COMP 363 - Algorithms: Homework #NUM

Due on DUE DATE

Leo Irakliotis

Nathan Hogg

Problem 1

Background

The goal of this assignment is to prove the time complexity of the mergesort algorithm. We are given the base complexity formula as:

$$T(n) = 2T(n/2) + f(n) \tag{1}$$

In this case, $T(n)$ is the time it takes to mergesort an array of size n, and $f(n)$ is the time it takes to assemble two partial solutions of size $n/2$ to a sorted array with n elements. Generally, $f(n) \approx n$.
For an array with $n = 8$, we can write:

$$T(8) = 2T(4) + f(8) \tag{2}$$

Then, $T(4) = 2T(2) + f(4)$; and $T(2) = 2T(1) + f(2)$; and finally $T(1) = f(1)$. Using the last finding, we can solve backwards:

$$\begin{aligned} T(2) &= 2T(1) + f(2) && (3)\\ &= 2f(1) + f(2) && (4) \end{aligned}$$

$$\begin{aligned} T(4) &= 2T(2) + f(4) && (5)\\ &= 2(2f(1) + f(2)) + f(4) && (6)\\ &= 4f(1) + 2f(2) + f(4) && (7) \end{aligned}$$

$$\begin{aligned} T(8) &= 2T(4) + f(8) && (8)\\ &= 2(4f(1) + 2f(2) + f(4)) + f(8) && (9)\\ &= 8f(1) + 4f(2) + 2f(4) + f(8) && (10) \end{aligned}$$

Given that $f(n) \equiv n$, the last equation can be rewritten as:

$$\begin{aligned} T(8) &\equiv 8 \times 1 + 4 \times 2 + 2 \times 4 + 8 && (11)\\ &= 32 = 8 \times 3 && (12)\\ &= 8(1 + log_2 8) && (13) \end{aligned}$$

We have shown that, for mergesort, $T(n) \equiv nlog_2 n$ (or, $\mathcal{O}(nlog_2 n)$,

Proof

We begin by writing:

$$\begin{aligned} T(n) &= 2T\Big(\frac{n}{2}\Big) + f(n)\\ &= 2\left[2T\Big(\frac{n}{4}\Big) + f\Big(\frac{n}{2}\Big)\right] + f(n)\\ &= 2\left[2\left[2T\Big(\frac{n}{8}\Big) + f\Big(\frac{n}{4}\Big)\right] + f\Big(\frac{n}{2}\Big)\right] + f(n)\\ &= 2\left[2\left[2\left[2T\Big(\frac{n}{16}\Big) + f\Big(\frac{n}{8}\Big)\right] + f\Big(\frac{n}{4}\Big)\right] + f\Big(\frac{n}{2}\Big)\right] + f(n) \end{aligned}$$