Introduction to Algorithms - Reading Notes & Selected Solutions

Nimrod Shneor

October 4, 2019

Chapter 2

Solutions to selected exercises:

2.1-1

$$A = [31, 41, 59, 26, 41]$$
$$A = [31, 41, 59, 26, 41]$$
$$A = [31, 41, 26, 59, 41]$$
$$A = [31, 26, 41, 59, 41]$$
$$A = [26, 31, 41, 59, 41]$$
$$A = [26, 31, 41, 41, 59]$$

2.1-4

Algorithm 1 BinaryAddition(A,B,n)

```
carry ← 0
for i ← 1 to n: do
    C[i] ← (A[i] + B[i] + C[i]) mod2
    carry ← A[i] * B[i]
end for
C[i + 1] ← carry
return C
```

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

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

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

□

Algorithm 2 BinarySearch(A,x)

2.3-5

```
l ← 0
r ← length(A)
while l < r − 1 do
  if x = A[(l+r)/2] then
    (l+r)/2
  end if
  if x > A[(l+r)/2] then
    l = A[(l+r)/2]
  else
    r = A[(l+r)/2]
  end if
end while
return -1
```

At each iteration of the while loop the distance between the two pointers - l, r - is halfed until the element is found or we return -1. The while loop will terminate once the two pointers are at distance two at which point either the element x is found or the loop will terminate. Thus the distance between the two pointers at each iteration i is percisly $\frac{length(A)}{2^i} = \frac{n}{2^i}$

At the time of the termination the distance between the two pointers is two, thus -

$$\begin{aligned}\frac{n}{2^i} &= 2\\ n &= 2^{i+1}\\ log(n) &= i + 1\\ log(n) - 1 &= i\\ \Theta(log(n)) &= i\end{aligned}$$