Problems for Recitation 14

1 TriMergeSort

We noted in lecture that reducing the size of subproblems is much more important to the speed of an algorithm than reducing the number of additional steps per call. Let's see if we can improve the $\Theta(n \log n)$ bound on `MergeSort` from lecture.

Let's consider a new version of `MergeSort` called `TriMergeSort`, where the size n list is now broken into *three* sublists of size $n/3$, which are sorted recursively and then merged. Since we know that floors and ceilings do not affect the asymptotic solution to a recurrence, let's assume that n is a power of 3.

1. How many comparisons are needed to merge three lists of 1 item each?

2. In the worst case, how many comparisons are needed to merge three lists of $n/3$ items, where n is a power of 3?

3. Define a divide-and-conquer recurrence for this algorithm. Let $T(n)$ be the number of comparisons to sort a list of n items.

4. We could analyze the running time of this using plug-and-chug, but let's try Akra-Bazzi. First, what is p?

5. Does the condition $|g'(x)| = O(x^c)$ hold for some $c \in N$?

6. Determine the theta bound on T(n) by integration.

7. Turns out that any equal partition of the list into a constant number of sublists $c > 1$ will yield the same theta bound. Can you see why?

Appendix

Theorem 1 (Akra-Bazzi, strong form)**.** *Suppose that:*

$$T(x) = \begin{cases} \text{is defined} & \text{for } 0 \leq x \leq x_0 \\ \displaystyle\sum_{i=1}^{k} a_i T(b_i x + h_i(x)) + g(x) & \text{for } x > x_0 \end{cases}$$

where:

- $a_1, \ldots, a_k$ *are positive constants*
- $b_1, \ldots, b_k$ *are constants between 0 and 1*
- x_0 *is "large enough" in a technical sense we leave unspecified*
- $|g'(x)| = O(x^c)$ *for some* $c \in \mathbb{N}$
- $|h_i(x)| = O(x / \log^2 x)$

Then:

$$T(x) = \Theta\left(x^p \left(1 + \int_1^x \frac{g(u)}{u^{p+1}}\, du\right)\right)$$

where p *satisfies the equation* $\sum_{i=1}^{k} a_i b_i^p = 1$.

Linear Recurrences

Find closed-form solutions to the following linear recurrences.

1. $$T_0 = 0$$
$$T_1 = 1$$
$$T_n = T_{n-1} + T_{n-2} + 1$$

2. $$S_0 = 0$$
$$S_1 = 1$$
$$S_n = 6S_{n-1} - 9S_{n-2}$$

6.042J / 18.062J Mathematics for Computer Science
Fall 2010