ACM ICPC strategy

Lukáš Poláček

October 13, 2011

This article is about my experience with ACM contests. I learned this from other people but mostly from my own and my team's mistakes.

These are the three most important facts you should bear in mind during the ACM ICPC contest:

1. It is a team competition and each team has 3 members.
2. Computer time is very expensive.
3. The beginning of the competition is the most important part of the contest.

1 General strategy

From these facts you can derive a very good strategy.

At the beginning of the contest, the quickest typist writes code templates and a few scripts (e.g. `'c'` for compiling with the right flags). During the same time, the two others skim through the problem set in search for an easy problem (one person going from the back, one from the front). As soon as a sufficiently easy problem is located (one that can be solved within, say, 10-15 minutes), it's given to the quickest typist along with a quick description. Due to ICPC scoring it's optimal to solve easier problems first.

The other two non-coding teammates continue skimming through the problem set. If something easier than the first problem is located (solvable in 5 minutes), it takes priority. After at least one member of the team has read each problem statement, the two non-coding teammates should discuss all the problems they have just read.

What should not happen in the beginning: After reading a problem I see that I can solve it. I don't read other problem statements and solve this problem for next 2 hours without success. But there were at least 2 other easier tasks that I didn't read. That's why reading all problems and discussion between members in the beginning is so important.

The team continues with solving problems. If someone knows how to solve a problem and nobody uses computer, he should code it. If the computer is not available, he should prepare the code on the paper.

2 Useful advices

You are now familiar with the general strategy. We continue with useful advices.

- If you are not sure about your solution, discuss it with your teammates. If stuck, explain problem to a teammate. Use good judgement if it's worth the interruption of him/her.
- If you have time, write code on the paper before you go the computer. It will save a lot of computer time of the whole team. You don't have to write whole code but try to focus on the most important parts of the program. For example if you are writing binary search, make sure you are using the right invariant.
- Do not debug code on the computer. Print your code and debugging output and debug on paper.
- If you are stuck on a problem, take a walk or go to the toilet. The best ideas come to mind here.
- If you keep getting WA on a problem, let it be for a while and try to solve another problem. Maybe you will get an idea how to solve it afterwards. Don't hesitate to do a complete rewrite of a solution. For most of the problems it can be done in 15 minutes.
- Is it easy to generate some large inputs, or inputs where you know the answer? If so, it may be worth doing so to test a bit more before submitting.
- When you're done with a problem, throw all papers concerning that problem onto the floor (problem set page, printouts, handwritten stuff). Saves some time searching for paper, and feels good.
- Look at the scoreboard every now and then. If there is a problem that all other teams solved, it should be easy.
- Keep track of all submissions on a sheet of paper, and keep track of who's working on what problem.
- Print early, print often. Print every time you submit.
- Don't forget the endgame strategy. When time is starting to run out you don't want all three people working on separate problems, but focus on one problem. Try to make sure that all people are still doing something useful (e.g. one person looking over the back of the coder, and another person trying to come up with tricky test cases). Knowing when to enter this mode can be difficult and in particular it takes some willpower to give up on those additional problems that one knows how to solve but just have to code up...

- So called "free submit" mode is to be used with caution. If it is not clear, its meaning is roughly "at this point, solving another problem is more important than any time penalty it can possibly cost us so let's just submit as soon as we have something that has a non-zero probability of getting accepted". Usually it is not entered until the last 30 minutes or so, but if you have a bad start with many incorrect submissions so that you already have a large penalty, it can be sometimes be entered very early (and sometimes exited if the outlook improves).

Acknowledgments

Thanks to Gunnar and Per for comments.