ELİF KIZILKAYA – 2018400108
ÜMMÜ SENA ÖZPINAR – 2019400279

Table of Contents

ELİF KIZILKAYA – 2018400108
ÜMMÜ SENA ÖZPINAR – 2019400279

THEORETICAL ANALYSIS

Basic operation is the comparison marked as (1)

Analyze B(n)

Let T_n be the set of all inputs of size n and τ(I) be the number of basic operations for input I. For all input I ∈ T_n, there are $\sum_{i=0}^{n-1} 1$ operations. Best case input I is also element of this set. Thus, B(n) = $\sum_{i=0}^{n-1} 1$ = n. B(n) ∈ Θ(n).

Analyze W(n)

Let T_n be the set of all inputs of size n and τ(I) be the number of basic operations for input I. For all input I ∈ T_n, there are $\sum_{i=0}^{n-1} 1$ operations. Worst case input I is also element of this set. Thus, W(n) = $\sum_{i=0}^{n-1} 1$ = n. W(n) ∈ Θ(n).

Analyze A(n)

Let T_n be the set of all inputs of size n and τ(I) be the number of basic operations for input I. For all input I ∈ T_n, there are $\sum_{i=0}^{n-1} 1$ operations. Thus, we can find A(n) using the formula A(n) = $\sum_{I \in T_n} \tau(I) * p(I)$ where τ(I) is the number of basic operations for input I as: $\sum_{i=0}^{n-1} 1$ = n. A(n) ∈ Θ(n).

Basic operations are the three assignments marked as (2)

Basic operation (2) is in 3 different, independent places in the algorithm. Let's divide our analysis into 3 parts:

1- If arr[i] = 0,
2- If arr[i] = 1,
3- If arr[i] = 2,

In the first part, the while loop executes when x1 ≥ 1. We know that n = 2^k – 1. Given x1 = n + 1 and x1 becomes $\lfloor x1/2 \rfloor$ at each iteration, we can show that:

x1 = 2^k – 1 + 1 = = 2^k

Since x1 is divided by 2 in each iteration, there must be k+1 iterations. From n = 2^k – 1,

k = $\log_2(n+1)$

Thus, this executes:
$\sum_{i=0}^{n-1} \sum_{j=1}^{n-1} [\log_2(n+1)+1]$

= $\sum_{i=0}^{n-1} (n-i)[\log_2(n+1)+1] = \frac{n*(n+1)}{2} \log_2(n+1) + \frac{n*(n+1)}{2} \in O(n^2 \log n)$

In the second part, the while loop executes when x2 > 0. We know that n = 2^k – 1. Given x2 = n + 1 and x2 becomes $\lfloor x2/2 \rfloor$ at each iteration, we can show that:

x2 = 2^k – 1 + 1 = = 2^k

ELİF KIZILKAYA – 2018400108
ÜMMÜ SENA ÖZPINAR – 2019400279

Since x2 is divided by 2 in each iteration, there must be k+1 iterations. From n = $2^k - 1$,

k = $\log_2(n+1)$

Thus, this executes:
$\sum_{i=0}^{n-1} \sum_{t2=1}^{n} \sum_{p2=1}^{n} [\log_2(n+1)+1]$

= $n^3 \log_2(n+1) \in O(n^3 \log n)$

In the third part, instead of a while loop, there are 2 for loops. The inner for loop's execution does not depend on x3.

This executes:
$\sum_{i=0}^{n-1} \sum_{t3=1}^{n} \sum_{p3=1}^{(t3)^2-1} 1 = \sum_{i=0}^{n-1}(0+4+9+16+\cdots+n^2) = n[\frac{n*(n+1)(2n+1)}{6}] \in O(n^4)$

According to these results, we can show the complexities of these parts as 1 < 2 < 3. Let us apply these results to the best, worst and average case as below.

Analyze B(n)

The best case is if arr[i] = 0 for all i ∈ {0,1,2}, $0 \le i \le n-1$. In this case, only the first if statement is executed and the complexity is $O(n^2 \log n)$ as shown above.

Analyze W(n)

The worst case is if arr[i] = 2 for all i ∈ {0,1,2}, $0 \le i \le n-1$. In this case, only the third if statement is executed and the complexity is $O(n^4)$ as shown above.

Analyze A(n)

In the average case, since for each i, the probability that arr[i] = 0 is 1/3, the probability that arr[i] = 1 is 1/3 and the probability that arr[i] = 2 is 1/3, we can calculate the average complexity using the formula A(n) = $\sum_{I \in T_n} \tau(I) * p(I)$ where τ(I) is the number of basic operations for input I as:
$\frac{1}{3}\sum_{i=0}^{n-1} \sum_{j=1}^{n-1} [\log_2(n+1)+1] + \frac{1}{3}\sum_{i=0}^{n-1} \sum_{t2=1}^{n} \sum_{p2=1}^{n} [\log_2(n+1)+1] +$
$\frac{1}{3}\sum_{i=0}^{n-1} \sum_{t3=1}^{n} \sum_{p3=1}^{(t3)^2-1} 1$
$= \frac{1}{3}\left[\frac{n*(n+1)}{2}\log_2(n+1) + \frac{n*(n+1)}{2}\right] + \frac{1}{3}n^3 \log_2(n+1) + \frac{1}{3}\, n[\frac{n*(n+1)(2n+1)}{6}] \in O(n^4)$

Basic operation is two assignments marked as (3)

Since basic operation (3) is executed only twice inside the second and third if statements, we can divide our analysis into 2 parts:

1- If arr[i] = 1,
2- If arr[i] = 2,

In the first part, the while loop executes when x2 > 0. We know that n = $2^k - 1$. Given x2 = n + 1 and x2 becomes $\lfloor x2/2 \rfloor$ at each iteration, we can show that:

x2 = $2^k - 1 + 1 = = 2^k$

ELİF KIZILKAYA – 2018400108
ÜMMÜ SENA ÖZPINAR – 2019400279

Since x2 is divided by 2 in each iteration, there must be k+1 iterations. From n = $2^k - 1$,

k = $\log_2(n+1)$

Thus, this executes:
$\sum_{i=0}^{n-1}\sum_{t2=1}^{n}\sum_{p2=1}^{n}[\log_2(n+1)+1]$

= $n^3\log_2(n+1) \in O(n^3 \log n)$

In the third part, instead of a while loop, there are 2 for loops. The inner for loop's execution does not depend on x3.

This executes:
$\sum_{i=0}^{n-1}\sum_{t3=1}^{n}\sum_{p3=1}^{(t3)^2-1} 1 = \sum_{i=0}^{n-1}(0+4+9+16+\cdots+n^2) = n[\frac{n*(n+1)(2n+1)}{6}] \in O(n^4)$

Let us apply these results to the best, worst, and average cases.

Analyze B(n)

Basic operation (3) is executed only inside the second and third if statements. In the case where arr[i] = 0 for all i ∈ {0,1,2}, $0 \le i \le n-1$, this operation is never executed. Hence this is the best-case scenario and its complexity is $O(1)$.

Analyze W(n)

The worst case is if arr[i] = 2 for all i ∈ {0,1,2}, $0 \le i \le n-1$. In this case, only the third if statement is executed and the complexity is $O(n^4)$ as shown above.

Analyze A(n)

In the average case, we do not have to take into account the case for when arr[i] = 0. Since probability that arr[i] = 1 is 1/3 and the probability that arr[i] = 2 is 1/3 , we can calculate the average complexity using the formula A(n) = $\sum_{I \in T_n} \tau(I) * p(I)$ where τ(I) is the number of basic operations for input I as:
$\frac{1}{3}\sum_{i=0}^{n-1}\sum_{t2=1}^{n}\sum_{p2=1}^{n}[\log_2(n+1)+1] + \frac{1}{3}\sum_{i=0}^{n-1}\sum_{t3=1}^{n}\sum_{p3=1}^{(t3)^2-1} 1$
$= \frac{1}{3}n^3\log_2(n+1) + \frac{1}{3}\, n[\frac{n*(n+1)(2n+1)}{6}] \in O(n^4)$

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

Since basic operation (4) is executed only twice inside the second and third if statements, we can divide our analysis into 2 parts:

1- If arr[i] = 1,
2- If arr[i] = 2,

In the first part, the number of executions is:
$\sum_{i=0}^{n-1}\sum_{t2=1}^{n}\sum_{p2=1}^{n} 1 = n^3 \in O(n^3)$

ELİF KIZILKAYA – 2018400108
ÜMMÜ SENA ÖZPINAR – 2019400279

In the second part, the number of executions is:
$\sum_{i=0}^{n-1} \sum_{t3=1}^{n} 1 = n^2 \in O(n^2)$

According to these results, we can show the complexities of these parts as 2 < 3. Hence;

Analyze B(n)

Basic operation (4) is executed only inside the second and third if statements. In the case where arr[i] = 0 for all i ∈ {0,1,2}, $0 \leq i \leq n-1$, this operation is never executed. Hence this is the best case scenario and its complexity is $O(1)$.

Analyze W(n)

The worst case is if arr[i] = 1 for all i ∈ {0,1,2}, $0 \leq i \leq n-1$. In this case, only the second if statement is executed and the complexity is $O(n^3)$ as shown above.

Analyze A(n)

In the average case, we don't have to take into account the case for when arr[i] = 0. Since probability that arr[i] = 1 is 1/3 and the probability that arr[i] = 2 is 1/3, we can calculate the average complexity using the formula A(n) = $\sum_{I \in T_n} \tau(I) * p(I)$ where τ(I) is the number of basic operations for input I as:
$\frac{1}{3}\sum_{i=0}^{n-1} \sum_{t3=1}^{n} 1 = + \frac{1}{3}\sum_{i=0}^{n-1} \sum_{t2=1}^{n} \sum_{p2=1}^{n} 1$
$= \frac{1}{3}n^2 + \frac{1}{3}n^3 \in O(n^3)$

Basic operation is the assignment marked as (5)

Analyze B(n)

Basic operation (5) is executed only inside the first if statement. In the cases where arr[i] = 1 and arr[i] = 2 for all i ∈ {0,1,2}, $0 \leq i \leq n-1$, this operation is never executed. Hence the best-case input is when arr[i] = 1 or arr[i] = 2 and its complexity is $O(1)$.

Analyze W(n)

The worst case is if arr[i] = 0 for all i ∈ {0,1,2}, $0 \leq i \leq n-1$. Its complexity is:
$\sum_{i=0}^{n-1} \sum_{t1=i}^{n-1} 1 = \sum_{i=0}^{n-1}(n-i) = \frac{n*(n+1)}{2} \in O(n^2)$

Analyze A(n)

In the average case, we don't have to take into account the case for when arr[i] = 1 and arr[i] = 2. Since probability that arr[i] = 0 is 1/3, we can calculate the average complexity using the formula A(n) = $\sum_{I \in T_n} \tau(I) * p(I)$ where τ(I) is the number of basic operations for input I as:
$\frac{1}{3}\sum_{i=0}^{n-1} n - i = \frac{1}{3}\frac{n*(n+1)}{2} \in O(n^2)$

ELİF KIZILKAYA – 2018400108
ÜMMÜ SENA ÖZPINAR – 2019400279

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)

To be a basic operation, it should be something repeated and characteristic to algorithm. In this case, (2) should be the basic operation in each if-else branch. This statement is repeated in each branch. The other operations are simple loop incrementations or comparisons that are not repeated or not characteristic

REAL EXECUTION

Best Case

N Size	Time Elapsed
1	0.000000
5	0.000000
10	0.000000
25	0.000607
50	0.001003
75	0.002969
100	0.006001
150	0.014029
200	0.024000
250	0.041001

Worst Case

N Size	Time Elapsed
1	0.000000
5	0.000000
10	0.000902
25	0.015022
50	0.225110
75	1.073590
100	3.395699
150	17.347764
200	54.496218
250	132.569160

ELİF KIZILKAYA – 2018400108
ÜMMÜ SENA ÖZPINAR – 2019400279

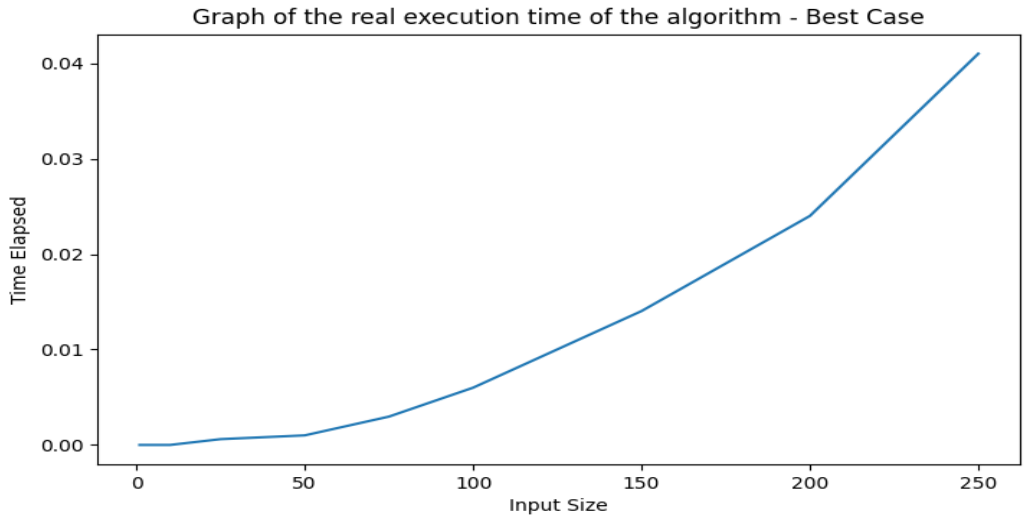
Average Case

N Size	Time Elapsed
1	0.000000
5	0.000000
10	0.000675
25	0.010174
50	0.096457
75	0.514675
100	1.433115
150	6.671943
200	19.830768
250	46.254731

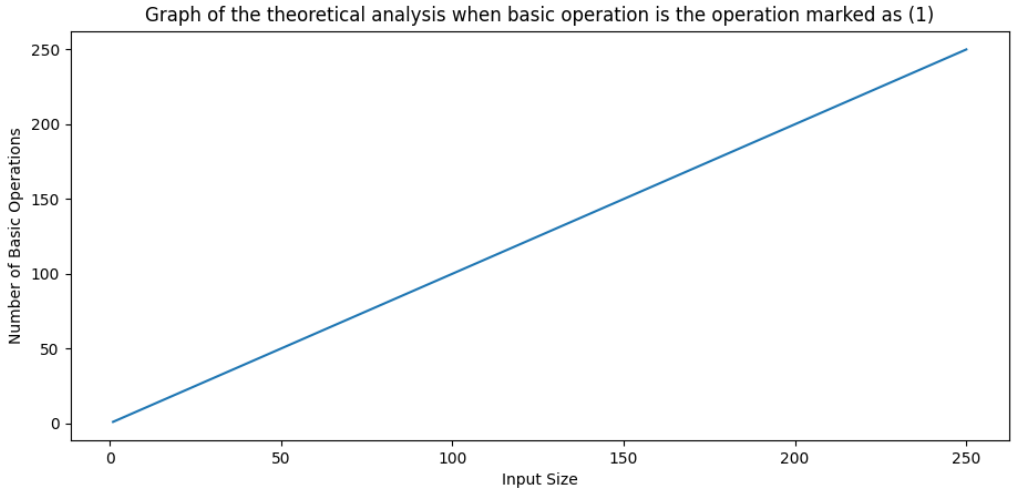
COMPARISON

Best Case

Graph of the real execution time of the algorithm

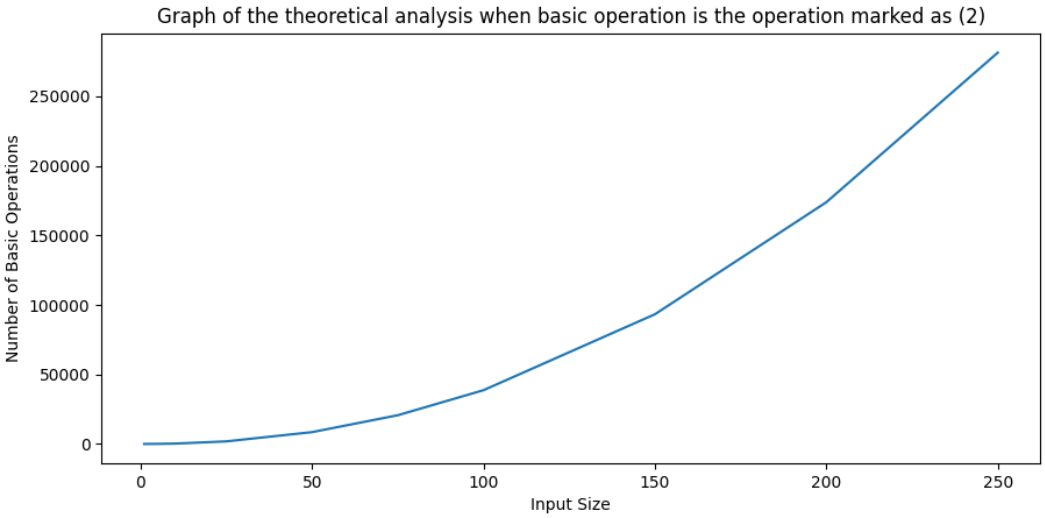


Graph of the theoretical analysis when basic operation is the operation marked as (1)

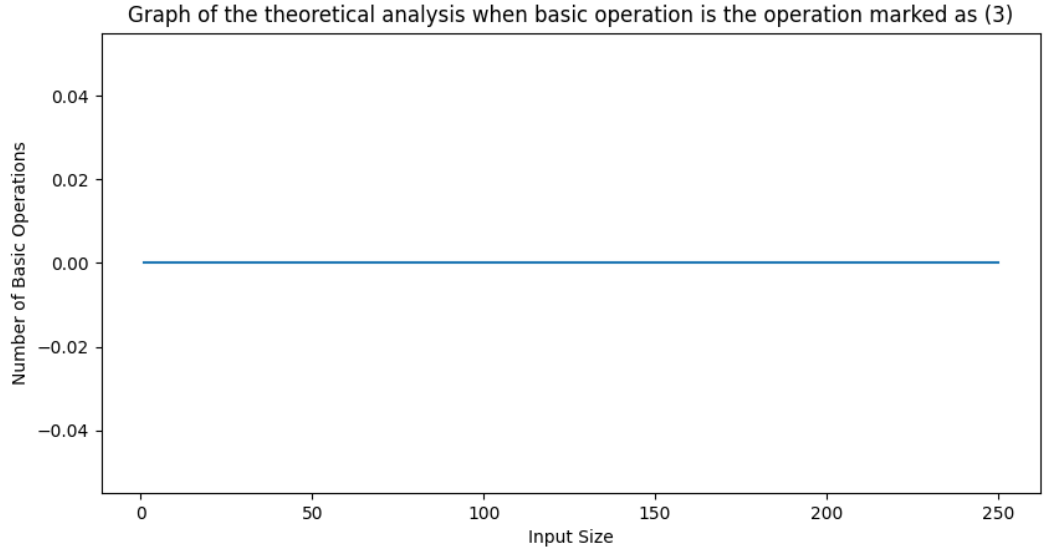


ELİF KIZILKAYA – 2018400108
ÜMMÜ SENA ÖZPINAR – 2019400279

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

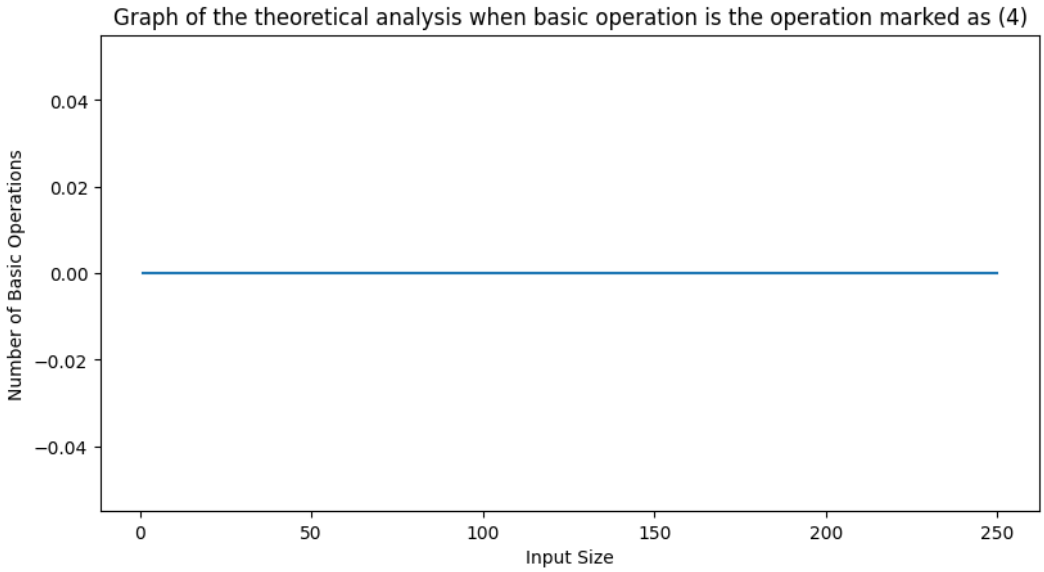


Graph of the theoretical analysis when basic operation is the operation marked as (3)

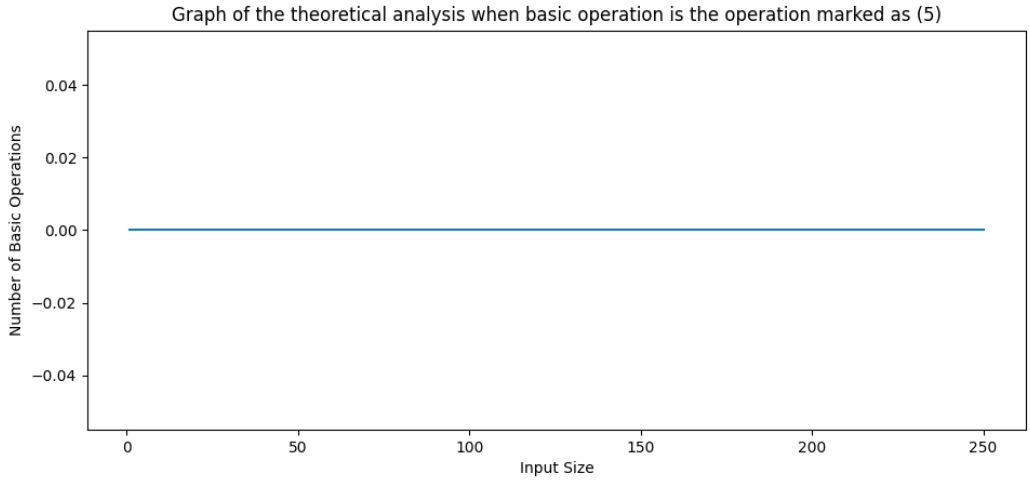


ELİF KIZILKAYA – 2018400108
ÜMMÜ SENA ÖZPINAR – 2019400279

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



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



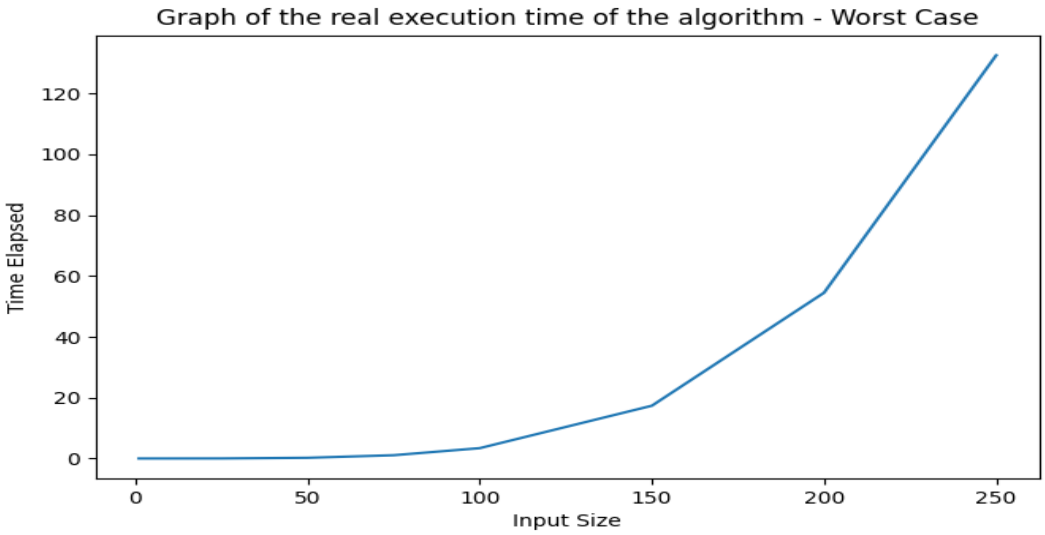
Comments

A basic operation is called realistic if the actual execution time on a computer is bounded by a linear function of the derived (analyzed) complexity. If we look at the graph that we chose as basic operation (2), the real execution time graph has similar appearances. Thus, we can say that their growth rates are similar. As a conclusion, we can see that we chose the right basic operation. The other graphs have very different growth rates than actual execution time of the algorithm.

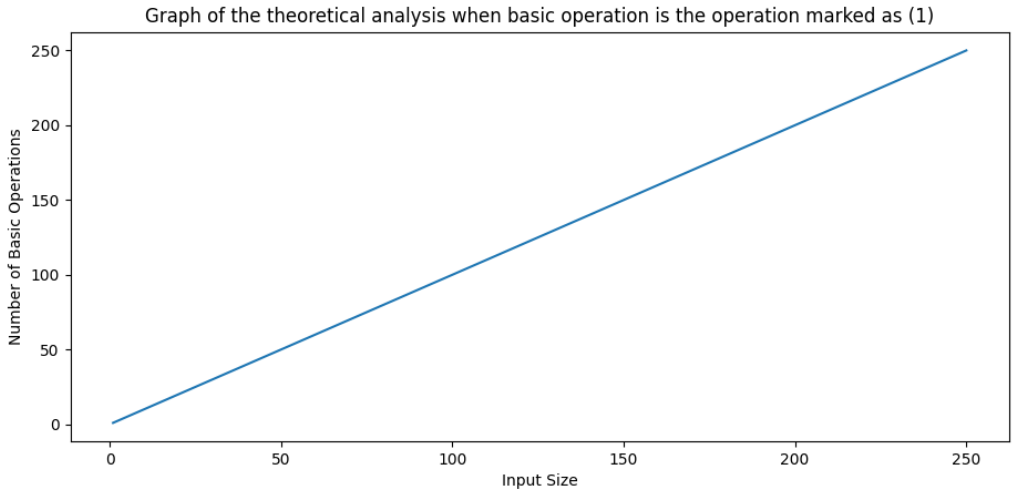
ELİF KIZILKAYA – 2018400108
ÜMMÜ SENA ÖZPINAR – 2019400279

Worst Case

Graph of the real execution time of the algorithm

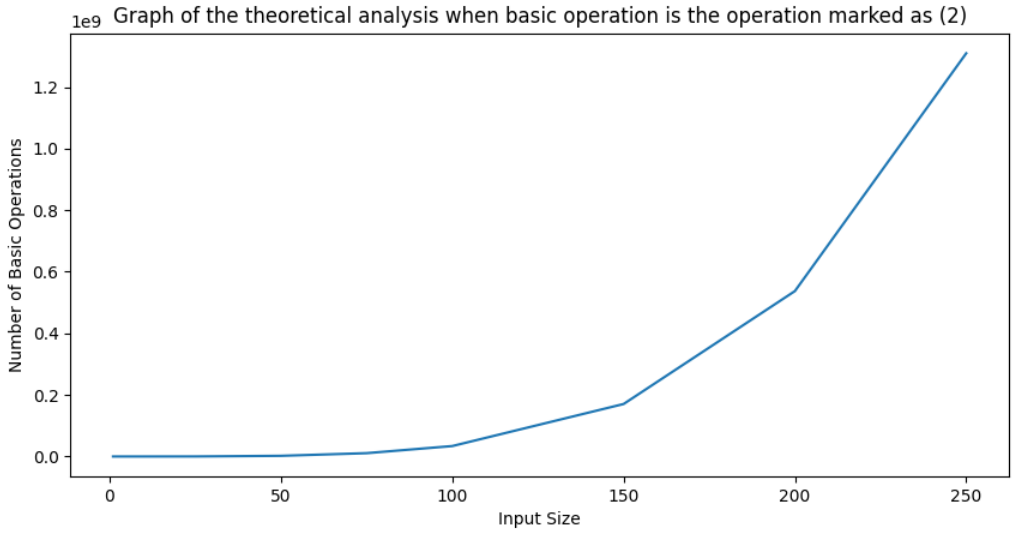


Graph of the theoretical analysis when basic operation is the operation marked as (1)

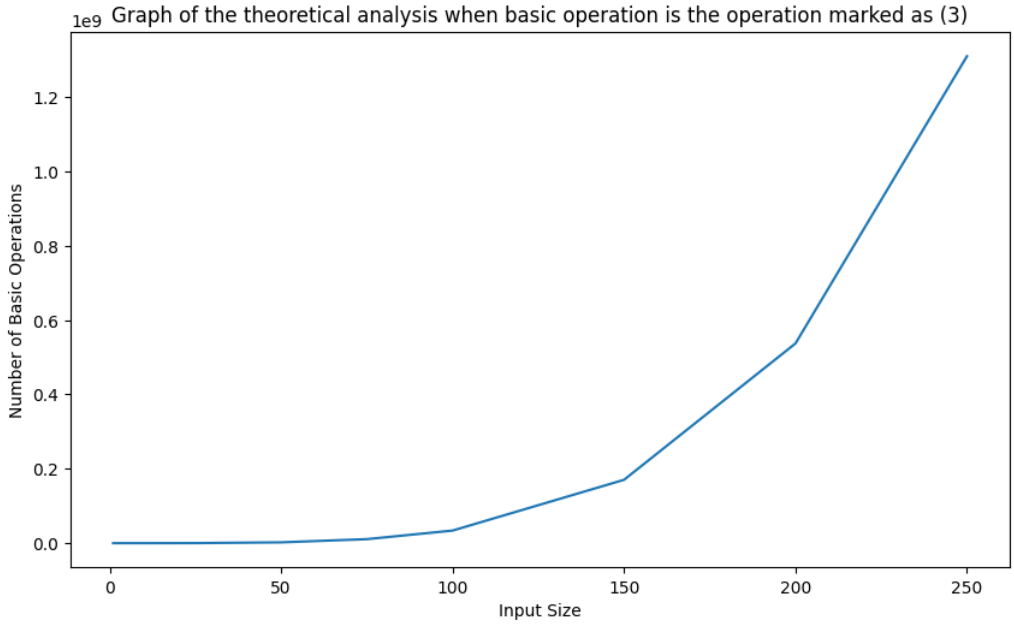


ELİF KIZILKAYA – 2018400108
ÜMMÜ SENA ÖZPINAR – 2019400279

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

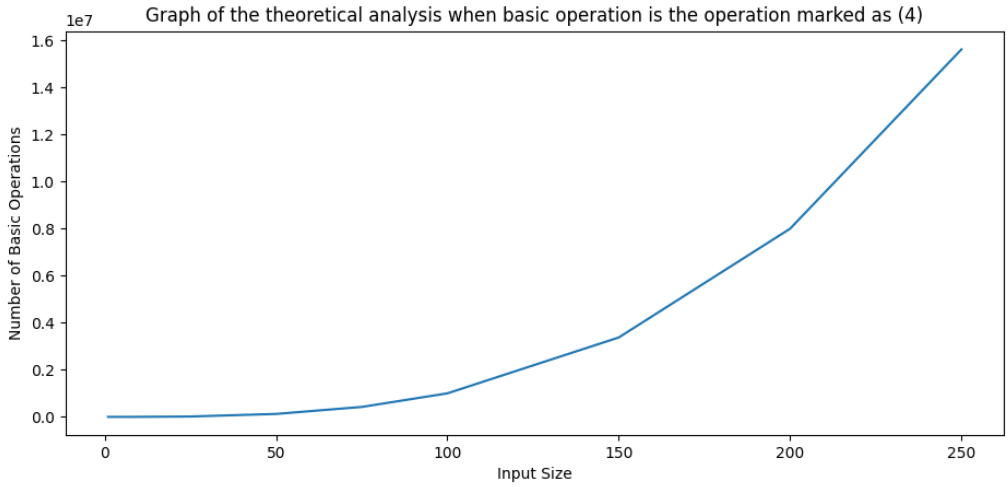


Graph of the theoretical analysis when basic operation is the operation marked as (3)

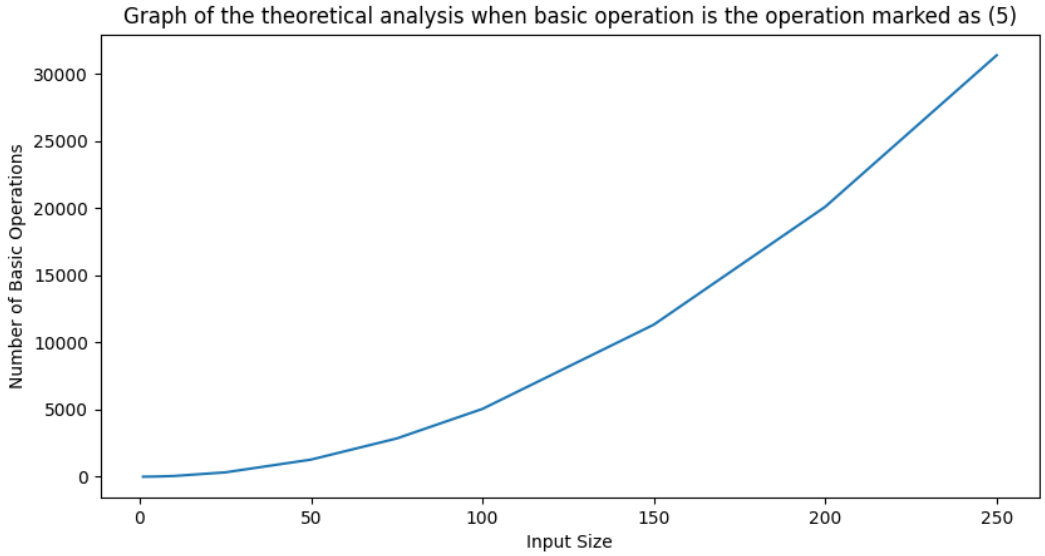


ELİF KIZILKAYA – 2018400108
ÜMMÜ SENA ÖZPINAR – 2019400279

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



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



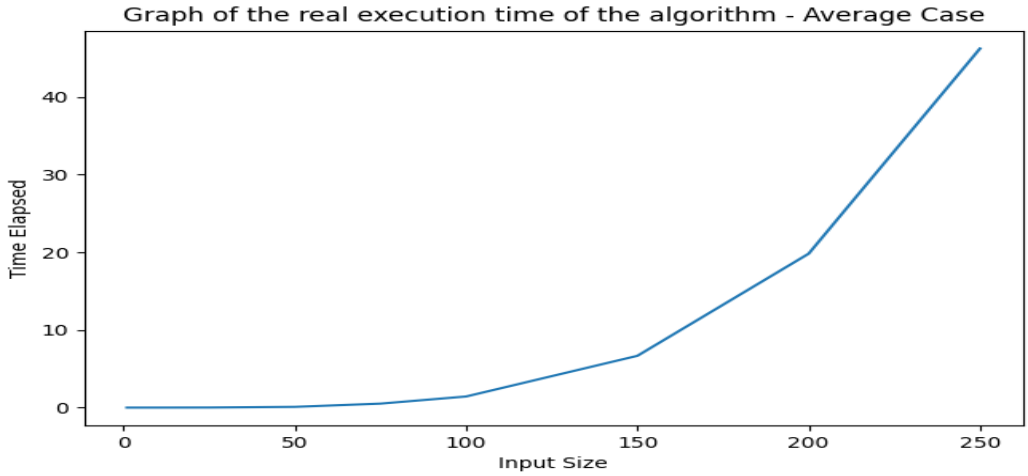
Comments

A basic operation is called realistic if the actual execution time on a computer is bounded by a linear function of the derived (analyzed) complexity. If we look at the graph that we chose as basic operation (2), the real execution time graph has similar appearances. Thus, we can say that their growth rates are similar. As a conclusion, we can see that we chose the right basic operation. However, in this case, the other (3), (4), and (5) numbered operations have also similar appearances as real execution time. We can attribute this situation to the fact that these functions contain similar elements. However, if we look at closely, we can say that (2) is the most similar one to the real execution time graph.

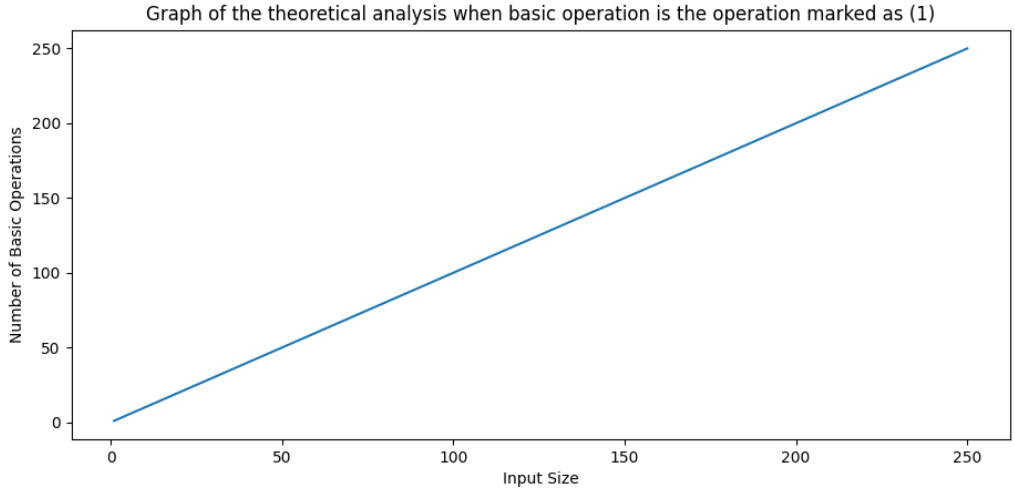
ELİF KIZILKAYA – 2018400108
ÜMMÜ SENA ÖZPINAR – 2019400279

Average Case

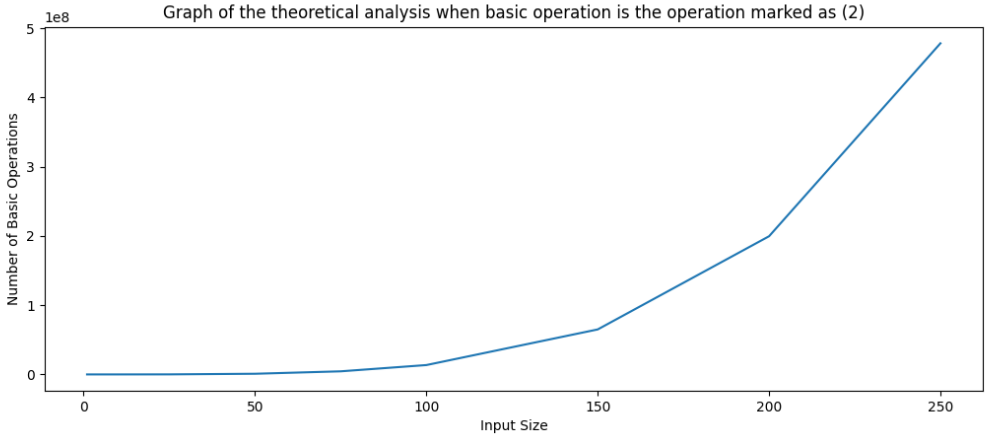
Graph of the real execution time of the algorithm



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

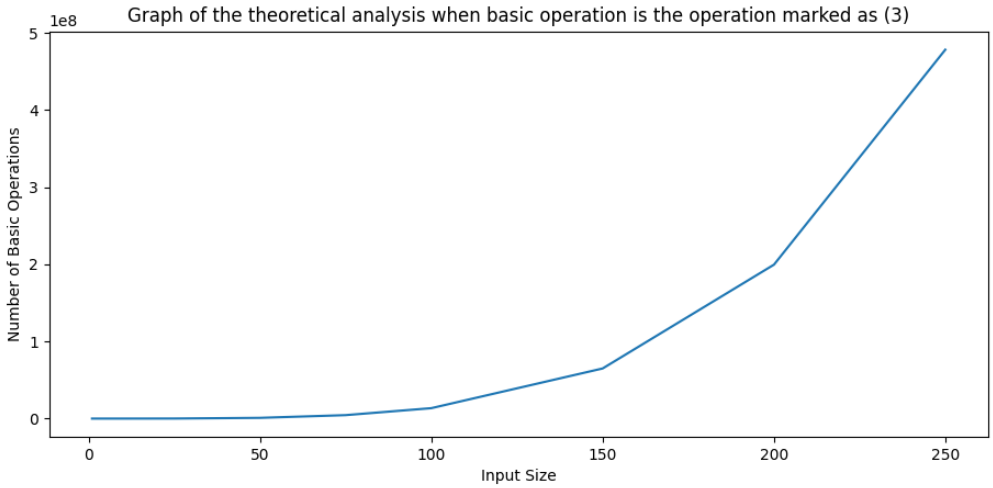


Graph of the theoretical analysis when basic operation is the operation marked as (2)

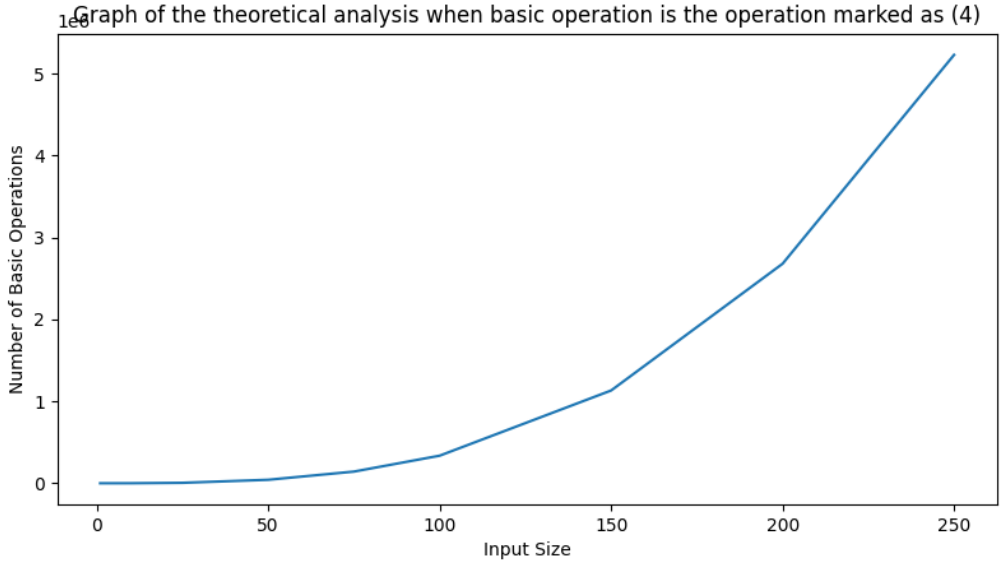


ELİF KIZILKAYA – 2018400108
ÜMMÜ SENA ÖZPINAR – 2019400279

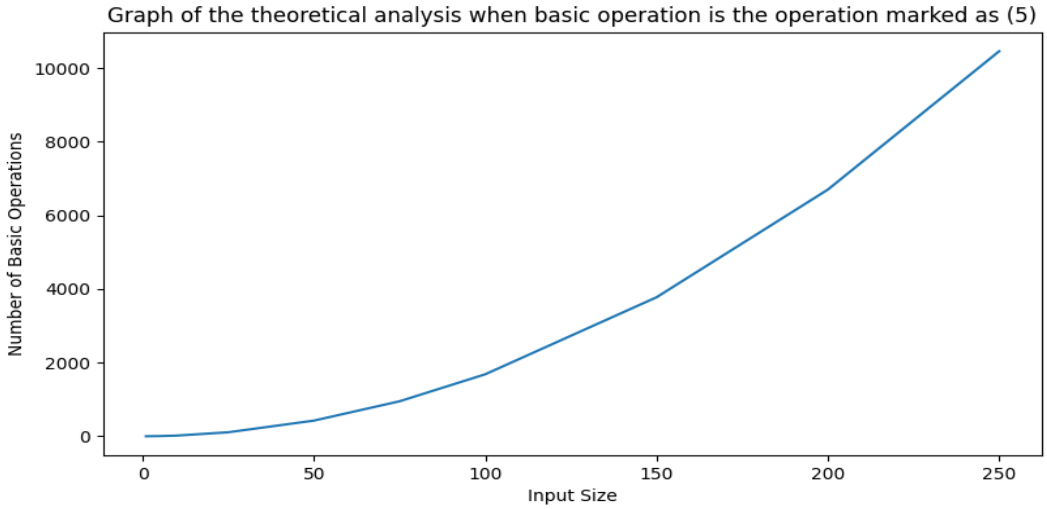
Graph of the theoretical analysis when basic operation is the operation marked as (3)



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



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



ELİF KIZILKAYA – 2018400108
ÜMMÜ SENA ÖZPINAR – 2019400279

Comments

A basic operation is called realistic if the actual execution time on a computer is bounded by a linear function of the derived (analyzed) complexity. If we look at the graph that we chose as basic operation (2), the real execution time graph has similar appearances. Thus, we can say that their growth rates are similar. As a conclusion, we can see that we chose the right basic operation. However, in this case, the other (3), (4), and (5) numbered operations have also similar appearances as real execution time. We can attribute this situation to the fact that these functions contain similar elements. However, if we look at closely, we can say that (2) is the most similar one to the real execution time graph.