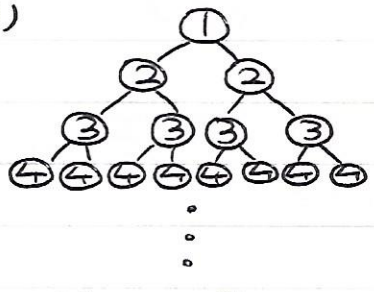
HW1 20191286 강현 ①

#1)



n개의 노드를 갖는 full binary tree, T_n

① full binary tree의 노드 개수가 n개이면 이 트리의 가장 마지막 노드가 있는 레벨 k는

$$k = \log_2(n+1)$$

과 같이 표현할 수 있다. ($\because n = 2^k - 1$)

② 각 노드에 대한 계산 비용이 해당 노드의 level이라고 하였으므로, level i를 제외 level 크하는 노드들의 계산 비용 합을 $f(i)$라고 하면

$$f(1) = 1 \times 2^0$$
$$f(2) = 2 \times 2^1$$
$$\vdots$$
$$f(i) = i \times 2^{i-1}$$

라는 규칙을 찾을 수 있다. 이는 level i에 있는 노드의 수가 2^{i-1}이기 때문이다.

③ 따라서, T_n에 대한 전체 루트 비용은 (→ 이하 $C(n)$) level이 1일 때부터 k일 때까지, 각 레벨에 있는 모든 노드의 계산 비용 합의 합, 즉 $f(1)$부터 $f(k)$까지 합한 비용

과 같으므로, $C(n) = \sum_{i=1}^{k} f(i)$ 를 표현이 가능하다. 이때, $\sum_{i=1}^{k} f(i)$는

$$\sum_{i=1}^{k} f(i) = 1\times2^0 + 2\times2^1 + 3\times2^2 + \cdots + K\times2^{K-1} \quad \cdots ①$$

이므로 ①에 2를 곱한

$$2\times\sum_{i=1}^{k} f(i) = \quad 1\times2^1 + 2\times2^2 + \cdots + (K-1)\times2^{K-1} + K\times2^K \quad \cdots ②$$

②에서 ①을 빼서

$$②-① = \sum_{i=1}^{k} f(i) = -1\times2^0 - 1\times2^1 - 1\times2^2 - \cdots - 1\times2^{K-1} + K\times2^K$$
$$= -(2^0 + 2^1 + \cdots + 2^{K-1}) + K\times2^K$$
$$= -\left(\frac{2^K - 1}{2-1}\right) + K\times2^K = K\times2^K - 2^K + 1 = (K-1)\times2^K + 1$$

이므로 증명할 수 있다.

④ $C(n)=\sum_{i=1}^{k} f(i)=(k-1)\times 2^{k}+1$ 를 n에 대한 식으로 고치기 위해 ①에서 구한 $k=\log_2(n+1)$ 을 대입하면

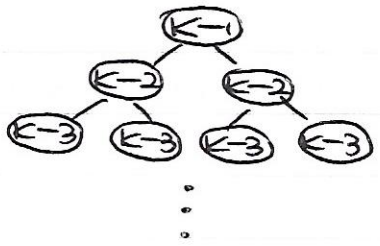
$$C(n)=(\log_2(n+1)-1)\times 2^{\log_2(n+1)}+1$$
$$=(n+1)(\log_2(n+1)-1)+1$$

이 된다. ☒

⑤ 이때의 시간복잡도를 Big-O 기호를 사용하여 표현하면

$O(n\log n)$ 또는 $O(n\ln n)$으로 나타낼 수 있다. ☒

#2)



level
1
2
3
k

n개의 노드를 갖는,
$k=\log_2(n+1)$의 level을 갖으며
모든 level이 full binary tree

① 문제에서, level i에 있는 모든 노드들의 계산량을 $f(i)$라고 하면,

$f(1)=(k-1)\times 2^{0}$
$f(2)=(k-2)\times 2^{1}$
⋮
$f(i)=(k-i)\times 2^{i-1}$
⋮
$f(k)=(k-k)\times 2^{k-1}$ 이다.

② 따라서, k에 대한 전체 계산량은 $\sum_{i=1}^{k} f(i)$로 구할 수 있고,

$$\sum_{i=1}^{k} f(i)=(k-1)\times 2^{0}+(k-2)\times 2^{1}+\cdots+(k-k)\times 2^{k-1} \quad \cdots ①$$

이때 ①에 2를 곱한

$$2\times\sum_{i=1}^{k} f(i)=\quad (k-1)\times 2^{1}+\cdots+(k-(k-1))\times 2^{k-1}+(k-k)\times 2^{k} \quad \cdots ②$$

②에서 ①을 빼면

$$② - ① = \sum_{i=1}^{k} f(i) = -(k-1)\times 2^0 + 1\times 2^1 + \cdots + 1\times 2^{k-1} + (k-k)\times 2^k$$

$$= -k+1 + (2^1 + \cdots + 2^{k-1})$$

$$= -k+1 + \left(\frac{2^{k-1}-1}{2-1}\right)$$

$$= 2^{k-1} - k$$

라고 정리할 수 있다.

③ 마찬가지로, $C(n)$을 n에 대한 식으로 표현하기 위해 $T_n = 2^{k-1} - k$에 $k = \log_2(n+1)$을 대입하면

$$C(n) = 2^{\log_2(n+1)-1} - \log_2(n+1)$$

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

가 된다. ☑

④ 이때의 시간복잡도는 Big-O 표현을 이용하여 $O(n)$으로 나타낼 수 있다. ☑