CENG 567 Design and Analysis of Algorithms Fall 2022

Homework 3

Released: Nov 9, 2022 Due:Nov 20, 2022

Instructions. You may work with other students, but you must individually write your solutions **in your own words**. If you work with other students or consult outside sources (such as Internet/book), cite your sources.

If you are asked to design an algorithm, provide:
(a) a description of the algorithm in English and, if helpful pseudocode,
(b) at least one worked example or diagram to show more precisely how your algorithm works,
(c) a proof (or indication) of the correctness of the algorithm,
(d) an analysis of the implementation of the algorithm.

Submissions. Submit a pdf file through odtuclass. LaTeX or Word typed submission is required.

1. Three-way sort. Consider the following pseudocode for a recursive sorting algorithm:

```
// Sorts the items from $L[i]$ through $L[j]$
procedure THREEWAYSORT( int[]$L$, int $i$, int $j$)
    if ($L[i] > L[j]$) then swap (i, j);
    end if
    if (($j - i + 1) > 2$) then
        $t = (j - i + 1)/3$;
        THREEWAYSORT( $L$, $i$, $j - t$);
        THREEWAYSORT($L, i + t, j$);
        THREEWAYSORT($L, i, j - t$);
    end if
end procedure
```

(a) Prove that this algorithm is correct, that is, the call THREEWAYSORT$(L, 0, L.length - 1)$ actually makes L sorted.

(b)Let $f(n)$ be the running time of THREEWAYSORT(L, i, j) when $j - i + 1 = n$. Write a recurrence for $f(n)$.

(c) Solve your recurrence to determine the worst-case running time of THREEWAYSORT on lists of size n. How does it compare to other sorting algorithms you know?

2. Recurrences. Give asymptotic upper and lower bounds in the following recurrences. Assume that $T(n)$ is bounded by a constant for $n \leq 3$. Make your bounds as tight as possible and justify your answers.
(a) $T(n) = 2T(n/3) + 2T(n/9) + n$

(b) $T(n) = \frac{1}{3}(T(n-1) + T(n-2) + T(n-3)) + cn$, with $c > 0$.

(c) $T(n) = T(n/2) + T(n/3) + T(n/6) + n$

(d) $T(n) = \sqrt{2n}T(\sqrt{2n}) + \sqrt{n}$ (Assume $T(n) \geq \sqrt{n}$)

3. Decimal to Binary. Recall that in class we discussed an algorithm that takes two n bit numbers and returns their product, in binary, in time $O(n^a)$ with $a = \log_2 3$ Assume the function `fastmultiply(x,y)` implements this algorithm. We'll use fast binary multiplication to convert numbers from decimal to binary. As representation, we will represent decimal numbers as strings and binary numbers using bits as usual. Given this representation, you can index decimal numbers, but are unable to multiply two decimal numbers together, without first converting to binary.
(a) We will first design an algorithm to convert the decimal number 10^n to binary. Assume that n is a power of 2.

```
procedure PWR2BIN(n)
    if n = 1 then return 1010 (decimal 10 in binary)
    else
        z = /* FILL ME IN */
        Return fastmultiply(z,z).
    end if
end procedure
```

What is the appropriate value for z? What is the running time of the algorithm?

(b) The next procedure is supposed to convert a decimal integer x with n digits into binary. Assume that n is a power of 2.

```
procedure DEC2BIN(x)
    if length(x) = 1 then return binary(x)
    else
        Split $x$ into $x_L$ the leading $n/2$ digits and $x_R$, the trailing $n/2$ digits.
        Return /* FILL ME IN */
    end if
end procedure
```

The function `binary(x)` performs a lookup into a table containing the binary value of all decimal numbers $0 \ldots 9$. What are we supposed to return? What is the running time of this algorithm?

4. Weighted path. You are given a rooted binary tree (each node has at most two children). For a simple path p between two nodes in the tree, let m_p be the node on the path that is highest (closest to the root). Define the weight of a path $w(p) = \Sigma_{u \in p} d(u, m_p)$, where d denotes the distance (number of edges on the path between two nodes). That is, every node on the path is weighted by the distance to the highest node on the path.

Design a divide and conquer algorithm that finds the maximum weight among all simple paths in the tree.