UMC201: Data Structures and Algorithms

Naman Mishra

August 2023

Contents

Lecture 01: A Warning

Tue 8 Aug '23

1 The Course

1.1 Instructor

Prof. C. Pandu Rangan.
Office hours: CDS 406, 2 pm onwards, Wednesday.

1.2 Grading

Harsh. Cluster-based absolute grading.

Tentative weightage for the theory component:

- **Assignments:** 30%.
- **Midsem:** 20%.
- **Endsem:** 50%.

1.3 References

(i) *Introduction to Algorithms* by Cormen, Leiserson, Rivest, Stein.

Good for pseudocode. Read fully over multiple semesters.

(ii) *Algorithms* by Jeff Erickson.
Great problem sets.

(iii) *Algorithm Design* by Kleinberg and Tardos.
Chatty style. Applied problems.

2 Analysis

Problem 2.1 (Binary Search)**.** Given an array A sorted by $\leq$ and a key k, return the index of k in A if it exists, else return -1.

Solution.

```
function BinarySearch(A[0..n − 1], k)
    p ← 0
    q ← n − 1
    while p ≤ q do
        m ← ⌊(p + q)/2⌋
        if x < A[m] then
            q ← m − 1
        else if x > A[m] then
            p ← m + 1
        else
            return m
    return −1
```

Proof of partial correctness. Let $C = p \leq q$ and $I = x \in A[0..n-1] \implies x \in A[p..q]$.

I is trivially true before the loop. C is false after the loop.

Suppose $C \wedge I$ is true at the beginning of the loop for some instance.

- If $x = k$, then p and q are not modified, and so $C \wedge I$ remains true. The program terminates with a correct result.
- If $x < A[m]$, then $x < A[m..n-1]$ since A is sorted. Thus if $x \in A$, then $x \in A[p..q] \setminus A[m..n-1]$ and so $x \in A[p..m-1]$. Thus $q \leftarrow m-1$ preserves I.
- If $x > A[m]$, then $x > A[0..m]$ since A is sorted. Thus if $x \in A$, then $x \in A[p..q] \setminus A[0..m]$ and so $x \in A[m+1..q]$. Thus $p \leftarrow m+1$ preserves I. □

Proof of termination. Consider two cases:

($x \in A$) By I, $x \in A[p..q]$ is always true. Thus $C \equiv p \leq q$ is always true. Thus $q - p$ is always a non-negative integer.

Let p_j, q_j be the values of p, q at the beginning of the j-th iteration. For the $(j+1)$-th iteration, we have that if $x = A[m]$, the procedure terminates. Otherwise,

$$\begin{aligned} q_{j+1} - p_{j+1} &= \begin{cases} \left\lfloor \frac{p_j+q_j}{2} \right\rfloor - 1 - p_j & x < A[m] \\ q_j - \left\lfloor \frac{p_j+q_j}{2} \right\rfloor - 1 & x > A[m] \end{cases} \\ &\leq q_j - p_j - 1 \\ &< q_j - p_j. \end{aligned}$$

Thus the procedure must terminate.

($x \notin A$) □

Alternative proof of termination. Let $T = q - p + 1$. T is a non-negative integer that decreases in each iteration of the loop. Thus the loop cannot run indefinitely. By C we have that $p \leq q$ at the beginning of each iteration.

$$\begin{aligned} T_j &= q_j - p_j + 1 \\ T_{j+1} &= q_{j+1} - p_{j+1} + 1 \\ &= \begin{cases} \left\lfloor \frac{p_j+q_j}{2} \right\rfloor - p_j & x < A[m] \\ q_j - \left\lfloor \frac{p_j+q_j}{2} \right\rfloor & x > A[m] \end{cases} \\ &\leq q_j - p_j \\ &< T_j. \end{aligned}$$

□

Complexity: We have that for the worst case, $T(1) = 1$ and $T(n) = 1 + T(\lfloor n/2 \rfloor)$ for $n > 1$.

Claim: $T(n) = \log_2(n) + 1$.

Proof. $P(n)$ be that $T(m) = \log_2(m) + 1$ for all $m \leq n$. Let $Q(n) = P(2^n)$.

$Q(0)$ is trivially true. // Induct. □

Solution (Alternative).

```
function BinarySearch(A[0..n − 1], k)
    p ← 0
    q ← n − 1
    while p < q do
        m ← ⌊(p + q)/2⌋
        if x ≤ A[m] then
            q ← m
        else
            p ← m + 1
    if x = A[p] then
        return p
    else
        return −1
```

Problem 2.2 (Buggy Binary)**.** Prove the incorrectness of the following algorithm for binary search.

```
1:  function BinarySearch(A[0..n − 1], k)
2:      p ← 0
3:      q ← n − 1
4:      while p < q do
5:          m ← ⌈(p + q)/2⌉
6:          if x ≤ A[m] then
7:              q ← m
8:          else
9:              p ← m + 1
10:     if x = A[p] then
11:         return p
12:     else
13:         return −1
```

Fix the algorithm without modifying line 5.