
- Plain old Pythagoras should do the trick.
- Print $w + h - \sqrt{w^2 + h^2}$ with sufficiently many digits.

D: Ducky Debugging

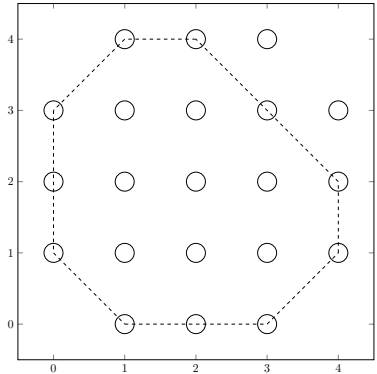
Problem Author: Mike de Vries

- Problem: help Bob debug his solution by imitating a rubber duck!
- Check the last character of every input line:
  - Is it a period (“`.`”)? Print “`*nod*`”
  - Is it a question (“`?`”)? Print “`Quack!`”
  - Is it an exclamation (“`!`”)? Exit program.
- Pitfall: Forgetting to flush output stream.

E: Eightgon

Problem Author: Mike de Vries; Timon Knigge

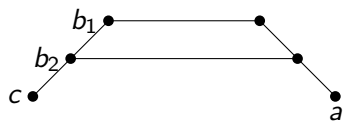
- Problem: how many regular, axis-aligned 8-gons can you construct from a given set of n lattice points?



E: Eightgon

Problem Author: Mike de Vries; Timon Knigge

- Problem: how many regular, axis-aligned 8-gons can you construct from a given set of n lattice points.
- Observation: can calculate all the number of ways to construct a partial octagon, starting from the top right, going anti-clockwise), keeping track of
  - Where you started (n possibilities)
  - Where you are now (n possibilities)
  - How many pieces you've used already (8 possibilities)
- These numbers have a recursive relation: $dp[a][c][k+1] = \sum_b dp[a][b][k]$ where the sum is over all c that are on the correct ray when viewed from b.
- For example: $dp[a][c][1] = 1$ iff the c'th point is on the ray going left from a.
- Can do a DP, with runtime $8n^3$: TLE!.



E: Eightgon

Problem Author: Mike de Vries; Timon Knigge

- Problem: how many regular, axis-aligned 8-gons can you construct from a given set of n lattice points.
- Second observation: can do the calculation $dp[a][c][k+1] = \sum_b dp[a][b][k]$ for all c at once!
- Reason: $dp[a][c][k]$ counts for all b on the ray from c.
- So: for each of the 8 possible directions, group the points by which line they lie on, and sort the points on each line.
- Then: for $dp[a][\cdot][k+1]$, go through every line, and go through all of their points in order.
- Complexity: $8n \log n$ for initial sorting and $8n^2$ for the DP.

F: Figure Skating

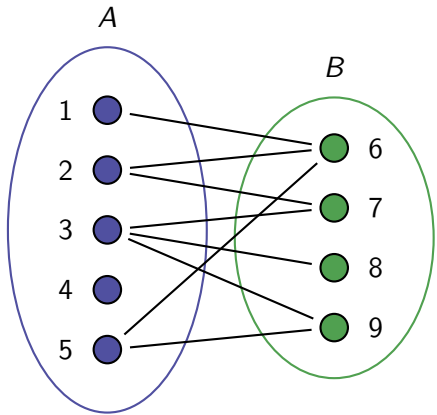
Problem Author: Timon Knigge

- Problem: find the contestant who rose the most places between predicted and final scoreboard.
- Loop over all contestants, keeping track of which contestant so far has the highest improvement.
- Also keep track of which position they ended up in, to break ties if necessary.

G: Group Project

Problem Author: Ragnar Groot Koerkamp; Timon Knigge

- Make as many pairs out of a group of N people, when they are split between groups A, B, and some mixed $A - B$ pairs are not allowed.
  - If at least one group has an even number of people, make pairs within the groups: $\lfloor \frac{n}{2} \rfloor$ pairs, which is optimal.
  - If both groups have an odd number of people, try to make one mixed pair, and the rest within the groups: in total $\frac{n}{2}$ pairs.
  - Only case where you cannot get $\lfloor \frac{n}{2} \rfloor$ pairs is two odd groups, where all of A connects to all of B.



G: Group Project

Problem Author: Ragnar Groot Koerkamp; Timon Knigge

How do you find out whether you have two odd-sized groups where every mixed pair is incompatible?

- Find the bipartite components with a graph traversal like BFS/DFS.
- Count the degrees: if
  - K nodes have degree $N - K$,
  - $N - K$ nodes have degree K,
  - and both K and $N - K$ are odd,

  you are in this situation.

  Special case: A and B have the same size, and every node has odd degree $N/2$.

H: Human Pyramid

Problem Author: Ludo Pulles

- Problem: in how many ways can we make a pyramid of height h with s strong people?
- Definitely a DP problem.
- Don't take horizontal layers, instead do *diagonals*!
- A diagonal consists of some strong people (possibly zero), and then some agile people (again, possibly zero) on top of them
- The ith diagonal can have at most one more strong person than the $i-1$th diagonal has.
- Let $\mathrm{dp}[i][j][k]$ be the number of ways to make a pyramid of height i with j strong people placed, and k the number of strong people in the current diogonal. Then

$$\mathrm{dp}[i][j][k] = \sum_{l=k-1}^{i} \mathrm{dp}[i-1][j-k][l].$$

- The answer is $\sum_{k=0}^{h} \mathrm{dp}[h][s][k]$.
- However, this gives an $O(H^5)$ algorithm: too slow!

H: Human Pyramid

Problem Author: Ludo Pulles

- You can solve this by slightly changing the dp array: let $\text{dp}[i][j][k]$ be the number of ways to make a pyramid of height i with j strong people placed, and *at least* k strong people in the current diagonal.
- Now
$$\text{dp}[i][j][k] = \text{dp}[i][j][k+1] + \text{dp}[i-1][j-k][k-1]$$
because there are either at least $k+1$ strong people in the current diagonal, or there are exactly k
- In the second case, we know that in the other diagonals there are precisely $j-k$ strong people, and the $i-1$th needs at least $k-1$ strong people to support the ith diagonal.
- The answer is $\text{dp}[\text{h}][\text{s}][0]$, with runtime $O(H^4)$.

I: In-place Sorting

Problem Author: Timon Knigge

- Problem: sort a list by flipping 6s and 9s.
- Go through the list one by one.
- If the current number cannot be made greater than the previous number, print "`impossible`".
- By flipping 6s and 9s, make the number as small as possible, but greater than the previous number.
  One possible strategy, assuming both numbers have same number of digits:
  - In the current number: replace all 6s with 9s.
  - From left to right, replace 9 by 6 if this does not make the current number smaller than the previous.
- If the new number has more digits than the previous number, replace all 9s with 6s.

J: Jam-packed

Problem Author: Jorke de Vlas

- Problem: distribute n jars over boxes (with capacity k) so that the box with the fewest jars is as full as possible.
- Idea: use the lowest number of boxes possible, which is

$$\#\text{boxes} = \left\lceil \frac{n}{k} \right\rceil .$$

- The average number of jars in a box is n divided by the number of boxes, hence the emptiest box will have at most

$$x = \left\lfloor \frac{n}{\#\text{boxes}} \right\rfloor$$

jars. We can achieve this by filling (n mod k) boxes with $x + 1$ jars and the rest with x jars.

K: Kangaroo Commotion

Problem Author: Abe Wits

Problem: Given a grid with $k + 2$ locations in it, and some blocked cells, jump in order to each location, where the absolute value of the acceleration between consecutive jumps is at most 1 in the x and y direction.

K: Kangaroo Commotion

Problem Author: Abe Wits

Problem: Given a grid with $k+2$ locations in it, and some blocked cells, jump in order to each location, where the absolute value of the acceleration between consecutive jumps is at most 1 in the x and y direction.

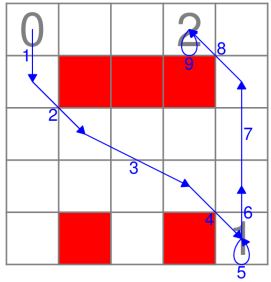


Figure: First sample input

K: Kangaroo Commotion

Problem Author: Abe Wits

- Use BFS with state (position, velocity, next location).
- The number of positions is $r \cdot c \leq 50^2 = 2\,500$.
- Velocity has $-50 \leq v_x, y_y \leq 50$, so this gives 101^2 states.
- There are $k + 1 \leq 6$ possibilities for the next location to be visited.
- The total number of states is:

$$(r \cdot c) \cdot (2r + 1)(2c + 1) \cdot (k + 1) \leq 153\,015\,000$$

- From each state, we can go to (up to) 9 other states.
- Trying all $9 \cdot 153\,015\,000 = 1\,377\,135\,000$ state transitions would give TLE, and storing all states may need too much memory.

K: Kangaroo Commotion

Problem Author: Abe Wits

- Observation: as acceleration is limited, maximum speed is limited. In particular, when accelarating $v_x = 0$ to $v_x = v_{\max}$ and decelerating back to $v_x = 0$, one moves at least

$$1 + 2 + \cdots + (v_{\max} - 1) + v_{\max} + (v_{\max} - 1) + \cdots + 1 = v_{\max}^2$$

  steps in the x-direction. So $v_{\max} \leq 7$.
- The number of states goes down to

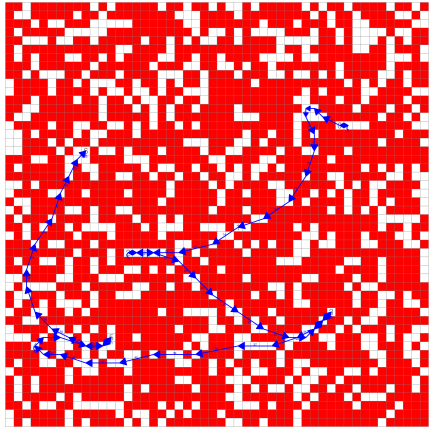
$$(r \cdot c) \cdot (2 \cdot 7 + 1)^2 \cdot (k + 1) \leq 3\,375\,000.$$

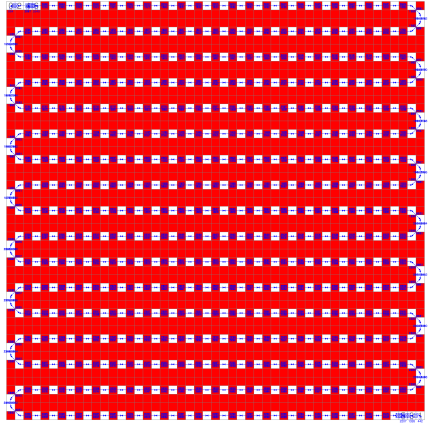
- We can easily visit all states and state transitions within the time bound.

K: Kangaroo Commotion

Problem Author: Abe Wits

Several secret test-cases. Number of steps needed on the right: 2649.





Some stats

- 1045 commits: up from 400 last year
- 356 secret testcases
- 221 jury solutions
- The number of lines the jury needed to solve all problems is

$$18 + 2 + 3 + 5 + 27 + 4 + 15 + 13 + 19 + 1 + 46 = 153$$

On average 13.9 lines per problem, up from 12.3 last year!

The Proofreaders

- Erik Baalhuis
- Nicky Gerritsen
- Rafael Kiesel
- Shane Minnema
- Michael Vasseur
- Kevin Verbeek
- Mees Vermeulen

The Jury

- Ruben Brokkelkamp
- Daan van Gent
- Ragnar Groot Koerkamp
- Joey Haas
- Freek Henstra
- Boas Kluiving
- Timon Knigge
- Ludo Pulles
- Maarten Sijm
- Harry Smit
- Pim Spelier
- Jorke de Vlas
- Mees de Vries
- Mike de Vries
- Wessel van Woerden