송자구 2022-2 기말시험

December 2022

1 문제 1

다음과 같은 입력이 있다: {금강산, 관악산, 내향산, 묘향산, 남산, 오대산, 지리산, 계룡산, 소요산}[1]

1. 위 입력에 대한 AVL Tree를 그리시오. 각 노드에 BF(Balance Factor)를 표시하시오.
2. 위 AVL Tree의 평균 검색 비용을 구하고, 그것이 최소 비용임을 보이시오.
3. 순서가 다른 입력에 대해서 AVL Tree는 유일한가? 증명하시오.

2 문제 2

1. 삽입 정렬(Insertion Sort)의 시간 복잡도를 구하라
2. 합병 정렬, 힙 정렬(Merge Sort, Heap Sort)의 시간 복잡도를 구하고 이를 보이시오
3. 퀵 정렬의 평균 시간 복잡도를 증명하여라
4. 최선의 경우 퀵 정렬의 시간 복잡도를 증명하여라

3 문제 3

그래프를 줌. 각 간선(Edge) 별로 가중치가 존재하고, A∼I까지 있던 거로 기억함

1. 위 그래프에 대한 인접 행렬(Adjency Matrix)를 구하라
2. 위 그래프에 대한 인접 리스트를 구하라.
3. 위 그래프의 A^+를 구하라.(힌트: 인접 리스트를 이용하라)
4. 위 그래프의 최소 신장 트리(Minimum Spanning Tree)를 구하라
5. 위 그래프의 G에서 시작하는 최소 비용 거리를 DFS를 이용하여 구하라

[1]순서는 임의로 결정하였고, 일부 산은 기억이 안 나서 임의로 집어넣음

4 문제 1 답

4.1 1.1 AVL Tree

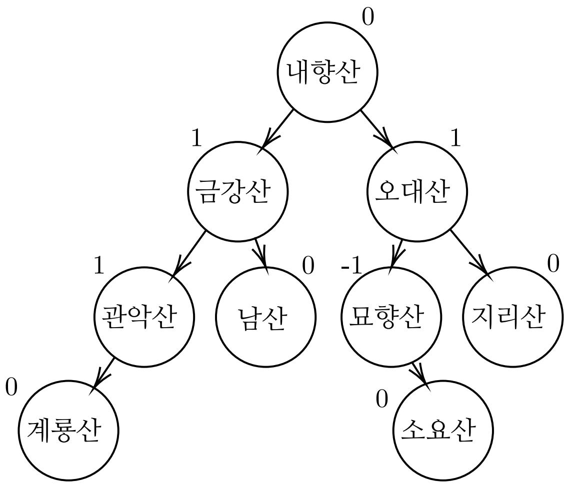


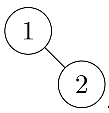
Figure 1: AVL tree

4.2 1.3 AVL tree 비유일성 증명

Answer: No. Not Unique. A balanced tree may have different order based on the order of operations made in order to get to it.

Counter Example:

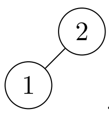
```
insert(1)
insert(2)
```

above will generate following AVL tree: 

. The result in no need for rebalancing.

consider another input:

```
insert(2)
insert(1)
```

above will generate following AVL tree: 

. The result is also no need for rebalancing.

Both are valid AVL trees with the same elements, but as you can see the form is not unique.

5 문제 2 답

5.1 2.1 삽입 정렬 시간 복잡도

Insert(e, a, i)는 최악의 경우 삽입 전 i+1번 비교해야 한다. 곧 Insert의 시간 복잡도는 $O(i)$. 한편 $i = j - 1 = 1, 2, \cdots, n - 1$일 때 Insert를 호출하므로 복잡도는 $O(\sum_{j=1}^{n-1}(i+1)) = O(n^2)$

5.2 2.2 합병 정렬 시간 복잡도

Now, let us follow up with the steps. our very own first step was to divide the input into two halves which comprised us of a logarithmic time complexity ie. $\log N$ where N is the number of elements.

our second step was to merge back the array into a single array, so if we observe it in all the number of elements to be merged N, and to merge back we use a simple loop which runs over all the N elements giving a time complexity of $O(N)$.

General Analysis

let T(N) for time complexity for problem size N, then $T(n) = \Theta(1) + 2T\left(\frac{1}{2}\right) + \Theta(n) + \Theta(1) = 2T\left(\frac{1}{2}\right) + \Theta(n)$

Now we can further divide the array into two halfs if size of the partition arrays are greater than 1. So,

$$\begin{aligned} \forall c \in \mathbb{R}, T(N) &= 2\left(2T\left(\frac{n}{4}\right) + \frac{cn}{2}\right) + cn \\ &= 4T\left(\frac{n}{4}\right) + 2cn \end{aligned} \tag{1}$$

For equation 1 we can say that where n can be subtituted to 2^k and the value of k is $\log N$ thus $T(n) = 2^k(T(\frac{n}{2^k})) + kcn$

Hence, $T(N) = NT(1) + N\log N = O(N\log N)$

5.3 2.2 힙 정렬 시간 복잡도

Assume that $2k^{-1} \leq n \leq 2^k$ for k level of tree, and n elements, then number of i-th level node is 2^{i-1}. At the first **for** loop, heap_sort function calls the Adjust function once for each node with a child. Thus time consumed for this loop equals for each level, number of nodes for that level multiplies max distance which each node can move. This value shall not exceed the following values on equation 2

$$\sum_{1\leq i\leq k} 2^{i-1}(k-1) = \sum_{1\leq i\leq k-1} 2^{k-i-1}i \leq n = \sum_{1\leq i\leq k-1} \frac{1}{2^i} < 2n = O(n) \tag{2}$$

5.4 2.3 퀵 정렬 평균 시간 복잡도

W.T.S: Let $T_{avg}(n)$ be the expected time for function quick_sort to sort a list with n records. Then $\exists k$ s.t. $T_{avg}(n) \leq kn\ln n$ for $n \geq 2$ which implies time complexity of quick sort algorithm is $O(n\log n)$

Prove: In the call to quick_sort(list, 1, n), the pivot gets placed at position j. This leaves us with the problem of sorting two sublists of size j-1 and n-j. The expected time for this is $T_{avg}(j-1) + T_{avg}(n-j)$. The remainder of the function clearly takes at most cn time for some constant c. Since j may take on any vlaues 1 to n with equal probability, we have equation 3 for $n \geq 2$.

$$T_{avg}(n) \leq cn + \frac{1}{n}\sum_{j=1}^{n}(T_{avg}(j-1) + T_{avg}(n-j)) = cn + \frac{2}{n}\sum_{j=0}^{n-1} T_{avg}(j) \tag{3}$$

We may assume that for $\forall b \in \mathbb{R}$, $T_{avg}(0) \leq b$ and $T_{avg}(1) \leq b$ so that $T_{avg}(0) + T_{avg}(1) \leq 2b$. We shall now show that $T_{avg}(n) \leq kn\ln n$ for $n \geq 2$ and $k = 2(b+c)$. The proof is by induction on n.

- Induction base: For $n = 2$, equation 3 yields $T_{avg}(2) \leq 2c + T_{avg}(0) + T_{avg}(1) \leq 2c + 2b \leq \overbrace{kn \ln 2}^{\because 1 < 2\ln 2}$
- Induction hypothesis: $T_{avg}(n) \leq kn \ln n$ for $1 \leq n < m$
- Induction Step: From equation 3 and the induction hypothesis we have equation 4

$$T_{avg}(m) \leq cm + \frac{4b}{m} + \frac{2}{m}\sum_{j=2}^{m-1} T_{avg}(j) \leq cm + \frac{4b}{m} + \frac{2k}{m}\sum_{j=2}^{m-1} j \ln j \tag{4}$$

Since $j \ln j$ is an increasing function of j, equation 4 yields equation 5

$$\begin{aligned} T_{avg}(m) &\leq cm + \frac{4b}{m} + \frac{2k}{m}\int_2^m x \ln x \, dx \\ &= cm + \frac{4b}{m} + \frac{2k}{m}\overbrace{\left(\frac{m^2 \ln m}{2} - \frac{m^2}{4}\right)}^{\text{use partial integral}} \\ &= cm + \frac{4b}{m} + km \ln m - \frac{km}{2} \\ &\leq km \ln m, \text{ for } m \geq 2 \blacksquare \end{aligned} \tag{5}$$

5.5 2.4 퀵 정렬 최선의 경우 시간 복잡도

When the list splits roughly into two equal parts each time,

$$\begin{aligned} \forall c \in \mathbb{R}, T(n) &\leq cn + 2T\left(\frac{n}{2}\right) \\ &\leq cn + 2\left(\frac{cn}{2} + 2T\left(\frac{n}{4}\right)\right) \\ &\leq 2cn + 4T\left(\frac{n}{4}\right) \\ &\vdots \\ &\leq cn \log_2 n + nT(1) = O(n \log_2 n) \end{aligned} \tag{6}$$

Note that $O(n \log_2 n)$ equals $O(n \log n)$ as $\log_2 n = \dfrac{\log n}{\log 2}$, but the $\log 2$ is contant.

위까지가 교재에 나온 증명이고, 아래는 제가 공부했던 더 엄밀한 증명입니다:

let T(n) be expected time for function Quicksort to sort an array with n elements and S(n) be expected time for comparison before recursive calling begin. But in Quicksort, as you linearly compare the elements so it is clear that $S(n) = n$. in Quicksort, $T(n) = \dfrac{2\{T(0) + T(1) + \cdots + T(n-1)\}}{n} + (n-1)$. Multiply both sides by n, then

$$nT(n) = 2\{T(0) + \cdots + T(n-2) + T(n-1)\} + n(n-1) \tag{7}$$

However about $n - 1$, you get

$$(n-1)T(n-1) = 2\{T(0) + \cdots + T(n-2)\} + (n-1)(n-2) \tag{8}$$

Now you substract equation 7 and 8 then you get equation 9

$$nT(n) - (n-1)T(n-1) = 2T(n-1) + 2(n-1) \tag{9}$$

Summarize equation 9 by $nT(n)$ then

$$nT(n) = (n+1)T(n-1) + 2(n-1) \tag{10}$$

Divide equation 10 by $n(n+1)$ then

$$\begin{aligned}
\frac{T(n)}{n+1} &= \frac{T(n-1)}{n} + \frac{2(n-1)}{n(n+1)} \\
&= \frac{T(n-1)}{n} + 2(n-1)\left(\frac{1}{n} - \frac{1}{n+1}\right)\overbrace{\left(\because \frac{1}{n(n+1)} = \frac{1}{n} - \frac{1}{n+1}\right)}^{\text{use } \frac{1}{AB} = \frac{1}{A} - \frac{1}{B}} \\
&= \frac{T(n-1)}{n} + \frac{2(n-1)}{n} - \frac{2(n-1)}{n+1}
\end{aligned} \tag{11}$$

As the same way, we can express $\dfrac{T(n-1)}{n}$ as equation 12

$$\frac{T(n-1)}{n} = \frac{T(n-2)}{n-1} + \frac{2(n-2)}{n-1} - \frac{2(n-2)}{n} \tag{12}$$

By equation 12, we can rewrite equation 11:

$$\frac{T(n)}{n+1} = \frac{T(n-2)}{n-1} + \frac{2(n-2)}{n-1} - \frac{2(n-2)}{n} + \frac{2(n-1)}{n} - \frac{2(n-1)}{n+1} \tag{13}$$

However again we can rewrite 13 by $\dfrac{T(n-2)}{n-1} = \dfrac{T(n-3)}{n-2} + \dfrac{2(n-3)}{n-2} - \dfrac{2(n-3)}{n-1}$ then

$$\frac{T(n-2)}{n-1} = \frac{T(n-3)}{n-2} + \frac{2(n-3)}{n-2} - \frac{2(n-3)}{n-1} \tag{14}$$

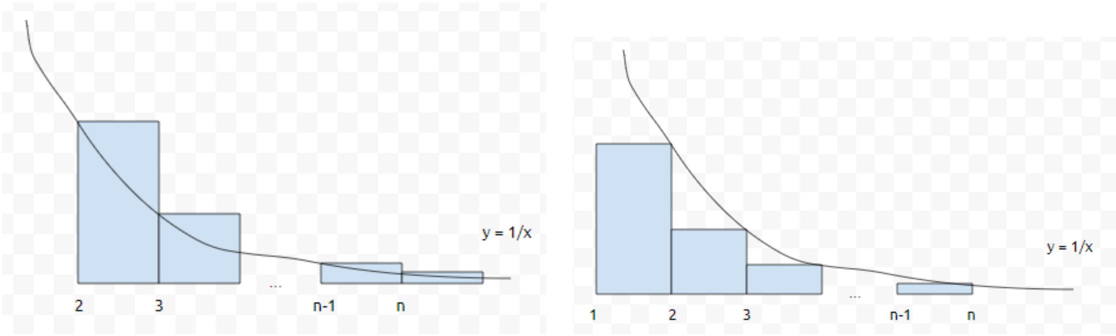
If we keep rewrite above, then we get

$$\frac{T(n)}{n+1} = \frac{T(n-k)}{n-k+1} + 2\left(\frac{1}{n-k+1} + ... + \frac{1}{n}\right) - \frac{2(n-1)}{n+1}, (\forall k \in \mathbb{N} \cap [1, n-1]) \tag{15}$$

when $k = n-1$, then

$$\begin{aligned}
\frac{T(n)}{n+1} &= \frac{T(1)}{2} + \frac{2}{2} + \frac{2}{3} + ... + \frac{2}{n} - \frac{2(n-1)}{n+1} \\
&= \cancelto{0}{\frac{T(1)}{2}} + 2\left(\frac{1}{2} + ... + \frac{1}{n}\right) - \cancelto{2}{\frac{2(n-1)}{n+1}} \\
&\approx 2\left(\frac{1}{2} + ... + \frac{1}{n}\right), (\because \lim_{n\to\infty} \frac{2(n-1)}{(n+1)} = 2, T(1) = 0)
\end{aligned} \tag{16}$$

Now let's see $2\left(\dfrac{1}{2} + ... + \dfrac{1}{n}\right)$, express it compare to $y = \dfrac{1}{x}(x \in \mathbb{R})$, then you get figure 2(a).



(a) From 2 to n+1, compare to $y = \frac{1}{x}$ (b) From 1 to n, compare to $y = \frac{1}{x}$

Figure 2: Comparison between $\left(\frac{1}{2} + ... + \frac{1}{n}\right)$ and function y

In figure 2(a), sum of the areas of all rectangles equals $\left(\frac{1}{2} + ... + \frac{1}{n}\right)$, and notice that the sum of the areas of all rectangles are greater than integral value of y from x=2 to x=n+1 so we get

$$\left(\frac{1}{2} + ... + \frac{1}{n}\right) = \sum_{k=2}^{n+1} \frac{1}{k} > \int_{2}^{n+1} \frac{1}{x} dx = \ln(n+1) - \ln 2 \tag{17}$$

On the other hand, by the -1 parallel movement to the x-axis of the rectangles, on figure 2(b), we can see

$$\left(\frac{1}{2} + ... + \frac{1}{n}\right) < \int_{1}^{n} \frac{1}{x} dx = \ln(n) - \cancelto{0}{\ln 1} \tag{18}$$

By equation 17 and 18, we get $\ln(n+1) - \ln 2 < \left(\frac{1}{2} + ... + \frac{1}{n}\right) < \ln(n) \Longrightarrow \sum_{k=2}^{n+1} \frac{1}{k} = \ln n$. And as the form of T(n) equals $2 \ln n$,

$$T(n) = 2(n+1)(\ln n) = \frac{2}{\log_2 e}(n+1)\log_2 n \Longleftrightarrow T(n) = O(n \log_2 n)\blacksquare \tag{19}$$

6 Notice

시험 잘 보십시오 후배님들