Karahan Sarıtaş – 2018400174
Furkan Özdemir – 2018400201

Table of Contents

Karahan Sarıtaş – 2018400174
Furkan Özdemir – 2018400201

THEORETICAL ANALYSIS

Basic operation is the comparison marked as (1)

Analyze B(n)

Regardless of the content of the list given as input, outer *for loop* will iterate from $i = 0$ to $i = n - 1$. In each iteration, comparison marked as (1) will be performed. Therefore in total, there will be n many comparison operations.

$$B(n) = n \in \theta(n)$$

Analyze W(n)

As stated above, regardless of the content of the input n many comparison operations will be performed.

$$W(n) = n \in \theta(n)$$

Analyze A(n)

As stated above, regardless of the content of the input n many comparison operations will be performed.

$$A(n) = \sum_{i=B(n)}^{W(n)} i = B(n) = W(n) = n \in \theta(n)$$

Basic operations are the two loop incrementations marked as (2)

Analyze B(n)

If $X[i] = 0$, then first *for loop* will be executed. If $X[i] = 1$, second *for loop* will be executed. Both for loops start from current i and go to $n - 1$. For any given i, number of incrementations performed in any loop: $(n - 1) - i$. Notice that if i = $(n - 1)$, there won't be any incrementation (but one iteration will be performed). At this point, I assume that the *for loop* in the pseudocode behaves like Python *for-loop*. For consistency, that is the behavior assumed throughout this project.

$$\sum_{i=0}^{n-1} (n-1) - i \;=\; (n-1) - 0 \;+\; (n-1) - 1 \;+\; \dots \;+\; (n-1) - (n-1)$$

$$= (n-1)n - (0 + 1 + 2 + \cdots + n - 1)$$

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

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

Karahan Sarıtaş – 2018400174
Furkan Özdemir – 2018400201

$$B(n) = \frac{(n-1)n}{2} \in \theta(n^2)$$

Since the number of incrementations for each loop is the same. There is no difference between $X[i] = 0$ and $X[i] = 1$.

Analyze W(n)

We explained that number of incrementations do not depend on the content of the input. Therefore worst-case time complexity will be the same.

$$W(n) = \frac{(n-1)n}{2} \in \theta(n^2)$$

Analyze A(n)

Again, if $X[i] = 0$ first loop will be executed. If $X[i] = 1$, second loop will be executed. Since they are exactly the same, which one goes first doesn't matter.

$$A(n) = \sum_{i=B(n)}^{W(n)} i = B(n) = W(n) = \frac{(n-1)n}{2} \in \theta(n^2)$$

Basic operation is the assignment marked as (3)

Analyze B(n)

In best case, $X[i] = 1 \quad \forall i \in \{0,1,2,\dots,n-1\}$, program never executes the inner part of the *if clause*. Therefore, assignment marked as (3) is never performed. We can say that the time complexity is O(1).

Analyze W(n)

In worst case, $X[i] = 0 \quad \forall i \in \{0,1,2,\dots,n-1\}$, program always executes the *if clause* of the *if-else* statement. There are nested for loops within the *if clause*, first loop goes from $j = i$ to $j = n-1$ one by one and second for loop goes from $k = n$ to $k = 1$, dividing by 2 in each step.

Assume that $n = 2^m, \; m \in \mathbb{N}$
Let's show the value of k in each iteration:

$$k = n, \quad k = \frac{n}{2^1}, \quad k = \frac{n}{2^2}, \quad \dots \quad k = \frac{n}{2^m}$$

(first iteration) ... ($(m+1)$. iteration)

In total, $m+1$ many iterations, and in each iteration one assignment operation will be performed.

$$n = 2^m, \;\; therefore\; m = log(n)$$

Karahan Sarıtaş – 2018400174
Furkan Özdemir – 2018400201

Outer loop goes from $j = i$ to $j = n - 1$. For any given i, number of iterations performed in this loop: $n - i$.

when $i = 0$, from $j = 0$ to $n - 1$ there are n iterations.
$i = 1$, from $j = 1$ to $n - 1$ there are $n - 1$ iterations.
…
$i =$ to $n - 1$, from $j = n - 1$ to $n - 1$ there is 1 iteration.

With the $m + 1$ assignment operations inside, in total:

$$W(n) = (m+1)\,[n + (n-1) + \cdots + 1]$$
$$W(n) = (\log(n)+1)\,[n + (n-1) + \cdots + 1]$$
$$W(n) = (\log(n)+1)\,\frac{(n+1)\cdot n}{2}$$
$$W(n) = \frac{log(n)n^2}{2} + log(n)\frac{n}{2} + \frac{n^2}{2} + \frac{n}{2} \in \theta(n^2\,log(n)) \quad for\ \ n = 2^m$$

- $\frac{log(n)n^2}{2} + log(n)\frac{n}{2} + \frac{n^2}{2} + \frac{n}{2}$ is eventually nondecreasing.
- $n^2\,log(n)$ is eventually nondecreasing.
- $n^2\,log(n)$ is $\theta - in$variant under scaling.
  - $(c\cdot n)^2\,log(cn) = c^2n^2[\log(c) + \log(n)] \in \theta(n^2\,log(n))$

Therefore, by Interpolation $W(n) \in \theta(n^2\,log(n))\ \ for\ \ n \in \mathbb{N}$

Analyze A(n)

For each i, there is $\frac{1}{3}$ possibility of having 0 and $\frac{2}{3}$ possibility of having 1. We have already shown that innermost *for-loop* performs $(\log(n)+1)$ operations when $X[i] = 0$. We also showed that for each i, outer loop iterates $n - i$ times.

when $i = 0$, from $j = 0$ to $n - 1$ there are n iterations. $(n - i)$
$i = 1$, from $j = 1$ to $n - 1$ there are $n - 1$ iterations. $(n - i)$
…
$i =$ to $n - 1$, from $j = n - 1$ to $n - 1$ there is 1 iteration $(n - i)$

To get the average time complexity, we should sum up $(\text{number of operations})$ * (probability) values for all $i \in \{0,1,2,\dots,n-1\}$. We stated in our best-case analysis that if $X[i] = 1$, number of operations performed will be zero. For any given i, if $X[i] = 0$, number of operations performed will be $(\log(n)+1)$. $(n + 1 - i)$.

Karahan Sarıtaş – 2018400174
Furkan Özdemir – 2018400201

$$\sum_{i=0}^{n-1} \frac{1}{3}[log(n)+1](n-i) = \frac{1}{3}[log(n)+1]\sum_{i=0}^{n-1}(n-i)$$

$$\frac{1}{3}[log(n)+1](n+(n-1)+\cdots+1) = \frac{1}{3}(log(n)+1)\frac{(n+1)\cdot n}{2}$$

$$A(n) \;=\; \frac{1}{3}(log(n)+1)\frac{(n+1)\cdot n}{2} \;\in \theta(n^2\,log(n))$$

Basic operations are the two assignments marked as (4)

Analyze B(n)

If we assume $X[i]=0 \quad \forall i \in \{0,1,2,\dots,n-1\}$, first assignment operation will be performed $\frac{log(n)n^2}{2} + log(n)\frac{n}{2} + \frac{n^2}{2} + \frac{n}{2}$ times. If we assume $X[i]=1 \quad \forall i \in \{0,1,2,\dots,n-1\}$, then second assignment operation will be performed $n\left(1+\frac{1}{2}+\frac{1}{3}+\cdots+\frac{1}{n}\right)\cdot\frac{(n+1)\cdot n}{2}$ times. It is proven in worst-case analysis below. Therefore for the best case, our input should consist of only 0's.

$$B(n) \in \theta(n^2\,log(n))$$

Analyze W(n)

For the worst case, we should either assume $X[i]=0 \quad \forall i \in \{0,1,2,\dots,n-1\}\; or\; X[i]=1 \quad \forall i \in \{0,1,2,\dots,n-1\}$. From above, we know that worst case time complexity of the first assignment is $\theta(n^2\,log(n))$. Now, let's assume $X[i]=1 \quad \forall i \in \{0,1,2,\dots,n-1\}$ and calculate the total number of assignment operations for the second one.

Inner loop starts from $t=1$ and goes to $t=n$. In each iteration, x starts from n and it's decremented by t until it becomes nonpositive. Let's say for each t, *while* loop iterates k many times. Due to the conditional, we know that:

$$n-kt \le 0$$
$$n \le kt$$
$$\frac{n}{t} \le k$$

$\frac{n}{t}$ can be non-integer. Therefore we should say $k = \left\lceil\frac{n}{t}\right\rceil$

Let's first solve for input sizes for which $\frac{n}{t}$ is integer for all t. We can denote such input sizes as follows: $n \in \{m \mid \forall t \in \{1,2,\dots,m\} \;\; m \equiv 0 (mod\; t)\}$

Total number of assignments:

$$\left[n+\frac{n}{2}+\frac{n}{3}+\cdots+\frac{n}{n}\right] = n\left(1+\frac{1}{2}+\frac{1}{3}+\cdots+\frac{1}{n}\right)$$

Karahan Sarıtaş – 2018400174
Furkan Özdemir – 2018400201

$(t = 1)$ $(t = n)$

Outer loop goes from $m = i$ to $m = n-1$. For any given i, number of iterations performed in this loop: $n - i$.

when $i = 0$, from $m = 0$ to $n-1$ there are n iterations.
$i = 1$, from $m = 1$ to $n-1$ there are $n-1$ iterations.
…
$i =$ to $n-1$, from $m = n-1$ to $n-1$ there is 1 iteration.

In total,

$$W(n) = n\left(1+\frac{1}{2}+\frac{1}{3}+\cdots+\frac{1}{n}\right)\cdot[n+(n-1)+\cdots+1]$$

$$W(n) = n\left(1+\frac{1}{2}+\frac{1}{3}+\cdots+\frac{1}{n}\right)\cdot\frac{(n+1)\cdot n}{2}$$

Harmonic series: $H(n) \in \theta(log(n))$ - Proof using Integration Technique:

$$\sum_{x=1}^{n}\frac{1}{x} = 1+\sum_{x=2}^{n}\frac{1}{x}$$

Since $\frac{1}{x}$ is a non-increasing function:

$$\int_{2}^{n+1}\frac{1}{x}dx \le \sum_{x=2}^{n}\frac{1}{x} \le \int_{1}^{n}\frac{1}{x}dx$$

$$ln\left(\frac{n+1}{2}\right) \le \sum_{x=2}^{n}\frac{1}{x} \le ln(n)$$

$$ln\left(\frac{n+1}{2}\right) + 1 \le H(n) \le ln(n) + 1$$

In conclusion: $W(n) = n\left(1+\frac{1}{2}+\frac{1}{3}+\cdots+\frac{1}{n}\right)\cdot\frac{(n+1)\cdot n}{2} \in \theta(n^3\, log(n))$

$$for\ \ n \in \{m \mid \forall t \in \{1,2,\ldots,m\}\ \ m \equiv 0 (mod\, t)\}$$

- $n\left(1+\frac{1}{2}+\frac{1}{3}+\cdots+\frac{1}{n}\right)\cdot\frac{(n+1)\cdot n}{2}$ is eventually nondecreasing.
- $n^3\, log(n)$ is eventually nondecreasing.
- $n^3\, log(n)$ is $\theta - in$variant under scaling.
    - $(c\cdot n)^3\, log(cn) = c^3 n^3[\log(c)+\log(n)] \in \theta(n^3\, log(n))$

Therefore, by Interpolation $W(n) \in \theta(n^3\, log(n)\,)\ for\ n \in \mathbb{N}$

As can be seen, $X[i] = 1 \quad \forall i \in \{0,1,2,\ldots,n-1\}$ yields a worse time complexity than $X[i] = 0 \quad \forall i \in \{0,1,2,\ldots,n-1\}$. Therefore worst time complexity should be calculated assuming $X[i] = 1$ for all i.

Analyze A(n)

To analyze average time complexity, we will consider the probability distribution. For each i, $X[i] = 0$ with a probability of $\frac{1}{3}$ and $X[i] = 1$ with a probability of $\frac{2}{3}$. We have shown in our average time complexity analysis for the previous part, if $X[i] = 0$, number of operations performed will be $[log(n) + 1](n - i)$. From our worst-case analysis for this part, we can say that if $X[i] = 1$, there will be $n\left(1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{n}\right)(n - i)$ operations in total.

So, for $i \;=\; i_0$ average number of operations:

$$\frac{1}{3}[log(n) + 1](n - i_0) + \frac{2}{3}n\left(1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{n}\right)(n - i_0)$$

In total, average number of operations:

$$A(n) = \sum_{i=0}^{n-1} \frac{1}{3}[log(n) + 1](n - i) + \frac{2}{3}n\left(1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{n}\right)(n - i)$$

$$A(n) \;=\; [\tfrac{1}{3}[log(n) + 1] \;+\; \tfrac{2}{3}n\left(1 + \tfrac{1}{2} + \tfrac{1}{3} + \cdots + \tfrac{1}{n}\right)]\ \textstyle\sum_{i=0}^{n-1}(n - i)$$

$$A(n) \;=\; [\tfrac{1}{3}[log(n) + 1] \;+\; \tfrac{2}{3}n\left(1 + \tfrac{1}{2} + \tfrac{1}{3} + \cdots + \tfrac{1}{n}\right)]\ \tfrac{(n+1)\cdot n}{2} \in \theta(n^3\, log(n))$$

IDENTIFICATION OF BASIC OPERATION(S)

Two assignment operations marked as (4) are the basic operations. Because they are the characteristic operations contributing the most to the total runtime of the given algorithm. First assignment operation marked as (3) is not enough on its own since we have to consider the operations performed when $X[i] = 1$.

Karahan Sarıtaş – 2018400174
Furkan Özdemir – 2018400201

REAL EXECUTION

Best Case

N Size	Time Elapsed
1	0.000 ms
10	0.000 ms
50	0.998 ms
100	5.983 ms
200	20.983 ms
300	53.444 ms
400	85.730 ms
500	130.651 ms
600	220.410 ms
700	325.295 ms

Worst Case

N Size	Time Elapsed
1	0.333 ms
10	0.969 ms
50	30.918 ms
100	272.860 ms
200	2600.837 ms
300	9570.347 ms
400	24409.351 ms
500	54111.419 ms
600	87529.315 ms
700	148560.024 ms

Average Case

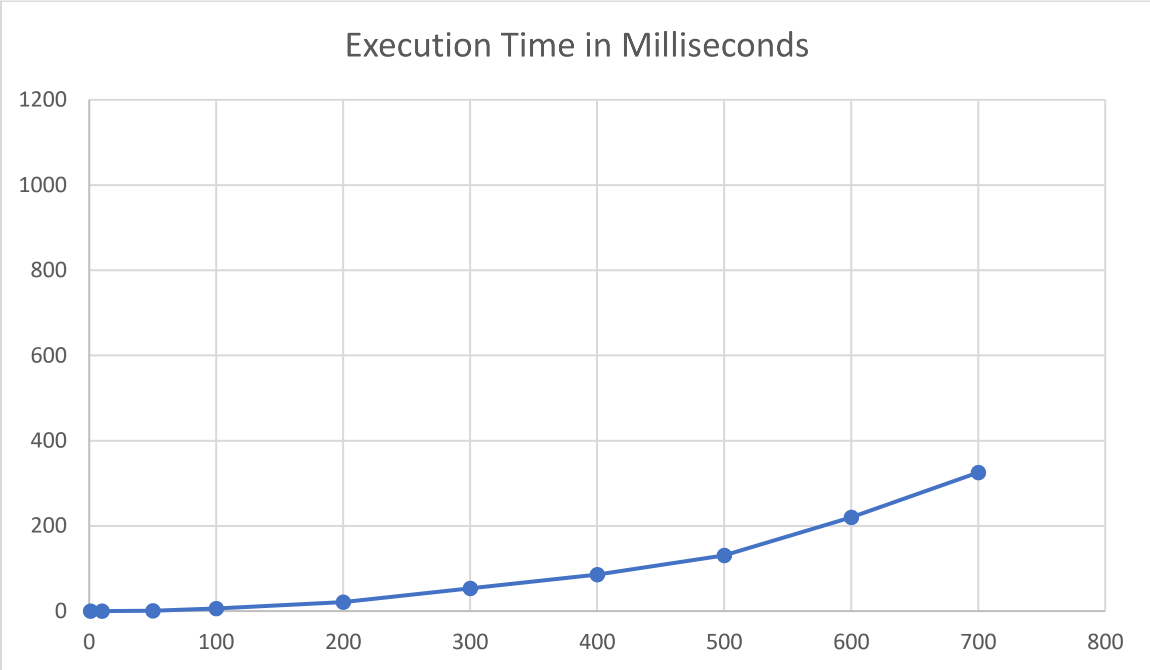
N Size	Time Elapsed
1	0.000 ms
10	0.321 ms
50	18.004 ms
100	187.574 ms
200	1820.527 ms
300	5845.871 ms
400	15070.106 ms
500	31565.533 ms
600	54148.284 ms
700	121735.798 ms

COMPARISON

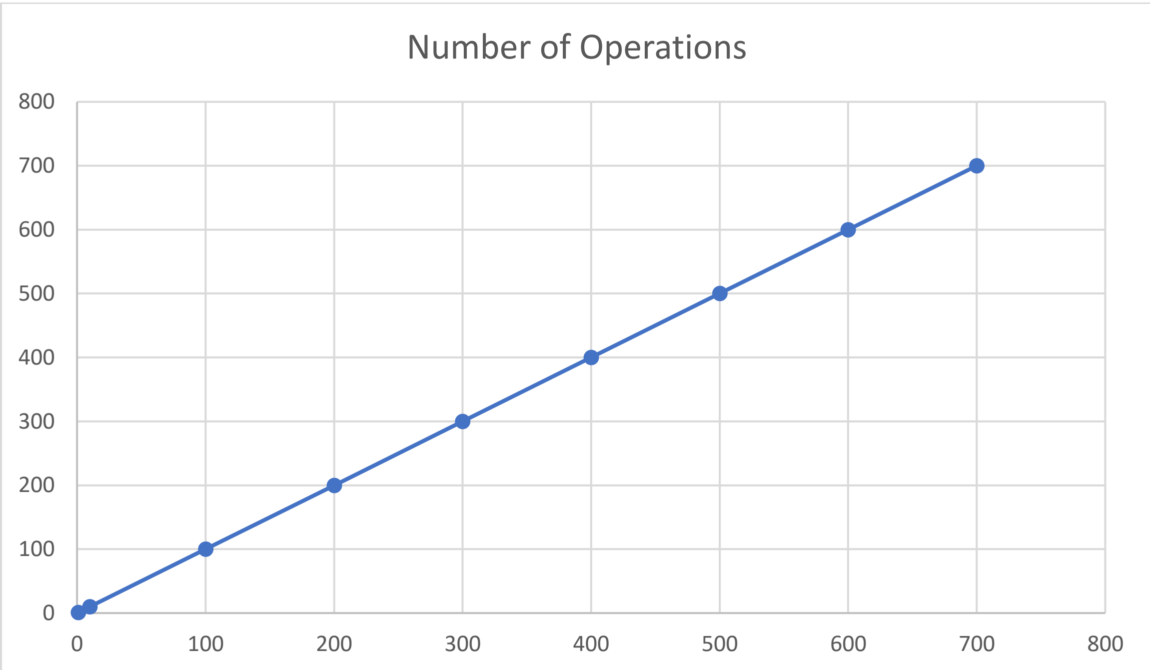
Best Case

Graph of the real execution time of the algorithm

Karahan Sarıtaş – 2018400174
Furkan Özdemir – 2018400201

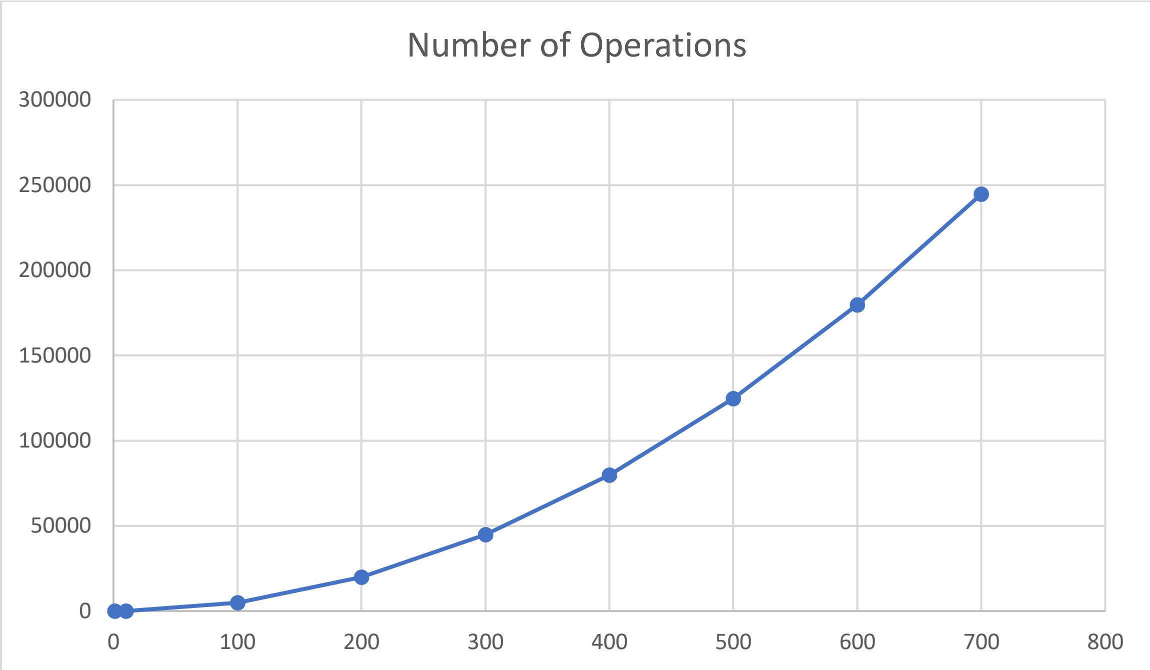


Graph of the theoretical analysis when basic operation is the operation marked as (1)

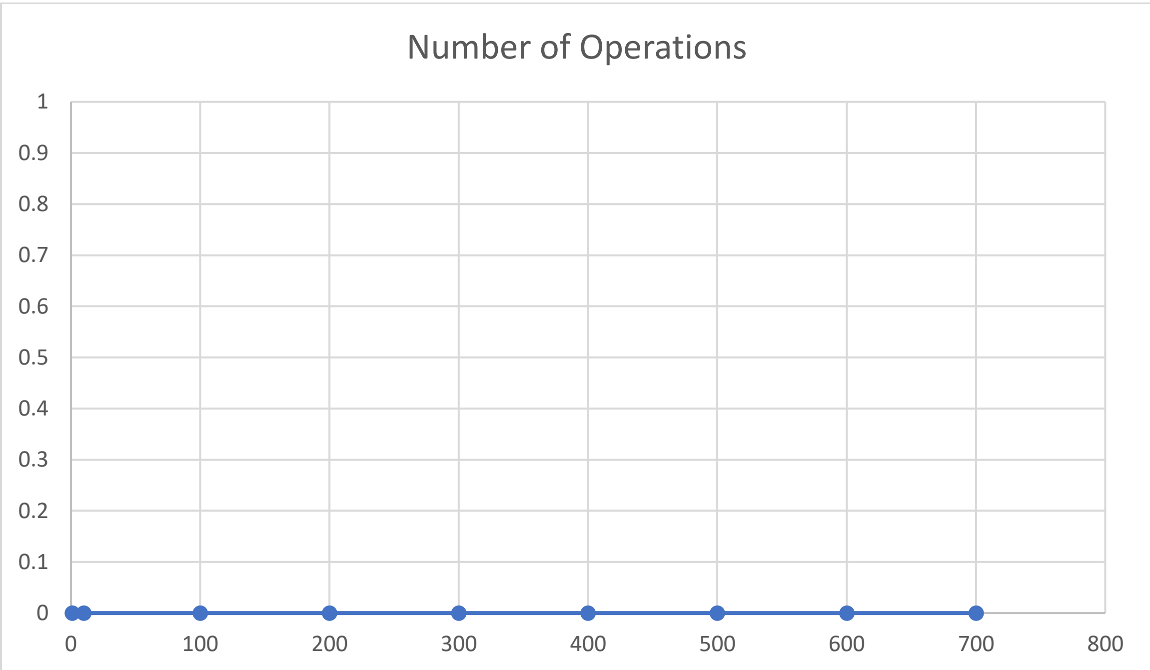


Graph of the theoretical analysis when basic operation is the operation marked as (2)

Karahan Sarıtaş – 2018400174
Furkan Özdemir – 2018400201

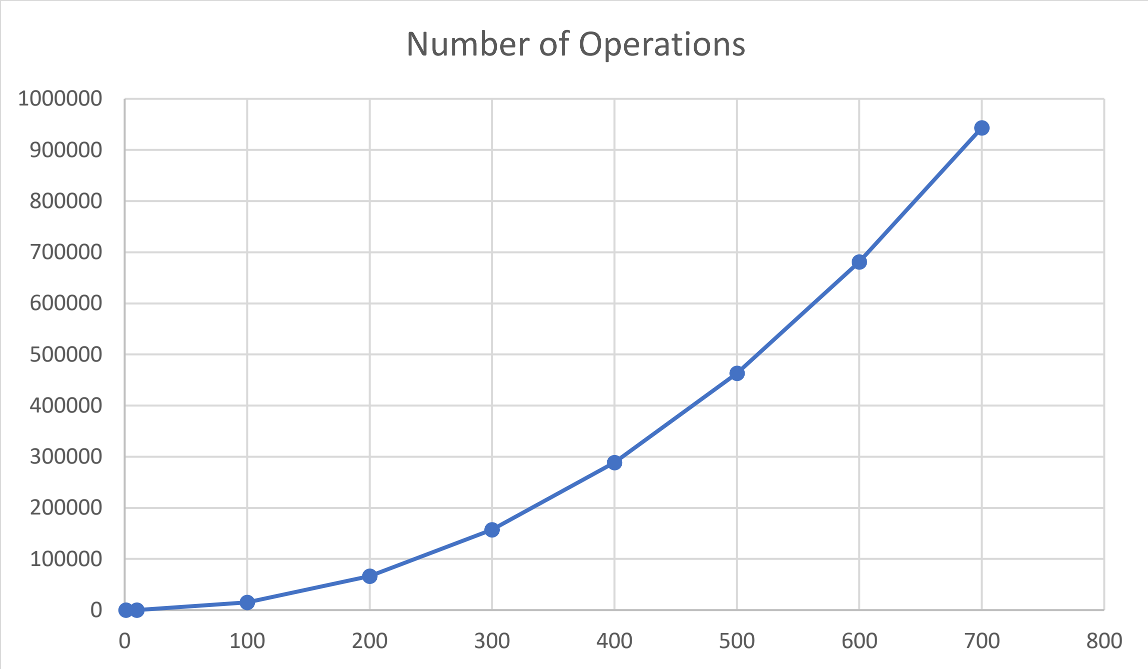


Graph of the theoretical analysis when basic operation is the operation marked as (3)



Graph of the theoretical analysis when basic operation is the operation marked as (4)

Karahan Sarıtaş – 2018400174
Furkan Özdemir – 2018400201

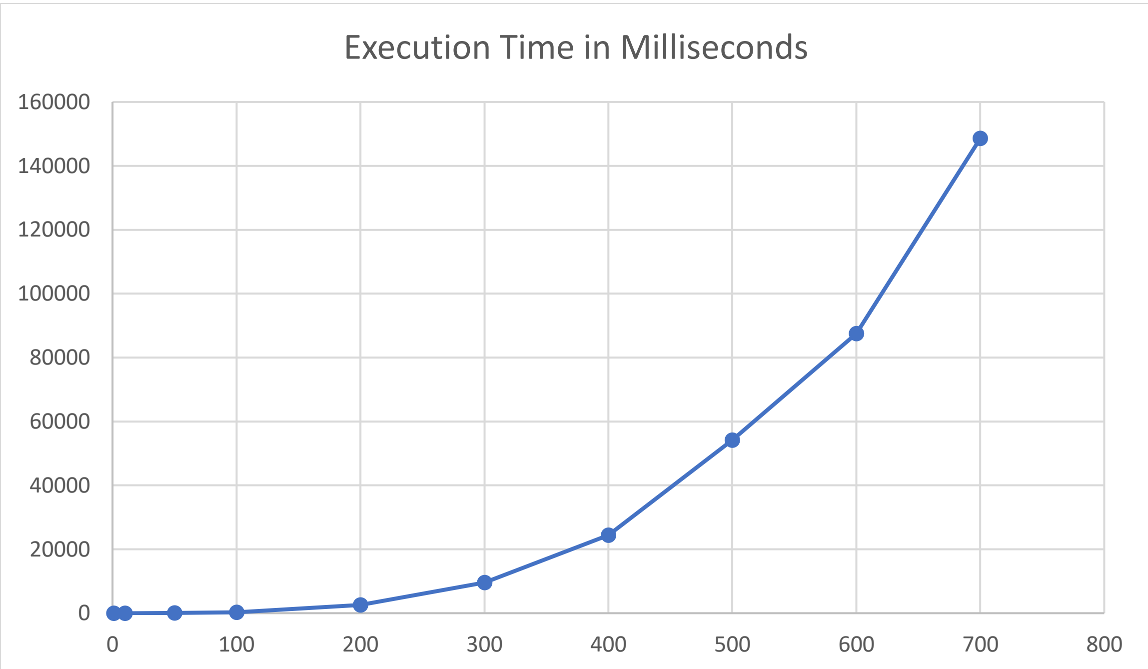


Comments

From the shape of the graph, it’s obvious that operations marked as (1) and (3) are not basic operations. When I compare the ratio of number of operations for two different input size with the ratio of total runtime for two different input size given in the first graph, I observe that basic operation should be the operation marked as (4).

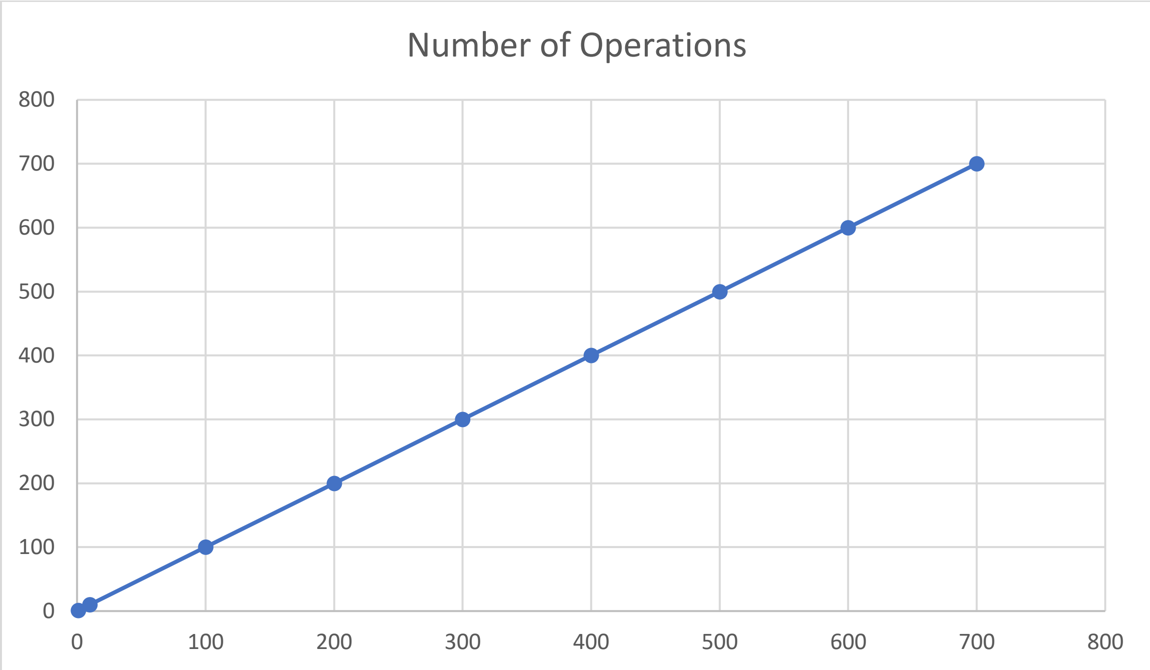
Worst Case

Graph of the real execution time of the algorithm

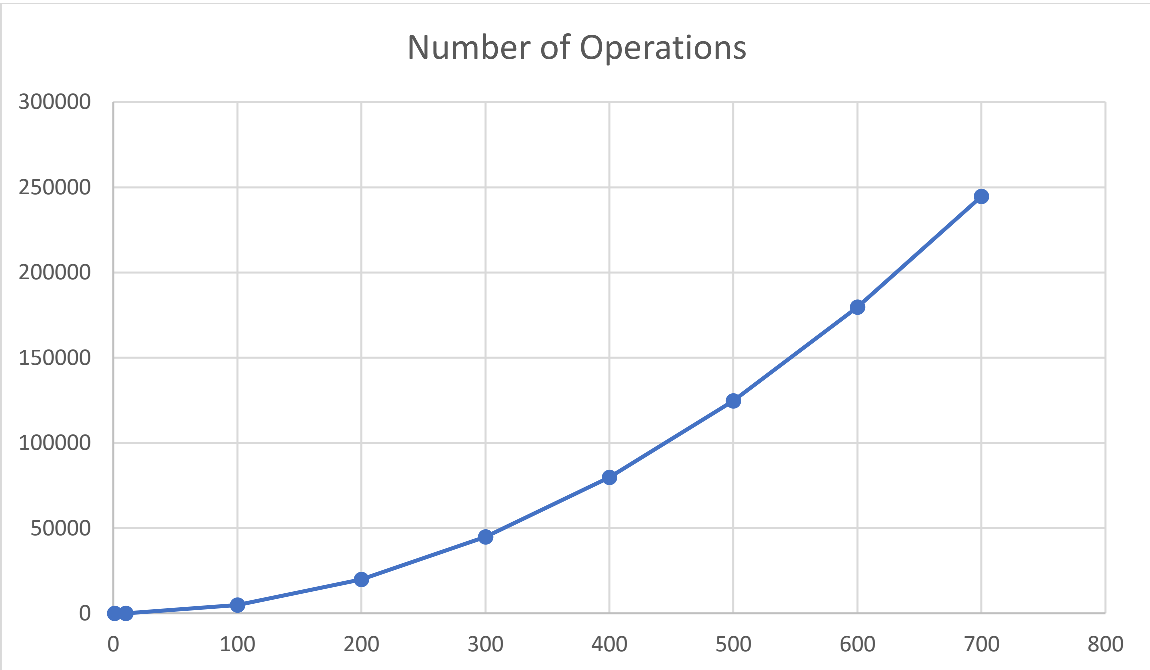


Graph of the theoretical analysis when basic operation is the operation marked as (1)

Karahan Sarıtaş – 2018400174
Furkan Özdemir – 2018400201

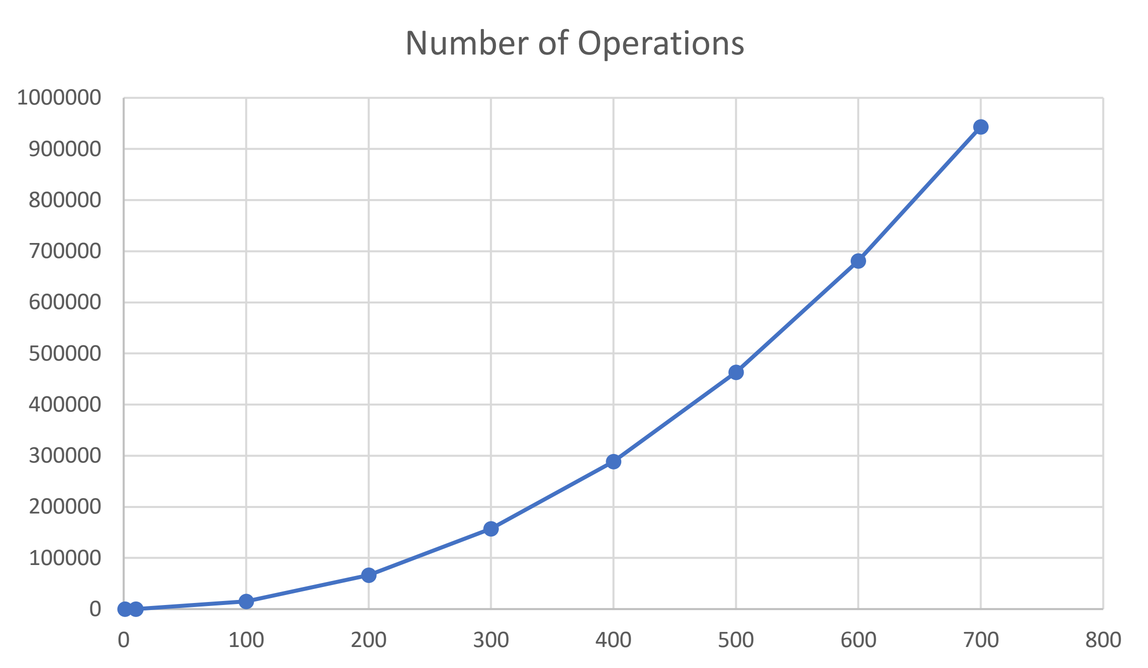


Graph of the theoretical analysis when basic operation is the operation marked as (2)

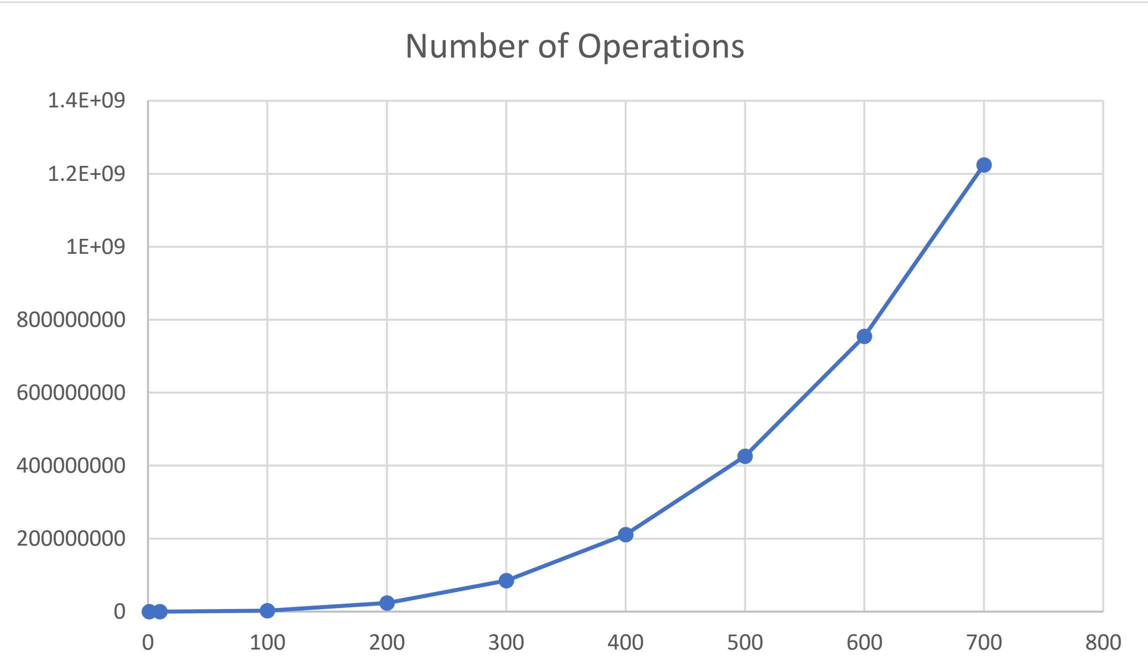


Graph of the theoretical analysis when basic operation is the operation marked as (3)

Karahan Sarıtaş – 2018400174
Furkan Özdemir – 2018400201



Graph of the theoretical analysis when basic operation is the operation marked as (4)



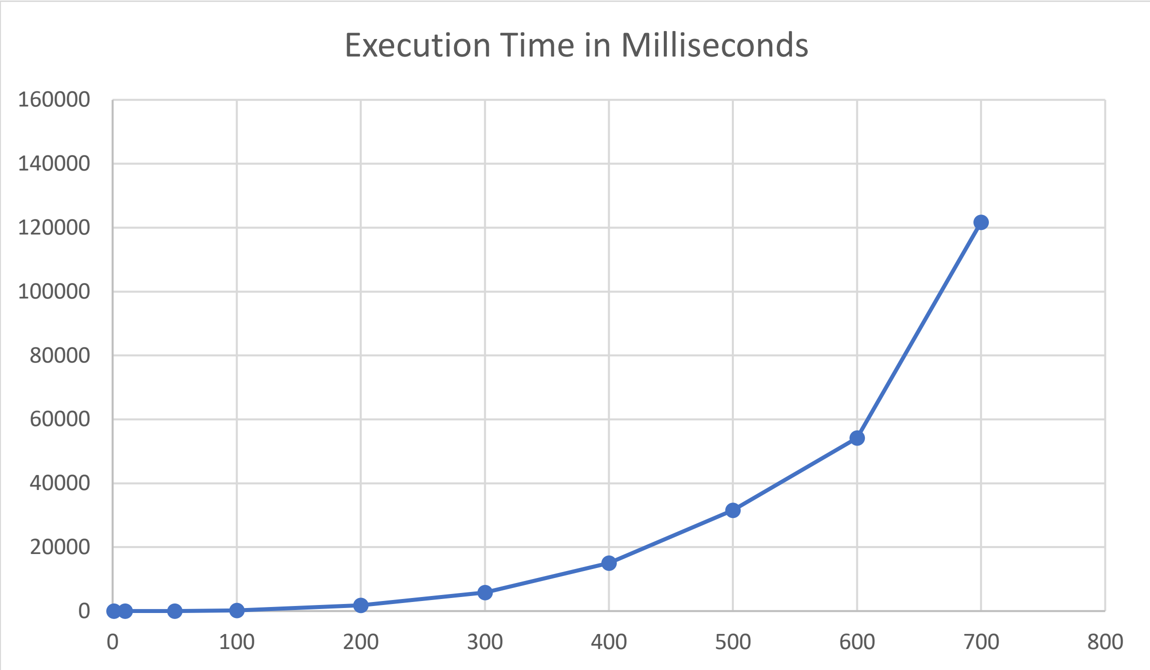
Comments

From the first graph to the fourth, number of basic operations in the theoretical analysis increases. When I compare the number of operations of each graph with the real execution time, I can conclude that basic operation should be the operation marked as (4).

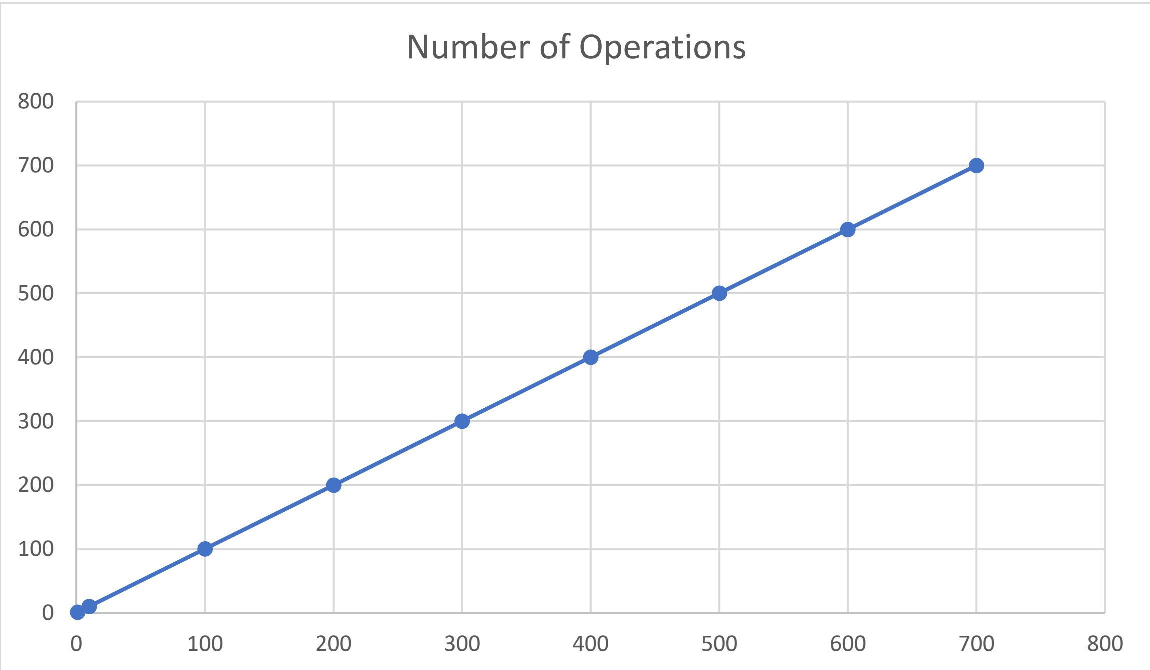
Average Case

Graph of the real execution time of the algorithm

Karahan Sarıtaş – 2018400174
Furkan Özdemir – 2018400201

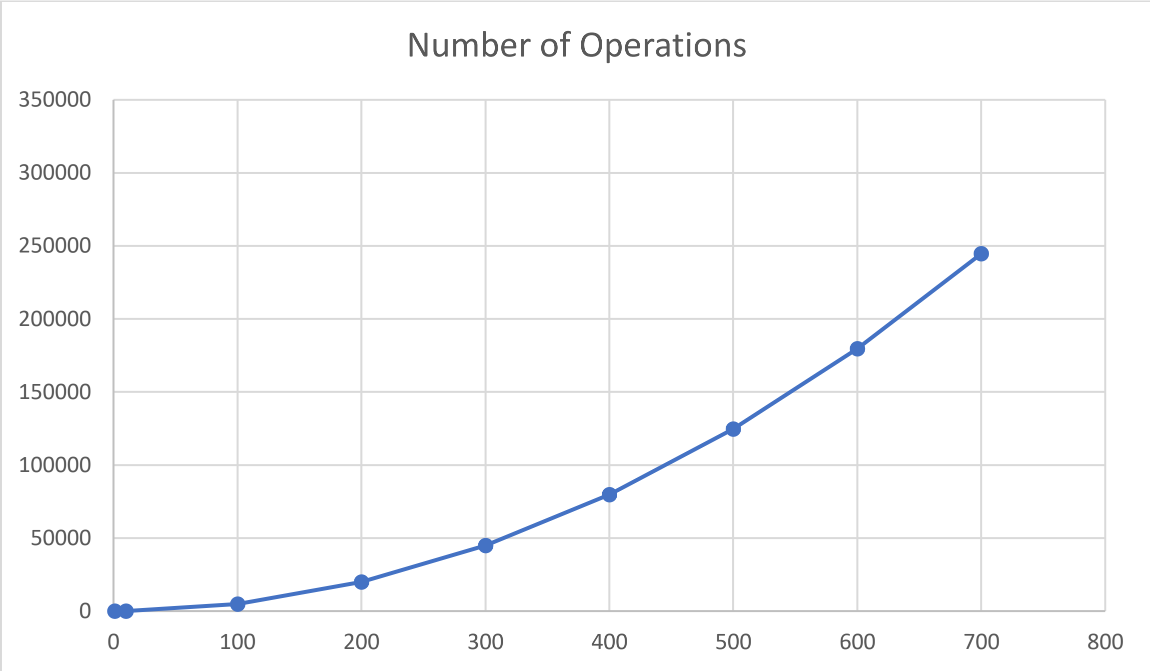


Graph of the theoretical analysis when basic operation is the operation marked as (1)

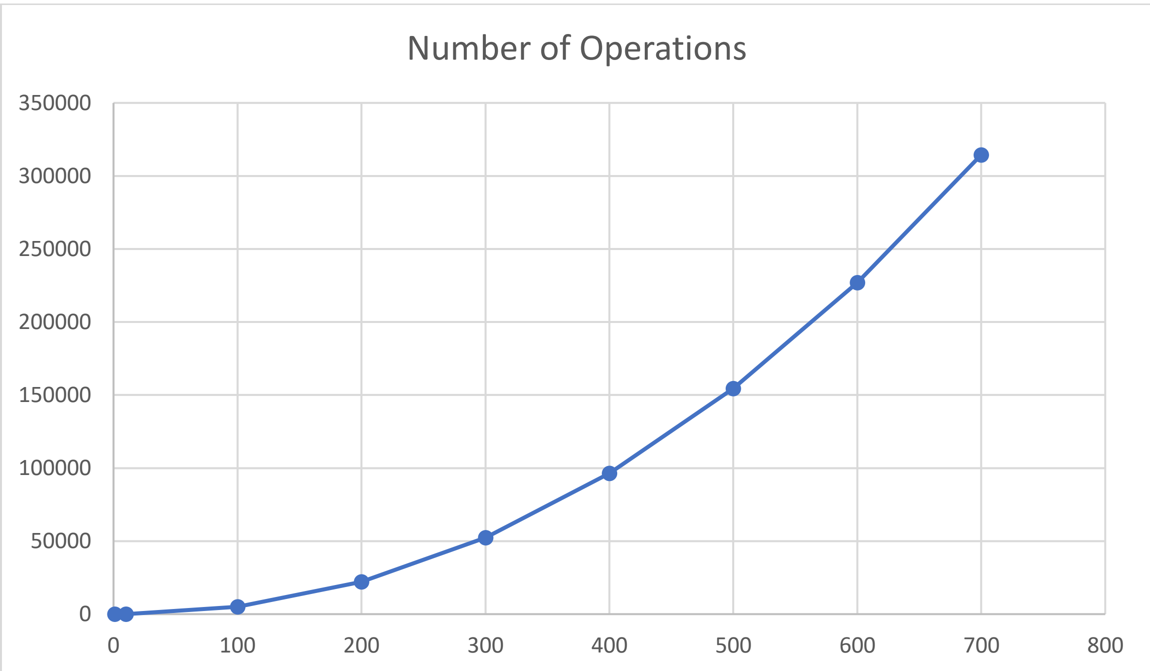


Graph of the theoretical analysis when basic operation is the operation marked as (2)

Karahan Sarıtaş – 2018400174
Furkan Özdemir – 2018400201

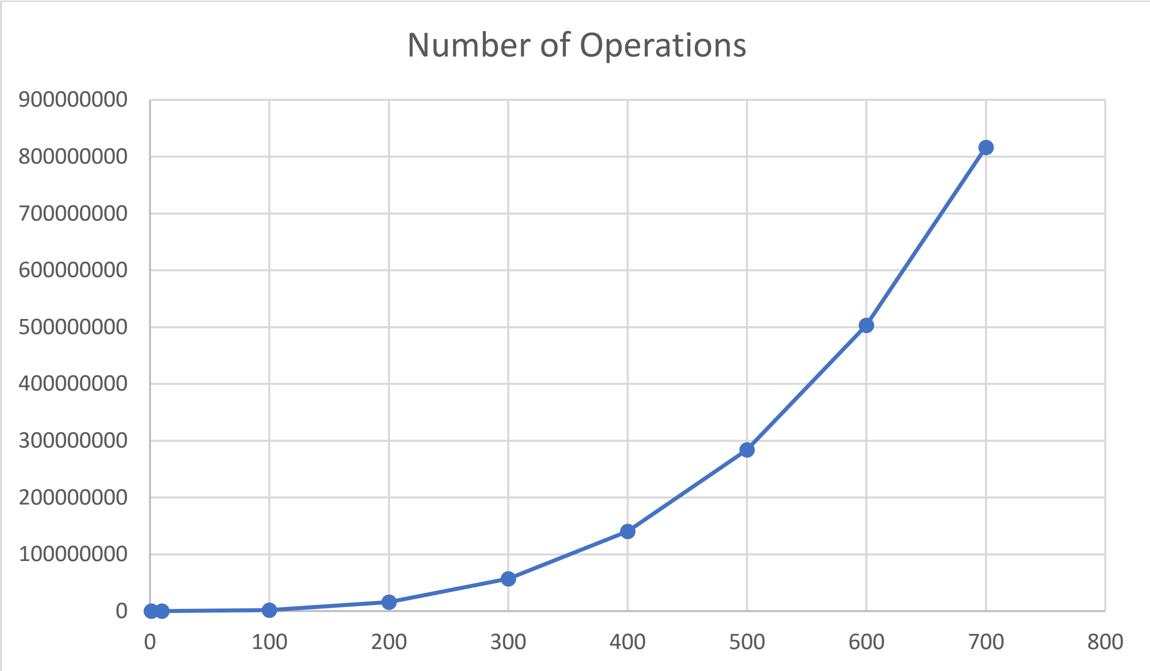


Graph of the theoretical analysis when basic operation is the operation marked as (3)



Graph of the theoretical analysis when basic operation is the operation marked as (4)

Karahan Sarıtaş – 2018400174
Furkan Özdemir – 2018400201



Comments

In average, number of operations performed, and real runtime are similar to those observed in the worst-case analysis. It's because of the probability distribution. In our analysis, we stated that $X[i] = 1$ leads to the worst-case, whereas $X[i] = 0$ leads to the best case. Since the probability of the latter is less than the probability of the former, it's reasonable to get an average runtime closer to the worst-case runtime.