NEERC 2010 Problem Review

© Roman Elizarov

Alignment of Code

- Read each line and split it into words
- Compute max width of each word on a line
- Write the resulting text with the required number of spaces between words

Binary Operation

- Write a procedure to compute for single digits:
  $a_0 * a * a * \ldots * a$
  \-----------/ *c* times
  - Repeat multiplies until it loops (after at most 10 muls)
  - Use offset and period length to compute the result
- Write a procedure to compute for single digits:
  $a_0 * a' * (a+1)' * \ldots * b'$
  \-------------------/ *d* times
  where +1 wraps to 0;
  and *a*' means "$a * a * \ldots * a$" *c* times (as above)
  - Compute loop similarly (can be as long as 100)

Binary Operation: altogether

- Compute the result digit by digit using two of the above procedures
  - For a digit i, round the number a up and b down to the nearest multiply of 10^i
  - Represent the number range [a,b] as:
    - X..XXaXX..X
    - X..XX(a+1)00..0
    - X..XXb00..0
    - X..XXbXX..X

Cactus Revolution

- Use DFS to find and enumerate all loops in the graph.
- Use DFS on a cactus to partition it:
  - Each partitioning procedure returns the remainder nodes that do not sum up to the target size of partition ($t = n/k$).
  - Partition a node by recursively partitioning the loops it is a part of (with the exception of a parent loop, if any) and it child nodes (with the exception of a parent node, if any)
  - Remainders must add up to less than target size

Cactus Revolution: loops

- Loops (without one node) are partitioned by recursively partitioning all nodes on a loop, then combining result.
- To combine the result we have to find an integer s ($0 <= s < t$), so that some number of first remainders sum up to s, some next ones sum up to $s + t$, next to $s + 2t$, etc.
  - s is a running sum of remainders is modulo t.
  - Find which sum module t is the most popular and try it as a candidate s
  - Treat zero reminders on a loop in a special way

Dome of Circus

- Assume we have a single point (x, y, z)
  - Let us define $v(h)$ = volume of a cone with height h going through point (x, y, z). This function can be computed with some basic geometry
  - Function $v(h)$ is convex
- For n points the volume is $\max(v(h))$.
  - It is also a convex function
  - The optimal dome's volume for the problem is the min of this function
  - It can be found using ternary search
  - The radius r can be found using volume and h

Evacuation Plan

- Sort the team's and shelter's locations
- The optimal assignment will assign consecutive ranges of teams to a shelter (after sort)
- Find the answer using two-parameter dynamic programming considering this sub-problem:
  - $P(u,v)$ – the total fuel required to match the first u teams to the first v shelters
  - $P(u,v) = |loc(u)\text{-}loc(v)| +$
    $\min(P(u\text{-}1,v\text{-}1), P(u\text{-}1,v))$

Factorial Simplification

- Sort all *p*s and *q*s
- Find all prime numbers up to max *p* and *q*, and also find the next prime number after that
- Compute the formula using representation of all numbers as products of primes in some power
  - Multiplication of numbers adds the powers
  - Division of numbers subtracts the powers
- The largest factorial factor in the result can be as large as the next prime minus 1.
  - Find the result by repeated division by smaller and smaller factorials

Game of 10

- Keep track of the game field after each move, including the number of filled cells in each row and column and their sums
- Use the following winning algorithm:
  - If you can make a winning move (close row or column with a sum of 10), then make it and declare “WIN”
  - If not, then take the previous opponent’s move (r, c, k) and make a move $(5 - r, c, 5 - k)$
  - Note that $10 = 1 + 2 + 3 + 4$

Hands of Poker

- The only relevant information for ranking is the list of card ranks (in the descending order) and a flag of whether it is a flush or not.
- Generate all possible representation in the above form (there are 7462 of those):
  - Each rank can occur at most 4 times
  - If any rank occurs more than once, it cannot be a flush
- Sort representations per problem statement
- Read hand, determine its representation and find its place in a previously sorted array of representations

Ideal Path

- Find the distance from all nodes to *n* with BFS
- Start with a set containing node 1
- On each steps:
  - Find the lowest color that can be used to go from a node of the current set to the node with one less distance to *n*
  - Find next set of nodes with one less distance to *n* going from the current set via the lowest possible color
- The resulting sequence of colors in the answer

Jungle Outpost

- Use binary search to find an answer
- Assume the answer is m. Enemy blows up m towers. Were the headquarters can be located to be protected after destruction of any m towers?
  - They can be located in some convex polygon
  - This convex polygon is an intersection of half planes going from point i to point $i+m+1$
  - Using a procedure to intersect convex polygon with a line we can figure if the resulting intersection is empty (headquarters cannot be made secure) or not

Jungle Outpost: alternative

- Instead of building convex polygon to check if it is empty…
  - Use simplex method to check if a set of n inequalities has a common solution in two variables x and y
  - Or use randomized methods

K-Graph Oddity

- If an on an odd K-Graph at least one node has the degree *strictly less* than *k*
- Run DFS starting from this node
- When backing out from DFS color each node
  - Each node in DFS tree will have strictly less than *k* children, so a unique color can be always found