Massachusetts Institute of Technology
6.854/18.415: Advanced Algorithms
David Karger

Handout 15
11/12/99

Course Project

This handout provides a brief description of the kinds of things I am looking for in the course project. The basic goal of the project is for you to independently apply some of the advanced algorithmic thinking you have (hopefully) been developing in this class. The course project involves a combination of goals—encountering and independently absorbing new material, doing some thinking about it, and presenting what you have done. Different projects will attain these goals in different proportions.

There are 3 main ways to approach the project.

Reading project. Find a few interesting but challenging theoretical papers on a coherent topic, and write up an exposition that synthesizes their ideas. You will be graded on how much clearer/more informative your exposition is than that of the original papers (so the original papers should be pretty tough). While you need not give all proofs in full detail, the reader should come away with a good understanding of why it all works.

Theoretical research. Develop an interesting new solution to an algorithmic problem. "Interesting" may mean more efficient or may mean simpler. While you will presumably need to do some background reading, there is no set lower bound on it so long as you are able to advance beyond what is known. This kind of project will be graded on the extent of the improvement on previous results.

Implementation project. We have studied many algorithms in theoretical form. Their behavior when implemented can be quite surprising. Grab one that interests you, implement and test it as above. This kind of project will be graded on how well you explain the algorithm you are implementing, what kind of interesting heuristics you developed, what interesting test cases you ran them against, and how well you interpret the results.

Algorithms papers tend to leave out a lot of little implementation details that turn out not to be so little when the time comes to implement. You may also explore implementation of heuristics that have "no theoretical value" but may lead to enormous improvements in practice. Once you have an implemented algorithm, you can test it to see how it performs in practice. This involves devising interesting "hard" inputs that make the algorithm perform poorly. Study of the algorithm's behavior may lead you to make changes in the algorithm and test them. A typical implementation paper says "we implemented algorithm X and it was awful, then we added heuristic Y and it was great!"

By cautious—implementations can take a lot of time. Make sure what you are trying to implement is a reasonably "small" algorithm. Give serious thought to test inputs—ideally, they will come from real-world problems, or will be specially designed to "stress" some aspect of the algorithm. You probably also need a control (i.e. dumb) algorithm to compare yours to.

Regardless of which route you take, the end result should be a roughly 10 page paper describing the results. Presentation quality will be a factor in your grade.

Some obvious question:

What topic should I work on? The best topic to pick is one you are interested in anyway. Many of you are already involved in some research project; some thought may reveal an algorithmic component. Perhaps your system is presently using a naive algorithm for X and could be improved by using a more sophisticated one. You can do background reading on the topic (reading project), develop a theoretical model of the problem and devise a solution (theoretical project) or take some extant algorithm and try it out (experimental project).

If you aren't working on anything, try browsing through what we've covered in class as well as papers (see below) to identify an area that sounds like fun.

You are **not** limited to the topics we have covered in class. Anything that uses combinatorial ideas and a theoretical framework to devise more efficient solutions is fair game.

Where can I find papers? There are numerous sources of papers on algorithms. The best work in the area is published in three conferences, each with yearly proceedings:

- The ACM Symposium on Theory of Computing (STOC).
- The IEEE Symposium on Foundations of Computer Science (FOCS).
- The ACM-SIAM Symposium on Discrete Algorithms (SODA).

There are lots of other sources, of course. There are also journals but they tend to be rather behind.

Is collaboration allowed? Yes, even encouraged—group projects are much more fun, and you will be able to get more done (more will be expected of groups). I recommend limiting group size to at most 3.