COS 423 Optional Final Project Proposal Due Mon. 4/13/09
Spring 2009 Project Due Tues. 5/12/09

No Collaboration

If you are seeking extra credit in this course, or you wish to make up for low scores on problem sets, you may do an optional project on a subject of your choosing. Topics will be by permission of the instructor. Please submit a one-page proposal by April 15 for me to approve. Your final submission should include a write-up of at most 10 pages (single-spaced, at least 11pt font) describing the problem, background information, your results and findings, and any conclusions you can draw. It should be written in the style of a conference research paper. Your project will count for up to 1/3 of your grade.

General possibilities for a project are:

(1) An experimental study of one or more data structures or algorithms we have studied in class, or one or more that we have not studied but that you find interesting. Especially interesting would be a comparison between two or more alternatives, such as:

Rank-pairing heaps vs. pairing heaps (vs. Fibonacci heaps vs. implicit heaps)
Rank-balanced trees vs. red-black trees (vs. AVL trees)

(2) An interactive demo of a data structure or algorithm that illustrates its workings.

(3) A survey paper on a topic related to some part of the course material. Your paper should cover at least three research papers.

(4) A research project of your own choosing. If you need help choosing a research topic, you may contact Prof. Tarjan or Sid for ideas.

The following is a research topic suggested by Sid, on which you may do your project if you choose.

Power utilization in tree-based wireless sensor networks. In a wireless sensor network consisting of a single base station and n sensors, the sensors may be arranged as a rooted tree to reduce the communication latency between the base and sensors. An appropriately balanced tree guarantees that there are $O(\log n)$ hops between any sensor and the base. However, it does not distribute the communication load evenly because sensors that are close to the root are used more often than sensors at higher depth. For example, the root of the tree is involved in every communication to or from a sensor. This results in non-uniform power utilization, causing some sensors to run out of battery power sooner than other sensors. Since battery replacement is often a manual process, all sensor batteries are replaced as soon as one of them dies, resulting in wasted power in many of the batteries.

One way to balance the communication load in the tree of sensors is to make the tree self-adjusting, similar to splay trees. In other words, we can view the sensors as nodes of a self-adjusting search tree that uses some heuristic to reorganize the nodes after an insertion, deletion, or access. For this application you can assume the sequence of operations consists of *n* insertions followed by one or more accesses (there are no deletions). Your task is to devise a self-adjusting heuristic that distributes the communication load evenly, where the load of a node is defined as the number of times an access sequence passes through that node, and "evenly" is left to you to define. For example, you may wish to minimize the variance of the load values of the *n* nodes. Devising a heuristic with provable theoretical properties may be a difficult task. We recommend that you implement a generic self-adjusting search tree using your favorite programming language and experiment with different heuristics first. For binary search trees, you can evaluate the splay heuristic you learned about in class: How well does it distribute load on a random access sequence? What is an example of an access sequence for which it performs poorly? How can you do better than regular splaying?

You are free to redefine the above problem, if you wish. Make sure to describe your formulation of the problem in your project proposal.