05-Report

June 19, 2020

1 Sorting: Homework 2

1.1 Exercise 1

- Generalize the `SELECT` algorithm to deal also with repeated values and prove that it still belongs to $\mathcal{O}(n)$.

1.2 Exercise 2

- Download the latest version of the code from [https://github.com/albertocasagrande/AD_sorting] and
  - Implement the `SELECT` algorithm of Ex. 1.
  - Implement a variant of the `QUICK SORT` algorithm using above mentioned `SELECT` to identify the best pivot for partitioning.
  - Draw a curve to represent the relation between the input size and the execution-time of the two variants of `QUICK SORT` (i.e, those of Ex. 2 and Ex. 1 31/3/2020) and discuss about their complexities.

1.3 Exercise 3

- (Ex. 9.3-1) In the algorithm `SELECT`, the input elements are divided into chunks of 5. Will the algorithm work in linear time if they are divided into chunks of 7? What about chunks of 3?

Solution:

We still know that the median of medians is less than at least 4 elements from half of the $\lceil n/7 \rceil$ so the number elements at least greater than m (median) is for chunks $\lceil n/7 \rceil$ is:

$$4\big(\lceil \frac{1}{2} \lceil \frac{n}{7} \rceil \rceil - 2\big) \geq \frac{2n}{7} - 8$$

An upper bound for the # of elements smaller or equal to m is: $n - (\frac{2n}{7} - 8) = \frac{5n}{7} + 8$

So the recurrence becomes:

$$T_s(n) = T(\lceil n/7 \rceil) + T(5n/7 + 8) + \Theta(n)$$

We can solve this by substitution method and show that it is linear. Selecting cn and $c'n$ representatives of $\prime(n)$ and $\Theta(n)$ and assume that $T_s(m) \geq cm$ for all $m < n$.

$$\begin{aligned} T(n) &\leq c\lceil n/7 \rceil + c(5n/7 + 8) + c'n && (1)\\ &\leq c(n/7 + 1) + c(5n/7 + 8) + c'n && (2)\\ &\leq 6/7cn + 9c + c'n && (3) \end{aligned}$$

Hence, $T(n) \leq cn$ for $c \geq 14c'$ when $n \geq 126$ and $T(n) \in \prime(n)$

For groups of 3, the number of elements greater than m, and the number of elements less than m, is at least

$$2\big(\lceil \frac{1}{2}\lceil \frac{n}{3} \rceil \rceil - 2\big) \geq \frac{n}{3} - 4$$

So the recurrence becomes:

$$T_s(n) = T(\lceil n/3 \rceil) + T(2n/3 + 4) + \Theta(n)$$

We can solve this by substitution method and show that it is non-linear. We guess that $T(n) > cn$

$$\begin{aligned} T(n) &\geq c\lceil n/3 \rceil + c(2n/3 + 4) + c'n && (4)\\ &\geq c(n/3) + c(2n/3 + 4) + c'n && (5)\\ &\geq cn + 4c + c'n && (6)\\ &\geq cn + 4c && (7) \end{aligned}$$

therefore we have that it grows more quickly than linear.

1.4 Exercise 4

- (Ex. 9.3-5) Suppose that you have a "black-box" worst-case linear-time subroutine to get the position in A of the value that would be in position $n/2$ if A was sorted. Give a simple, linear-time algorithm that solves the selection problem for an arbitrary position i.

A : Array, p,r are the subarray indices, i is the position.

```
def Select(A, l, r, i)
    if l = r
        return A[l]
    m <- MEDIAN(A,p,r)
    p <- PARTITION(A,m)
    k <- p - l + 1  // length of the first half of the original array
    // i is half the length of the array, return the element coming from the median finding bla
    if i = k
        return A[p]
```

```
// i is less than half the length of the original array, recurse on the first half
elseif i < k
    return SELECT(A,l,p-1,i)
// i is greater than half the length of the original array, recurse on the second half
return SELECT(A,p+1,r,i-k)
```

The recurrence for the worst-case running time is $T(n) \leq T(n/2) + \prime(n) \in \prime(n)$

1.5 Exercise 5

- Solve the following recursive equations by using both the recursion tree and the substitution method:

First two questions were solved during the lectures, so solutions will be provided from lecture notes.

1.5.1 a) $T_1(n) = 2 * T_1(n/2) + \mathcal{O}(n)$

Recursion Tree

In recursion tree, we have 2^i nodes for level i and height of tree is $\log n$ cost for each level can be calculated with $2^i \frac{cn}{2^i}$ so total cost can be represented like following.

$T_1(n) \leq \sum_{i=0}^{\log n} 2^i \frac{cn}{2^i}$ which equals to $cn \log n \in \mathcal{O}(n \log n)$

Substitution Method

We guess that $T_1(n) \in \mathcal{O}(n \log n)$. We select representatives for $\mathcal{O}(n \log n)$ and $\mathcal{O}(n)$ as $cn \log n$ and $c'n$

$$
\begin{aligned}
T_1(n) &= 2 * T_1(n/2) + c'n && (8)\\
&\leq 2\frac{cn}{2} log n/2 + c'n && (9)\\
&\leq cn \log n - cn \log 2 + c'n \text{ (if } cn \log 2 + c'n \text{ is smaller or equal to zero)} && (10)\\
&\leq cn \log n && (11)\\
T_1(n) &\leq cn \log n \in \mathcal{O}(n \log n) && (12)
\end{aligned}
$$

1.5.2 b) $T_2(n) = T_2(\lceil n/2 \rceil) + T_2(\lfloor n/2 \rfloor) + \Theta(1)$

Recursion Tree

In this recursion tree which is not complete because of ceiling and floor operations. Height of left side is smaller than $\log 2n$ and height of right is larger than $\log \frac{n}{2}$ By choosing c as a represantative of $\Theta(1)$ we have the following boundaries for cost.

$$T_2(n) \geq \sum_{i=0}^{\log \frac{n}{2}} c2^i \tag{13}$$
$$\geq c\frac{2^{log\frac{n}{2}+1}-1}{2-1} \tag{14}$$
$$\geq c(2^{\log n - \log 2 + 1} - 1) \tag{15}$$
$$\geq cn - c \in \Omega(n) \tag{16}$$

$$T_2(n) \leq \sum_{i=0}^{\log 2n} c2^i \tag{17}$$
$$\leq c(2^{log2n+1} - 1) \tag{18}$$
$$\leq c(2^{\log n + 2} - 1) \tag{19}$$
$$\leq 4cn - c \in \mathcal{O}(n) \tag{20}$$

So $T_2(n) \in \Theta(n)$

Substitution Method

First we guess that $T_2(n) \in \mathcal{O}(n)$ and we select representatives for $\mathcal{O}(n)$ and $\Theta(1)$ as cn and 1.

By assuming $\forall m < n$ and $T_2(m) \leq cm$ we want to prove that $T_2(n) \leq cn$

$$T_2(n) = T_2(\lceil n/2 \rceil) + T_2(\lfloor n/2 \rfloor) + 1 \tag{21}$$
$$T_2(n) \leq c\lceil n/2 \rceil + c\lfloor n/2 \rfloor + 1 \tag{22}$$
$$\leq cn + 1 \tag{23}$$

which doesn't prove $T_2(n) \leq cn$ because $cn + 1 \not\leq cn$

The problem is that we selected the wrong representative for $\mathcal{O}(n)$ so we choose a new representative for $\mathcal{O}(n)$ as $cn - d$

Again by assuming $\forall m < n$ and $T_2(m) \leq cm - d$ we want to prove that $T_2(n) \leq cn - d$

$$T_2(n) = T_2(\lceil n/2 \rceil) + T_2(\lfloor n/2 \rfloor) + 1 \tag{24}$$
$$T_2(n) \leq c\lceil n/2 \rceil - d + c\lfloor n/2 \rfloor - d + 1 \tag{25}$$
$$\leq cn - 2d + 1 \tag{26}$$
$$\tag{27}$$

If $1 - d \leq 0$ then $T_2(n) \leq cn - d$ and we can say that $T_2(n) \in \mathcal{O}(n)$

Now we guess that $T_2(n) \in \Omega(n)$ and cn, 1 are the representatives for $\Omega(n)$ and $\Theta(1)$.

$$T_2(n) \geq c\lceil n/2 \rceil - d + c\lfloor n/2 \rfloor - d + 1 \tag{28}$$
$$\geq cn + 1 \geq cn \text{ (so we can prove that } T_2 \in \Omega(n)) \tag{29}$$

Hence we can say that $T_2(n) \in \Theta(n)$

1.5.3 c) $T_3(n) = 3 * T_3(n/2) + \mathcal{O}(n)$

Recursion Tree

Here for every level, we have 3^i nodes where i is the level and for each level cost is $3^i \frac{cn}{2^i}$. Height of the tree is $\log n$. So the total cost is:

$$T_3(n) \leq cn \sum_{i=0}^{\log n} (\frac{3}{2})^i \tag{30}$$
$$= cn\Big(\frac{(\frac{3}{2})^{\log n} - 1}{\frac{3}{2} - 1}\Big) = 2c(n^{\log 3} - n) \tag{31}$$

Hence we can say that $T_3(n) \in \mathcal{O}(n^{\log 3})$

Substitution Method

We guess that $T_3(n) \in \mathcal{O}(n^{\log 3})$ and we select representatives for $mathcalO(n^{\log 3})$ and $mathcalO(n)$ as $cn^{\log 3}$ and $c'n$.

By assuming $\forall m < n$ and $T_3(m) \leq cm^{\log 3}$ we want to prove that $T_3(n) \leq cn^{\log 3}$

$T_3(n) \leq 3(c(n/2)^{\log 3}) + c'n = cn^{\log 3} + c'n$ from here we can't prove our assumption so we have to change our representative like previous exercise. Therefore new representative is $cn^{\log 3} - dn$

$T_3(n) \leq 3(c(n/2)^{\log 3} - dn/2) + c'n = cn^{\log 3} - dn/2 + c'n$ so for $-dn/2 + c'n \leq 0 -> d \geq 2c'$ we prove that $T_3(n) \in \mathcal{O}(n^{\log 3})$

1.5.4 d) $T_4(n) = 7 * T_4(n/2) + \Theta(n^2)$

Recursion Tree

Here for every level, we have 7^i nodes where i is the level and for each level cost is $7^i c\big(\frac{n}{2^i}\big)^2$. Height of the tree is $\log n$. So the total cost is:

$$T_4(n) \leq\geq cn^2 \sum_{i=0}^{\log n} (\frac{7}{4})^i \tag{32}$$

$$= cn^2\Big(\frac{(\frac{7}{4})^{\log n} - 1}{\frac{7}{4} - 1}\Big) \tag{33}$$

$$= \frac{4}{3}cn^2\Big((\frac{7}{4})^{\log n} - 1\Big) \tag{34}$$

$$= \frac{4}{3}c\Big(n^2(n)^{\log \frac{7}{4}} - n^2\Big) \tag{35}$$

$$= \frac{4}{3}c\Big((n)^{\log 7} - n^2\Big) \tag{36}$$

Hence we can say that $T_4(n) \in \Theta(n^{\log 7})$

Substitution Method

We guess that $T_4(n) \in \mathcal{O}(n^{\log 7})$ and we select representatives for $\mathcal{O}(n^{\log 7})$ and $\Theta(n^2)$ as $cn^{\log 7} - dn^2$ and $c'n^2$.

By assuming $\forall m < n$ and $T_4(m) \leq cm^{\log 7} - dm^2$ we want to prove that $T_4(n) \leq cn^{\log 7} - dn^2$

$T_4(n) \leq 7c(\frac{n}{2})^{\log 7} - d(\frac{n}{2})^2 + c'n^2 = cn^{\log 7} - \frac{dn^2}{4} + c'n^2$ So for $-\frac{dn^2}{4} + c'n^2 \leq 0 -> d \geq 4c'$ we prove that $T_4(n) \in \mathcal{O}(n^{\log 7})$

Now for lower boundary we need to update our representative as $cn^{\log 7}$ so we have $T_4(n) \geq 7c(\frac{n}{2})^{\log 7} + c'n^2 = cn^{\log 7} + c'n^2$ which is $\geq$ than $cn^{\log 7}$ so we can say that $T_4(n) \in \Omega(n^{\log 7})$.

Hence we proved that $T_4(n) \in \Theta(n^{\log 7})$