EMRE KILIÇ
KRİSTİNA TRAJKOVSKİ

Table of Contents

EMRE KILIÇ
KRİSTİNA TRAJKOVSKİ

THEORETICAL ANALYSIS

Basic operation is the comparison marked as (1)

Analyze B(n)

For best case, the number of basic operations is the same for any input array.
There are $B(n) = \sum_{i=0}^{n-1} 1 = n$ basic operations. So $B(n) \in \theta(n)$.

Analyze W(n)

For worst case, the number of basic operations is the same for any input array.
There are $W(n) = \sum_{i=0}^{n-1} 1 = n$ basic operations. So $W(n) \in \theta(n)$.

Analyze A(n)

For average case, the number of basic operations is follows:
$A(n) = \sum_{k=B(n)}^{W(n)} k * p_k = \sum_{k=n}^{n} k * 1 = n.$ So $A(n) \in \theta(n)$.
Note that $p_k = 1$ since there is only one possibility in the sample space that the number of basic operations is n.

EMRE KILIÇ
KRİSTİNA TRAJKOVSKİ

Basic operations are the two assignments marked as (2)

- Number of basic operations for arr[i] = 0 :

There is one basic operation in the inner most loop, so we may start with 1 as the right-most element in the sum representation. The operation is inside a loop which takes values from i to n-1. Therefore, the number of basic operations for arr[i] = 0 is:

$$\tau_i(0) = \sum_{j=i}^{n-1} 1 = (n - 1 - i + 1) * 1 = n - i$$

- Number of basic operations for arr[i] = 1:

The number of basic operations for arr[i] = 1 has the same structure as above, for it has a loop which takes values from i to n-1 and has one basic operation in the loop. Therefore, the number of basic operations for arr[i] = 1 is:

$$\tau_i(1) = \sum_{m=i}^{n-1} 1 = (n - 1 - i + 1) * 1 = n - i$$

- Number of basic operations for arr[i] = 2:

There are no basic operations labeled with (2) for arr[i]=2, so number of basic operations for this case is $\tau_i(2) = 0$.

Analyze B(n)

From the analysis of basic operations for each element type from above, clearly the best case happens when there are no basic operations executed at all. Therefore, the best case happens when for all i arr[i] = 2 (the input I is 222222…). The number of basic operations is then:

$\sum_{i=0}^{n-1} 0 = 0$ and clearly the best case is constant $B(n) \in \theta(1)$

Analyze W(n)

From the analysis of basic operations for each element type from above, clearly the worst case happens when all arr[i] are either 0 or 1 (0010111100… etc). Then number of basic operations for arr[i] is n-1 and total number of operations is:

$\sum_{i=0}^{n-1}(n - i) = n + (n - 1) + \cdots + 1 = \frac{n(n-1)}{2} = \frac{n^2}{2} - \frac{n}{2}$ and clearly worst case is quadratic $W(n) \in \theta(n^2)$.

Analyze A(n)

For the set of all inputs of size n T_n, number of basic operations for input $I \in T_n$, $\tau(I)$, is calculated as:

$$A(n) = E[\tau(I)]$$

For each $I_i \in I$, the probability of an element is independent and $p(I_i) = \frac{1}{3}$. Note that an element can only take 3 values, $I_i \in \{0,1,2\}$. So, we can rewrite $A(n)$ as:

$$A(n) = E[\sum_{i=0}^{n-1} \tau_i(I_i)] = \sum_{i=0}^{n-1} E[\tau_i(I_i)] = \sum_{i=0}^{n-1} \tau_i(0) * p(0) + \tau_i(1) * p(1) + \tau_i(2) * p(2)$$

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

$$= \frac{2n^2 - n^2 + n}{3} = \frac{n^2 + n}{3}$$

We can say that the average case is quadratic $A(n) \in \theta(n^2)$.

EMRE KILIÇ
KRİSTİNA TRAJKOVSKİ

Basic operations are the two comparisons marked as (3)

- Number of basic operations for arr[i] = 0 :

The basic operation is a part of the inner most loop. It will be executed as many times as the loop returns true plus one time it returns false. There is one comparison. Since the loop starts as k = n and executes until k>0 and reduces k as $\frac{k}{3}$, for simplicity of the analysis let

$$m = \lfloor \log_3 k \rfloor + 1 \ (or\ in\ other\ words)\ k \approx 3^{m-1},$$

$$(in\ reality\ k \geq 3^{m-1}, but\ in\ this\ example\ all\ the\ values\ of\ k\ between\ 3^m\ and\ 3^{m+1}\ for\ some\ m \in Z\ have\ the\ same\ number\ of\ executions\ due\ to\ integer\ division)$$

$$so\ at\ each\ iteration\ \frac{k}{3} = 3^{m-1}$$

$$and\ the\ comparison\ returns\ true\ while\ k > 0\ and\ m > 0$$

So just for the purpose of counting the number of basic operations, we may rewrite the inner most loop as:

for $m \leftarrow 1\ to\ \lfloor \log_3 k \rfloor + 1\ do$

…

end

Also, we must not forget that the comparison chosen as basic operation will be made one extra time for the false return in the while loop and cause the termination of the loop. So, the number of basic operations in the inner most loop is: $(\sum_{m=0}^{\lfloor \log_3 n \rfloor + 1} 1) + 1$

Adding the other loops yields:

$$\tau_i(0) = \sum_{j=i}^{n-1} \sum_{m=1}^{\lfloor \log_3 n \rfloor + 1} (1) + 1 = \sum_{j=i}^{n-1} (\lfloor \log_3 n \rfloor + 2) = (n-i)\ (\lfloor \log_3 n \rfloor + 2)$$

- Number of basic operations for arr[i] = 1:

There are no basic operations labeled with (3) for arr[i]=1, so number of basic operations for this case is $\tau_i(1) = 0$.

- Number of basic operations for arr[i] = 2:

For the purposes of counting the number of basic operations, we may rewrite the while loop as for loop which yields:

for $p \leftarrow 0\ to\ n-1\ do$

…

end

Also, we must not forget that the comparison chosen as basic operation will be made one extra time for the false return in the while loop and cause the termination of the loop. So the number of basic operations in the loop is: $(\sum_{p=0}^{n-1} 1) + 1$

Therefore:

$$\tau_i(2) = \sum_{p=0}^{n-1} (1) + 1 = n + 1$$

EMRE KILIÇ
KRİSTİNA TRAJKOVSKİ

Analyze B(n)

From the analysis of basic operations for each element type from above, clearly the best case happens when there are no basic operations executed at all. Therefore, the best case happens when for all i arr[i] = 1 (the input I is 1111111…). The number of basic operations is then:

$\sum_{i=0}^{n-1} 0 = 0$ and clearly the best case is constant $B(n) \in \theta(1)$

Analyze W(n)

Since $\tau_i(0)$ varies between $\lfloor \log_3 n \rfloor + 2$ and $n * \lfloor \log_3 n \rfloor + 2$, depending on i $\tau_i(0)$ may be larger or less than $\tau_i(2)$ which is always n+1. Therefore the worst case input looks like 0000…0002222… To see where exactly the input switches from one to the other we need to see for which i in terms of n $\tau_i(2) > \tau_i(0)$.

$$n + 1 > (n - i)(\lfloor \log_3 n \rfloor + 2)$$

$$\frac{n+1}{\lfloor \log_3 n \rfloor + 2} > n - i$$

$$i > n - \frac{n+1}{\lfloor \log_3 n \rfloor + 2}$$

Since

Therefore:

$$W(n) = \sum_{i=0}^{n-\left\lfloor \frac{n+1}{\lfloor \log_3 n \rfloor+2} \right\rfloor} \tau_i(0) + \sum_{n-\left\lfloor \frac{n+1}{\lfloor \log_3 n \rfloor+2} \right\rfloor+1}^{n-1} \tau_i(2)$$

$$= \sum_{i=0}^{n-\left\lfloor \frac{n+1}{\lfloor \log_3 n \rfloor+2} \right\rfloor} (n - i)(\lfloor \log_3 n \rfloor + 2) + \sum_{n-\left\lfloor \frac{n+1}{\lfloor \log_3 n \rfloor+2} \right\rfloor+1}^{n-1} n + 1$$

$$= (\lfloor \log_3 n \rfloor + 2)(n \sum_{i=0}^{n-\left\lfloor \frac{n+1}{\lfloor \log_3 n \rfloor+2} \right\rfloor} 1 - \sum_{i=0}^{n-\left\lfloor \frac{n+1}{\lfloor \log_3 n \rfloor+2} \right\rfloor} i + (n + 1) \sum_{n-\left\lfloor \frac{n+1}{\lfloor \log_3 n \rfloor+2} \right\rfloor+1}^{n-1} 1$$

$$= (\lfloor \log_3 n \rfloor + 2)(n * (n^2 - \left\lfloor \frac{n+1}{\lfloor \log_3 n \rfloor + 2} \right\rfloor + 1) - \frac{1}{2} * (n - \left\lfloor \frac{n+1}{\lfloor \log_3 n \rfloor + 2} \right\rfloor) * (n - \left\lfloor \frac{n+1}{\lfloor \log_3 n \rfloor + 2} \right\rfloor - 1)) + (n + 1)(\left\lfloor \frac{n+1}{\lfloor \log_3 n \rfloor + 2} \right\rfloor + 1)$$

$$= (\lfloor \log_3 n \rfloor + 2)(\frac{2n^3 - n^2 + 5n - \left\lfloor \frac{n+1}{\lfloor \log_3 n \rfloor + 2} \right\rfloor^2 + \left\lfloor \frac{n+1}{\lfloor \log_3 n \rfloor + 2} \right\rfloor + 2}{2})$$

The dominant term of the right term is n^3. So, we can reduce the terms by eliminating low order terms and constants in order to find asymptotic behaviour to

$$W(n) = n^3 * \lfloor log\, n \rfloor$$

We can say that $\lfloor log\, n \rfloor \in \theta(log\, n)$ since

$$\exists c_1, c_2, n_0 \quad s.t. \quad c_1 * log\, n \leq \lfloor log\, n \rfloor \leq c_2 * log\, n \quad \forall n \geq n_0$$

We know $log\, n - 1 \leq \lfloor log\, n \rfloor \leq log\, n$

so find $c_1, n_0 \;\; s.t \;\; c_1 * log\, n \leq log\, n \; - 1 \quad \forall n \geq n_0$

$c_1 \leq \frac{log\, n - 1}{log\, n} \;\; \forall n \geq n_0$ when n grows $\frac{log\, n-1}{log\, n}$ grows too. So $c_1 \leq \frac{log\, n_0 - 1}{log\, n_0} \quad \forall n \geq n_0$

by choosing $c_2 = 1$ and $c_1 = \frac{log\, n_0 - 1}{log\, n_0}$ the statement satisfies.

Finally, the worst-case complexity is $W(n) \in \theta(n^3 * log\, n)$

Analyze A(n)

The average case can be found by the formula derived above:

$$A(n) = E[\tau(I)] = \sum_{i=0}^{n-1} \tau_i(0) * p(0) + \tau_i(1) * p(1) + \tau_i(2) * p(2)$$

$$= \sum_{i=0}^{n-1} (n-i)(\lfloor \log_3 n \rfloor + 2) * \frac{1}{3} + (n+1) * \frac{1}{3}$$

$$= \frac{1}{3} * (\sum_{i=0}^{n-1} (n-i)(\lfloor \log_3 n \rfloor + 2) + \sum_{i=o}^{n-1} (n+1))$$

$$= \frac{1}{3} * ((\lfloor \log_3 n \rfloor + 2) * \sum_{i=0}^{n-1} (n-i) + n^2 + n)$$

$$= \frac{1}{3} * ((\lfloor \log_3 n \rfloor + 2)(n^2 - \sum_{i=0}^{n-1} i) + n^2 + n)$$

$$= \frac{1}{3} * ((\lfloor \log_3 n \rfloor + 2)(n^2 - (\frac{n^2 - n}{2})) + n^2 + n)$$

$$= \frac{1}{3} * ((\lfloor \log_3 n \rfloor + 2)(\frac{n^2 + n}{2}) + n^2 + n)$$

$$= \frac{1}{6} * ((\lfloor \log_3 n \rfloor + 2) * (n^2 + n) + 2n^2 + 2n)$$

$$= \frac{\lfloor \log_3 n \rfloor * n^2 + \lfloor \log_3 n \rfloor * n + 4n^2 + 4n}{6}$$

The dominant term here is $\lfloor \log_3 n \rfloor * n^2$. We can say that $\lfloor log\, n \rfloor \in \theta(log\, n)$ since

$$\exists c_1, c_2, n_0 \quad s.t. \quad c_1 * log\, n \leq \lfloor log\, n \rfloor \leq c_2 * log\, n \quad \forall n \geq n_0$$

We know $log\, n - 1 \leq \lfloor log\, n \rfloor \leq \; log\, n$

so find $c_1, n_0 \;\; s.t \;\; c_1 * log\, n \leq log\, n \; - 1 \quad \forall n \geq n_0$

$c_1 \leq \frac{log\, n - 1}{log\, n} \;\; \forall n \geq n_0$ when n grows $\frac{log\, n-1}{log\, n}$ grows too. So $c_1 \leq \frac{log\, n_0 - 1}{log\, n_0} \quad \forall n \geq n_0$

by choosing $c_2 = 1$ and $c_1 = \frac{log\, n_0 - 1}{log\, n_0}$ the statement satisfies.

Finally, we can say the average complexity is $A(n) \in \theta(n^2 * log\, n)$.

Basic operations are the three assignments marked as (4)

- Number of basic operations for arr[i] = 0 :

There is one basic operation in the inner most loop, so we may start with 1 as the right-most element in the sum representation. The calculation of the while loop executions will be very similar to the example above ($\tau_i(0)$ for "basic operations are the three assignments marked as (3)") keeping in mind that we should count only the cases where the comparison in the while loop returns true. Therefore, the number of basic operations for arr[i] = 0 is:

$$\tau_i(0) = \sum_{j=i}^{n-1} \sum_{m=1}^{\lfloor \log_3 n \rfloor + 1} 1 = \sum_{j=i}^{n-1} (\lfloor \log_3 n \rfloor + 1) = (n-i)\ (\lfloor \log_3 n \rfloor + 1)$$

- Number of basic operations for arr[i] = 1:

There is one basic operation in the inner most loop, so we may start with 1 as the right-most element in the sum representation. The inner most loop decrements z by t at each iteration, so just for the purpose of calculating the number of basic operations in the loop we may rewrite it as

for z← $0\ to\ \left\lceil \frac{n}{t} \right\rceil - 1\ do$
…
end

Also, the previous loop determining t may be rewritten as:

for t← $1\ to\ n\ do$
…
end

The rest of the loops are straight forward, therefore the number of basic operations for arr[i]=1 is:

$$\tau_i(1) = \sum_{m=i}^{n-1} \sum_{l=m}^{n-1} \sum_{t=1}^{n} \sum_{z=0}^{\left\lceil \frac{n}{t} \right\rceil - 1} 1 = \sum_{m=i}^{n-1} \sum_{l=m}^{n-1} \sum_{t=1}^{n} \left\lceil \frac{n}{t} \right\rceil$$

We may conclude:

$$\sum_{m=i}^{n-1} \sum_{l=m}^{n-1} \sum_{t=1}^{n} \frac{n}{t} \le \sum_{m=i}^{n-1} \sum_{l=m}^{n-1} \sum_{t=1}^{n} \left\lceil \frac{n}{t} \right\rceil \le \sum_{m=i}^{n-1} \sum_{l=m}^{n-1} \sum_{t=1}^{n} (\frac{n}{t} + 1)$$

Let H(n) denote the sum of first n terms of the harmonic series.
Then:

$$\sum_{m=i}^{n-1}\sum_{l=m}^{n-1} n\,H(n) \;\le \sum_{m=i}^{n-1}\sum_{l=m}^{n-1}\sum_{t=1}^{n}\left\lceil\frac{n}{t}\right\rceil \le \sum_{m=i}^{n-1}\sum_{l=m}^{n-1} n\,H(n)+n$$

$$\sum_{m=i}^{n-1}(n-m)\,n\,H(n) \;\le \sum_{m=i}^{n-1}\sum_{l=m}^{n-1}\sum_{t=1}^{n}\left\lceil\frac{n}{t}\right\rceil \le \sum_{m=i}^{n-1}(n-m)(n\,H(n)+n)$$

Simplify the first sum:

$$\sum_{m=i}^{n-1}(n-m)\,n\,H(n) = \;n^2H(n)\sum_{m=i}^{n-1}1 - nH(n)\sum_{m=i}^{n-1}m =$$

$$(n-i)n^2H(n) - nH(n) * \frac{n^2-3n+2-i^2+i}{2}$$

Simplify the last sum:

$$\sum_{m=i}^{n-1}(n-m)(\,n\,H(n)+n) = \;(n^2H(n)+n)\sum_{m=i}^{n-1}1 - (nH(n)+n)\sum_{m=i}^{n-1}m =$$

$$(n-i)(n^2H(n)+n) - (nH(n)+n) * \frac{n^2-3n+2-i^2+i}{2}$$

Therefore number of basic operations for arr[i]=1 :

$$(n-i)n^2H(n) - nH(n) * \frac{n^2-3n+2-i^2+i}{2} \le \tau_i(1) \le$$
$$(n-i)(n^2H(n)+n) - (nH(n)+n) * \frac{n^2-3n+2-i^2+i}{2}$$

- Number of basic operations for arr[i] = 2:

There is one basic operation in the inner most loop, so we may start with 1 as the right-most element in the sum representation. For the purposes of counting the number of basic operations, we may rewrite the while loop containing p as for loop which yields:

for $p \leftarrow 0\ to\ n-1\ do$

$\ldots$

end

Then the number of basic operations for arr[i]=2 is:

$$\tau_i(2) = \sum_{p=0}^{n-1}\sum_{j=0}^{p^2-1} 1 = \sum_{p=0}^{n-1} p^2 = \frac{n(n-1)(2n-1)}{6} = \frac{2n^3-3n^2+n}{6}$$

EMRE KILIÇ
KRİSTİNA TRAJKOVSKİ

Analyze B(n)

Since number of basic operations for arr[i]=0 varies between $\lfloor\log_3 n\rfloor + 1$ and $n\lfloor\log_3 n\rfloor + n$, it definitely executes less operations compared to arr[i]=1 and arr[i]=2. Therefore the best case happens when arr[i] = 0 for all i. Then the number of basic operations is:

$$\sum_{i=0}^{n-1}(n-i)\left(\lfloor\log_3 n\rfloor + 1\right) = \left(n\lfloor\log_3 n\rfloor + n\right)\sum_{i=0}^{n-1} 1 - \left(\lfloor\log_3 n\rfloor + 1\right)\sum_{i=0}^{n-1} i =$$

$$n^2\lfloor\log_3 n\rfloor + n^2 - \left(\lfloor\log_3 n\rfloor + 1\right)\frac{(n-1)(n-2)}{2}$$

Clearly the dominant term is $n^2\lfloor\log_3 n\rfloor$. We have proven above that floor function of logn is element of θ logn, and similarly we may say B(n)∈ $\theta(n^2 logn)$.

Analyze W(n)

From the calculations above, we see that for arr[i]=1 the number of basic operations varies between $n^2H(n)\ and\ n^3H(n)$. Knowing that $H(n) \approx logn$, number of basic operations for arr[i]=1 approximately varies between $n^2logn\ and\ n^3logn$. We also know that for arr[i]=2 the number of basic operations is of order n^3. So for large i, arr[i]=2 has the most operations, and for small i arr[i]=1 has the most operations. Therefore the worst case has the form 1111…112222…

Let X be the last i in terms of n where $\tau_i(1) > \tau_i(2)$. Then the number of operations for the worst case is:

$$\sum_{i=0}^{X}(n-i)(n^2H(n)+n) - (nH(n)+n) * \frac{n^2 - 3n + 2 - i^2 + i}{2} + \sum_{i=X+1}^{n-1}\frac{2n^3 - 3n^2 + n}{6}$$

The largest (dominant) term will clearly be the term from the beginning approximately: Xn^3H(n), since we know X is in terms of n but cannot be larger than n-1, and H(n) is approximately logn, the complexity of the worst case is W(n)∈ $\theta(n^4 logn)$.

Analyze A(n)

The average case can be found by the formula deriven above:

$$A(n) = E[\tau(I)] = \sum_{i=0}^{n-1} \tau_i(0) * p(0) + \tau_i(1) * p(1) + \tau_i(2) * p(2)$$

$$= \sum_{i=0}^{n-1} \frac{(n-i)(\lfloor \log_3 n \rfloor + 1)}{3} + \frac{(n-i) * n^2 H(n)}{3} - \frac{nH(n)(n^2 - 3n + 2 - i^2 + i)}{6} + \frac{2n^3 - 3n^2 + n}{18}$$

$$= \sum_{i=0}^{n-1} \frac{6n\lfloor \log_3 n \rfloor + 6n - 6i\lfloor \log_3 n \rfloor - 6i + 6n^3 H(n) - 6in^2 H(n) - 3n^3 H(n) + 9n^2 H(n)}{18} + \frac{-6nH(n) + 3i^2 nH(n) + inH(n) + 2n^3 - 3n^2 + n}{18}$$

$$= \sum_{i=0}^{n-1} \frac{6n\lfloor \log_3 n \rfloor + 6n + 6n^3 H(n) - 3n^3 H(n) + 9n^2 H(n) - 6nH(n) + 2n^3 - 3n^2 + n}{18} + \sum_{i=0}^{n-1} \frac{-6i\lfloor \log_3 n \rfloor - 6i - 6in^2 H(n) + 3i^2 nH(n) + inH(n)}{18}$$

$$= \frac{6n^2\lfloor \log_3 n \rfloor + 6n^2 + 6n^4 H(n) - 3n^4 H(n) + 9n^3 H(n) - 6n^2 H(n) + 2n^4 - 3n^3 + n^2}{18} + \frac{-6\lfloor \log_3 n \rfloor(n^2 + n) - 6(n^2 + n) - 6n^2 H(n)(n^2 + n) + nH(n)(n^2 + n)}{36} + \frac{3n^2 H(n)(n+1)(2n+1)}{108}$$

$$= \frac{n}{6} + \frac{2n^2}{9} - \frac{n^3}{6} + \frac{n^4}{9} + \frac{n\lfloor \log_3 n \rfloor}{6} + \frac{n^2\lfloor \log_3 n \rfloor}{6} - \frac{2n^2 H(n)}{9} + \frac{n^3 H(n)}{2} + \frac{n^4 H(n)}{18}$$

The dominant term is $\frac{n^4 H(n)}{18}$, since harmonic series has complexity of $H(n) \in \theta(logn)$. We can say that the average complexity is $A(n) \in \theta(n^4 logn)$.

EMRE KILIÇ
KRİSTİNA TRAJKOVSKİ

IDENTIFICATION OF BASIC OPERATION(S)

The operation labeled as (4) is definitely the basic operation. It defines the algorithm, for it gathers all of the loops and executions, and as we have shown in all of the analyses, it has the largest amount of executions compared to (1), (2), and (3). As we will show with graphs in the next section, the results gathered by the analysis closely define the asymptotic behavior of the real time execution as well, so the conclusion is also supported by experimental evidence. Another point is that it shows the difference between different types of inputs realistically, while (1), (2), and (3) are very extreme in that matter (1 is input independent, 2 and 3 do not give the real best case image).

REAL EXECUTION

Best Case

N Size	Time Elapsed
1	0.0000005
5	0.0000031
10	0.0000091
20	0.0000291
30	0.0000942
40	0.0001709
50	0.0002508
60	0.0003397
70	0.0004718
80	0.0005820
90	0.0009472
100	0.0011539
110	0.0014179
120	0.0016549
130	0.0019479
140	0.0022278
150	0.0025828

EMRE KILIÇ
KRİSTİNA TRAJKOVSKİ

Worst Case

N Size	Time Elapsed
1	0.0000010
5	0.0000281
10	0.0003431
20	0.0048862
30	0.0253611
40	0.0782232
50	0.1960173
60	0.3970137
70	0.7485931
80	1.2696068
90	2.0384960
100	3.1446300
110	4.6136451
120	6.5458889
130	9.1026609
140	12.211936
150	16.162601

Average Case

N Size	Time Elapsed
1	0.0000009
5	0.0000118
10	0.0001184
20	0.0017537
30	0.0100558
40	0.0326318
50	0.0797204
60	0.1481916
70	0.2759008
80	0.5063664
90	0.8087902
100	1.2884994
110	1.8014128
120	2.8256861
130	3.7426544
140	4.5465607
150	6.5673308

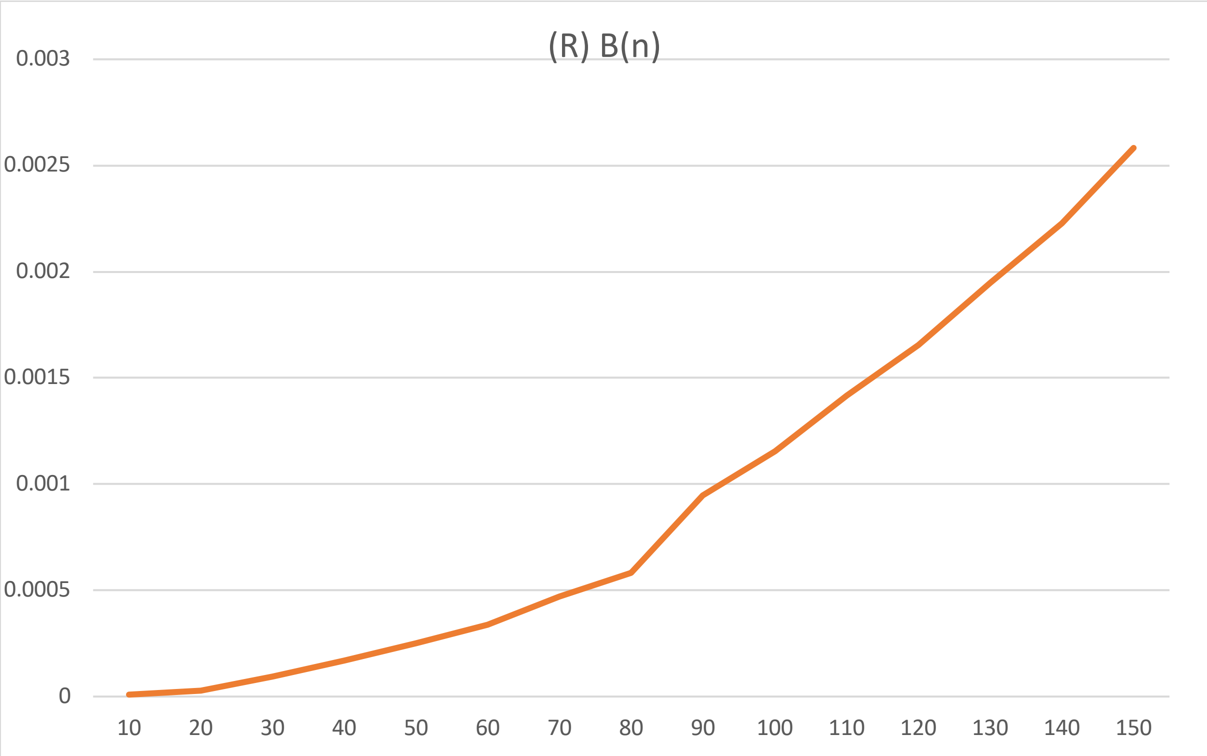
EMRE KILIÇ
KRİSTİNA TRAJKOVSKİ

COMPARISON

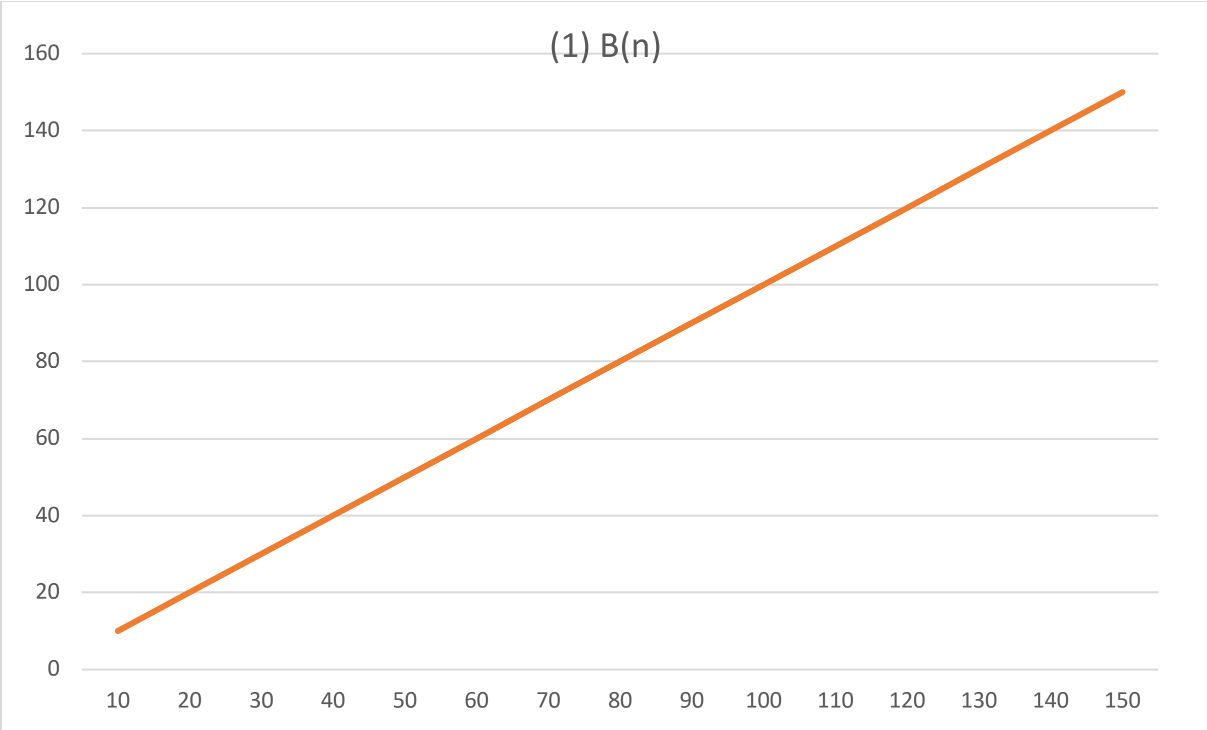
NOTE: ***all graphs have 'n' as their x axis, the real execution time graphs have seconds as their y axis, and the rest of the theoretical analysis graphs have the number of basic operations as their y axis.***

Best Case

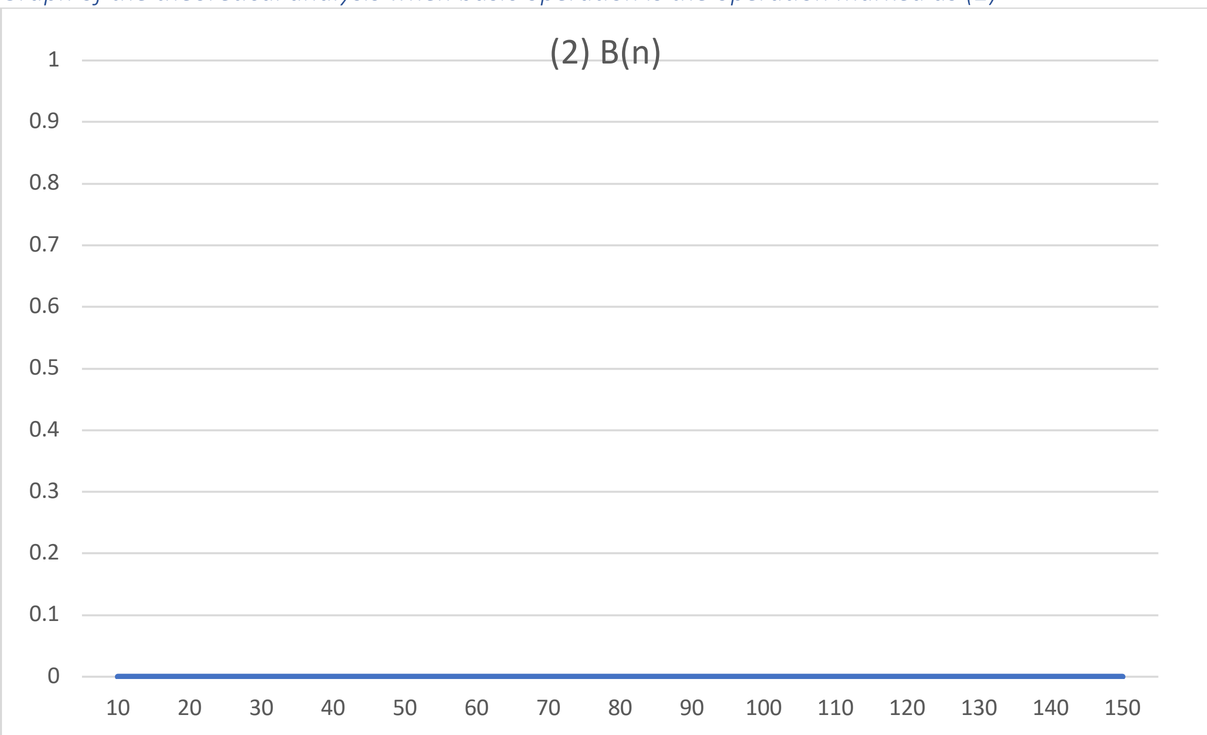
Graph of the real execution time of the algorithm



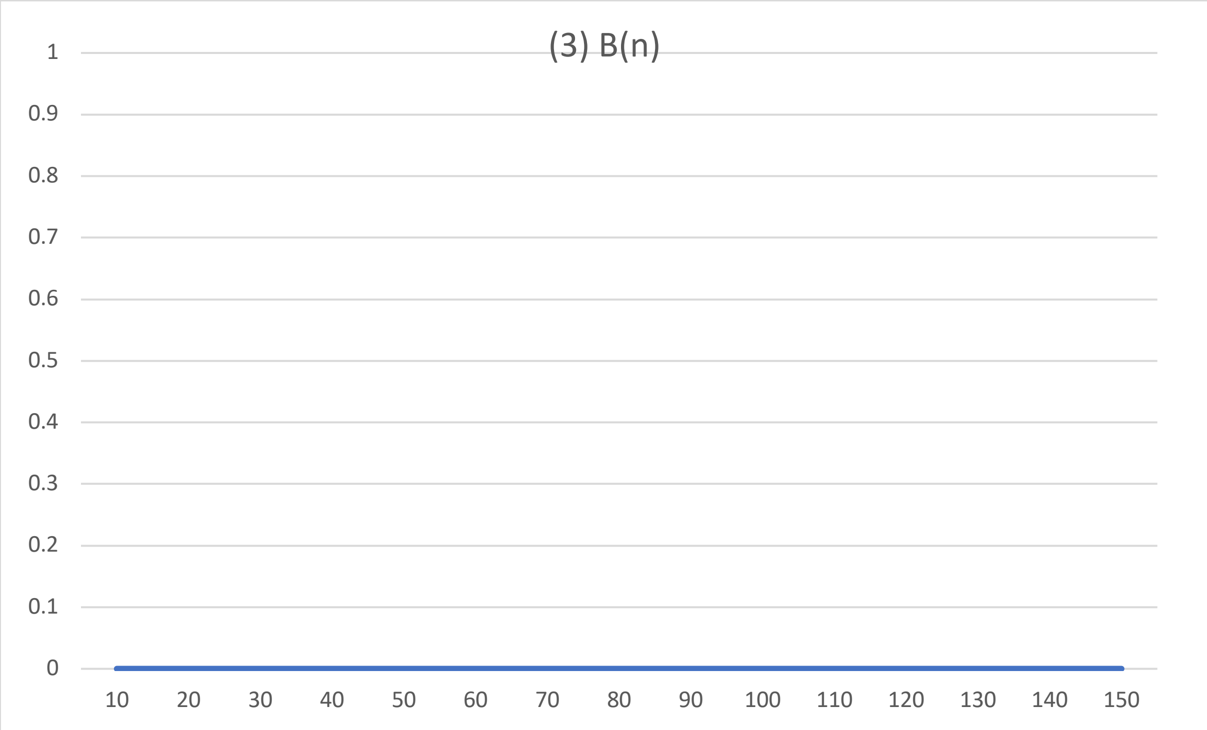
Graph of the theoretical analysis when basic operation is the operation marked as (1)



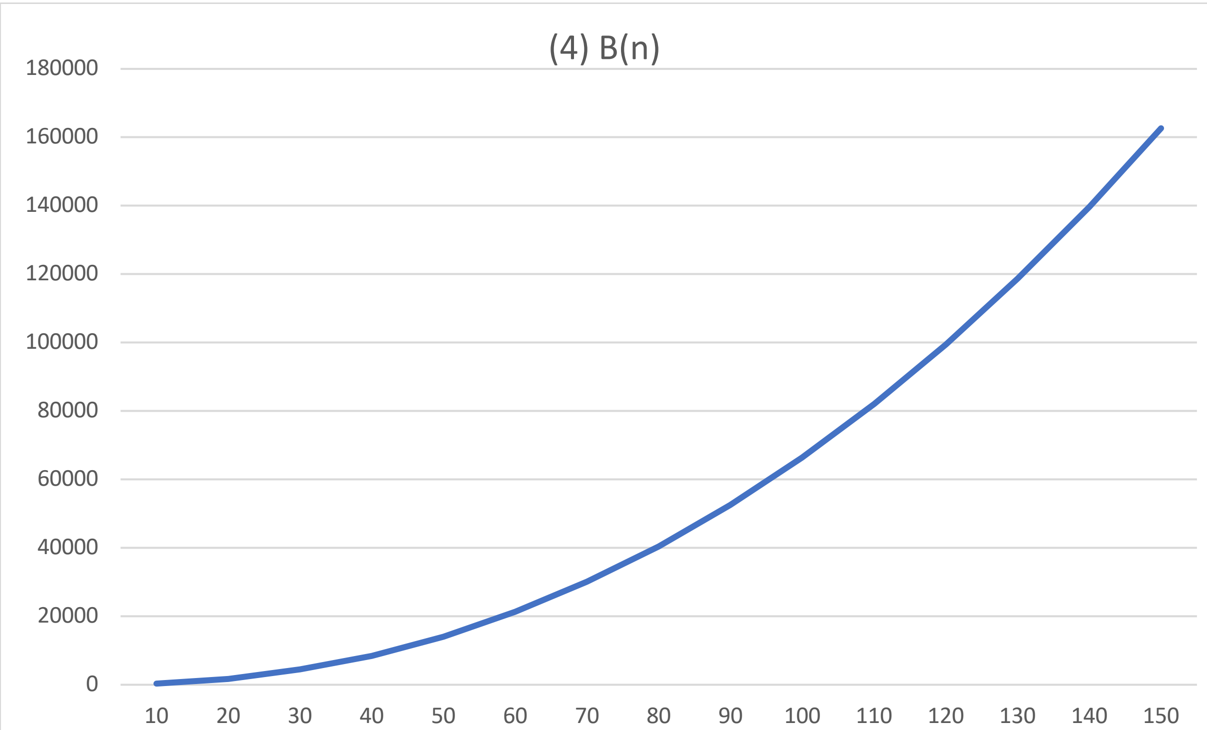
Graph of the theoretical analysis when basic operation is the operation marked as (2)



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

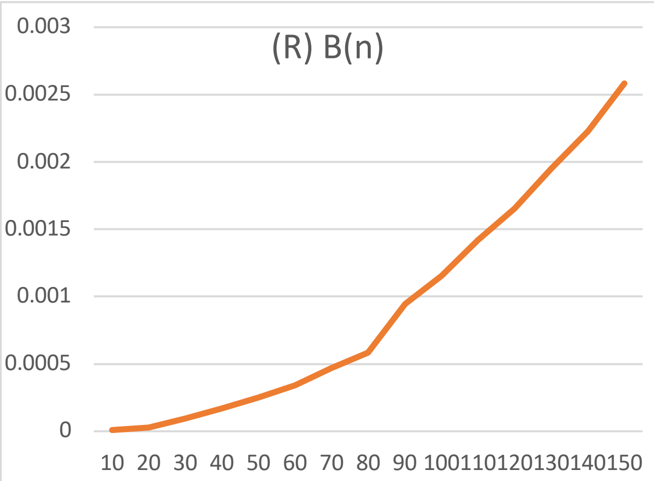


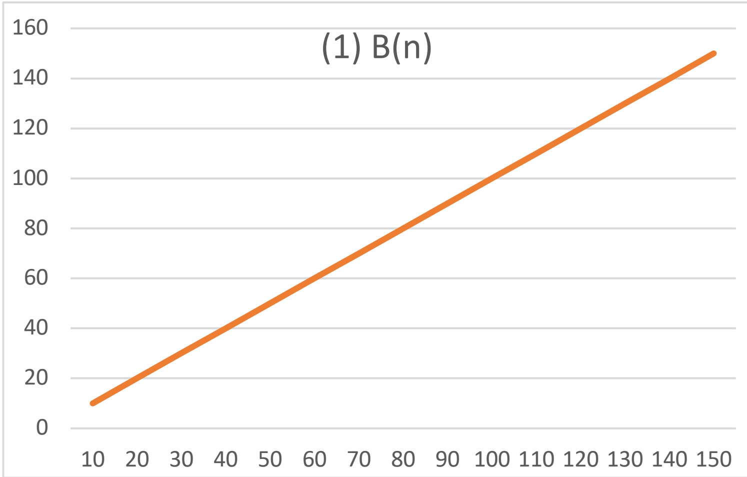
Graph of the theoretical analysis when basic operation is the operation marked as (4)



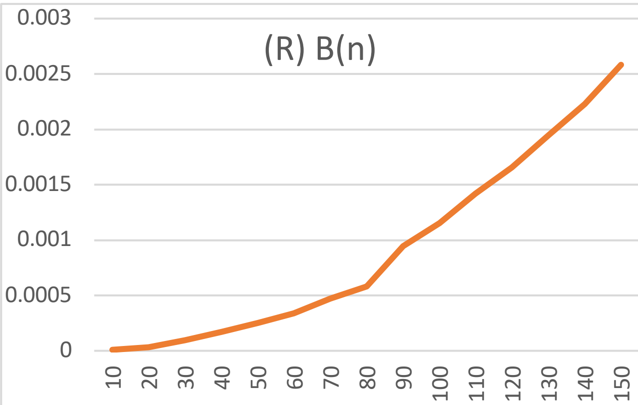
EMRE KILIÇ
KRİSTİNA TRAJKOVSKİ

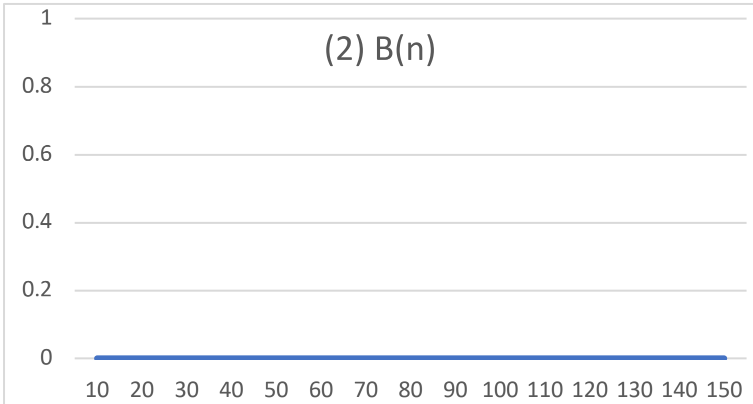
Comments





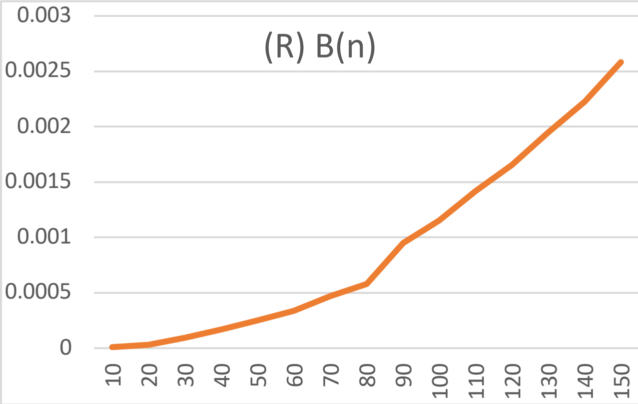
- Comparing the real experiment best case data to the case (1) clearly shows that they are not compatible, for the best case in (1) grows linearly, and the real best case does not. The difference in the shape shows that they do not grow at the same rate and therefore choosing (1) as the real basic operation would be wrong.

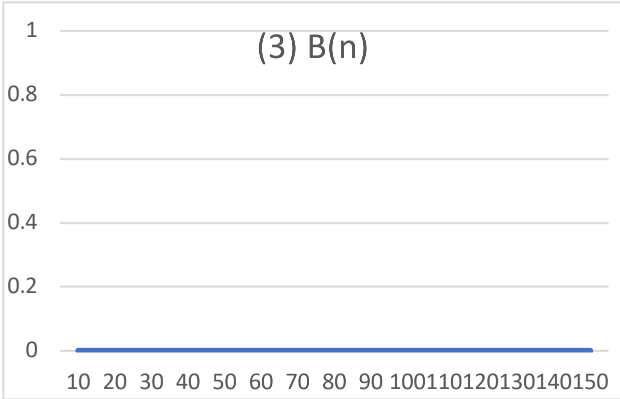




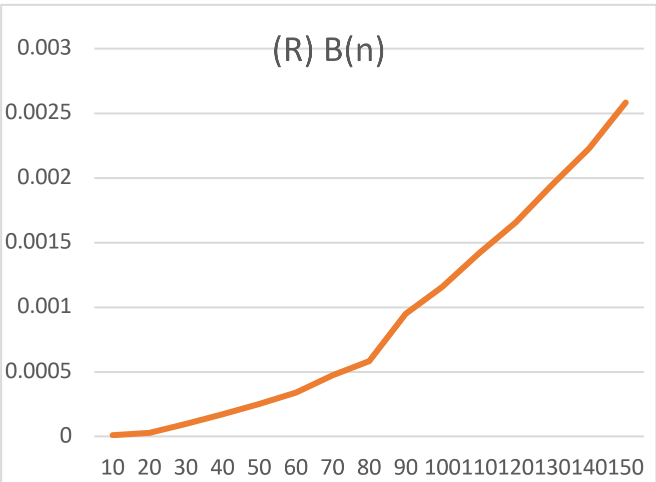
- Comparing the real experiment best case data to the case (2) clearly shows that they are not compatible, for the best case in (2) is constantly 0 and the real data has growth. The difference in the shape shows that they do not grow at the same rate and therefore choosing (2) as the real basic operation would be wrong.

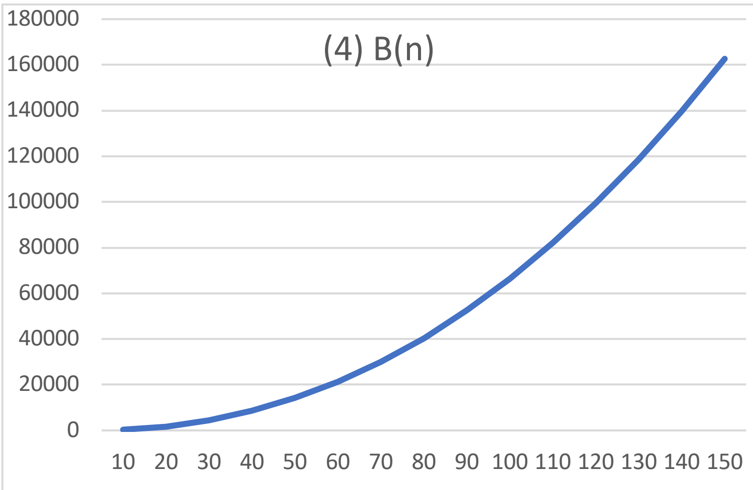
EMRE KILIÇ
KRİSTİNA TRAJKOVSKİ





- Comparing the real experiment best case data to the case (3) clearly shows that they are not compatible, for the best case in (3) is constantly 0 and the real data has growth. The difference in the shape shows that they do not grow at the same rate and therefore choosing (3) as the real basic operation would be wrong.

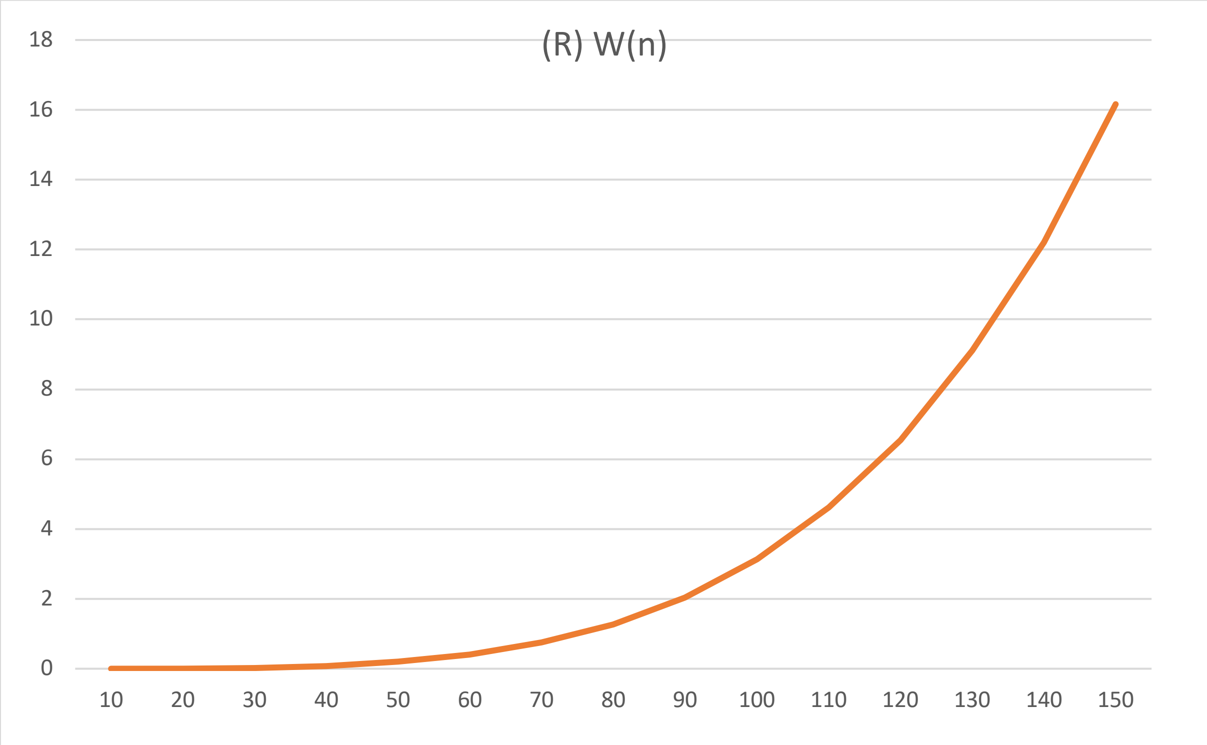




- The expectations have been met by the experimental analysis, for the choice of basic operation as (4) defines the best case behavior in the closest matter by far. Comparing the best cases of the four choices together with the real time results confirm that our analysis for the best case was most likely correct. The best case indeed grows similarly to $n^2 logn$.

Worst Case

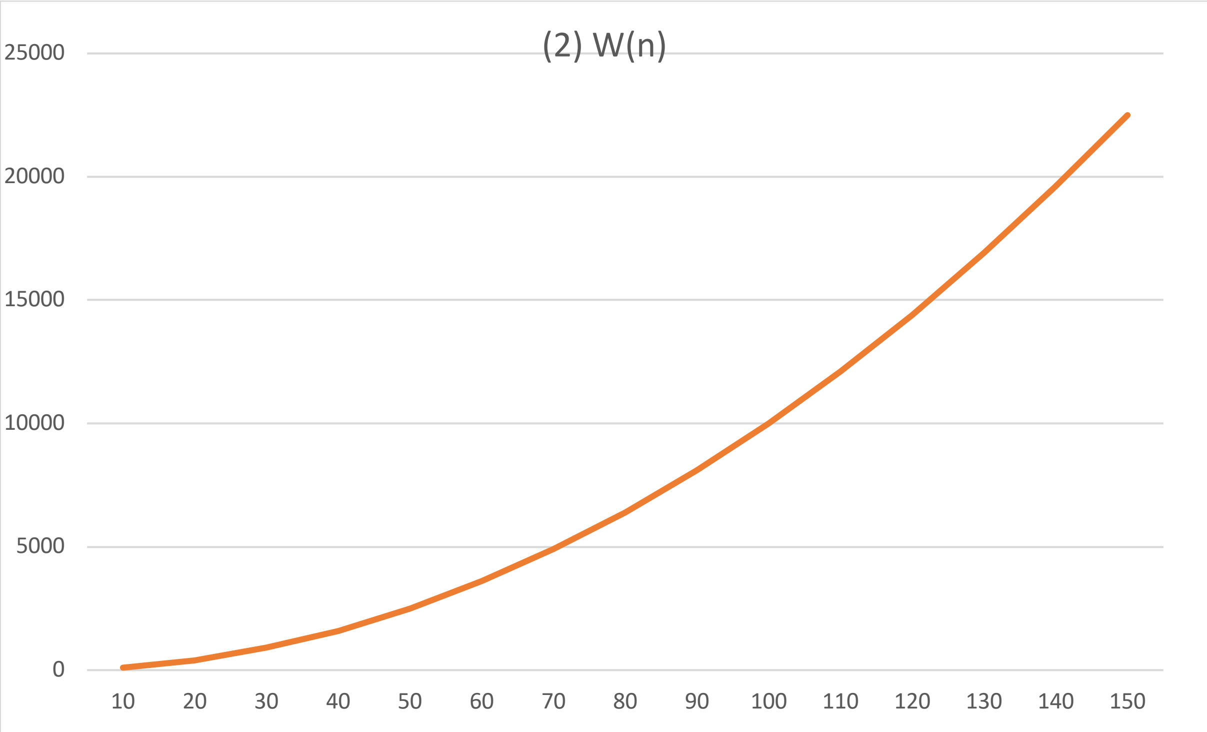
Graph of the real execution time of the algorithm



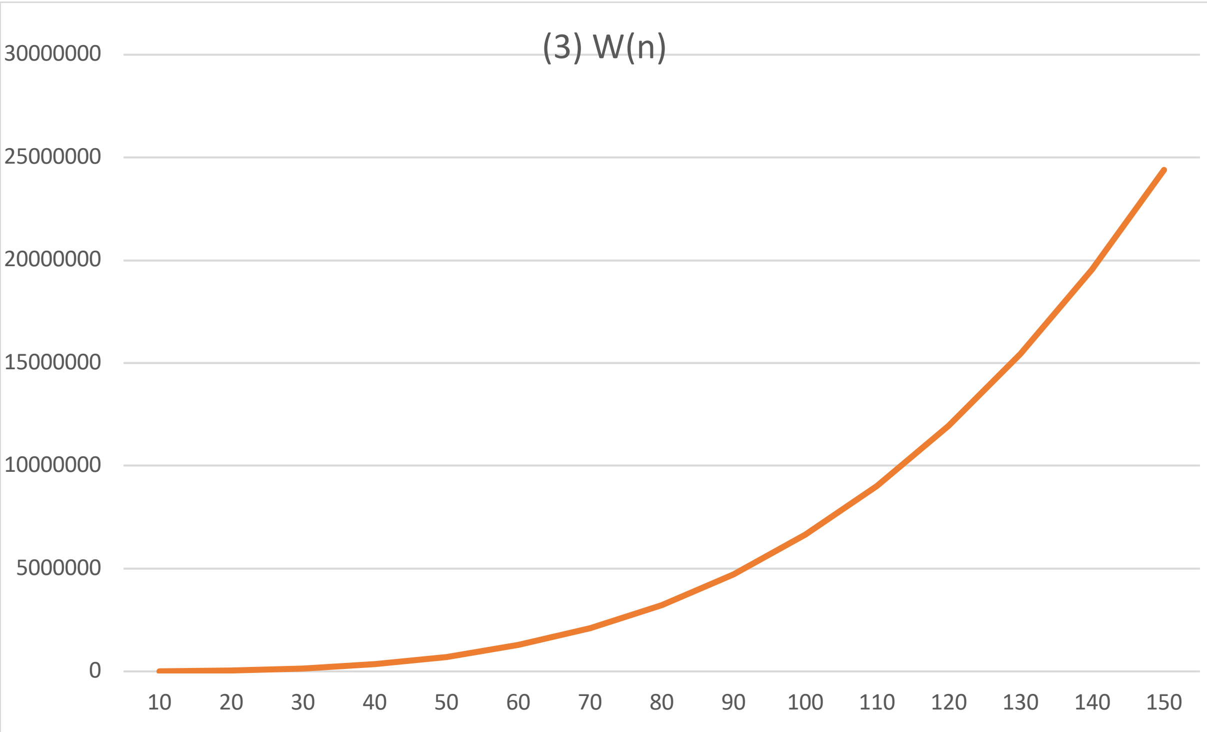
Graph of the theoretical analysis when basic operation is the operation marked as (1)

EMRE KILIÇ
KRİSTİNA TRAJKOVSKİ

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

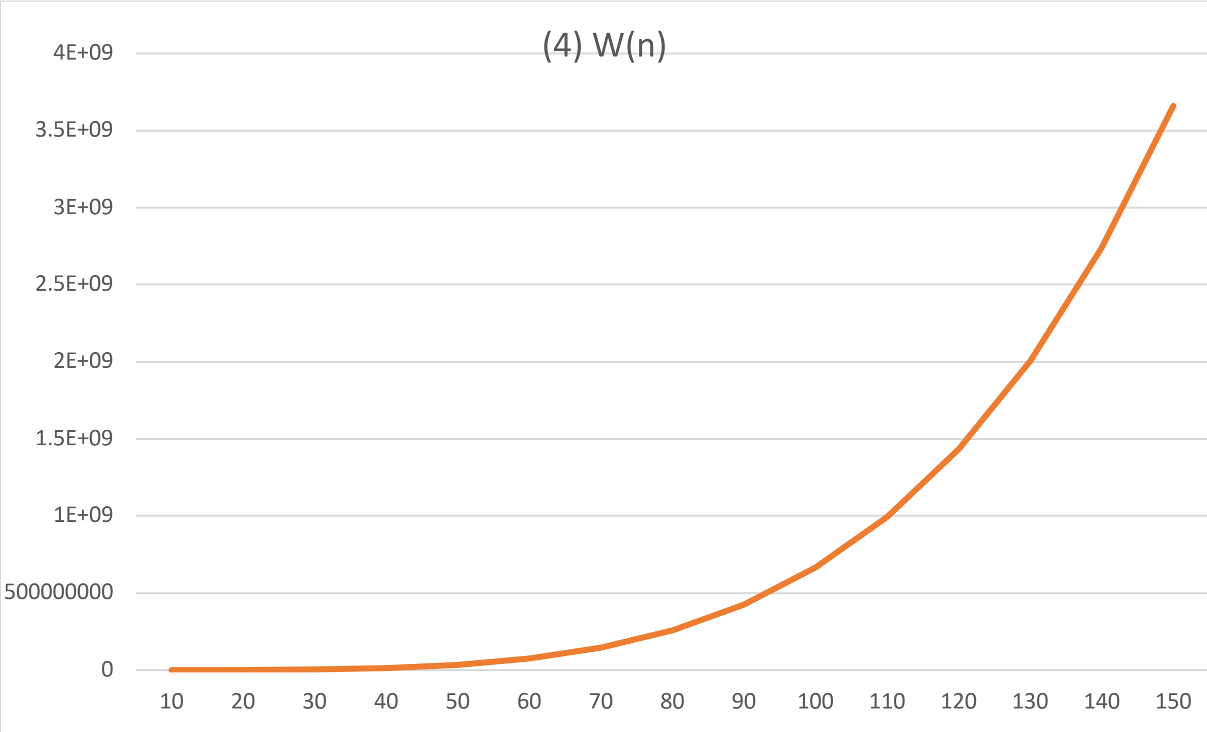


Graph of the theoretical analysis when basic operation is the operation marked as (3)



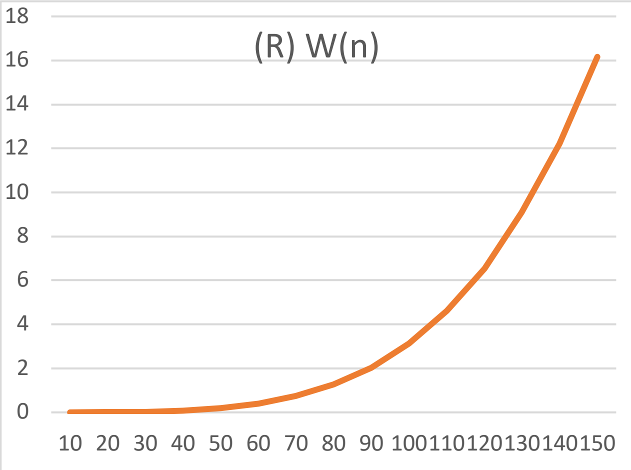
EMRE KILIÇ
KRİSTİNA TRAJKOVSKİ

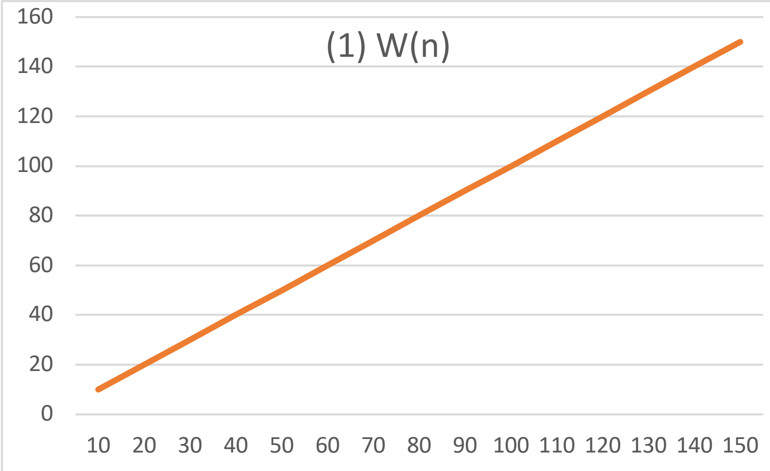
Graph of the theoretical analysis when basic operation is the operation marked as (4)



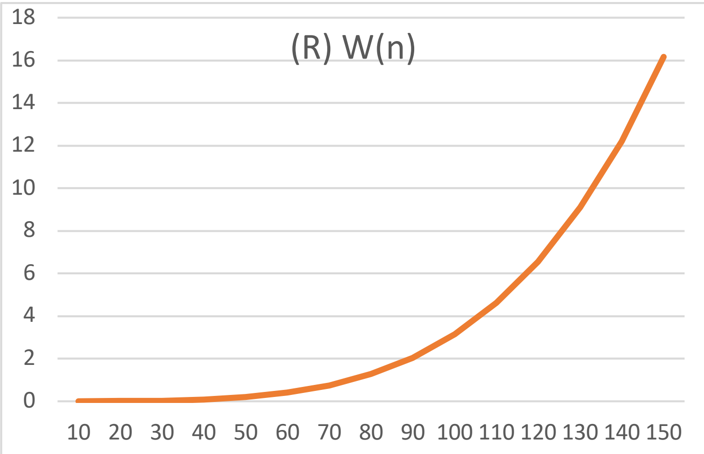
EMRE KILIÇ
KRİSTİNA TRAJKOVSKİ

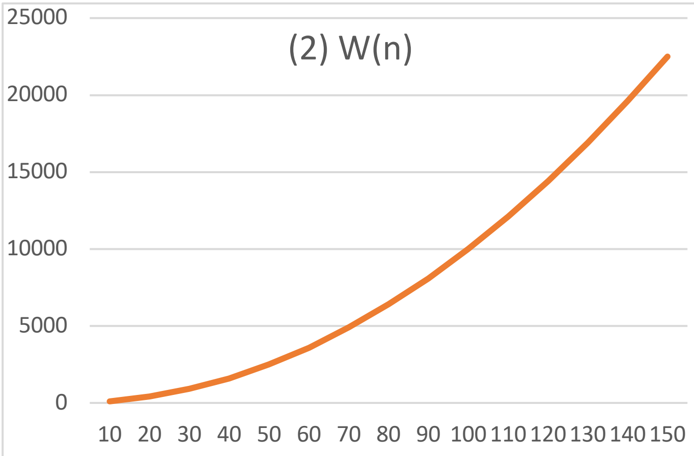
Comments





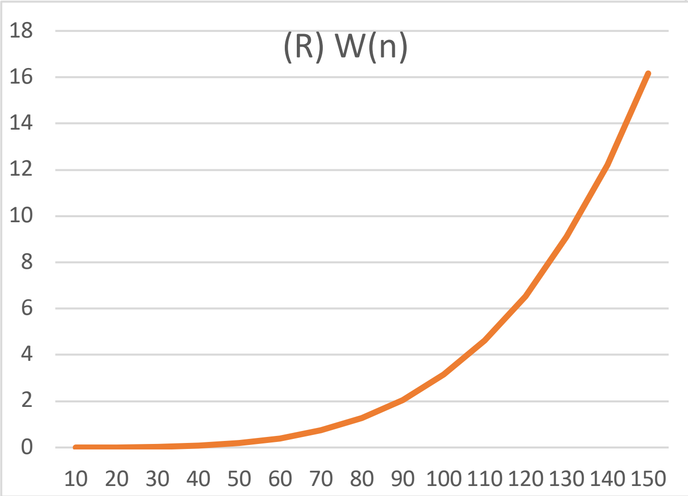
- Similarly to the comparison of B(n), the worst cases of the real data and case (1) have very different growths, for worst case in (1) grows linearly while the real worst case does not. The difference in the shape shows that they do not grow at the same rate and therefore choosing (1) as the real basic operation would be wrong.

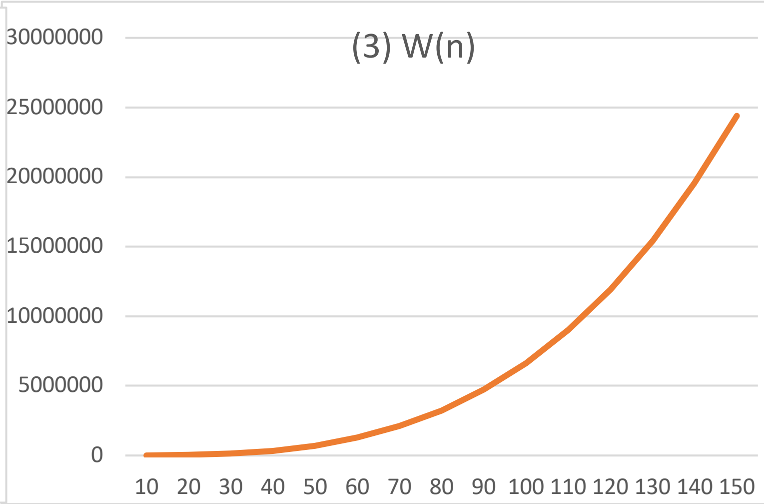




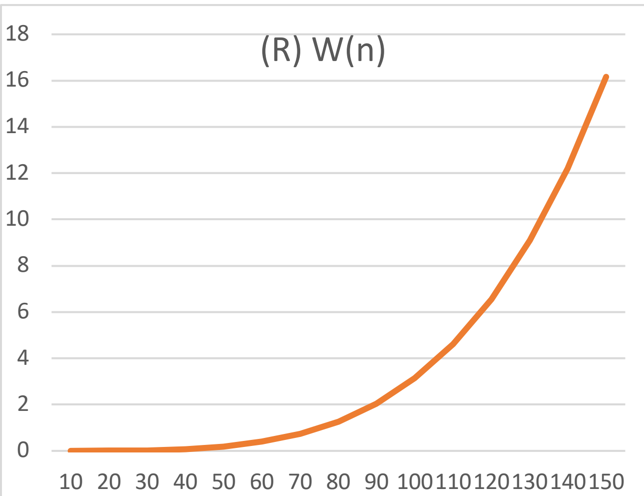
- From the graphs, we may conclude that the real worst-case data grows faster for larger n than what we see in (2). Although definitely closer than case (1), the shape of case(2) would still not depict the behavior of the real time worst case and it would be a wrong choice for the basic operation

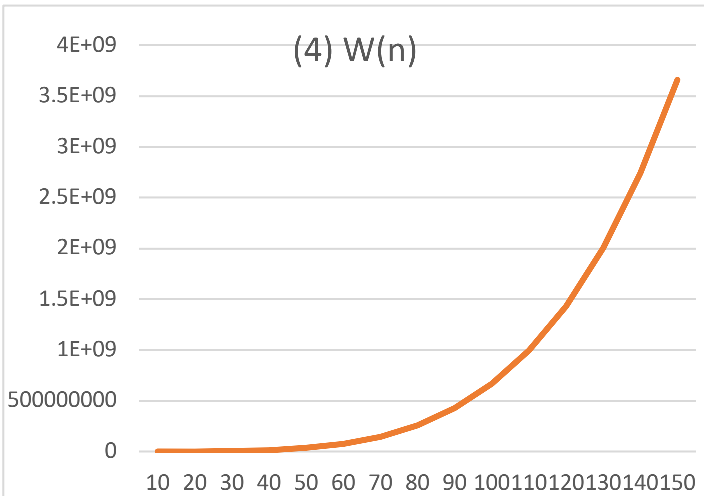
EMRE KILIÇ
KRİSTİNA TRAJKOVSKİ





- Similarly to the previous comparison, the real worst case growth clearly grows faster for large n than in the worst case of (3). Choosing (3) as the basic operation would not realistically depict the real behavior and would be a wrong choice.



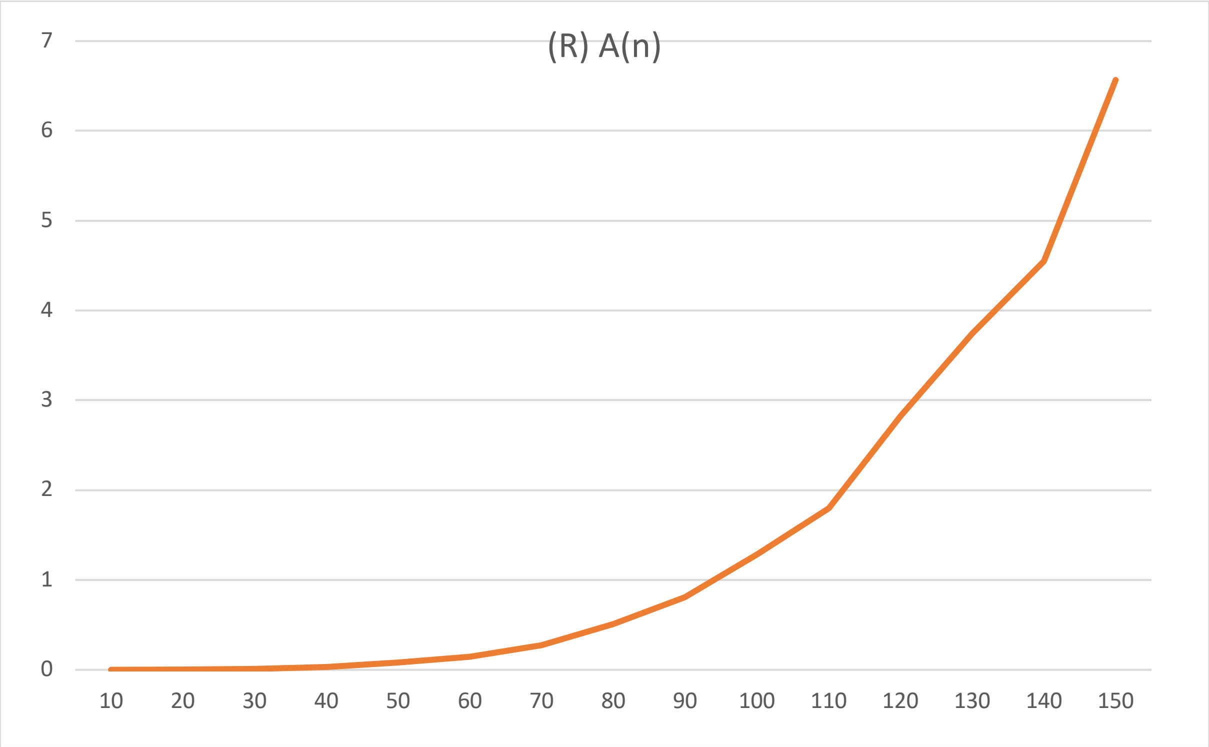


- Similarly to the comparison of the B(n), the choice of basic operations (4) most closely defines the real behavior of the algorithm. Comparing the graphs as well, our worst case analysis was most likely correct; clearly the worst case of the real algorithm grows similarly to $n^4 logn$.

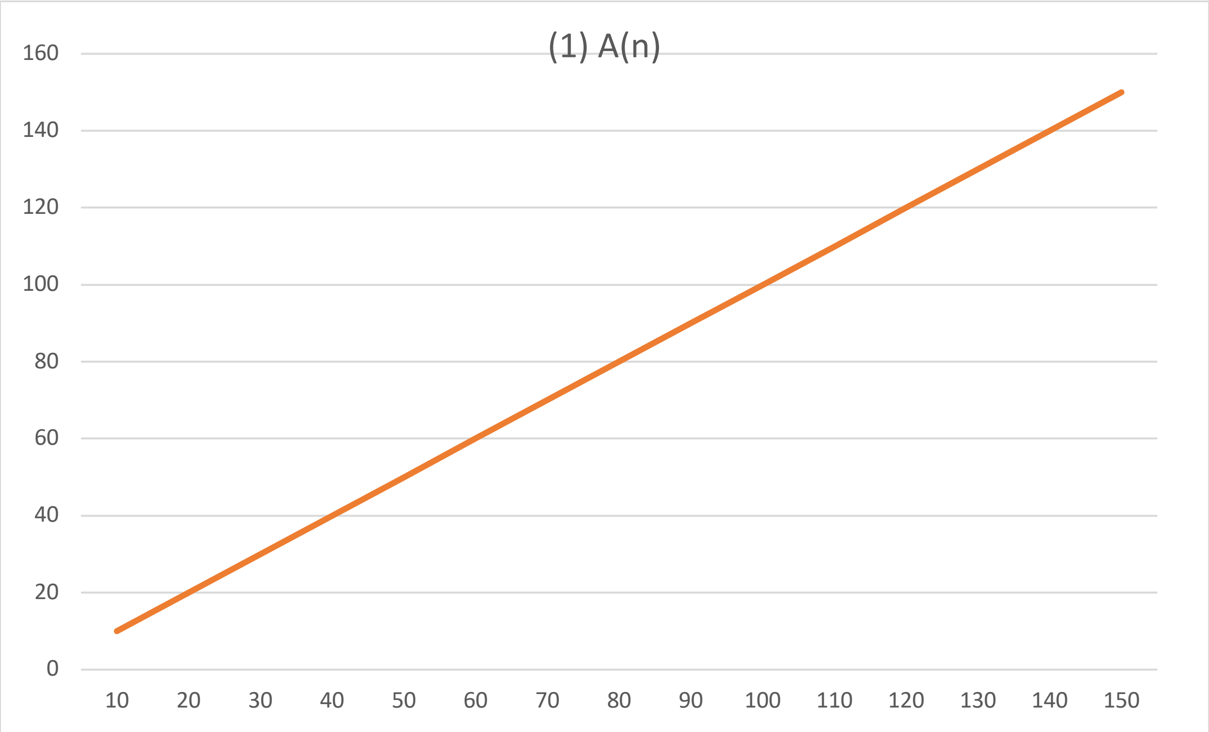
EMRE KILIÇ
KRİSTİNA TRAJKOVSKİ

Average Case

Graph of the real execution time of the algorithm

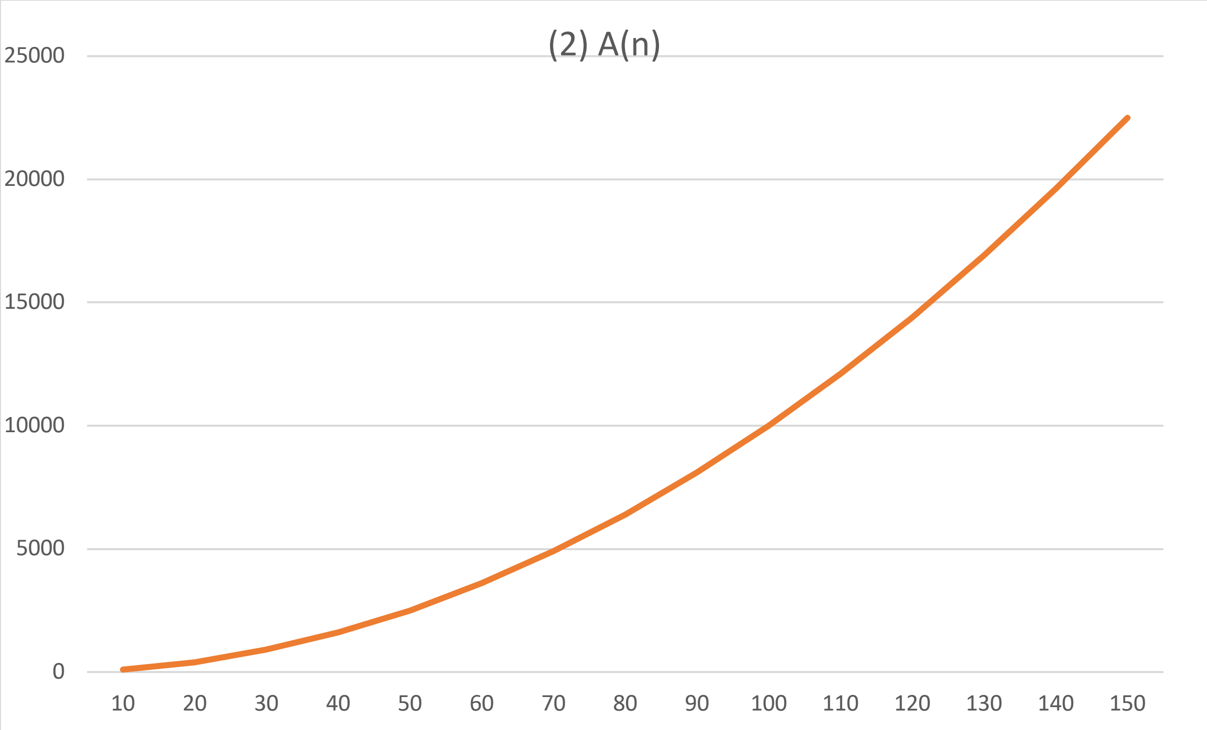


Graph of the theoretical analysis when basic operation is the operation marked as (1)

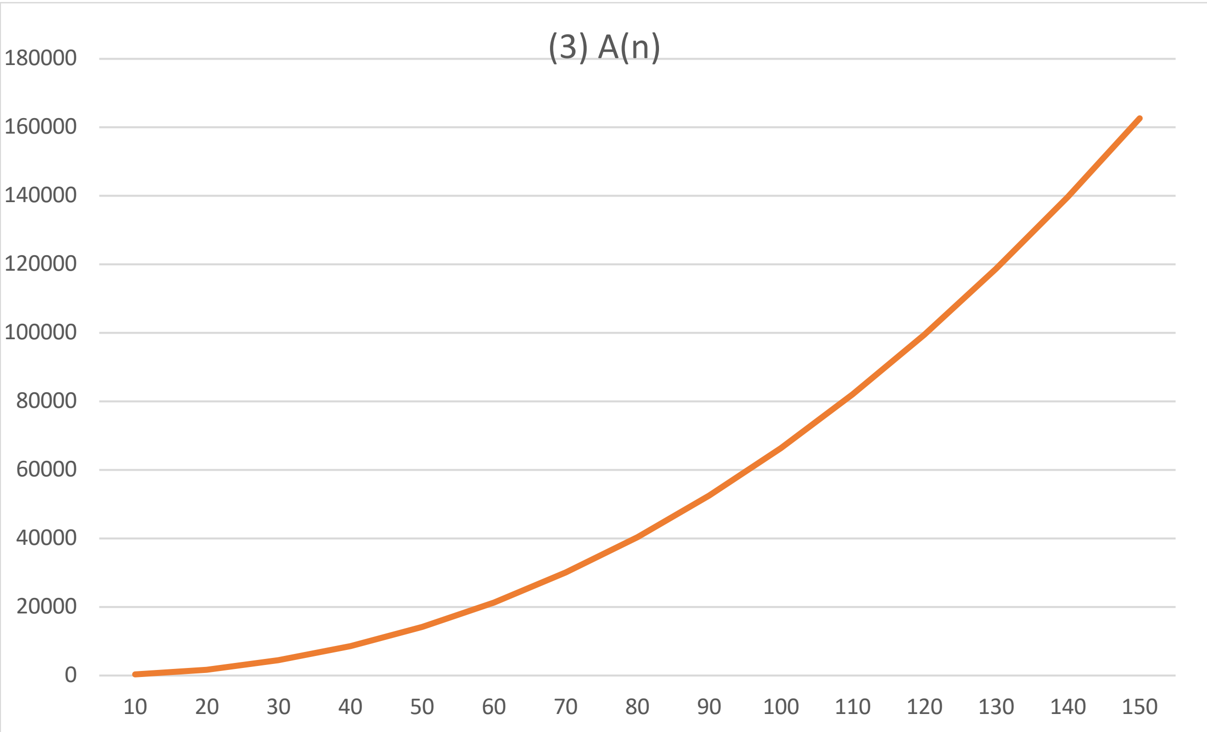


EMRE KILIÇ
KRİSTİNA TRAJKOVSKİ

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

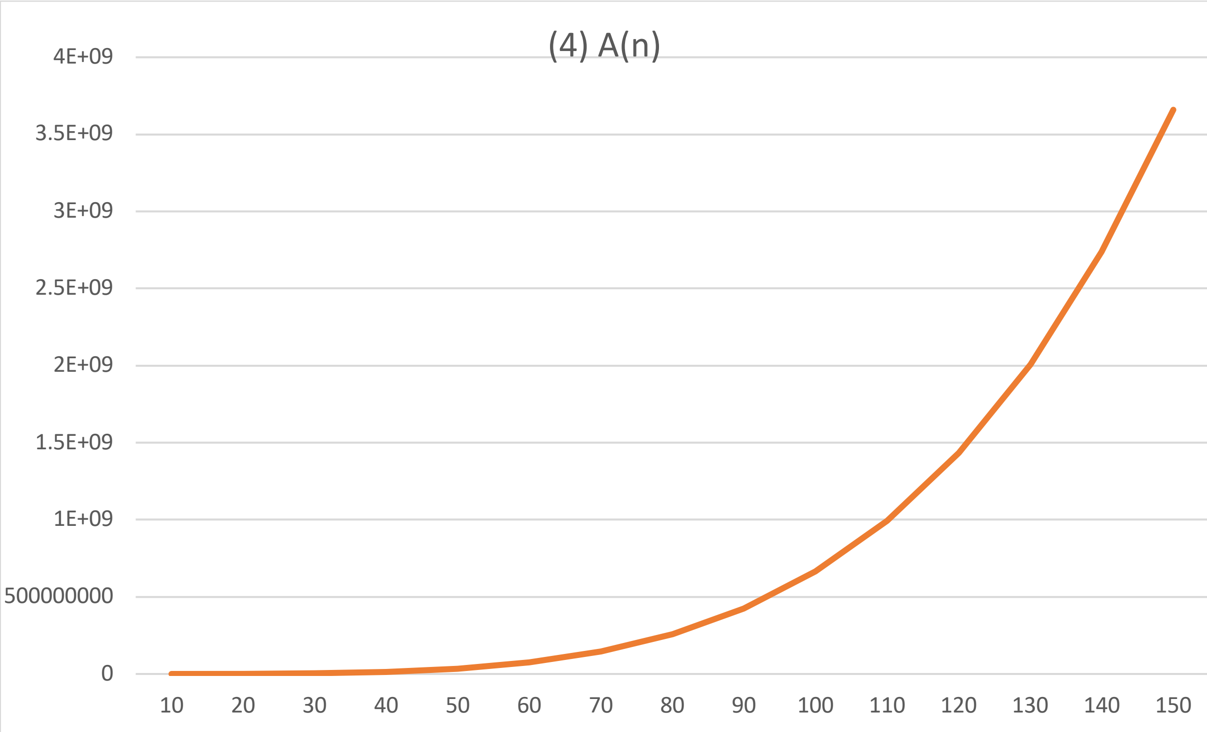


Graph of the theoretical analysis when basic operation is the operation marked as (3)



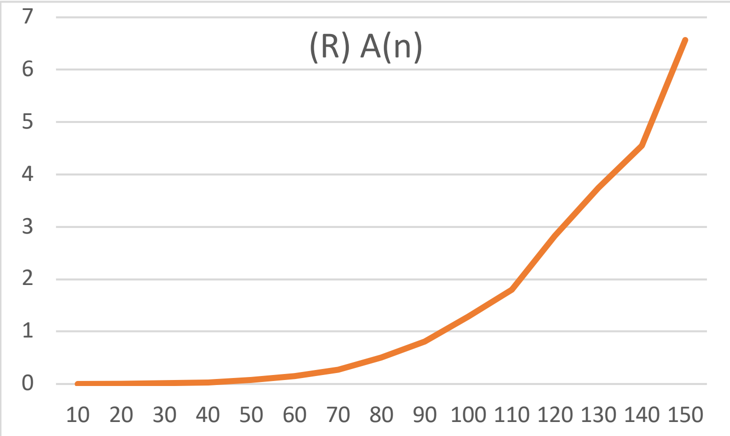
EMRE KILIÇ
KRİSTİNA TRAJKOVSKİ

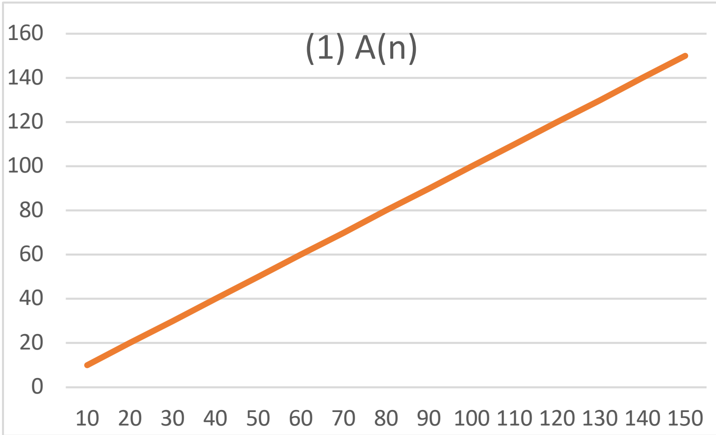
Graph of the theoretical analysis when basic operation is the operation marked as (4)



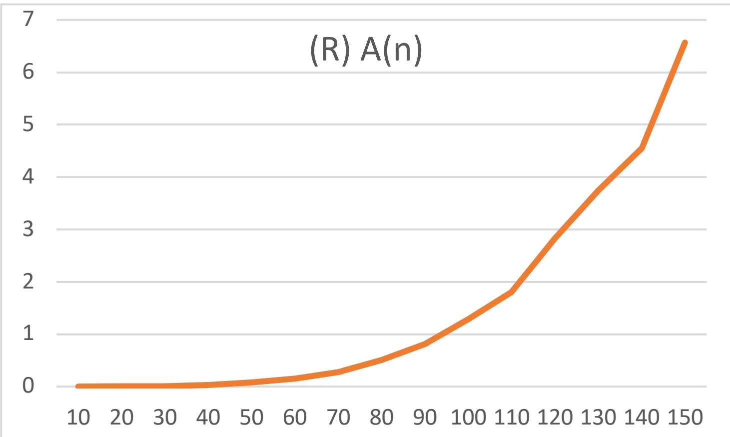
EMRE KILIÇ
KRİSTİNA TRAJKOVSKİ

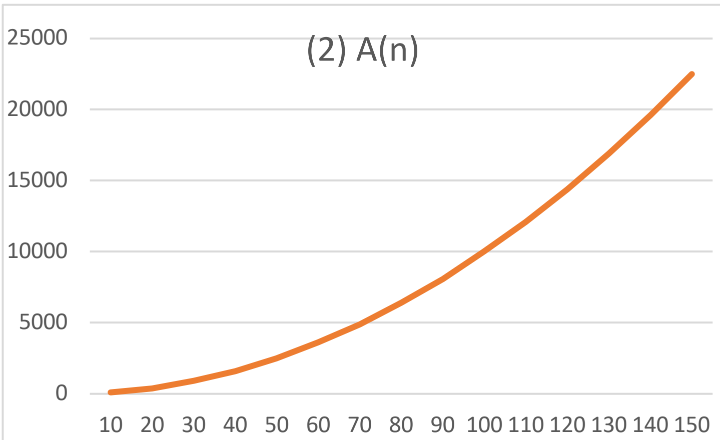
Comments





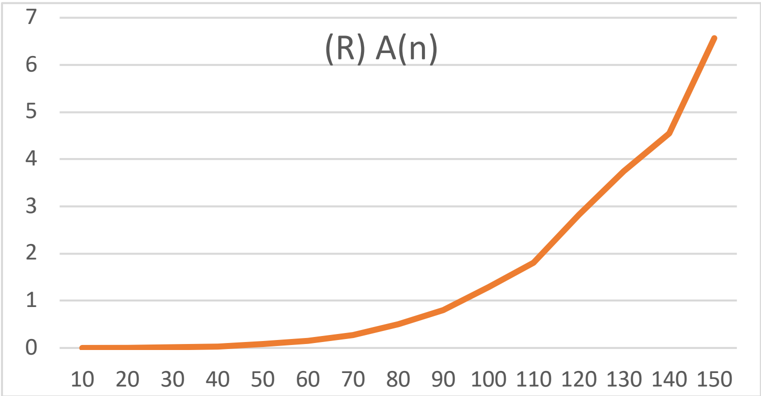
- Similarly to the comparison of W(n), the average cases of the real data and case (1) have very different growths, for average case in (1) grows linearly while the real average case does not. The difference in the shape shows that they do not grow at the same rate and therefore choosing (1) as the real basic operation would be wrong.

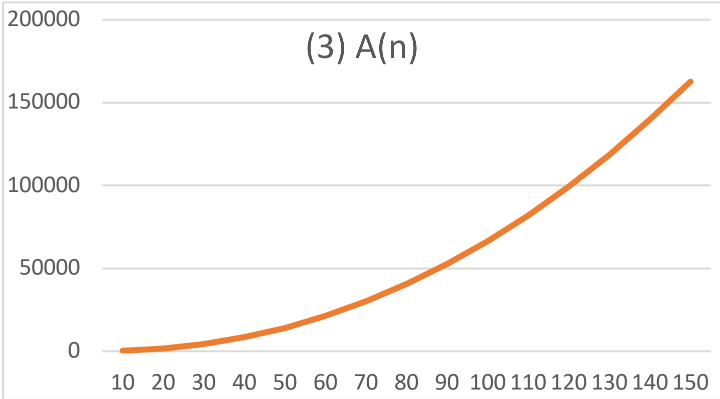




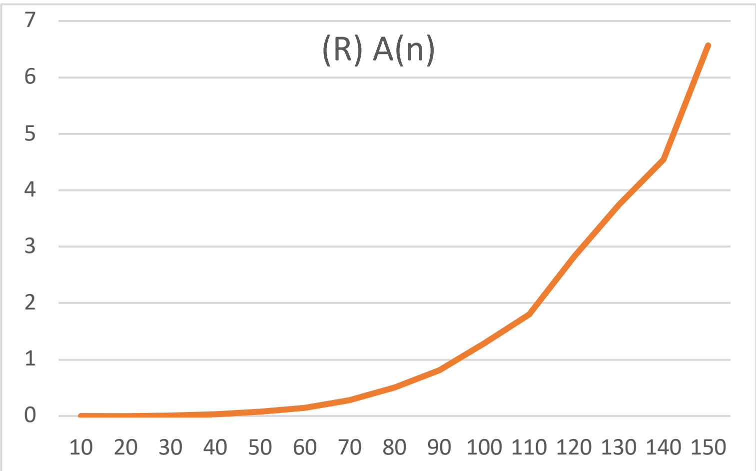
- Similarly to the comparison of W(n), the average cases of the real data and case (1) have somewhat different growths, for average case in (2) grows slower than the real average case. The difference in the shape shows that they do not grow at the same rate and therefore choosing (2) as the real basic operation would be wrong.

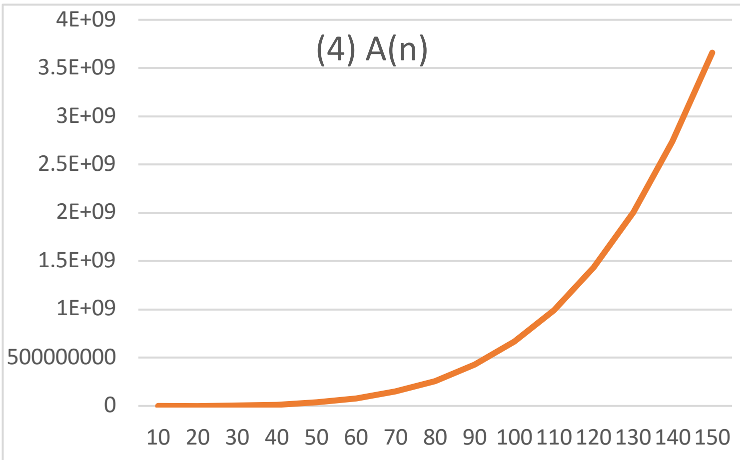
EMRE KILIÇ
KRİSTİNA TRAJKOVSKİ





- Similarly to the comparison of W(n), the average cases of the real data and case (3) have somewhat different growths, for average case in (3) grows slower than the real average case. The difference in the shape shows that they do not grow at the same rate and therefore choosing (3) as the real basic operation would be wrong.





- Although the average case is more prone to fluctuations in the real execution (we may see that from the graph not being as smooth as others), still the choice of (4) as the basic operations relates the closest to the real experiment data. Clearly the average case of the real algorithm grows similarly to $n^4 logn$ and our analysis is most likely correct.