KINGDOM OF SAUDI ARABIA
Ministry of Higher Education
Al-Imam Mohammad University
College of Computer & Information Sciences

المملكة العربية السعودية
وزارة التعليم العالي
جامعة الإمام محمد بن سعود الإسلامية
كلية علوم الحاسب والمعلومات

First term 1442/2020

Algorithm design and analysis (CS- 215)

BSCS- Section: 171

Algorithms Project

Submitted By

Student 1 Abdulelah Alanzi (439015929)

Student 2 Rayan almengash (439016742)

Student 3 Saleh alsaeed (439016667)

Supervisor

D. Hashimi Ahmed Nasser

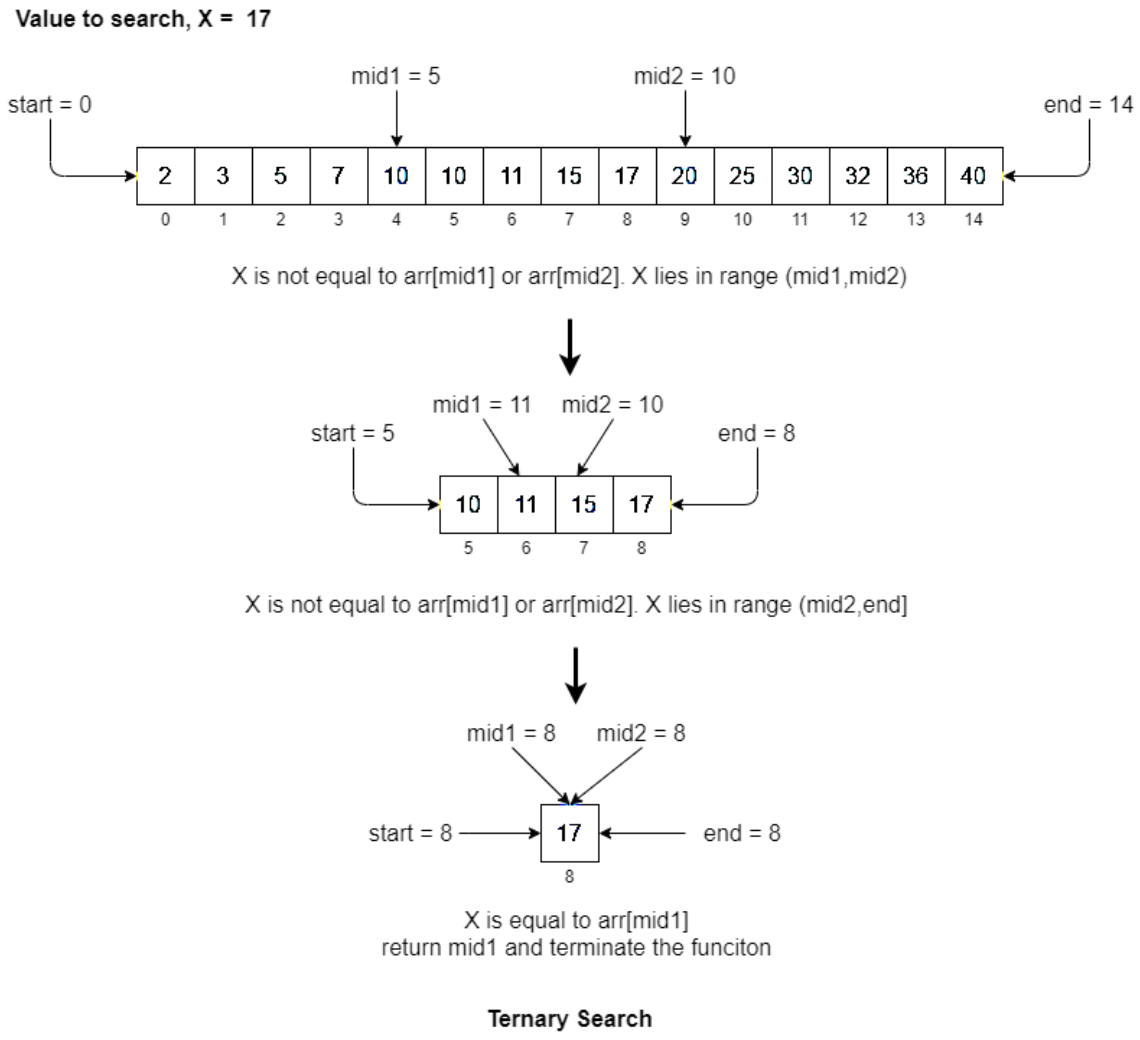
Date: 25/11/2020

Introduction.

In this project we will talk about the theoretical part of the Ternary search algorithm which includes the design and the recurrence relation and the analysis then we will move into the practical part which will include the implementation of the linear search algorithm, binary search algorithm and Ternary search algorithm then we will do the analysis using tables and graph.

Theoretical Part.

The Design of the Ternary search algorithm



Pseudocode of the TSA:

```
arr: Array we want to search
x: value we want to search in arr
Start: start of the array
End: end of the array
Integer TSA( arr , x , start, end)
        WHILE start <= end DO
                middleOne := (end - start)/3 + start
                middleTwo := 2*(end - start)/3 + start
                IF x == arr[middleOne] THEN
                        return middleOne
                ELSE IF x == arr[middleTwo] THEN
                        return middleTwo
                ELSE IF x < arr[middleOne] THEN
                            TSA(arr, x, start, middleOne-1)
                ELSE IF x < arr[middleTwo] THEN
                            TSA(arr, x, middleOne+1, middleTwo–1)
                ELSE
                            TSA(arr, x, middleTwo+1, end)

                END ELSEIF
        END WHILE
        return -1
END FUNCTION
```

The Recurrence Relation of the Ternary search algorithm.

Recurrence relation in the best case is **T(n) = T(n/3) + O(1)**

In the worst case it will be **T(n) = T(2n/3) + O(1)**

The analysis of the Ternary search algorithm

Using Master theorem, it will result in O(Log n)

Practical Part.

This practical part will be about the implementation and the output

1- Linear Search Algorithm

The Implementation of the LSA.

```
   public static int sequenceSearch( int l, int r, int key,Integer arr[])
{
      sequenceCount++;
      if (r < l)
         return -1;
      if (arr[l] == key)
         return l;
      if (arr[r] == key)
         return r;
      return sequenceSearch( l+1, r-1, key,arr);
}
```

2- Binary Search Algorithm

The Implementation of the BSA.

```
   public static int binarySearch(int l,int r, int key ,Integer arr[])
{
    binaryCount++;

    if (r >= l && l<arr.length-1) {

        int mid = l + (r - l) / 2;

        if (arr[mid] == key)
            return mid;
        if (arr[mid] > key)
            return binarySearch( l, mid - 1, key,arr);
        return binarySearch(mid + 1, r, key,arr);
    }
    return -1;
}
```

3- Ternary Search Algorithm

The Implementation of the TSA.

```java
   public static int ternarySearch(int l, int r, int key, Integer [] ar)
  {
    if (r >= l) {

        int middleOne = l + (r - l) / 3;
        int middleTwo = r - (r - l) / 3;

        if (ar[middleOne] == key)
        {    tenaryCount++;
            return middleOne;
        }
        if (ar[middleTwo] == key)
        {    tenaryCount++;
            return middleTwo;
        }

        if (key < ar[middleOne]) {
             tenaryCount++;
            return ternarySearch(l, middleOne - 1, key, ar);
        }
        else if (key > ar[middleTwo]) {
             tenaryCount++;
            return ternarySearch(middleTwo + 1, r, key, ar);
        }
        else {

           tenaryCount++;
            return ternarySearch(middleOne + 1, middleTwo - 1, key, ar);
        }
    }
    return -1;
}
```

The run Screen shot of the TSA, BSA and LSA algorithm.

```
---- When N = 100 ----
[Best case] Tenary is 2 , Binary is 3 , Sequence is 1
[Avg case] Tenary is 3 , Binary is 5 , Sequence is 22
[Worst case] Tenary is 5 , Binary is 5 , Sequence is 50

---- When N = 200 ----
[Best case] Tenary is 3 , Binary is 3 , Sequence is 6
[Avg case] Tenary is 4 , Binary is 6 , Sequence is 58
[Worst case] Tenary is 5 , Binary is 6 , Sequence is 98

---- When N = 300 ----
[Best case] Tenary is 3 , Binary is 4 , Sequence is 3
[Avg case] Tenary is 4 , Binary is 7 , Sequence is 70
[Worst case] Tenary is 6 , Binary is 7 , Sequence is 136

---- When N = 400 ----
[Best case] Tenary is 3 , Binary is 4 , Sequence is 9
[Avg case] Tenary is 4 , Binary is 7 , Sequence is 83
[Worst case] Tenary is 6 , Binary is 7 , Sequence is 198

---- When N = 500 ----
[Best case] Tenary is 3 , Binary is 7 , Sequence is 4
[Avg case] Tenary is 5 , Binary is 7 , Sequence is 108
[Worst case] Tenary is 6 , Binary is 7 , Sequence is 237

---- When N = 600 ----
[Best case] Tenary is 2 , Binary is 7 , Sequence is 14
[Avg case] Tenary is 5 , Binary is 8 , Sequence is 174
[Worst case] Tenary is 6 , Binary is 8 , Sequence is 292

---- When N = 700 ----
[Best case] Tenary is 1 , Binary is 5 , Sequence is 11
[Avg case] Tenary is 5 , Binary is 8 , Sequence is 159
[Worst case] Tenary is 6 , Binary is 8 , Sequence is 335

---- When N = 800 ----
[Best case] Tenary is 3 , Binary is 7 , Sequence is 26
[Avg case] Tenary is 5 , Binary is 8 , Sequence is 178
[Worst case] Tenary is 7 , Binary is 8 , Sequence is 392

---- When N = 900 ----
[Best case] Tenary is 5 , Binary is 3 , Sequence is 14
[Avg case] Tenary is 5 , Binary is 8 , Sequence is 219
[Worst case] Tenary is 7 , Binary is 8 , Sequence is 448

---- When N = 1000 ----
[Best case] Tenary is 3 , Binary is 5 , Sequence is 8
[Avg case] Tenary is 6 , Binary is 8 , Sequence is 195
[Worst case] Tenary is 7 , Binary is 8 , Sequence is 461
```

This will run the three algorithms in different N sizes from 100 to 1000 and search for 20 different random generated numbers and show the comparisons.

Table for the **Best case** in each algorithm.

Strategy	N= 100	N= 200	N= 300	N= 400	N= 500	N= 600	N= 700	N= 800	N= 900	N= 1000
TSA	2	3	3	3	3	2	1	3	5	3
BSA	3	3	4	4	7	7	5	7	3	5
LS	1	6	3	9	4	14	11	26	14	8

Table for the **Average case** in each algorithm.

Strategy	N= 100	N= 200	N= 300	N= 400	N= 500	N= 600	N= 700	N= 800	N= 900	N= 1000
TSA	3	4	4	4	5	5	5	5	5	6
BS	5	6	7	7	7	8	8	8	8	8
LS	22	58	70	83	108	174	159	178	219	195

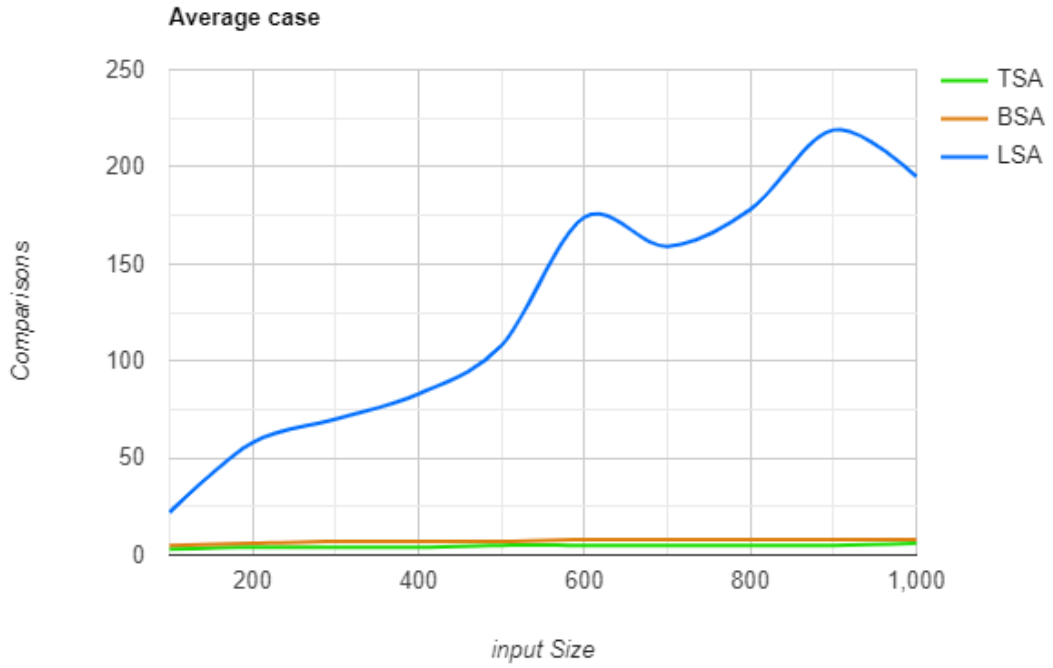
Table for the **Worst case** in each algorithm.

Strategy	N= 100	N= 200	N= 300	N= 400	N= 500	N= 600	N= 700	N= 800	N= 900	N= 1000
TSA	5	5	6	6	6	6	6	7	7	7
BSA	5	6	7	7	7	8	8	8	8	8
LS	50	98	136	198	237	292	335	392	448	461

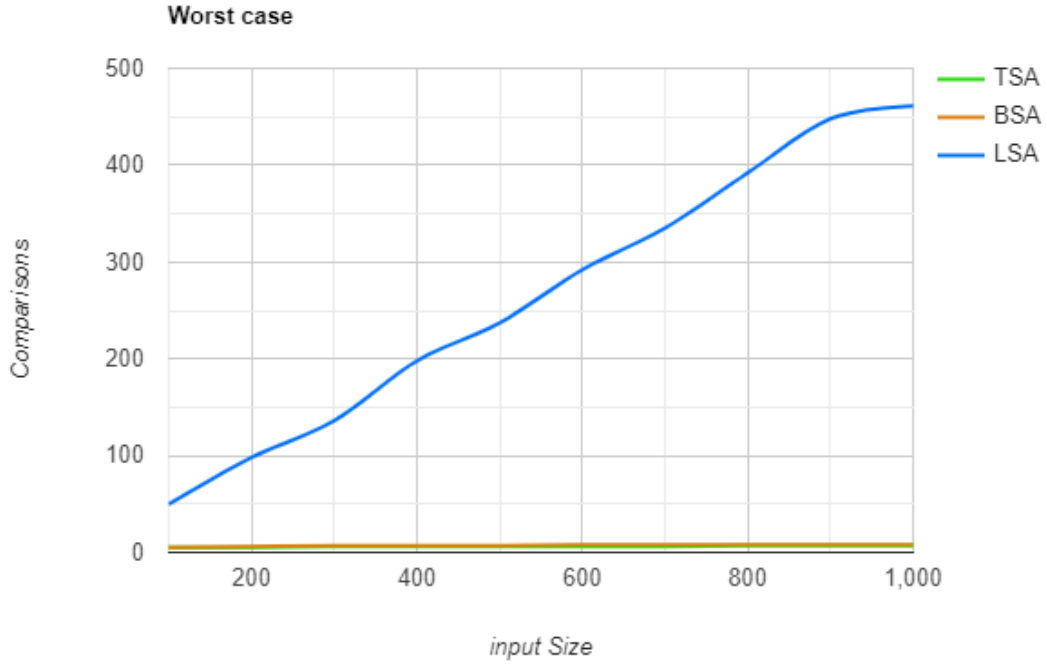
Graphs

Graph for the best case table A

Graph for the average case table B.



Graph for the worst case table C



The theoretical part and the practical part are correlated here,

because if we take only the practical part in this project we can derive the theoretical part easily and it will match the theoretical part provided here.

In Ternary Search, we divide our array into three parts by taking two mid and discard two-third of our search space at each iteration. So, at first look it seems that ternary search might be faster than binary search as its time complexity should be.

In ternary and binary at each iteration ternary search makes at most 4 comparisons as compared to binary search which only makes 2 comparisons. So, it seems the ternary search does a smaller number of comparisons as it makes Log3n recursive calls, but binary search makes Log2n recursive calls.

However, in the sequence search it does more comparisons than the ternary and binary, that is because it is linear and it makes N recursive calls.

Conclusion

In conclusion, this project explained the theoretical part of the TSA which includes the design, recurrence relation and the analysis of it, also it explained the practical part of the TSA, BSA and LSA including pictures and outputs, then this project provided tables, graph, and the correlation between them and the explanation of the algorithms.