Hamburg Collegiate Programming Contest at TUHH

Christian Renner and Gustav Munkby

Solution Outline

7th July, 2010

Institute of Telematics
Hamburg University of Technology
TUHH

The King of Atlantis

- Places on the island and the paths are a directed graph
- Times of travelling are edge weights
- Check whether there is a time path from each node to the light house shorter than or equal to T (DFS, Dijkstra)
- Doing this search N times could be too slow
- Better: Invert edge direction and find shortest path from light house to all other places. Solution ready in one run.
- Runtime $\mathcal{O}(N^2 + M)$ or $\mathcal{O}(N \log N + M)$ (if using a heap)

New Bees in Town

- Find (x, y), so that
  $\sum_i h_i \cdot (x - x_i)^2 + h_i \cdot (y - y_i)^2 = \text{min!} \quad \left(h_i = \left\lceil \frac{f_i}{10} \right\rceil\right)$
- Can be solved by computing gradient:
  - $x = \frac{\sum_i h_i \cdot x_i}{\sum_i h_i}$
  - $y = \frac{\sum_i h_i \cdot y_i}{\sum_i h_i}$
- Read in data, compute the three sums, calculate (x, y)
- Watch your data types (`double`, `long long`)
- Runtime $\mathcal{O}(N)$

Carnival Planner

- Traffic network is a directed graph.
- Remove nodes (or mark as removed) in carnival.
- Strongly connected components are isolated parts.
- Tarjan's algorithm
- Runtime: $\mathcal{O}(I + R + C)$

No Dollars at Hand

- Read in exchanges and put them into list
- During read-in, sum up available dollars. If less than what you need, impossible.
- Sort list using exchange ratio (best € to $ conversions first)
- Sum up $ and € in sorted list, until you reach the point where you would have more dollars than you need (using ratios here might lead to rounding errors!)
- Fill up the gap and compute needed euros (round up)
- Finally, check your budget!
- Output using `fixed` and `setprecision(2)`
- Runtime: Sorting avg. $\mathcal{O}(N \log N)$, Calculating sums $\mathcal{O}(N)$

Easy Cash

- $\text{gain}(0) = 0$
- $\text{gain}(1) = M_1$
- $\text{gain}(i) = \max(M_i + \text{gain}(i-2), \text{gain}(i-1))$
- Try all possibilities $\Rightarrow$ exponential growth, infeasible
- Simple dynamic programming
- Use `long long`
- Runtime: $\mathcal{O}(2^B)$ without DP, $\mathcal{O}(B)$ with DP.
- Storage: $\mathcal{O}(1)$, since we only need two at the time.

Flower Power

- Flowers and pots are bipartite graph
- Edges between flowers and pots that go along
- Find maximum matching between flowers and pots (e.g., Ford-Fulkerson)
- Output number of flowers - matching number
- Runtime: $\mathcal{O}(\min(F, P) \cdot F \cdot P)$
- In large data sets, a hashmap for the colors could be useful

Hamburg Collegiate Programming Contest at TUHH

Christian Renner and Gustav Munkby

Solution Outline

7th July, 2010

Institute of Telematics
Hamburg University of Technology

TUHH