ACM ICPC — Training Session V

C. Maria Keet

Department of Computer Science, University of Cape Town, South Africa
mkeet@cs.uct.ac.za

September 27, 2014

Today

- Some recap on strategy, problems
- SWERC 2013 Regional—use icpc archive to submit
- Try to solve as many as possible in 3.5h
- Then we'll discuss the solutions

Approach

- Read description
- What is the task?
- What is given?
  - data, variables, constraint, examples
- Solve the problem
  - 'essentially solve it'
    - input data space
    - recognise underlying core issues
    - similarity with other problems
    - result: a solution on paper, knowing **what** needs to be done
  - solve it algorithmically (**how** to do it—e.g., a sort, OSPF, complexity, ...)
  - code and test it, i.e., **do** it and verify solution

Types of puzzles

- Regarding the core problem
  A. maths-y (e.g., probabilities, geometry)
  B. algorithmically/general (still an elegant solution)
  C. seeing patterns, and brute force
- For the algorithms: which class of algorithm would be needed?
  i. e.g.: simple sorting, searching, graphs, numerical, combinatorial, sets, strings, geometry?
  ii. within the class, which type? e.g., geometry: convex hulls, range search, polygons, shape similarity, ...

Important problem-solving approaches in computing

CS Complete search (brute force)
- solve problem searching the entire search space

D&C Divide & Conquer
- make problem 'simpler' by dividing into sub-problems (usually half the size)

Gr Greedy
- make locally optimal choice at each step

DP Dynamic Programming
- problem that has overlapping subproblems and optimal substructure

Tips for competitive programming

- General tips:
  - Type Code Faster
  - Quickly Identify Problem Types
  - Do Algorithm Analysis
  - Master Programming Languages
  - Master the Art of Testing Code
  - Practice and More Practice
- Know your problem solving paradigms in CS: complete search, divide and conquer, greedy, dynamic programming

Competitive programming 1, by Steven and Felix Halim: `https://sites.google.com/site/stevenhalim/`

Teams

- Your team overall needs to have several skills: problem solving, designing, coding
- Only one computer per team!
- This leaves lots of room for interesting team make-up and team strategy
- Choosing which problems to solve first, splitting up the problems among the members, not all focusing on the same problem at once, etc requires a team strategy
- Being able to solve those problems, recognising basic algorithms and repurposing them for the task, and programming are skills that require practice

Teams

- Your team overall needs to have several skills: problem solving, designing, coding
- Only one computer per team!
- This leaves lots of room for interesting team make-up and team strategy
- Choosing which problems to solve first, splitting up the problems among the members, not all focusing on the same problem at once, etc requires a team strategy
- Being able to solve those problems, recognising basic algorithms and repurposing them for the task, and programming are skills that require practice

More on cooperation and strategy

- Team manager needs to keep track of time
- Pair programming
- If stuck at coding, let another team member use the computer
- Don't sit idly by—work on another problem, make test cases, design a solution
- Discuss problem with another team member
- ...

Problem A–Mixing colours

- Counting 'forward' requires one to test *all* options, which is going to give you a TLE in most cases
- Idea: count 'backward' from possible final solution to start colours, check probability, take highest.
- How to count backward? invert the rules; e.g.:
  - instead of `Blue Yellow` $\rightarrow$ `Green`
  - use `Green` $\rightarrow$ `Blue Yellow` and `Green` $\rightarrow$ `Yellow Blue`
- It's a context-free grammar then
- Efficient way to implement that is the CYK algorithm

Problem B–It can be arranged

- Requested: minimum number of rooms that need to be rented
- Use maximum flow from S to T, and C set to sum of all classes required by the courses
- Then check if the capacities from S to each course N_i has been satisfied
- Run a binary search with $C = [1, nrOfClasses]$ to find the minimum value of C that satisfies all courses

Problem C–Shopping malls

- Different stretches wit different costs
- Set up directed graph with costs per segment
- Compute the shortest pat from A to B
- i.e.: typical graph & shortest path problem

Problem D–Decoding the hallway

- Complicated description, but 'easy' task!
- Test sub-string membership

Problem E–Joe is learning to speak

- Main choice is how to store the words and phrases Joe knows, and querying them
- Would a list work, an array, hash table?
- Note: used in computational linguistics, known as n-grams

Problem F–Odd and even zeros

- Brute force computing all factorials each time for each input number will take too much time
- A 'cleverer' option: scan the input for the largest number, compute factorials for all up till that one, store each output, check each on even number of zeros, then process input to get the answer (how many have an even number of zeros)
- But is it really feasible to compute $n!$ when $n \leq 10^{18}$?
- A mathsy option, using number theory (see answer sheet).

Problem F–Odd and even zeros

- Brute force computing all factorials each time for each input number will take too much time
- A 'cleverer' option: scan the input for the largest number, compute factorials for all up till that one, store each output, check each on even number of zeros, then process input to get the answer (how many have an even number of zeros)
- But is it really feasible to compute $n!$ when $n \leq 10^{18}$?
- A mathsy option, using number theory (see answer sheet).

Problem G–VivoParc

- This is a nice instance of the graph colouring problem
- Take the enclosures as nodes, visibility as edges, species as colour

Problem H–Binary tree

- see SWERC 2013 solutions.pdf
- `http://users.dsic.upv.es/swerc/problems.php`
- .
- .

I–Trending topic

- see SWERC 2013 solutions.pdf
- .
- .
- .

J–Cleaning the hallway

- see SWERC 2013 solutions.pdf
- .
- .

Scheduled training dates

- Aug 2: 9:30-15:30
- Aug 16: 9:30-15:30
- Aug 30: 9:30-15:30
- Sept 13: 9:30-15:30
- Sept 26: deadline for registering for the regionals
- Sept 27: 9:30-15:30
- **Date of the regionals: 4 October, 2014**