NCPC 2012
Presentation of solutions

Heads of Jury: Jon Marius Venstad and Tommy Färnqvist

2012-10-06

NCPC Jury

Problem authors

- Andreas Björklund (ARM)
- Ulf Lundström (KTH)
- Lukáš Poláček (KTH)
- Daniel Espling
- Tommy Färnqvist (Linköping University)
- Matias Holte (UIO)
- Christian Neverdal Jonassen (NTNU)
- Michał Pilipczuk (UIB)
- Fredrik Svensson (Autoliv Electronics)
- Pehr Söderman (KTH)

A – Aaah!

Solution

- The input is guaranteed to be only 'a's and 'h's, so simply compare the length of the first and second lines of input.

D – Doorman

Solution

- Observe that Bruno can actually let people pass him in the order they are lined up, *if* he tells a person about to be given entrance to the club of the gender so far dominating the club to wait behind him for a while. As soon as he tells another person to wait behind him, the person already waiting may now enter the club.
- This way it is easier to see that Bruno looses count when the absolute difference of the genders of the people actually let in to the club equals Bruno's max, and the person waiting behind him is of the same gender as the one first in line (if any).
- So simply compute the absolute difference for increasing prefixes of the string. When it gets 2 above Bruno's max output the prefix length - 2. If it gets 1 above Bruno's max when the prefix equal the whole queue, output length - 1, otherwise output the length.

G – Galactic

Problem

Given a set of lines, how many extra lines will you have to add to make them split the plane in at least W parts with infinite area?

Solution

- Remove all duplicate lines.
- Check if there are enough lines already.
  - If all (N) unique lines are parallel there will be $N + 1$ areas.
  - Otherwise there will be $2N$ infinite areas.
- If you add M lines you can get $2(N + M)$ infinite areas, so if you need to add lines you will need $\texttt{max}\left(1, \left\lceil \frac{W}{2} \right\rceil - N\right)$ of them.

H – Horror List

Solution

- Database is an undirected, unweighted graph
- Create a special node H, connected to all movies on the horror list.
- Find shortest path from H to all movies, output the longest of these paths

J – Juice

Solution

- Solve greedily by checking the houses in order of increasing demand:
- Assume an optimal solution O and the greedy solution G agree on which of the first $i-1$ demands to meet, but not on the i-th. Clearly G meets this requirement, as it is greedy, so O does not. Then there is a $k > i$ such that the k-th demand is met in O, and not in G, and where the k-th demand shares a maximum number of edges with the i-th. If swapping the k-th demand for the i-th in O produces an infeasible solution, then O must meet some j-th demand that shares more edges with the i-th demand than the k-th does. Because of how k was chosen, this means that $j < i$, which contradicts the assumption that O and G agreed on all demands $j < i$.
- DP solutions were also accepted – for the effort.

C - Cookie Selection (1/2)

- Using e.g. `std::nth_element` repeatedly is to slow.

Solution

- Keep two priority queues, implemented with heaps, with cookie diameter as priority. One maxPQ, and one minPQ.
- If a cookie with diameter d arrives:
  - If d is strictly larger than the minimum element of minPQ:
    - Insert d into to minPQ.
    - If the size of minPQ is strictly larger than the size of maxPQ + 1, transfer the minimum element from minPQ to maxPQ.
  - Else:
    - Insert d into to maxPQ.
    - If the size of maxPQ is strictly larger than the size of minPQ, transfer the maximum element from maxPQ to minPQ.

C - Cookie Selection (2/2)

Solution

- To send a cookie:
  - Remove the minimum item m from the minPQ.
  - If the sizes of minPQ and maxPQ differ, transfer the maximum element from maxPQ to minPQ.
  - Print out m.
- This way, each operation takes time $O(\log_2(\text{number of cookies currently in the holding area}))$.

E - Eco-driving

Solution

- Guess maximum turning angle. Do binary search.
- Apply shortest path, with the assumed restriction.
  - Vertices are pairs $(v, previous)$ – a vertex and the previously visited vertex.
  - The angle can be computed by e.g.
    - using scalar product between vectors or
    - complex numbers and their arguments.

I – Infiltration

Solution

1. Observe that no pair of words except "our" and "rum" can match the same encrypted word. That helps in bounding the running time for the following algorithm.
2. Try to match the known words recursively to the encrypted words, remembering the partial substitution mapping learnt so far.
3. If there is a mapping in which every letter of the encrypted text is mapped, remember the mapping.
4. Whenever you find a new matching of the known words such that all letters are given, see if it is another mapping than the one you already found. If so output "Impossible".
5. Otherwise if you found a unique mapping decrypt the input encrypted text with the found unique mapping, or output "Impossible" if no mapping were found.

B – Bread Sorting

Number of inversions: number of pairs that are in the wrong order, e.g. when 5 is before 2.

Solution

- Each flip of 3 breads changes the number of inversions by $-2, 0$ or 2. Parity stays the same.
- Move the first element to its position, then 2nd, 3rd etc. The last two elements may be in the wrong order – the permutation has wrong parity.
- Calculate the parity of both permutations in $O(n \log n)$ by counting inversions or in $O(n)$ by counting cycles.

L – Luggage

Problem

Find maximum constant conveyor belt speed that does not cause any collisions when the bags are dropped onto the luggage carousel.

Solution

- Each pair of bags will have some intervals of speeds for which they collide; one interval for each number of turns the luggage carousel makes between the two bags are dropped.
- Find all such speed intevals. They are $\left(\frac{|x_i - x_j|}{nL+1}, \frac{|x_i - x_j|}{nL-1}\right)$, where x_i is the position of bag i, n is the number of turns and L is the carousel length.
- Sort the intevals to find the highest speed ($0.1 \leq v \leq 10$) which is not covered by an interval.

K – Kindergarten

Solution

Reduction to 2SAT:

1. Loop over (or even better, binary search for) the asked bound T.
2. For each guessed value T, encode the resulting problem as a 2SAT instance with one variable per kid, with true meaning one of the two possible teachers, and false the other.
3. Set up all pairwise conflicts as binary clauses.
4. Solve with standard algorithm for 2SAT.

F – Food Review (1/2)

Problem

Given are graphs G and H. Want to make G Eulerian by adding edges from H.

If G is connected

- Want to match vertices of odd degree and add the shortes path in $(G \cup H)$ between matched vertices.
- Find the minimum cost matching on odd vertices in the graph $(G \cup H)^*$ – "transitive closure" of $(G \cup H)$, edges are shortest paths.
- Possible in $O(2^n)$ using dynamic programming over subsets.

F – Food Review (2/2)

If G is not connected

- Make G connected by adding a spanning tree between components of G. There are at most $c^{c-2}2^{n/2}$ spanning trees (c is the number of components of G).
- Other solutions with dynamic programming in $O(2.82)^n$ and $O(2^n)$.