Algorithm Design
Assignment 2

Hana Esfandiar
Mohammad Afshari

April 14, 2022

▶Algorithm

In order to find the minimum f, we use binary search algorithm.

In *find_minimum_f* function, we use binary search algorithm to find an f. We start from the middle. Then we pass that middle number to *find_possible_f* function. This function will calculate:

$$f + \left\lfloor \frac{f}{m} \right\rfloor + \left\lfloor \frac{f}{m^2} \right\rfloor + \left\lfloor \frac{f}{m^3} \right\rfloor + \dots$$

until $\left\lfloor \frac{f}{m^i} \right\rfloor = 0$. The *find_possible_f* function returns the result of the above expression. If the result is equal to l(number of lines that we want to write before deadline) then we found the minimum f. If the result is less than l it means that the minimum f should be larger than this f so we have to apply the binary search on the right half of the sequence. If the result is grater than l, we have to apply the binary search on the left half of the sequence because there might be an smaller f.

▶Analyzing the Time Complexity

In the *find_minimum_f* function we have a while loop that in each iteation, We halve the sequence until we reach one number and this will be done in *logl* division. In each iteration of this while loop, we call the *find_possible_f* function. There is a also a while loop in this function. This while loop will stop when $\left\lfloor \frac{f}{m^i} \right\rfloor = 0$ which means m^i should be grater than f.
Suppose the number of lines that we want to write is n.

▷Time Complexity of find_possible_f

Stop condition of while loop: $m^i > f$

$$\begin{aligned} &\log m^i > \log f \\ &i \times \log m > \log f \\ &i > \frac{\log f}{\log m} \\ &i > \log_m^f \end{aligned}$$

So the while loop will run $\log_m^f$ times.
Since $f <= n$ then we can write: $\log_m^n$ and since for $m >= 2$, $\log_m^n < \log_2^n$ we can write $\log_2^n$ instead of $\log_m^n$. In conclusion, the while loop run $\log n$times.

▷Final Time Complexity

We explained that the while loop in *find_minimum_f* function runs $\log n$ times. Suppose $T(n)$ is the number of times that the line 14 of code runs. So we havs:

$$T(n) = \sum_{j=1}^{\log n} \sum_{i=1}^{\log n} 1 = \sum_{j=1}^{\log n} \log n$$

$$T(n) = \sum_{j=1}^{\log n} \log n = \log n \log n$$

$$T(n) \in O(\log n \log n)$$