Ömer Şükrü Uyduran – 2018400234
Bilal Atım – 2019400168

Table of Contents

Ömer Şükrü Uyduran – 2018400234
Bilal Atım – 2019400168

THEORETICAL ANALYSIS

Basic operation is the comparison marked as (1)

Only 1 basic operation is executed at each iteration of the most outer for loop. It depends only on the length of the input. Therefore, regardless of the order of array and the values in the array, n basic operations are executed throughout the algorithm. No matter what the input content is, the number of executed basic operation is n for all cases. Therefore:

Analyze B(n)

B(n) ∈ θ(n)

Analyze W(n)

W(n) ∈ θ(n)

Analyze A(n)

Since for every case the number is the same, their average is also n. A(n) ∈ θ(n)

Basic operations are the three assignments marked as (2)

When the program gets in the first if block, the for loop inside iterates n – i times. Inside this for loop, while loop iterates log(n + 1) + 1 times. So, the basic operation is executed (n – i)*(log(n + 1) + 1) times whenever the program gets in this if block.

When the program gets in the second if block, the outer for loop inside the if statement iterates n times and the inner for loop inside that iterates n times for each iteration of that. The while loop iterates log(n + 1) + 1 times for each. Therefore, the basic operation is executed n^2 *(log(n + 1) + 1) times whenever the program gets in this if block.

When the program gets in the third if statement, the outer for loop in this if block iterates n times and the inner for loop iterates $(t3)^2$ times where t3 is the current iteration number. So, the basic operation is executed $\sum_{k=0}^{n} k^2 = \frac{n*(n+1)*(2n+1)}{6}$ times whenever the program gets in the last if block.

The program may get in only 1 of these three if blocks in each iteration of the most outer loop.

Analyze B(n)

Since even the biggest value of the (n – i)*(log(n+1) + 1) is lower than n^2 *(log(n + 1) + 1) and lower than $\frac{n*(n+1)*(2n+1)}{6}$ when n goes to infinity, the best case is that all elements of the input array is 0. In that case the basic operation is executed $\sum_{k=0}^{n-1}(n-k)*(\log(n+1)+1) = \frac{n*(n+1)*(\log(n+1)+1)}{2} = (\frac{n^2}{2})*\log(n+1) + \frac{n^2}{2} + (\frac{n}{2})*\log(n+1) + \frac{n}{2}$ times in total. By excluding the lower order terms and constant coefficients we get that B(n) ∈ θ($n^2 logn$).

Analyze W(n)

Since $\frac{n*(n+1)*(2n+1)}{6}$ is greater than both than n^2*(log(n + 1) + 1) and the biggest value of (n – i)*(log(n+1) + 1) when n goes to infinity, the worst case is that all elements in the input array is 2. In that case, the basic operation is executed $\frac{n^2*(n+1)*(2n+1)}{6} = \frac{n^4}{3} + \frac{n^3}{2} + \frac{n^2}{6}$ times in total. By excluding the lower order terms and constant coefficients we get that W(n) ∈ θ(n^4).

Ömer Şükrü Uyduran – 2018400234
Bilal Atım – 2019400168

Analyze A(n)

The expected number of executed basic operation is $\sum_{i=0}^{n-1} \sum_{k=B(i)}^{W(i)} k * P(k)$, where B(i) is the best case and W(i) is the worst case for only the ith iteration and P(k) is the probability of k. This expression equals to:

$$\sum_{i=0}^{n-1} (n - i) * [\log(n + 1) + 1] * \frac{1}{3} + (n^2 * log(n + 1) + n^2) * \frac{1}{3} + (\frac{n^3}{3} + \frac{n^2}{2} + \frac{n}{6}) * \frac{1}{3}$$

since the value of the ith index of the input array is equally likely to be one of the values 0, 1, or 2 (1/3 for each). Above, the number of executions of basic operations for each case is explained and they are placed in the equation.

This equation is equal to:

$$\frac{1}{3}[[[log(n + 1) + 1] \sum_{i=0}^{n-1} n - i] + [\frac{n^3}{3} + n^2 * log(n + 1) + \frac{3n^2}{2} + \frac{n}{6}] \sum_{i=0}^{n-1} 1]$$

Also, since $\sum_{i=0}^{n-1} n - i = \sum_{i=0}^{n-1} n - \sum_{i=0}^{n-1} i = \frac{n^2 + n}{2}$, and $\sum_{i=0}^{n-1} 1 = n$; this equals to:

$$\frac{n^4}{9} + \frac{n^3 log(n + 1)}{3} + \frac{n^3}{2} + \frac{n^2 log(n + 1)}{6} + \frac{2n^2}{9} + \frac{n log(n + 1)}{6} + \frac{n}{6}$$

When we exclude the lower order terms and constant coefficients, we get that: $A(n) \in \theta(n^4)$.

Basic operation is two assignments marked as (3)

When the program gets in the first if block, basic operation isn't executed anytime. Therefore, the basic operation is executed 0 times whenever the program gets in this if block.

When the program gets in the second if block, the outer for loop inside the if statement iterates n times and the inner for loop inside that iterates n times for each iteration of that. The while loop iterates log(n + 1) + 1 times for each. Therefore, the basic operation is executed n^2 *(log(n + 1) + 1) times whenever the program gets in this if block.

When the program gets in the third if statement, the outer for loop in this if block iterates n times and the inner for loop iterates $t3^2$ times where t3 is the iteration number. So, the basic operation is executed $\sum_{k=0}^{n} k^2 = \frac{n*(n + 1)*(2n + 1)}{6}$ times whenever the program gets in the last if block.

The program may get in only 1 of these three if blocks in each iteration of the most outer loop.

Analyze B(n)

Since 0 is lower than n^2 *(log(n + 1) + 1) and lower than $\frac{n*(n + 1)*(2n + 1)}{6}$ when n goes to infinity, the best case is that all elements of the input array is 0. In that case the basic operation is executed 0 times in total. Since it is constant, $B(n) \in \theta(1)$.

Analyze W(n)

Since $\frac{n*(n + 1)*(2n + 1)}{6}$ is greater than both than n^2*(log(n + 1) + 1) and the biggest value of (n – i)*(log(n+1) + 1) when n goes to infinity, the worst case is that all elements in the input array is 2. In that case, the basic operation is executed $\frac{n^2*(n + 1)*(2n + 1)}{6} = \frac{n^4}{3} + \frac{n^3}{2} +$

$\frac{n^2}{6}$ times in total. By excluding the lower order terms and constant coefficients we get that W(n) ∈ θ(n^4).

Analyze A(n)

The expected number of executed basic operation is $\sum_{i=0}^{n-1} \sum_{k=B(i)}^{W(i)} k * P(k)$, where B(i) is the best case and W(i) is the worst case for only the ith iteration and P(k) is the probability of k. This equals to:

$$\sum_{i=0}^{n-1} [0 * \frac{1}{3} + (n^2 * log(n+1) + n^2) * \frac{1}{3} + (\frac{n^3}{3} + \frac{n^2}{2} + \frac{n}{6}) * \frac{1}{3}]$$

since the value of the ith index of the input array is equally likely to be one of the values 0, 1, or 2 (1/3 for each). Above, the number of executions of basic operations for each case is explained and they are placed in the equation.

This equation is equal to:

$$\frac{1}{3}[\frac{n^3}{3} + n^2 * log(n+1) + \frac{3n^2}{2} + \frac{n}{6}] \sum_{i=0}^{n-1} 1$$

Also, since $\sum_{i=0}^{n-1} 1$ = n; this equals to:

$$\frac{n^4}{9} + \frac{n^3 log(n+1)}{3} + \frac{n^3}{2} + \frac{n^2}{18}$$

When we exclude the lower order terms and constant coefficients, we get that: A(n) ∈ θ(n^4).

Basic operations are the two loop incrementations marked as (4)

When the program gets in the first if block, basic operation isn't executed anytime. Therefore, the basic operation is executed 0 times whenever the program gets in this if block.

When the program gets in the second if block, the outer for loop inside the if statement iterates n times and the inner for loop inside that iterates n times for each iteration of that and 1 basic operation is executed for each. Therefore, the basic operation is executed n^2 times whenever the program gets in this if block.

When the program gets in the third if statement, the outer for loop in this if block iterates n times and 1 basic operation is executed for each. So, the basic operation is executed n times whenever the program gets in the last if block.

The program may get in only 1 of these three if blocks in each iteration of the most outer loop.

Analyze B(n)

Since even 0 is lower than n^2 *(log(n + 1) + 1) and lower than $\frac{n*(n+1)*(2n+1)}{6}$ when n goes to infinity, the best case is that all elements of the input array is 0. In that case the basic operation is executed 0 times in total. Since it is constant, B(n) ∈ θ(1).

Analyze W(n)

Since n^2 is greater than both n and 0 when n goes to infinity, the worst case is that all elements in the input array is 1. In that case, the basic operation is executed n^3 times in total. By excluding the lower order terms and constant coefficients we get that W(n) ∈ θ(n^3).

Ömer Şükrü Uyduran – 2018400234
Bilal Atım – 2019400168

Analyze A(n)

The expected number of executed basic operation is $\sum_{i=0}^{n-1} \sum_{k=B(i)}^{W(i)} k * P(k)$, where B(i) is the best case and W(i) is the worst case of the ith iteration and P(k) is the probability of k. This equals to:

$$\sum_{i=0}^{n-1} 0 * \frac{1}{3} + n^2 * \frac{1}{3} + n * \frac{1}{3}$$

And this is equal to $\frac{n^3}{3} + \frac{n^2}{3}$. When we exclude the lower order terms and constant coefficients, we get that: A(n) ∈ θ(n^3).

Basic operation is the assignment marked as (5)

When the program gets in the first if block, the for loop inside iterates n – i times. The basic operation is executed only 1 time for each iteration. So, the basic operation is executed n – i times whenever the program gets in this if block.

When the program gets in the second and third if block, basic operation isn't executed anytime. Therefore, the basic operation is executed 0 times whenever the program gets in these if blocks.

The program may get in only 1 of these three if blocks in each iteration of the most outer loop.

Analyze B(n)

Since 0 is lower than $the\ smallest\ value\ of\ (n - i)$ when n goes to infinity, the best case is that all elements of the input array is 1 or 2. In that case the basic operation is executed 0 times in total. Since it is constant, B(n) ∈ θ(1).

Analyze W(n)

Since even the smallest value of (n – i) is greater than 0 when n goes to infinity, the worst case is that all elements of the input array is 0. In that case the basic operation is executed $\sum_{i=0}^{n-1} n - i = \sum_{i=0}^{n-1} n - \sum_{i=0}^{n-1} i = \frac{n^2}{2} + \frac{n}{2}$ times and by excluding the constant coefficients and lower order terms we get that W(n) ∈ θ(n^2).

Analyze A(n)

The expected number of executed basic operation is $\sum_{i=0}^{n-1} \sum_{k=B(i)}^{W(i)} k * P(k)$, where B(i) is the best case and W(i) is the worst case of the ith iteration and P(k) is the probability of k. This equals to:

$$\sum_{i=0}^{n-1} (n - i) * \frac{1}{3} + 0 * \frac{2}{3}$$

since the value of the ith index of the input array is equally likely to be one of the values 0, 1, or 2 (1/3 for each). Above, the number of executions of basic operations for each case is explained and they are placed in the equation.

This equation is equal to: $\frac{1}{3}\sum_{i=0}^{n-1} n - i$. And also, since $\sum_{i=0}^{n-1} n - i = \sum_{i=0}^{n-1} n - \sum_{i=0}^{n-1} i = \frac{n^2+n}{2}$ this equals to:

Ömer Şükrü Uyduran – 2018400234
Bilal Atım – 2019400168

$$\frac{n^2}{6} + \frac{n}{6}$$

When we exclude the lower order terms and constant coefficients, we get that: A(n) ∈ θ(n^2).

IDENTIFICATION OF BASIC OPERATION(S)

Here, state clearly which operation(s) in the algorithm must be the basic operation(s). Also, you should provide a simple explanation about why you have decided on the basic operation you choose. (1-3 sentences)

Basic operations in algorithm must be operations marked as (2) because these operations are the most important operations in the algorithm. Beside that, the most repetitive operations are operations (2) in best, worst and average cases. Also, operations marked as (2) also contributes most to the total execution time.

REAL EXECUTION

Best Case

N Size	Time Elapsed
1	0 ns
5	0 ns
10	0 ns
25	0 ns
50	1001100 ns
75	1999700 ns
100	4004900 ns
150	18749300 ns
200	35446800 ns
250	41975100 ns

Worst Case

N Size	Time Elapsed
1	0 ns
5	0 ns
10	0 ns
25	13014100 ns
50	167997300 ns
75	809961100 ns
100	2559300000 ns
150	12652156800 ns
200	40145051400 ns
250	97347465100 ns

Ömer Şükrü Uyduran – 2018400234
Bilal Atım – 2019400168

Average Case

N Size	Time Elapsed
1	0 ns
5	0 ns
10	332766 ns
25	6996266 ns
50	78012600 ns
75	463710166 ns
100	1127769700 ns
150	5796366400 ns
200	16181681500 ns
250	35625950033 ns

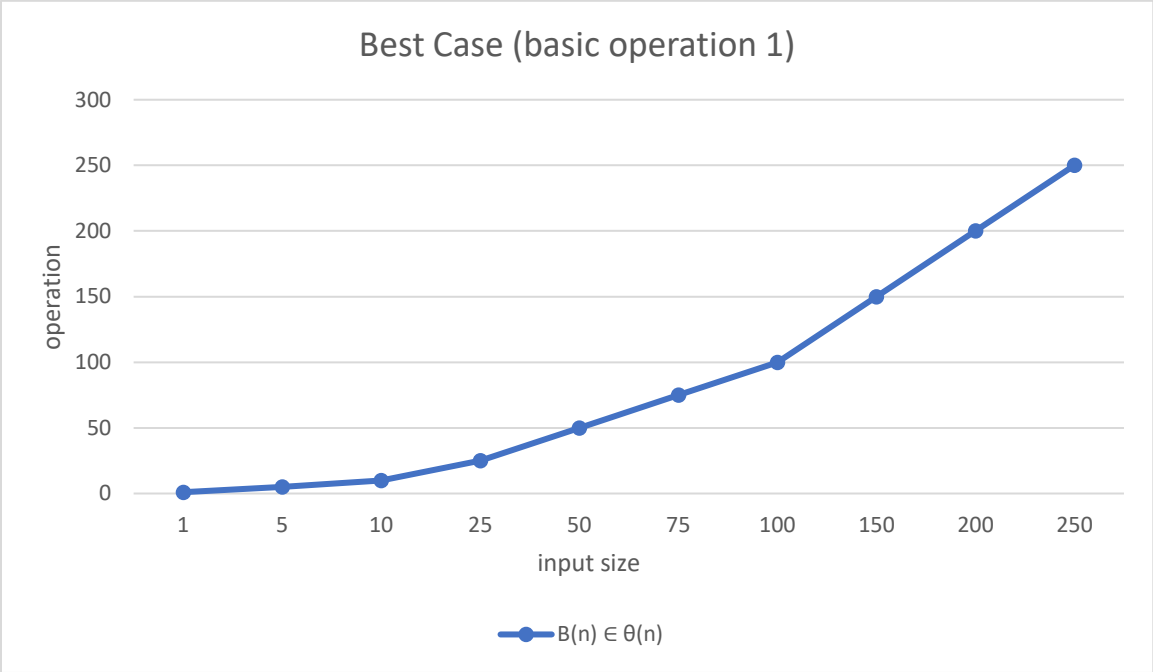
COMPARISON

Best Case

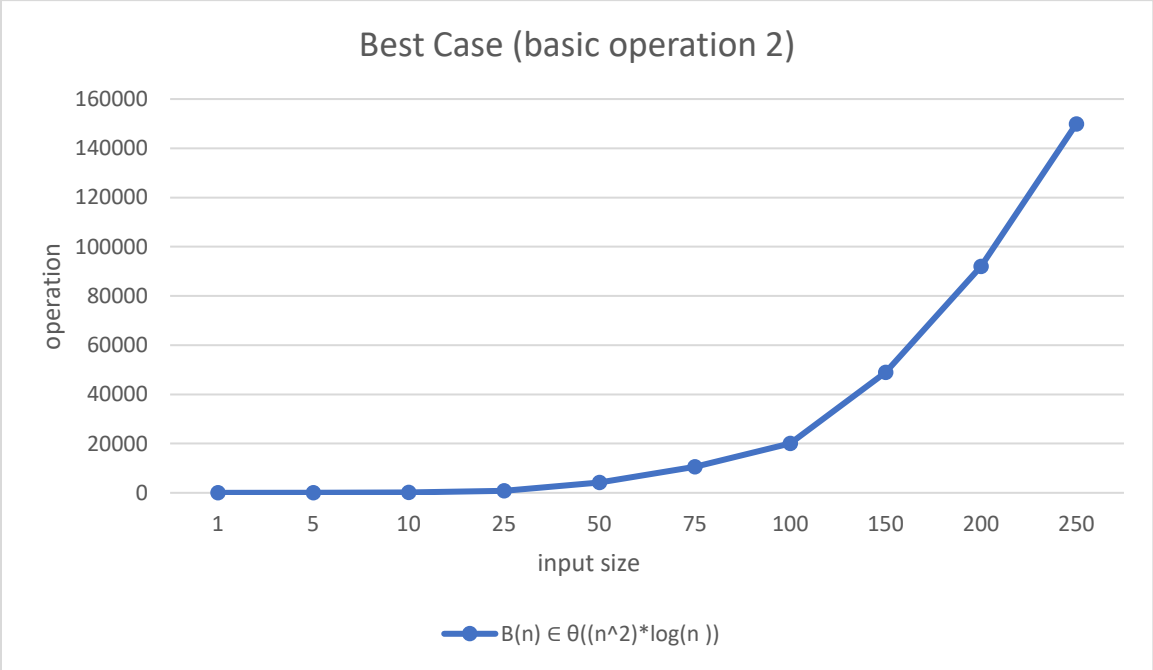
Graph of the real execution time of the algorithm

Ömer Şükrü Uyduran – 2018400234
Bilal Atım – 2019400168

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

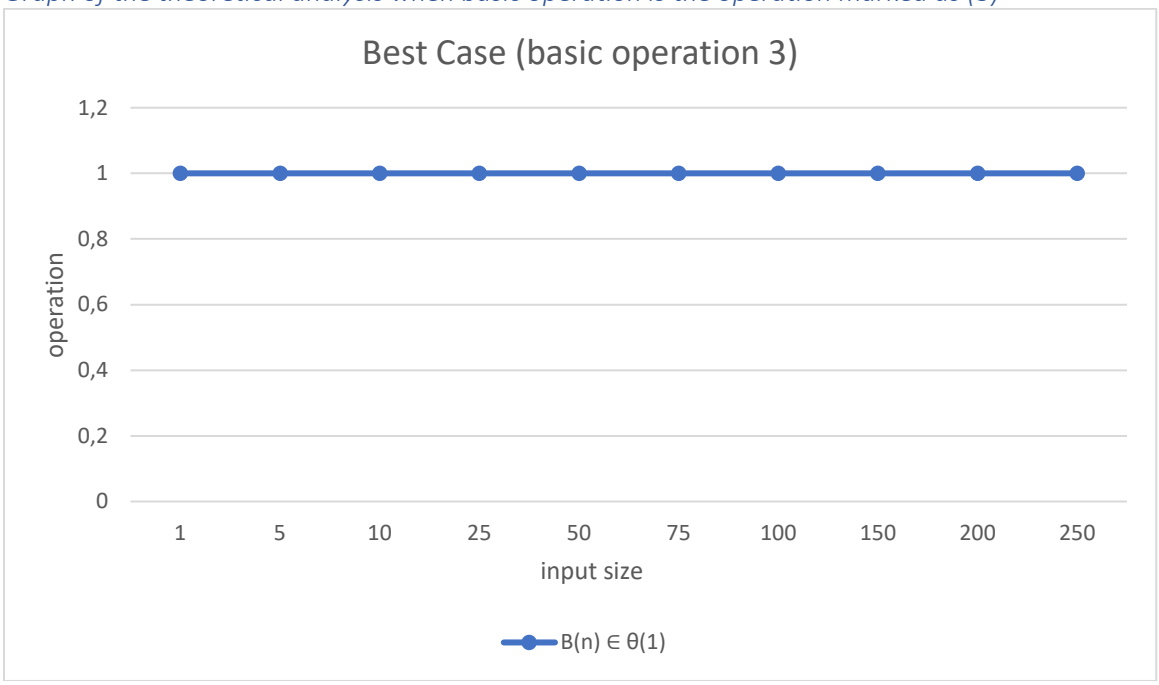


Graph of the theoretical analysis when basic operation is the operation marked as (2)

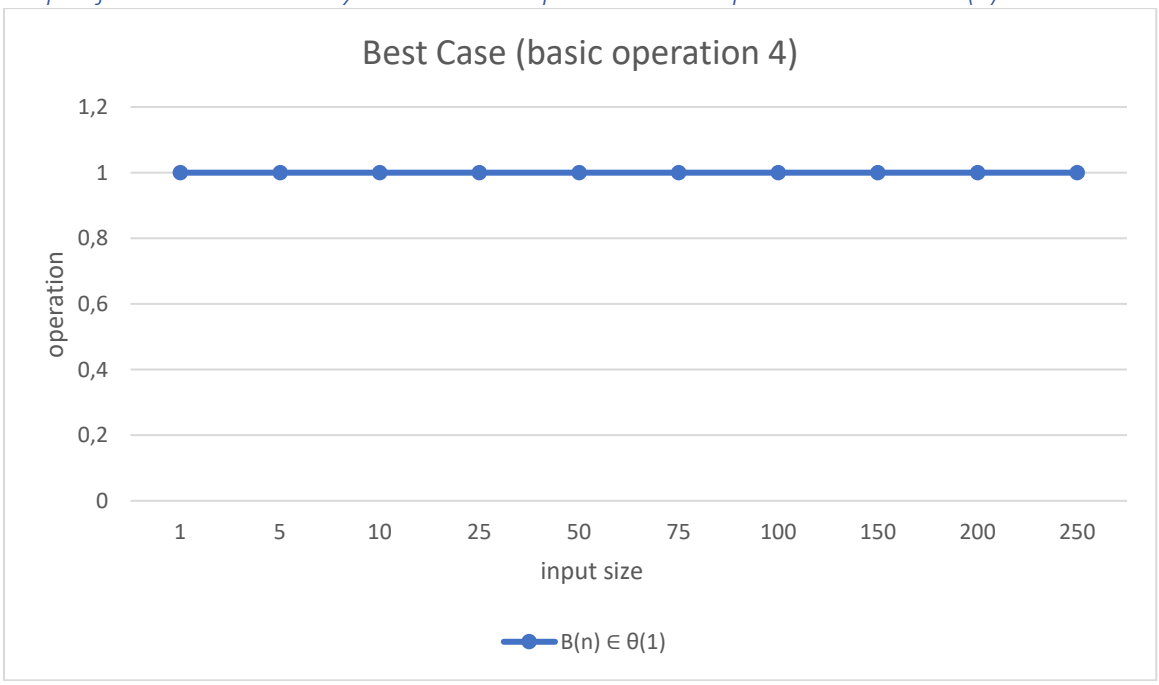


Ömer Şükrü Uyduran – 2018400234
Bilal Atım – 2019400168

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

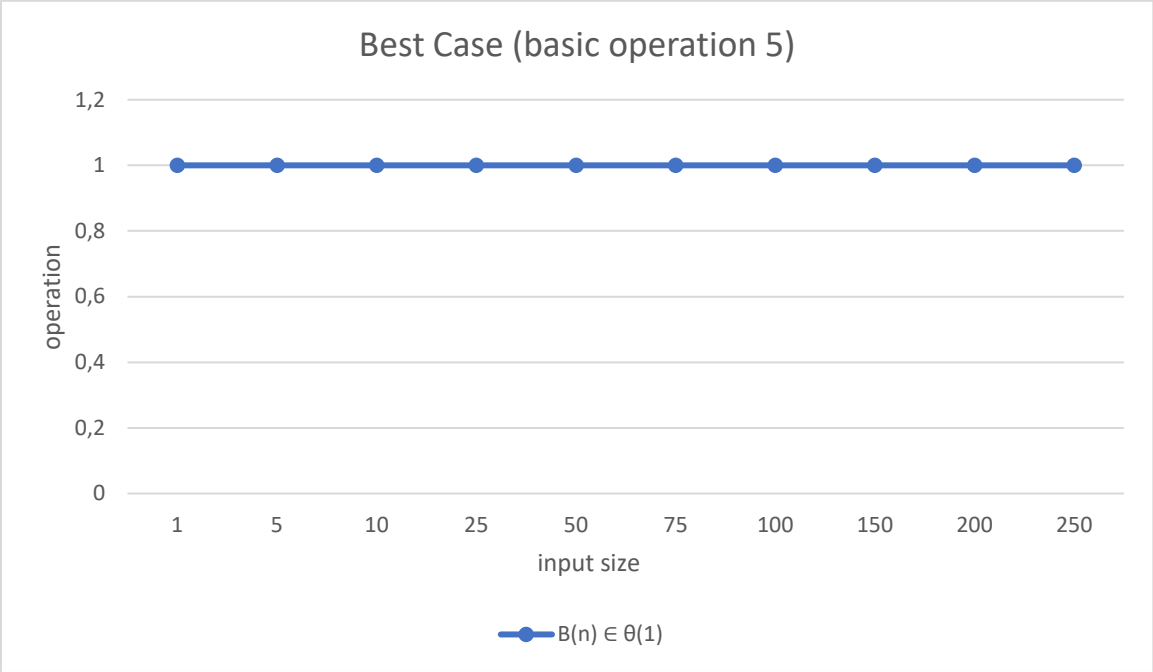


Graph of the theoretical analysis when basic operation is the operation marked as (4)



Ömer Şükrü Uyduran – 2018400234
Bilal Atım – 2019400168

Graph of the theoretical analysis when basic operation is the operation marked as (5)



Comments

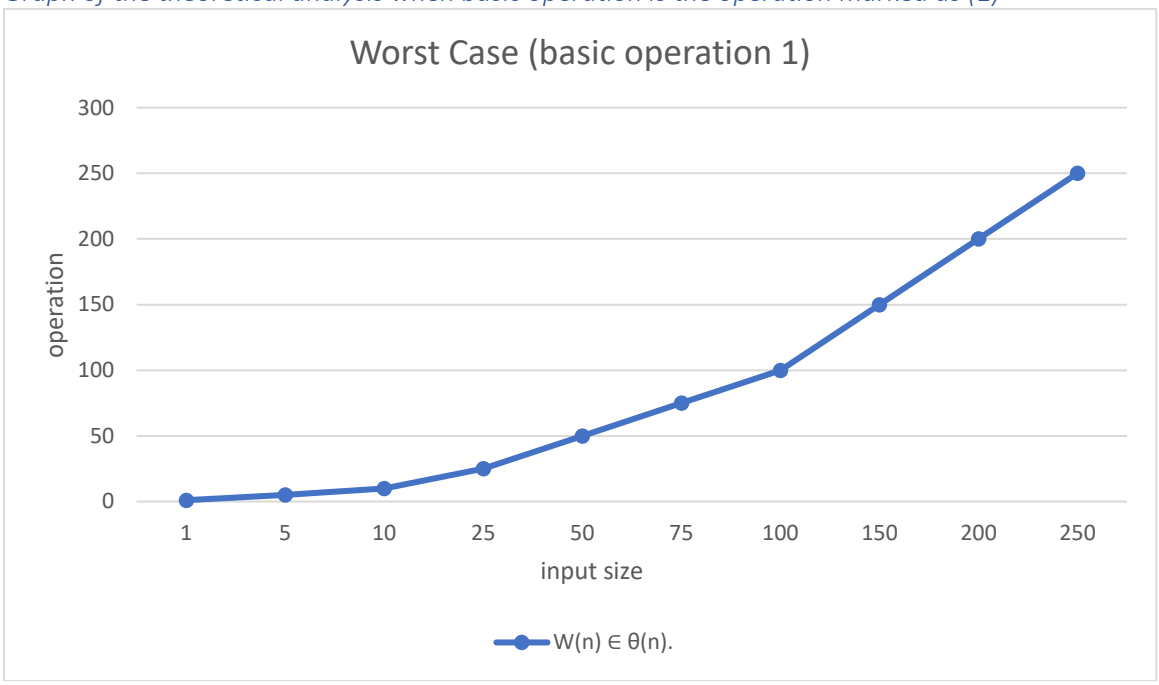
When we analyze the graphs, the graphs of operation marked as (1) and (2) are close to the real execution graph. The shape of the graphs are similar to the real execution time graph, however the graph of basic operation marked as (2) is closer to the real execution graph in terms of the values too. The other graphs are far from being similar to the real execution graph both in terms of the shape and the values.

Worst Case

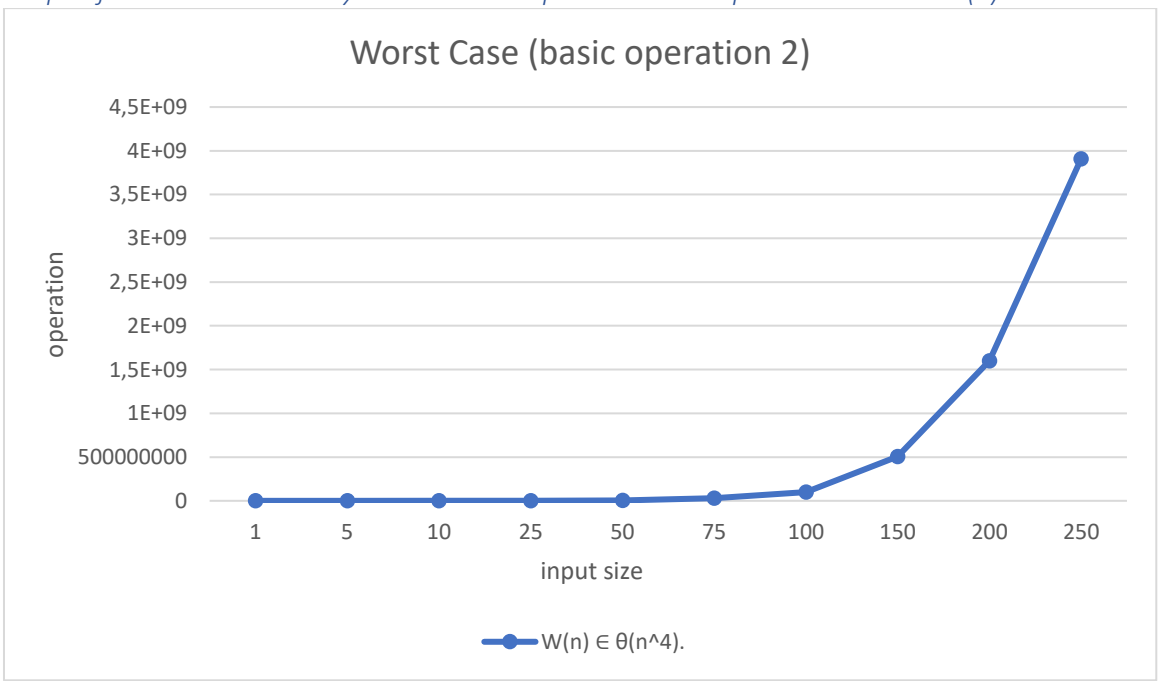
Graph of the real execution time of the algorithm

Ömer Şükrü Uyduran – 2018400234
Bilal Atım – 2019400168

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

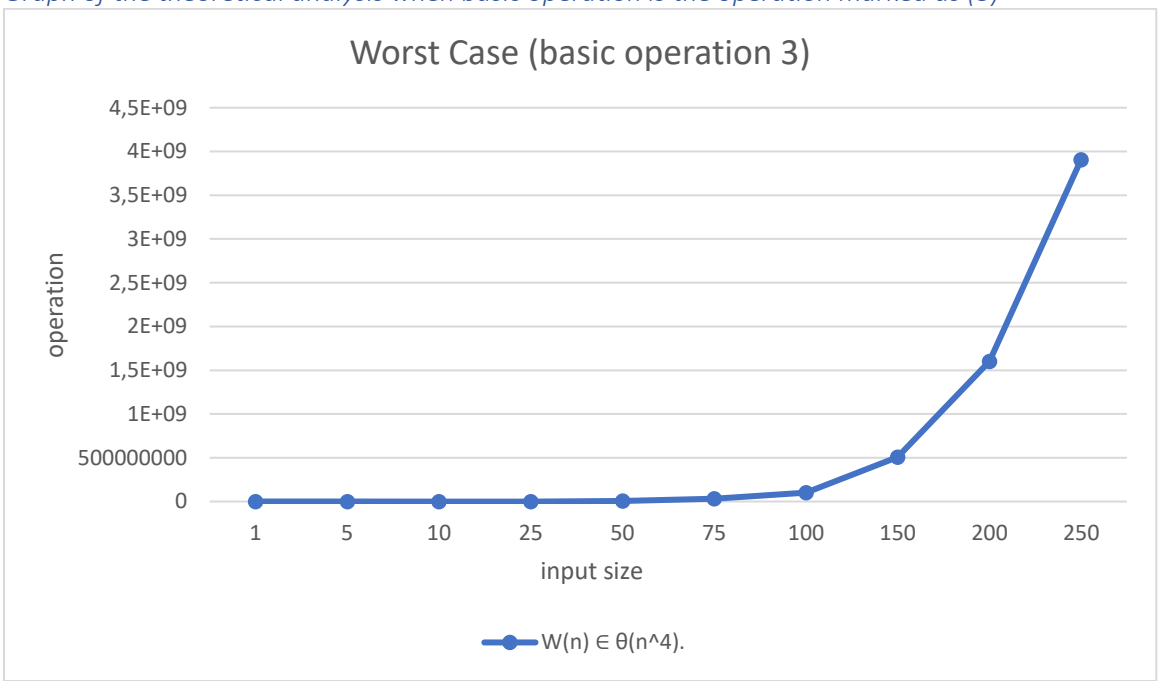


Graph of the theoretical analysis when basic operation is the operation marked as (2)

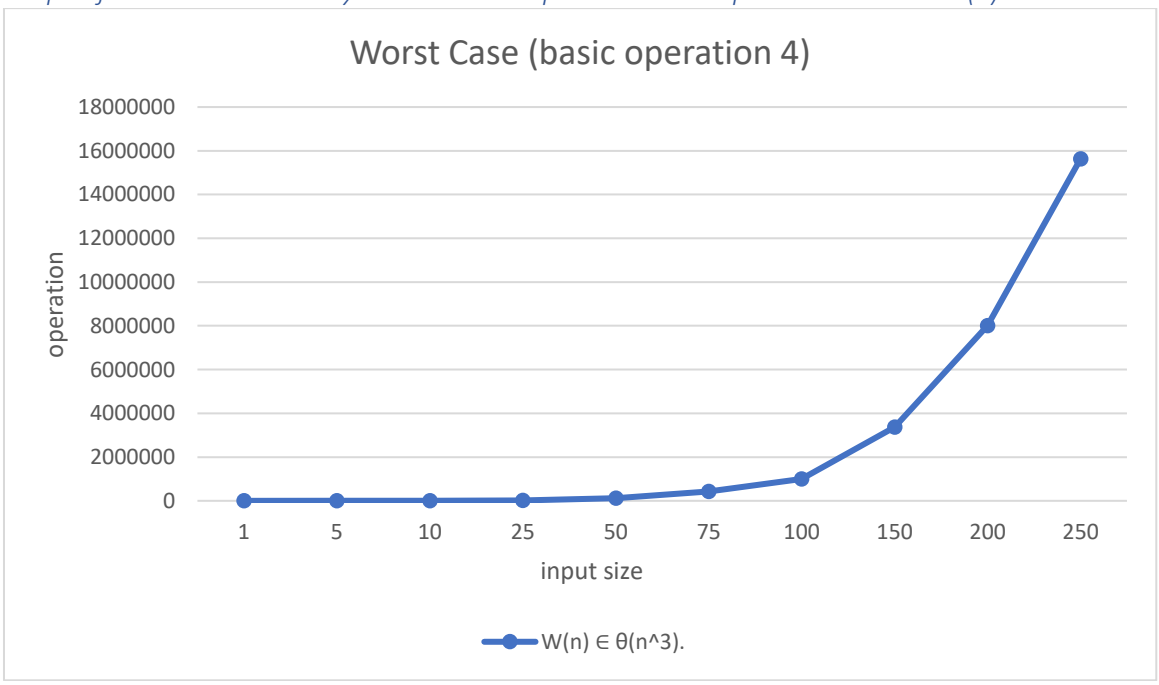


Ömer Şükrü Uyduran – 2018400234
Bilal Atım – 2019400168

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

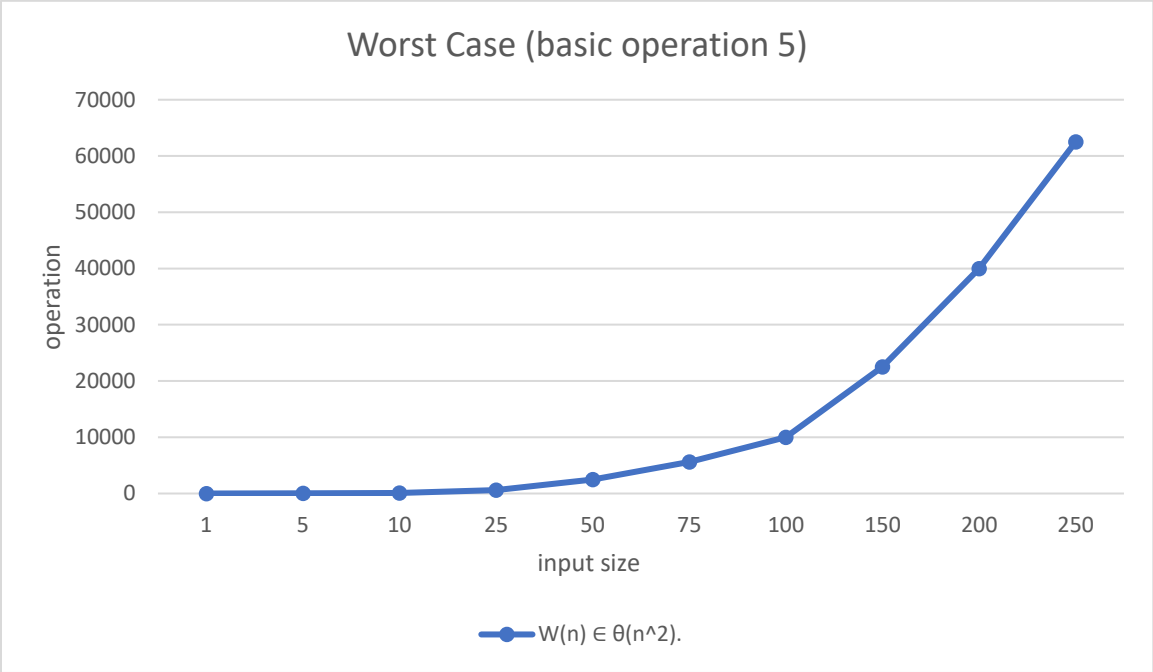


Graph of the theoretical analysis when basic operation is the operation marked as (4)



Ömer Şükrü Uyduran – 2018400234
Bilal Atım – 2019400168

Graph of the theoretical analysis when basic operation is the operation marked as (5)



Comments

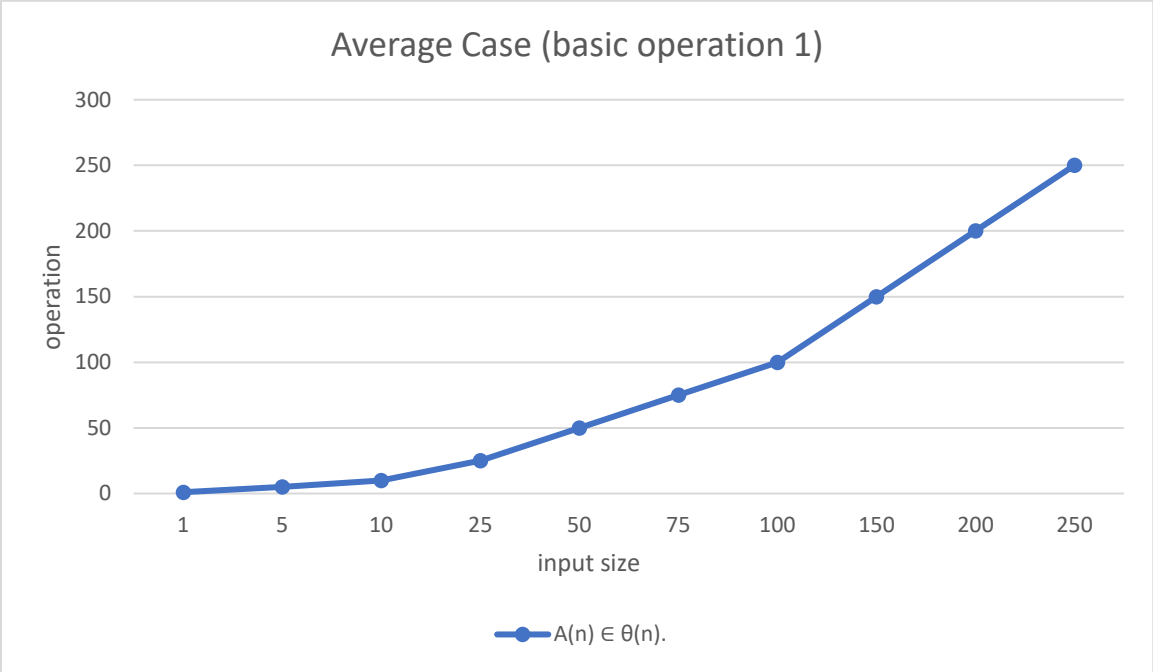
All graphs other than the one whom belong to operation marked as 1 are similar to the real execution graph in terms of the shape. However only the graphs of operations marked as (2) and (3) are close to the real execution graph in terms of values. This situation supports operation (2) to be a the basic operation.

Average Case

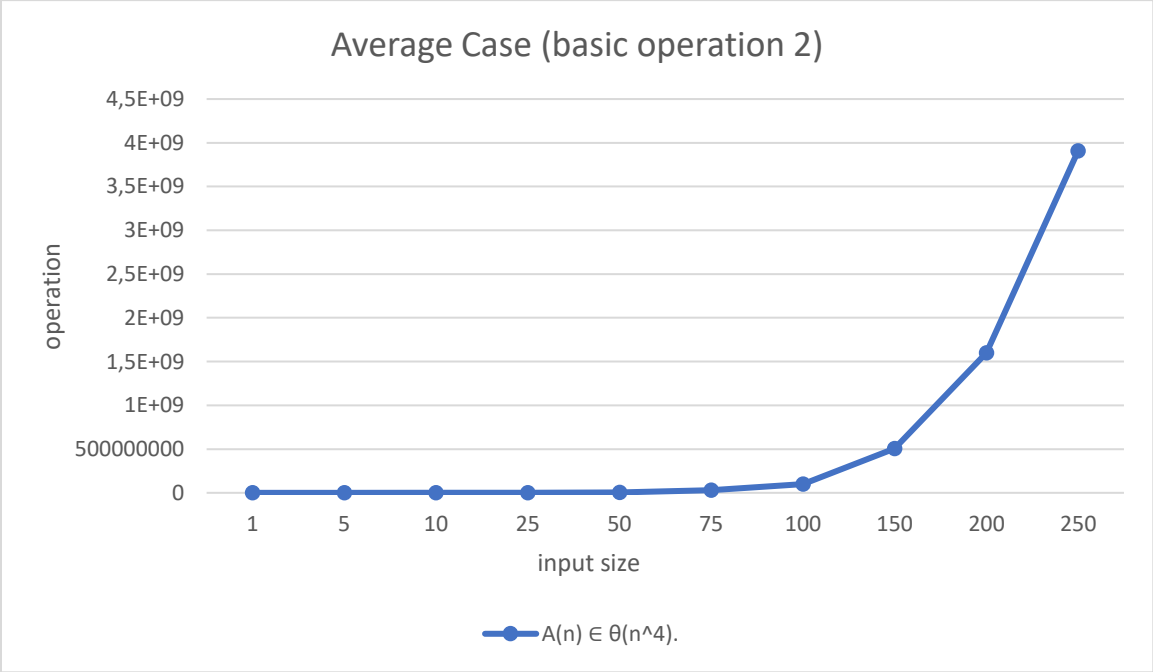
Graph of the real execution time of the algorithm

Ömer Şükrü Uyduran – 2018400234
Bilal Atım – 2019400168

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

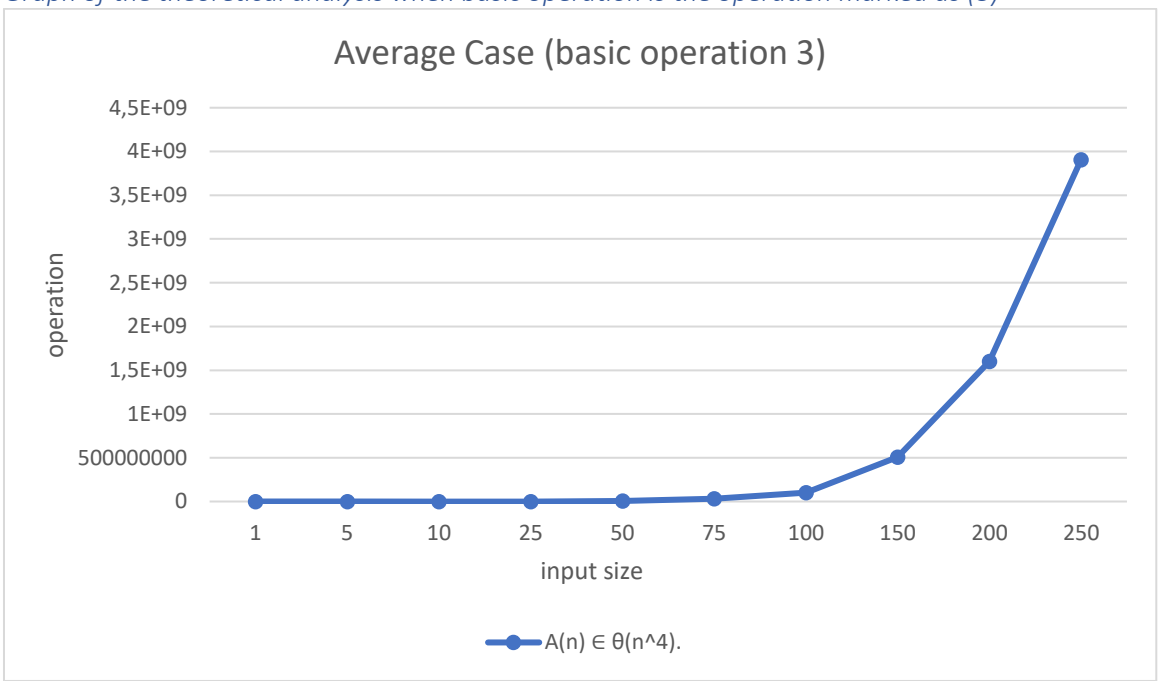


Graph of the theoretical analysis when basic operation is the operation marked as (2)

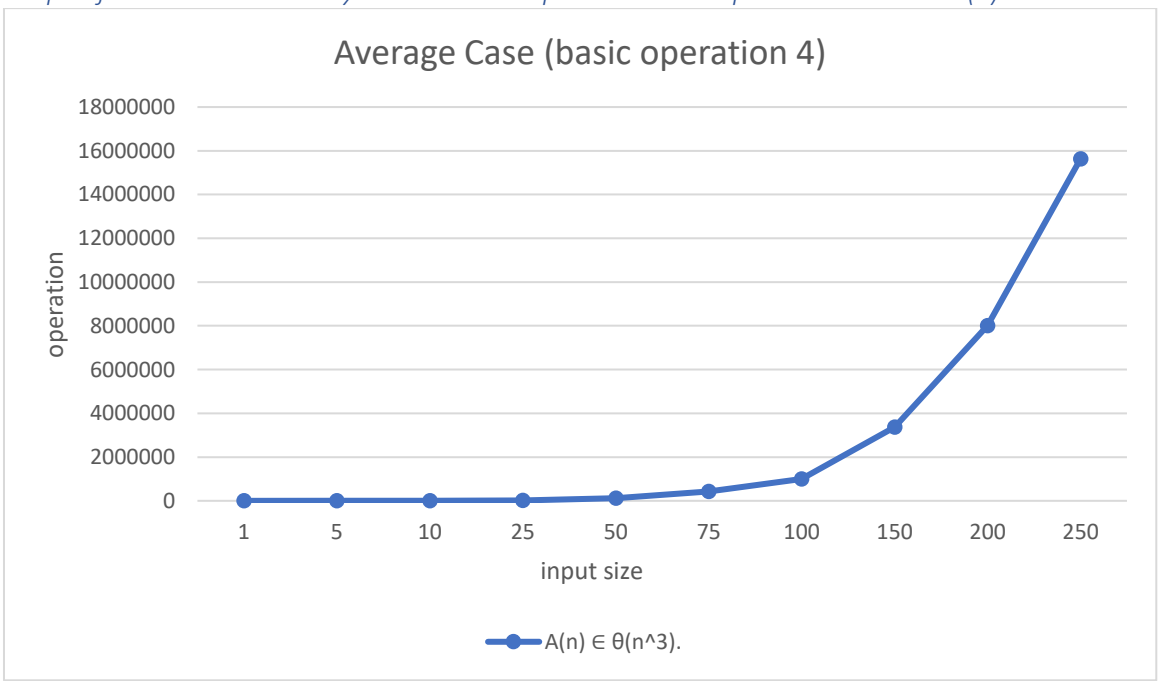


Ömer Şükrü Uyduran – 2018400234
Bilal Atım – 2019400168

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

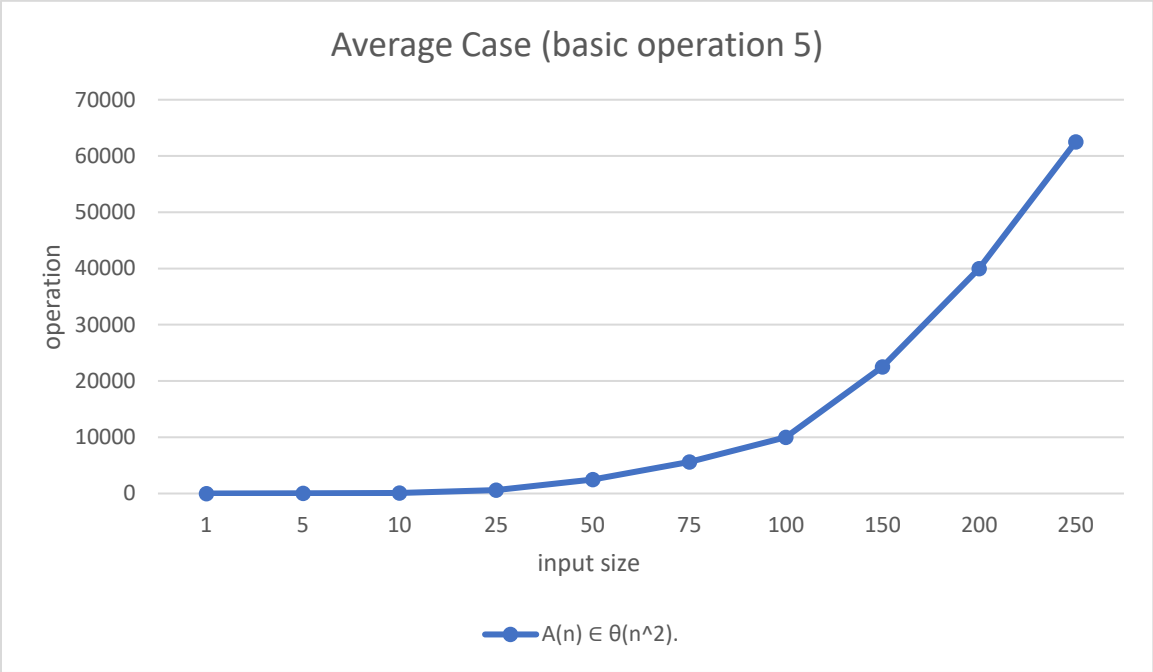


Graph of the theoretical analysis when basic operation is the operation marked as (4)



Ömer Şükrü Uyduran – 2018400234
Bilal Atım – 2019400168

Graph of the theoretical analysis when basic operation is the operation marked as (5)



Comments

All graphs other than the one whom belong to operation marked as 1 are similar to the real execution graph in terms of the shape. However only the graphs of operations marked as (2) and (3) are close to the real execution graph in terms of values. This situation supports operation (2) to be a the basic operation.

When we consider best, worst, and average case; operation (2) is the most suitable choice for the basic operation.