Exercises

4.1 Does binary search use the divide-and-conquer strategy?

4.2 Multiplying two n bit numbers u and v straightforwardly requires $O(n^2)$ steps. By using the divide-and-conquer approach, we can split the number into two equal parts and compute the product by the following method:

$$uv = (a \cdot 2^{n/2} + b) \cdot (c \cdot 2^{n/2} + d) = ac \cdot 2^n + (ad + bc) \cdot 2^{n/2} + bd.$$

If $ad + bc$ is computed as $(a + b)(c + d) - ac - bd$, what is the computing time?

4.3 Prove that in quick sort, the maximum stack needed is $O(\log n)$.

4.4 Implement the Fast Fourier Transform algorithm based upon the divide-and-conquer approach. Compare it with the straightforward approach.

4.5 Implement the rank finding algorithm based upon the divide-and-conquer approach. Compare it with the straightforward approach.

4.6 Let $T\left(\frac{n^2}{2^r}\right) = nT(n) + bn^2$, where r is an integer and $r \geq 1$. Find $T(n)$.

4.7 Read Section 3–7 of Horowitz and Sahni (1978) for Strassen's matrix multiplication method based upon divide-and-conquer.

4.8 Let

$$T(n) = \begin{cases} b & \text{for } n = 1 \\ aT(n/c) + bn & \text{for } n > 1. \end{cases}$$

where a, b and c are non-negative constants.
Prove that if n is a power of c then

$$T(n) = \begin{cases} O(n) & \text{if } a < c \\ O(n \log_a n) & \text{if } a = c \\ O(n^{\log_c a}) & \text{if } a > c. \end{cases}$$

4.9 Prove that if $T(n) = mT(n/2) + an^2$, then $T(n)$ is satisfied by

$$T(n) = \begin{cases} O(n^{\log m}) & \text{if } m > 4 \\ O(n^2 \log n) & \text{if } m = 4 \\ O(n^2) & \text{if } m < 4. \end{cases}$$

4.10 A very special kind of sorting algorithm, also based upon divide-and-conquer, is the odd-even merge sorting, invented by Batcher (1968). Read Section 7.4 of Liu (1977) for this sorting algorithm. Is this sorting algorithm suitable as a sequential algorithm? Why? (This is a famous parallel sorting algorithm.)

4.11 Design an $O(n \log n)$ time algorithm to find the longest monotonically increasing subsequence of a sequence of n numbers.