2015 Rocky Mountain Regional Programming Contest

Solution Sketches

Credits

- Darko Aleksic
- Howard Cheng
- Zachary Friggstad
- Warren MacEvoy

Popular Vote

- Straightforward
- Just sum up the votes, and find the maximum and see if it is unique and more than half of the total.
- Minor detail: be careful with integer division.

Card Game

- You could represent this as a 2-SAT instance (clause = card, variables = face), but the instances are too big.
- Let the different face values be nodes in a graph.
- Add an undirected edge connecting the two faces of each card (multiple edges and self loops are allowed).
- Faces that are not in the same connected component can be treated independently.
- In a component, there is no issue unless there is a cycle. A single cycle is okay, but not more. Self-loops count as cycles.

Amazing Race

- Similar to dynamic programming solution for travelling salesman. Complexity: $O(n^2 \cdot 2^n)$
- For each subset of locations S and each $u \in S$, we want to compute

  $f(S, u) =$ shortest time to visit all locations in S and ending at u.
- The recursion is

  $$f(S, u) = \min_{v \in S \setminus \{u\}} (f(S \setminus \{u\}, v) + \text{dist}(v, u) + t_u.$$

  care needs to be taken to exclude v if we cannot finish the task at u by going through v.
- Maximize the points obtained over all subsets S and $u \in S$ such that $f(S, u) + \text{dist}(u, \text{dest}) \leq T$.

Scaling Recipes

- Straightforward: just follow the “recipe”!
- Can be done completely with integer arithmetic to avoid rounding errors, but not needed here.

Space Junk

- Two spheres touch/collide if the distance between their center is less than the sum of their radii.
- Represent the position of the center of each sphere parametrically as:

$$p(t) = (x, y, z) + t(v_x, v_y, v_z)$$

- Write the equation:

$$\|p_1(t) - p_2(t)\|^2 \leq (r_1 + r_2)^2$$

Expand everything and collect like terms in t gives a quadratic inequality

$$at^2 + bt + c \leq 0.$$

Space Junk (cont.)

- Since $a \geq 0$, the problem is equivalent to finding the roots of the quadratic equation $at^2 + bt + c = 0$.
- Only the smaller root matters because the two spheres do not touch initially.
- Some special cases:
  - no roots ($b^2 - 4ac < 0$)
  - collision before current time
  - $a = 0$

  No collision in all of these cases.
- Collision/no collision detection can be done in integer-only computations to avoid round-off errors. Unnecessary here.

A Classy Problem

- To compare two class names, work backwards and compare them until there is a difference.
- If one class name is shorter than the other, assume the rest of the shorter name is “middle”.
- Slightly tricky to pick the class names apart.

Rubik's Revenge in . . . 2D!? 3D?

- Standard state space search:
  - each state is a configuration of the puzzle
  - there is an edge from one state to another if it can be reached in one move
- Standard breadth-first search solves the problem but it is too slow:
  - there are $\frac{16!}{(4!)^4} \approx 63 \times 10^6$ states.

Rubik's Revenge in . . . 2D!? 3D? (cont.)

- A bidirectional search can cut down the search space significantly. (Roughly square root of the original search space.)
- Search backwards from the goal for up to 6 moves (half the maximum number of moves). Record the distance of each state reachable from the goal in 6 moves or fewer.
- Search forward from the initial state until it reaches one of states discovered in the previous step.
- The total distance is the sum of the distance from the initial state and the distance to the goal.

- We want to find the smallest base $b \geq 4$ such that

$$n \equiv 3 \bmod b$$

- Or in other words, $n - 3$ is divisible by b.
- So just find the smallest divisor ≥ 4 of $n - 3$.
- Trial division of all b up to $\sqrt{n-3}$ is sufficient.

Matrix Keypad

- Observation: if we have two buttons pressed that are not on the same row or the same column, then all four corners of the “rectangle” formed are indeterminate.
- Once we have the rectangles, we make sure that all other “corners” are connected. Otherwise it is impossible.
- The only time when we know for sure a button is pressed is when all the buttons pressed are in the same row or same column.

I’ve Been Everywhere, Man

- Straightforward, can be done in number of ways.
- Easiest to use a “set” data type to collect the city names and ask for the final size of the set.

Bundles of Joy

- Dynamic programming.
- For each bundle, the cheapest way to buy that collection of cakes is either:
  - buy the bundle itself
  - buy a collection of smaller bundles that make up the bundle
- Care needs to be taken when two bundles have the same cakes.
- To determine the collection of smaller bundles involved, it is sufficient to do it "brute force" and check every other bundle. Complexity $O(m^2 n)$.
- There are faster ways to determine the collection of smaller bundles involved, but it's not needed for this problem. It is possible to solve the problem in linear time (i.e. proportional to the sum of items in each bundle).