Introduction to Algorithms - Reading Notes & Selected Solutions

Nimrod Shneor

October 3, 2019

Chapter 1

Solutions to selected exercises:

2.1-1

$$
\begin{aligned}
A &= [31, 41, 59, 26, 41] \\
A &= [31, 41, 59, 26, 41] \\
A &= [31, 41, 26, 59, 41] \\
A &= [31, 26, 41, 59, 41] \\
A &= [26, 31, 41, 59, 41] \\
A &= [26, 31, 41, 41, 59]
\end{aligned}
$$

21.1-2

Algorithm 1 Non-Increasing-Insertion-Sort(A)

```
for i ← 2 to length(A) do
    for j ← i to 2 do
        if A[j] < A[j-1] then
            k ← A[j-1]
            A[j-1] ← A[j]
            A[j] ← k
        end if
    end for
end for
```

2.1-4

- Input: two n-bit numbers A, B.
- Output: the sum of A, B - an $n + 1$-bit number.

Algorithm 2 BinaryAddition(A,B,n)

```
carry ← 0
for i ← 1 to n do
    C[i] ← A[i]+B[i]+carry (mod 2)
    carry ← A[i] * B[i]
end for
C[n+1] ← carry
return C
```

2.2-3 Define $X =$ The number of elements checked in a “brute force” linear search.

Than $X \in \{1...n\}$ and the average number of elements checked in a linear search is exactly:

$$E[X] = \sum_{i=1}^{n} \frac{1}{n} i = \frac{1}{n} \sum_{i=1}^{n} i = \frac{n(n-1)}{2n} = \Theta(n)$$

The worst case is where the last element of the array is the one searched for - resulting in an $\Theta(n)$ run time.

2.3-3

$$T(n) = \begin{cases} 2 & n = 2 \\ 2T(\frac{n}{2}) + n & n > 2 \end{cases}$$

Q: Proof by induction that if n is an exact power of two (that is $n = 2^k$ for some constant $k \geq 1$) than $T(n) = nlogn$.

Proof: By induction.

Base case: for $k = 1$ than $T(n) = 2 = 2log2 = nlogn$

Assumption: Assume the above holds for all integers up to $k > 1$.

Induction step: We now prove the statement for $n = 2^{k+1}$.

Plugging in to the formula

$$\begin{aligned} T(n) &= 2T(\frac{n}{2}) + n = 2T(\frac{2^{k+1}}{2}) + 2^{k+1} \\ &= 2T(2^k) + 2^{k+1} = 2 * 2^k log2^k + 2^{k+1} \\ &= k2^{k+1} + 2^{k+1} = (k+1)2^{k+1} = 2^{k+1} log2^{k+1} \\ &= nlogn \end{aligned}$$

$\square$