# DSA. Problem solutions. Week 4.

Maksim Al Dandan

April 21, 2024

## 1 Task 1

### 1.1 Statement

Compute asymptotic worst case time complexity of the solve procedure:

(i) Express the running time of the solve procedure $T(n)$ as a recurrence relation.

(ii) Find the asymptotic complexity of $T(n)$ using the master theorem.

(iii) Specify which case of the master theorem applies (if any).

(iv) Write down conditions for this case that need to be checked (write down specialized version for a particular recurrence).

(v) Prove that the conditions are satisfied. For asymptotic notation, each property used in a proof must be either proven explicitly or properly referenced (e.g. by citing [Cormen] or a particular Lecture/Tutorial slide).

```
1   /* A is a 0-indexed array
2    * n is the number of elements in A */
3   solve(A, n):
4     helper(A, 0 , n - 1)
5
6   helper(A, l, r):
7     if l > r:
8       return
9     else:
10      k := ⌈(r − l + 1)/4⌉
11      for j from 0 to 3 /* inclusive */
12        helper(A, l + j * k, min(r, l + (j + 1) * k))
13      for a from l to r
14        m := i
15        for b from l to r
16          if A[b] ≥ A[m]:
17            m = b
18        exchange A[m] with A[a]
```

### 1.2 Solution

Running time of the "solve" procedure: $T(n) = 4T(\frac{n}{4}) + n^2$

We divide array by four parts and call the helper function itself only once within the for loop for exactly 4 times. That's way $a = b = 4$.

$f(n) = n^2$ is because the code from the 13th line till the end takes asymptotic worst case time complexity $O(n^2)$.

Then let's try to apply the Master theorem.

$a = 4, b = 4, f(n) = n^2, n^{\log_b a} = n^{\log_4 4} = n$

**[Case 1]** $\implies f(n) = n^2 = O(n^{1-\epsilon})$, $\epsilon > 0 :$ $\epsilon$ doesn't exist

**[Case 2]** $\implies f(n) = n^2 = \Theta(n \times \log^k n)$, $k \geq 0 :$ $k$ doesn't exist

**[Case 3]** $\implies f(n) = n^2 = \Omega(n^{1+\epsilon})$, $\epsilon > 0 : \epsilon = 0.5$ (exist), then we nee to check *regularity condition.*

$$4 \times (\frac{n}{4})^2 \leq cn^2 \tag{1}$$

$$c \geq \frac{1}{4}, c < 1 \tag{2}$$

For instance, $c = 0.5$. Hence, *regularity condition* holds and, thus, we can compute $T(n)$.
$T(n) = \Theta(f(n)) = \Theta(n^2)$.

**Answer:** $T(n) = \Theta(n^2)$**.**

# 2 Task 2

## 2.1 Statement

For each of the following recurrences, apply the master theorem [Cormen, Theorem 4.1] yielding a closed form formula for the asymptotic complexity of $T(n)$. Assume that $T(n) = 1$ when $n < 10$ for all recurrences below. You must specify the following:

- Can the theorem be applied to the recurrence?
- If yes, then
  - (i) Which case of the master theorem applies?
  - (ii) Which conditions for this case that need to be checked (write down specialized version for a particular recurrence).
  - (iii) Prove that the condition is satisfied. For asymptotic notation, each property used in a proof **must** be either proven explicitly or properly referenced (e.g. by citing [Cormen] or a particular Lecture/Tutorial slide).
  - (iv) Provide asymptotic complexity for $T(n)$ using $\Theta$-notation.
- Otherwise, provide an explicit justification, explaining why the theorem cannot be applied.

(a) $T(n) = 2T(n/3) + \log_2 n$

(b) $T(n) = 7T(n/49) + \sqrt{n} \cdot \log_2^2 n$

(c) $T(n) = 4T(n/3) + n^2$

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

(e) $T(n) = \frac{1}{2}T(2n) + n \log_2 n$

## 2.2 Solution

### 2.2.1 Point (a)

$a = 2, b = 3, f(n) = \log_2 n, n^{\log_b a} = n^{\log_3 2} = n \approx n^{0.631}$
**[Case 1]** $\implies f(n) = \log_2 n = O(n^{0.631-\epsilon})$, $\epsilon > 0 : \epsilon = 0.1$ (exist), then we can compute $T(n)$.

$T(n) = \Theta(n^{\log_3 2})$

### 2.2.2 Point (b)

$a = 7, b = 49, f(n) = \sqrt{n} \times \log_2^2 n, n^{\log_b a} = n^{\log_{49} 7} = \sqrt{n}$
**[Case 1]** $\implies f(n) = \sqrt{n} \times \log_2^2 n = O(n^{\frac{1}{2}-\epsilon})$, $\epsilon > 0 : \epsilon$ doesn't exist
**[Case 2]** $\implies f(n) = \sqrt{n} \times \log_2^2 n = \Theta(\sqrt{n} \times \log^k n)$, $k \geq 0 : k = 2$ (exist), then we can compute $T(n)$.

$T(n) = \Theta(\sqrt{n} \times \log_2^3 n)$

### 2.2.3 Point (c)

$a = 4, b = 3, f(n) = n^2, n^{\log_b a} = n^{\log_3 4} \approx n^{1.262}$
**[Case 1]** $\implies f(n) = n^2 = O(n^{1.262-\epsilon})$, $\epsilon > 0 :$ $\epsilon$ doesn't exist
**[Case 2]** $\implies f(n) = n^2 = \Theta(n \times \log^k n)$, $k \geq 0 :$ $k$ doesn't exist
**[Case 3]** $\implies f(n) = n^2 = \Omega(n^{1.262+\epsilon})$, $\epsilon > 0 :$ $\epsilon = 0.1$ (exist), then we nee to check *regularity condition*.

$$4 \times (\frac{n}{3})^2 \leq cn^2 \tag{1}$$

$$c \geq \frac{4}{9}, c < 1 \tag{2}$$

For instance, $c = \frac{5}{9}$. Hence, *regularity condition* holds and, thus, we can compute $T(n)$.

$T(n) = \Theta(f(n)) = \Theta(n^2)$.

### 2.2.4 Point (d)

$a = 2, b = 1, f(n) = n$
Hence, cannot be computed because by the formula of Master theorem $b$ should be constant and $b > 1$.

### 2.2.5 Point (e)

$a = \frac{1}{2}, b = \frac{1}{2}, f(n) = n \log_2 n$
Hence, cannot be computed because by the formula of Master theorem $b > 1$.

## 2.3 Answer

(a) $T(n) = \Theta(n^{\log_3 2})$
(b) $T(n) = \Theta(\sqrt{n} \times \log_2^3 n)$
(c) $T(n) = \Theta(n^2)$
(d) No answer
(e) No answer

# References

[Cormen] T. H. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein. Introduction to Algorithms, Fourth Edition. The MIT Press 2022