Design and Analysis of Algorithms (2022 Spring)
Problem Set 1: Solution

Due Date: Feb. 21st, 2022

1. (a) By main theorem (with $a = 4, b = 2, d = 2$), since $\log_b a = d$, $T(n) = O(n^d \log n) = O(n^2 \log n)$.

   (b)

$$\begin{aligned} T(n) &= T(n-1) + n \\ &= T(n-2) + (n-1) + n \\ &= \cdots \\ &= T(1) + 2 + \cdots + n \\ &= T(1) + \frac{n(n+1)}{2} - 1, \end{aligned}$$

   therefore $T(n) = O(n^2)$.

   (c) By main theorem (with $a = 3, b = 3, d = 3$), since $\log_b a < d$, $T(n) = O(n^d) = O(n^3)$.

   (d)

$$\begin{aligned} T(n) + 1 &= 2(T(n-1) + 1) \\ &= 4(T(n-2) + 1) \\ &= \cdots \\ &= 2^{n-1}(T(1) + 1), \end{aligned}$$

   therefore $T(n) = O(2^n)$.

   (e) By main theorem (with $a = 4, b = 2, d = 1$), since $\log_b a > d$, $T(n) = O(n^{\log_b a}) = O(n^2)$.

   (f)

$$\begin{aligned} T(n) &= 2T(n/2) + n \log n \\ &= 4T(n/4) + 2 \cdot \frac{n}{2} \log \frac{n}{2} + n \log n \\ &= \cdots \\ &= nT(1) + n \log \frac{n}{2^{\log n - 1}} + \cdots + n \log \frac{n}{2} + n \log n \\ &= O(n) + n \log^2 n - n(1 + 2 + \cdots + (\log n - 1)) \\ &= O(n) + \frac{n \log n(\log n + 1)}{2}, \end{aligned}$$

   therefore $T(n) = O(n \log^2 n)$.

2. Let the sequence be $a_1, a_2, \cdots, a_n$. If $a_k = a_1 + (k-1)d$, the missing number is greater than a_k, otherwise it's less than a_k. We have $T(n) = T(n/2) + O(1)$, by main theorem $O(\log n)$ time complexity.

```
L ← 1, R ← N
while L < R do
    M ← (L+R)/2
    if a_M − a_{M−1} > d return a_M − d
    if a_{M+1} − a_M > d return a_M + d
    if a_M − a_1 = d(M − 1) then
        L ← M
    else
        R ← M
    end if
end while
```

3. (a) With 1/3 probability, the pivot is between 1/3 and 2/3 of the sequence, dividing it into two parts, each with no more than $2n/3$ elements. With 2/3 probability (when the pivot is "bad"), the remaining part takes no more time than $T(n)$.

$$T(n) \leq \frac{1}{3} \cdot 2T(2n/3) + \frac{2}{3} \cdot T(n) + O(n).$$
$$T(n) \leq 2T(2n/3) + O(n).$$

By main theorem, $T(n) \leq n^{\log_{3/2} 2} \leq n^{1.71}$.

   (b) In average case,

$$\begin{aligned} T(n) &= \frac{1}{n}\sum_{i=0}^{n-1}(T(i) + T(n-1-i)) + n - 1 \\ &= \frac{2}{n}\sum_{i=1}^{n-1} T(i) + n - 1. \end{aligned}$$

We have $nT(n) - (n-1)T(n-1) = 2T(n-1) + 2(n-1)$, or equivalently,

$$\frac{T(n)}{n+1} - \frac{T(n-1)}{n} = \frac{2(n-1)}{n(n+1)} \leq \frac{2}{n+1}.$$

Summing up from 1 to n we have

$$\frac{T(n)}{n+1} - \frac{T(1)}{2} \leq \sum_{i=2}^{n+1} \frac{2}{i} \leq 2\int_1^n \frac{1}{x}\,\mathrm{d}x = O(\log n),$$

therefore $T(n) = O(n \log n)$.

4. If we divide the sequence into two parts, there are 3 possible cases for the optimal sub sequence:

   - In the left part.
   - In the right part.
   - Crossing two parts.

The first two cases are subtasks, and the third one can be solved in $O(n)$ time. We have $T(n) = 2T(n/2) + O(n)$, by main theorem $O(n \log n)$ time complexity.

---

function LCIS($a_1, a_2, \cdots, a_n$)
    $(l_1, r_1) \leftarrow$ LCIS($a_1, a_2, \cdots, a_{n/2}$)
    $(l_2, r_2) \leftarrow$ LCIS($a_{n/2+1}, \cdots, a_{n-1}, a_n$)
    if $a_{n/2} \le a_{n/2+1}$ **then**
        $l_3 \leftarrow n/2, r_3 \leftarrow n/2 + 1$
        while $a_{l_3-1} \le a_{l_3}$ **do**
            $l_3 \leftarrow l_3 - 1$
        end while
        while $a_{r_3+1} \ge a_{r_3}$ **do**
            $r_3 \leftarrow r_3 + 1$
        end while
    end if
    return the best of $(l_1, r_1), (l_2, r_2), (l_3, r_3)$
end function
