CS 5413 Assignment

Problems 4-3 in the text, page 120

Problems 4-4.b, f, g, h, I, j

4-2 Parameter-passing costs

Throughout this book, we assume that parameter passing during procedure calls takes constant time, even if an N-element array is being passed. This assumption is valid in most systems because a pointer to the array is passed, not the array itself. This problem examines the implications of three parameter-passing strategies:

1. Arrays are passed by pointer. Time $= \Theta(1)$.
2. Arrays are passed by copying. Time $= \Theta(N)$, where N is the size of the array.
3. Arrays are passed by copying only the subrange that might be accessed by the called procedure. Time $= \Theta(n)$ if the subarray contains n elements.

Consider the following three algorithms:

a. The recursive binary-search algorithm for finding a number in a sorted array (see Exercise 2.3-6).

b. The MERGE-SORT procedure from Section 2.3.1.

c. The MATRIX-MULTIPLY-RECURSIVE procedure from Section 4.1.

Give nine recurrences $T_{a1}(N,n), T_{a2}(N,n), \ldots, T_{c3}(N,n)$ for the worst-case running times of each of the three algorithms above when arrays and matrices are passed using each of the three parameter-passing strategies above. Solve your recurrences, giving tight asymptotic bounds.

4-3 Solving recurrences with a change of variables

Sometimes, a little algebraic manipulation can make an unknown recurrence similar to one you have seen before. Let's solve the recurrence

$$T(n) = 2T\left(\sqrt{n}\right) + \Theta(\lg n) \tag{4.25}$$

by using the change-of-variables method.

a. Define $m = \lg n$ and $S(m) = T(2^m)$. Rewrite recurrence (4.25) in terms of m and $S(m)$.

b. Solve your recurrence for $S(m)$.

c. Use your solution for $S(m)$ to conclude that $T(n) = \Theta(\lg n \lg\lg n)$.

d. Sketch the recursion tree for recurrence (4.25), and use it to explain intuitively why the solution is $T(n) = \Theta(\lg n \lg\lg n)$.

Solve the following recurrences by changing variables:

e. $T(n) = 2T(\sqrt{n}) + \Theta(1)$.

f. $T(n) = 3T(\sqrt[3]{n}) + \Theta(n)$.

4-4 More recurrence examples

Give asymptotically tight upper and lower bounds for $T(n)$ in each of the following recurrences. Justify your answers.

a. $T(n) = 5T(n/3) + n \lg n$.

b. $T(n) = 3T(n/3) + n/\lg n$.

c. $T(n) = 8T(n/2) + n^3\sqrt{n}$.

d. $T(n) = 2T(n/2 - 2) + n/2$.

e. $T(n) = 2T(n/2) + n/\lg n$.

f. $T(n) = T(n/2) + T(n/4) + T(n/8) + n$.

g. $T(n) = T(n-1) + 1/n$.

h. $T(n) = T(n-1) + \lg n$.

i. $T(n) = T(n-2) + 1/\lg n$.

j. $T(n) = \sqrt{n}\,T(\sqrt{n}) + n$.

4-5 Fibonacci numbers

This problem develops properties of the Fibonacci numbers, which are defined by recurrence (3.31) on page 69. We'll explore the technique of generating functions to solve the Fibonacci recurrence. Define the ***generating function*** (or ***formal power series***) $\mathcal{F}$ as

$$\begin{aligned}\mathcal{F}(z) &= \sum_{i=0}^{\infty} F_i z^i \\ &= 0 + z + z^2 + 2z^3 + 3z^4 + 5z^5 + 8z^6 + 13z^7 + 21z^8 + \cdots ,\end{aligned}$$

where F_i is the ith Fibonacci number.

a. Show that $\mathcal{F}(z) = z + z\mathcal{F}(z) + z^2\mathcal{F}(z)$.