Dean Meluban
COMP 251 - A1
260764246

10/8/19

Collaborator: Jiahui Peng

3.1) O(1): When adding a prime # to the heap, a new node is added to the bottom of the heap. By heap definition, insertion is O(1).

3.2) O(log n): When adding a number that satisfies $j = p^{\alpha}$, we must re-construct the heap. Thus, the complexity is essentially the complexity of Max-Heapify which is O(log n)

3.3) Proof

$N = \sqrt{n} \times \log_2 n$

$$\lim_{n \to \infty} \frac{\log_2 n}{\sqrt{n}} = 0$$

$\hookrightarrow \log_2 n < \sqrt{n}$

$\therefore \sqrt{n}(\log_2 n) < \sqrt{n}(\sqrt{n})$ (mult both sides by $\sqrt{n}$)

$\therefore \sqrt{n}(\log_2 n) < n$

$N < o(n)$

Explanation: We know there exists at most $\sqrt{n}$ primes. Beause there exist $\log_2 n$ numbers that fit the form $j = 2^m, m \in \mathbb{Z}$, $\log_3 n$ numbers that fit the form $j = 3^m, m \in \mathbb{Z}$, $\log_5 n$ numbers that fit the form $j = 5^m, m \in \mathbb{Z}$, and we know $\log_2 n > \log_3 n > \log_5 n > \dots$ etc, the upper bound must be $\log_2 n \times \sqrt{n}$. Hence $N < n$, and N is asymptotically negligble to n.

3.4) To compute LCM using a heap equipped w/ algo, we must use either op_1 or op_2. We know it takes $\sqrt{n}$ time to compute whether the integer is prime.

<u>Case 1</u>

If $j \in \mathbb{Z}$ satisfies the equation $j = p^d$, operation 2 will be executed. Time complexity of op_2 is $O(\sqrt{n} + \log n + 1)$ where $\sqrt{n}$ is prime check, $(\log n)$ is reconstructing heap, and 1 is multiplying the LCM.

We also know that the amount of numbers in this case will be less than $\sqrt{n}$. Hence,

$$O(\sqrt{n}(\sqrt{n} + \log n + 1)) = O(n + n\log n + \sqrt{n}) = O(n)$$

<u>Case 2</u>

If j is prime, operation 1 will be used. Time complexity for op_1 is $O((\sqrt{n}+1)(\underbrace{\sqrt{n} - n}_{*})) = O(n\sqrt{n} - n + n - \sqrt{n}) = O(n\sqrt{n})$. We multiply by * b/c we know thats how many primes will use this operation

<u>Case 3</u>

If j does not use either operation, time complexity is $O((n - \sqrt{n})(\sqrt{n})) = O(n\sqrt{n} - n)$
$= O(n\sqrt{n})$

—

Combining all time complexities for each case yields

$$O(n\sqrt{n}) + O(n) + O(n\sqrt{n})$$
$$= O(n\sqrt{n} + n + n\sqrt{n})$$
$$= O(2n\sqrt{n} + n)$$
$$= O(n\sqrt{n})$$

□