UNIVERSITY OF CAMBRIDGE

Computer Laboratory

Algorithms II — Exercises for students

Academic year 2012–2013

Michaelmas term 2012

`http://www.cl.cam.ac.uk/Teaching/1213/AlgorithI/`

`frank.stajano--algs1@cl.cam.ac.uk`

(course-related emails sent to *this* address will be given higher priority)

Revised 2012 edition

Revision 13 of 2013-02-03 20:27:30 +0000 (Sun, 03 Feb 2013).

Exercises from the course handout

Your supervisor may also point you at suitable sections of past exam questions. Past exam questions are available from the course web site.

Exercise 1
As a comparison, what is the most efficient algorithm you can think of to merge two binary heaps? What is its complexity?

Exercise 2
Draw a binomial tree of order 4.

Exercise 3
Give proofs of each of the stated properties of binomial trees (trivial) and heaps (harder until you read the next paragraph—try before doing so).

Exercise 4
Explain with an example why, if the "peculiar constraint" were not enforced, it would be possible for a node with n descendants to have more than $O(\lg n)$ children.

Exercise 5
Assume that `insert()`, `delete()`, `min()`, `max()`, `pred()`, `succ()` all have complexity $O(f(n))$. Define `extractMin()`, `first()` and `decreaseKey()` in terms of the previous primitives, without exceeding $O(f(n))$ complexity.

Exercise 6
How much do insertion and deletion cost for the two-level tree? Why?

Exercise 7
Sketch a picture of a proto-vEB tree of size $u = 16$ representing the set 2, 3, 4, 5, 7, 14, 15.

Exercise 8
Prove that the recurrence

$$T(u) = 2T(\sqrt{u}) + O(1)$$

has the solution

$$T(u) = O(\lg u).$$

Exercise 9
Prove that the recurrence

$$T(u) = 2T(\sqrt{u}) + O(\lg \sqrt{u})$$

has the solution

$$T(u) = O(\lg u \lg \lg u).$$

Exercise 10
Deceptively difficult. Do not skip.
Sketch a picture of a vEB tree of size $u = 16$ representing the set 2, 3, 4, 5, 7, 14, 15. OK to study the textbook and handout first, but keep them closed while doing this exercise. No matter how good you are, you will almost certainly get some details wrong. That's OK. Check the textbook *after* having drawn your solution and mark in red all the items that are different, understanding why. Then *the next day* do the exercise again, with textbook closed. Iterate until you get no errors. Will take several days (no shame in that).
You may think you understand vEB trees but you actually don't until you successfully complete this exercise.

Exercise 11
If we are so obsessed with keeping down the height of all these trees, why don't we just maintain all trees at height ≤ 1 all along?

Exercise 12
Build a "7-bridge" graph with this property. Then look up Euler and Königsberg and check whether your graph is or isn't isomorphic to the historical one. (Hint: you don't *have* to reconstruct the historical layout but note for your information that the river Pregel, which traversed the city of Königsberg, included two islands, which were connected by some of the 7 bridges.) Finally, build the *smallest* graph you can find that has this property.

Exercise 13
Draw in the margin an example of each of the following:

1. An anti-reflexive directed graph with 5 vertices and 7 edges.
2. A reflexive directed graph with 5 vertices and 12 edges.
3. A DAG with 8 vertices and 10 edges.
4. An undirected tree with 8 vertices and 10 edges.
5. A tree that is *not* "an undirected graph without cycles".
6. A graph without cycles that is *not* a tree.

Actually, I cheated. Some of these can't actually exist. Which ones? Why?

Exercise 14
Draw a random DAG in the margin, with 9 vertices and 9 edges, and then perform a depth-first search on it, marking each vertex with two numbers as you proceed—the discovery time (the time the vertex went grey), then a slash and the finishing time (the time it went black). Then draw the linearized DAG by arranging the vertices on a line in reverse order of their finishing time and reproducing the appropriate arrows between them. Do the arrows all go forward?

Exercise 15
Develop a proof of the correctness of the topological sort algorithm. *(Requires some thought.)*

Exercise 16
Find, by hand, a minimum spanning tree for the graph drawn in CLRS3 figure 23.4.(a) (trying not to look at nearby figures). Then see if you can find any others.

Exercise 17
Starting with fresh copies of the graph you used earlier, run these two algorithms on it by hand and see what you get. Note how, even when you reach the same end result, you may get to it via wildly different intermediate stages.

Exercise 18
Give a formal proof of the intuitive "triangle inequality"

$$v.\delta \leq u.\delta + w(u, v)$$

(where $w(u, v)$ is the weight of the edge from vertex u to vertex v) but covering also the case in which there is actually no path between s and v.

Exercise 19
Write out explicitly the elements of matrix $L^{(0)}$.

Exercise 20
To what matrix would $L^{(0)}$ map? What is the role of that matrix in the corresponding algebraic structure?

Exercise 21
Draw suitable pictures to illustrate and explain the previous comments. *Very easy; but failure to do this, or merely seeing it done by someone else, will make it unnecessarily hard to understand what follows.*

Exercise 22
Prove that T_1/T_∞ is the average amount of work that can be performed in parallel at each step of the critical path.

Exercise 23
Prove that, if $P > T_1/T_\infty$, the computation cannot achieve perfect linear speedup.

Exercise 24
Construct a minimal case of determinacy race with two threads accessing variable x but only one of them writing to it. Show at least two ways of sequencing the machine code instructions that will cause different results.

Exercise 25
If you can solve this one without help, you are pretty good.
As written, several listings in chapter 27 of CLRS3 (third printing) contain an unintended determinacy race. Which listings? Where is the race? How could such a bug occur?
(When I noticed this, I wrote to Professor Leiserson, who wrote the chapter, and he confirmed I had found a severe bug, which will be fixed in a future printing.)

Exercise 26

- Draw a 3D picture of $\vec{p_1}, \vec{p_2}$ and $\vec{p_3}$.
- Draw a 2D picture of $\vec{p_1}, \vec{p_2}$ and the parallelogram.
- Prove that the absolute value of the determinant of the matrix $\begin{pmatrix} x_1 & x_2 \\ y_1 & y_2 \end{pmatrix}$, equal to $x_1y_2 - x_2y_1$, gives the magnitude of $\vec{p_3}$ and that its sign says whether $\vec{p_3}$ "comes out" of the plane ($\odot$) or "goes into" it ($\otimes$).

Exercise 27
Sketch an algorithm (not the best possible: just any vaguely reasonable algorithm) to compute the convex hull; then do a rough complexity analysis. Stop reading until you've done that. *(Requires thought.)*

Exercise 28
How can you sort a bunch of points by their polar coordinates using efficient computations that don't include divisions or trigonometry? (Hint: use cross products.) Don't read beyond this box until you've found a way.

Exercise 29
Imagine a complex CAD situation in which the set contains about a million points. Which of the two algorithms we have seen would you use to compute the convex hull if you expected it to have about 1,000 vertices? And what is the rough boundary value (for the expected number of vertices of the hull) above which you would use one algorithm instead of the other? (Is this a trick question? If so, why?)