

Istanbul Technical University



Analysis of Algorithms I

BLG 335E

Homework 1

Tevfik Özgü

150180082

05 December 2020

1. CONTENT OF TABLE

| | | |
|--------|--|----|
| 1. | CONTENT OF TABLE | 2 |
| 2. | FIGURES..... | 3 |
| 3. | TABLES..... | 4 |
| 4. | EQUATIONS | 5 |
| 5. | INTRODUCTION | 6 |
| 6. | PART 1 | 7 |
| 7. | PART 2 | 9 |
| 1.1. | UPPER BOUNDS AND PROOFS..... | 9 |
| 1.1.1. | Best Case | 9 |
| 1.1.2. | Worst Case..... | 10 |
| 1.1.3. | Average Case..... | 11 |
| 1.2. | DIFFERENT APPROACH | 13 |
| 1.2.1. | Explanation and Simulation..... | 13 |
| 1.2.2. | Sorting Algorithms That Fit These Approach | 16 |
| 1.3. | RUNNING AND COMMENTING ALGORITHM ON UNSORTED DATASET | 16 |
| 1.3.1. | Running on 10 Sales | 17 |
| 1.3.2. | Running on 100 Sales..... | 17 |
| 1.3.3. | Running on 1000 Sales..... | 17 |
| 1.3.4. | Running on 10K Sales | 18 |
| 1.3.5. | Running on 100K Sales | 18 |
| 1.3.6. | Running on 500K Sales | 19 |
| 1.3.7. | Running on 1M Sales | 19 |
| 1.3.8. | Average Running Times of Different Input Sizes and Plotting..... | 20 |
| 1.4. | RUNNING AND COMMENTING ALGORITHM ON SORTED DATASET | 22 |
| 1.4.1. | Comparing Sorted and Unsorted Dataset..... | 23 |
| 1.4.2. | Other Input Cases that Give the Similar Result | 23 |
| 1.4.3. | Solution | 23 |
| 8. | REFERENCES | 24 |

2. FIGURES

| | |
|---|----|
| Figure 1. Unsorted Dataset With 50 Inputs | 7 |
| Figure 2. Result of Sorting Operation | 8 |
| Figure 3. Best Case of Quick Sort | 10 |
| Figure 4. Worst Case of Quick Sort | 11 |
| Figure 5. Average Case of Quick Sort | 12 |
| Figure 6. Initial Array | 13 |
| Figure 7. First Step..... | 14 |
| Figure 8. Second Step | 14 |
| Figure 9. Third Step..... | 14 |
| Figure 10. After Changing Sale 2 and Sale 3 | 15 |
| Figure 11. Final Array..... | 15 |
| Figure 12. Four Sales of Same Country Name | 16 |
| Figure 13. N-Running Time Graph..... | 21 |
| Figure 14. N-Running Time Relation of Sorted Dataset..... | 22 |

3. TABLES

| | |
|--|----|
| Table 1. Sorting Times of 10 Sales..... | 17 |
| Table 2. Sorting Times of 100 Sales..... | 17 |
| Table 3. Sorting Times of 1000 Sales..... | 18 |
| Table 4. Sorting Times of 10K Sales | 18 |
| Table 5. Sorting Times of 100K Sales | 19 |
| Table 6. Sorting Times of 500K Sales | 19 |
| Table 7. Sorting Times of 1M Sales..... | 20 |
| Table 8. Average Running Times of Unsorted Dataset..... | 20 |
| Table 9. Relation Between Running Time and Upper Bound | 21 |
| Table 10. Average Running Times of Sorted Text | 22 |
| Table 11. Worst Case Upper Bound vs Running Time | 23 |

4.EQUATIONS

| | |
|---|----|
| Equation 1. Relation of Merge Sort in Best Case | 9 |
| Equation 2. Simple Best-Case Equation..... | 9 |
| Equation 3. Best Case Equation when Divided to N | 9 |
| Equation 4. Simplified Version of Best Case | 9 |
| Equation 5. Best Case Equation of $N=N/2$ | 9 |
| Equation 6. Best Case Equation of $N=N/4$ | 9 |
| Equation 7. Last Steps Where Base Case is Handled | 10 |
| Equation 8. Final Approach..... | 10 |
| Equation 9. Best Case Upper Bound | 10 |
| Equation 10. Worst Case Recurrence Equation..... | 10 |
| Equation 11. Worst Case Upper Bound..... | 11 |
| Equation 12. Recurrence Steps | 11 |
| Equation 13. Worst Case Upper Bound Equation | 11 |
| Equation 14. Average Case Equation..... | 11 |
| Equation 15. Recurrences of Average Case | 12 |
| Equation 16. Longest Subproblem..... | 12 |
| Equation 17. Equation which Gives Subproblem Depth..... | 12 |

5. INTRODUCTION

In this homework, there is given a dataset where the global sales of different products are reported. In dataset there are country, units sold, item type, total profit and Order ID is given. It was asked to sort the sales according to country name. If the sales have same country it was asked to sort according to total profits by descending. For this homework Quick Sort algorithm will be used.

In the first part there will be given the results of the not sorted and sorted results with number of 50 sales.

In the second part of this homework firstly asymptotic upper bound for the best case, worst case and average case will be given and they will be proved by using recurrence equations.

Then it will be discussed if the result of sorting initially by profit then sort the sorted list according to country name is same. Also, there will be given 3 example algorithms where it satisfies this condition.

After that there will be calculated the average running time of the algorithm with different inputs on the dataset which is not sorted. Average running time will be calculated by finding the average of 10 calculation with same input amount.

And finally, there will be done same calculation with the previous step. But this will be done on sorted dataset which was handled at the previous parts. Results of these calculations will be compared with the calculations which was handled by unsorted data set. And at the last, the solution for sorting sorted dataset faster is provided.

6. PART 1

In this part of the homework there will be given of the unsorted and sorted results of the dataset with 50 input will be shown. Source codes are given in the 'main.cpp' file which was sent to homework files.

Unsorted dataset is given at Figure 1.



```
BLG335E_2020_HW1 - zsh - 87x50
Algeria Cosmetics 761723172 9669 1.68115e+06
Belgium Personal Care 222504317 2827 70844.6
Brunei Cereal 153842341 4222 374027
Canada Cosmetics 368977391 7464 1.29777e+06
China Office Supplies 198927056 5791 731114
Czech Republic Cosmetics 726137769 9157 1.59213e+06
Djibouti Clothes 880811536 562 41273.3
Dominica Household 274930989 7044 1.1674e+06
Dominican Republic Baby Food 824714744 274 26265.6
Estonia Household 835696351 9976 1.65332e+06
Ethiopia Cosmetics 807785928 662 115102
Finland Household 757257401 8148 1.35037e+06
France Cosmetics 324669444 5758 1.00114e+06
Ghana Office Supplies 601245963 896 113120
Haiti Office Supplies 485070693 2052 259065
India Snacks 440306556 5349 294944
Israel Beverages 371502530 4709 73742.9
Kazakhstan Snacks 710296428 1352 74549.3
Kuwait Household 459386289 1466 242960
Lebanon Meat 704205024 8770 501644
Malaysia Beverages 955894076 9154 143352
Mauritius Clothes 349235904 5520 405389
Morocco Clothes 667593514 4611 338632
Nicaragua Household 573998582 7791 1.2912e+06
Oman Cosmetics 358570849 7937 1.38001e+06
Papua New Guinea Clothes 647164094 9092 667716
Papua New Guinea Meat 940995585 360 20592
Samoa Household 937431466 5657 937535
Sao Tome and Principe Clothes 548299157 2760 202694
Serbia Clothes 925136649 7348 539637
Seychelles Beverages 425793445 507 9349.02
Singapore Snacks 176461303 7676 423255
Slovakia Beverages 174590194 3973 62217.2
Solomon Islands Household 101328551 4225 700209
South Africa Fruits 443368995 1593 3839.13
Sri Lanka Fruits 830192887 1379 3323.39
Taiwan Fruits 732588374 8034 19361.9
Tanzania Cosmetics 739008080 7768 1.35062e+06
Tanzania Fruits 156530129 9599 23133.6
Tanzania Beverages 659878194 1476 23114.2
The Bahamas Personal Care 246248090 9137 228973
Togo Cosmetics 563681733 4806 835619
Turkmenistan Vegetables 116205585 6670 421077
Uganda Cosmetics 842238795 6031 1.04861e+06
Uganda Personal Care 539471471 451 11302.1
United Arab Emirates Office Supplies 425418365 9603 1.21238e+06
United Kingdom Cosmetics 135178029 1038 180477
Vanuatu Fruits 571997869 5735 13821.3
Vietnam Personal Care 314505374 7984 200079
Zimbabwe Office Supplies 953361213 9623 1.2149e+06
```

Figure 1. Unsorted Dataset With 50 Inputs

And when quick sort is used as stated at the introduction part, this dataset will be sorted according to country by ascending order and if country is same according to total profit by descending order this dataset will be used. Result of these operations are given at Figure 2.

```
BLG335E_2020_HW1 — -zsh — 87x50
Algeria Cosmetics 761723172 9669 1.68115e+06
Belgium Personal Care 222504317 2827 70844.6
Brunei Cereal 153842341 4222 374027
Canada Cosmetics 368977391 7464 1.29777e+06
China Office Supplies 198927056 5791 731114
Czech Republic Cosmetics 726137769 9157 1.59213e+06
Djibouti Clothes 880811536 562 41273.3
Dominica Household 274930989 7044 1.1674e+06
Dominican Republic Baby Food 824714744 274 26265.6
Estonia Household 835696351 9976 1.65332e+06
Ethiopia Cosmetics 807785928 662 115102
Finland Household 757257401 8148 1.35037e+06
France Cosmetics 324669444 5758 1.00114e+06
Ghana Office Supplies 601245963 896 113120
Haiti Office Supplies 485070693 2052 259065
India Snacks 440306556 5349 294944
Israel Beverages 371502530 4709 73742.9
Kazakhstan Snacks 710296428 1352 74549.3
Kuwait Household 459386289 1466 242960
Lebanon Meat 704205024 8770 501644
Malaysia Beverages 955894076 9154 143352
Mauritius Clothes 349235904 5520 405389
Morocco Clothes 667593514 4611 338632
Nicaragua Household 573998582 7791 1.2912e+06
Oman Cosmetics 358570849 7937 1.38001e+06
Papua New Guinea Clothes 647164094 9092 667716
Papua New Guinea Meat 940995585 360 20592
Samoa Household 937431466 5657 937535
Sao Tome and Principe Clothes 548299157 2760 202694
Serbia Clothes 925136649 7348 539637
Seychelles Beverages 425793445 597 9349.02
Singapore Snacks 176461303 7676 423255
Slovakia Beverages 174590194 3973 62217.2
Solomon Islands Household 101328551 4225 700209
South Africa Fruits 443368995 1593 3839.13
Sri Lanka Fruits 830192887 1379 3323.39
Taiwan Fruits 732588374 8034 19361.9
Tanzania Cosmetics 739008080 7768 1.35062e+06
Tanzania Fruits 156530129 9599 23133.6
Tanzania Beverages 659878194 1476 23114.2
The Bahamas Personal Care 246248090 9137 228973
Togo Cosmetics 563681733 4806 835619
Turkmenistan Vegetables 116205585 6670 421077
Uganda Cosmetics 842238795 6031 1.04861e+06
Uganda Personal Care 539471471 451 11302.1
United Arab Emirates Office Supplies 425418365 9603 1.21238e+06
United Kingdom Cosmetics 135178029 1038 180477
Vanuatu Fruits 571997869 5735 13821.3
Vietnam Personal Care 314505374 7984 200079
Zimbabwe Office Supplies 953361213 9623 1.2149e+06
```

Figure 2. Result of Sorting Operation

After sorting the dataset, new dataset which is name sorted.txt is created and sorted dataset is written to that text file. Now in the next parts, some comments will be done and it will be tried to explain the results.

7. PART 2

1.1. Upper Bounds and Proofs

In this part there will be different operations accordingly. In the first part best, worst and average cases will be explained. They will be proved by using recurrences.

By Using Recurrence Equations

1.1.1. Best Case

In the best case it will be thought that selected pivot will be the median of the current list. When pivot is selected as median list will be divided into 2 parts. So, the next part of the list will be $N/2$ length where N is the previous list length but 2 different lists. Recurrence Equation can be shown as Equation 1.

$$T(N) = \begin{cases} 0, & N = 0,1 \\ 2T(N/2) + N, & N > 1 \end{cases}$$

Equation 1. Relation of Merge Sort in Best Case

So from this point on, it can be said that recurrence equation is as in Equation 2

$$T(N) = 2T(N/2) + N$$

Equation 2. Simple Best-Case Equation

When all equation is divided to N equation yields to as in Equation 3.

$$\frac{T(N)}{N} = \frac{N}{N} + \frac{2T(N/2)}{N}$$

Equation 3. Best Case Equation when Divided to N

It can be simplified to form at Equation 4.

$$\frac{T(N)}{N} = 1 + \frac{T(N/2)}{N/2}$$

Equation 4. Simplified Version of Best Case

For $N = N/2$ that means it is divided 2 equal parts which yields to Equation 5.

$$\frac{T(N/2)}{N/2} = 1 + \frac{T(N/4)}{N/4}$$

Equation 5. Best Case Equation of $N=N/2$

When $N = N/4$ equation yields to Equation 6.

$$\frac{T(N/4)}{N/4} = 1 + \frac{T(N/8)}{N/8}$$

Equation 6. Best Case Equation of $N=N/4$

In this point intermediate steps are not shown but directly $N = 2$ situation is shown to explain base step. This equation can be shown at Equation 7.

$$\frac{T(2)}{2} = 1 + \frac{T(1)}{1}$$

Equation 7. Last Steps Where Base Case is Handled

So total N can be written as in Equation 8.

$$\frac{T(N)}{N} = 1 + 1 + 1 + \dots + 1 = \log_2^N$$

Equation 8. Final Approach

Finally, best case upper bound is as in Equation 9.

$$T(N) = N \log_2^N$$

Equation 9. Best Case Upper Bound

Visualization of the best case of Quick Sort is given at Figure 3.

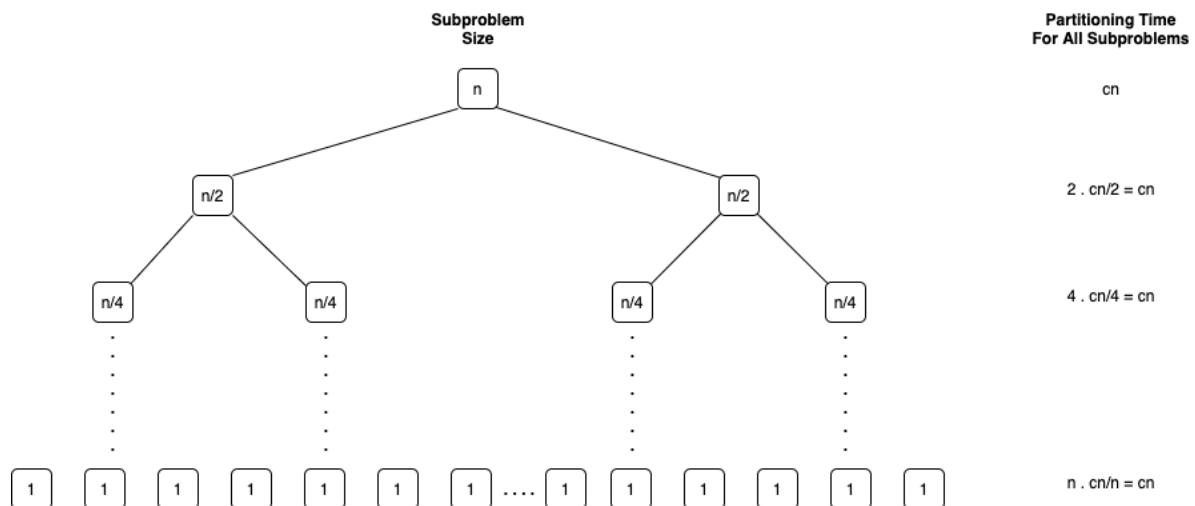


Figure 3. Best Case of Quick Sort

As a conclusion **best case upper bound is $O(N \log_2^N)$.**

1.1.2. Worst Case

In the worst case it is clear when pivot must be selected as highest or lowest number of the current subproblem list. So, when highest or lowest number is selected, there becomes 2 subproblems which has length 0 and another has length $n-1$. This equation can be shown as in Equation 10.

$$T(N) = \begin{cases} 0, & N = 0,1 \\ N + T(N - 1), & N > 1 \end{cases}$$

Equation 10. Worst Case Recurrence Equation

So, from this point on it can be said that worst case upper bound is as in Equation 11.

$$T(N) = N + T(N - 1)$$

Equation 11. Worst Case Upper Bound

Recurrence steps are shown at Equation 12.

$$T(N - 1) = (N - 1) + T(N - 2);$$

$$T(N - 2) = (N - 2) + T(N - 3);$$

$$T(N - 3) = (N - 3) + T(N - 2);$$

...

$$T(3) = 3 + T(2);$$

$$T(2) = 2 + T(1) = 2$$

Equation 12. Recurrence Steps

So finally, worst case upper bound can be calculated and written as in Equation 13.

$$T(N) = N + (N - 1) + (N - 2) + \dots + 3 + 2 \cong \frac{N(N - 1)}{2} \cong \frac{N^2}{2}$$

Equation 13. Worst Case Upper Bound Equation

Visualization of worst case of Quick sort is given at Figure 4.

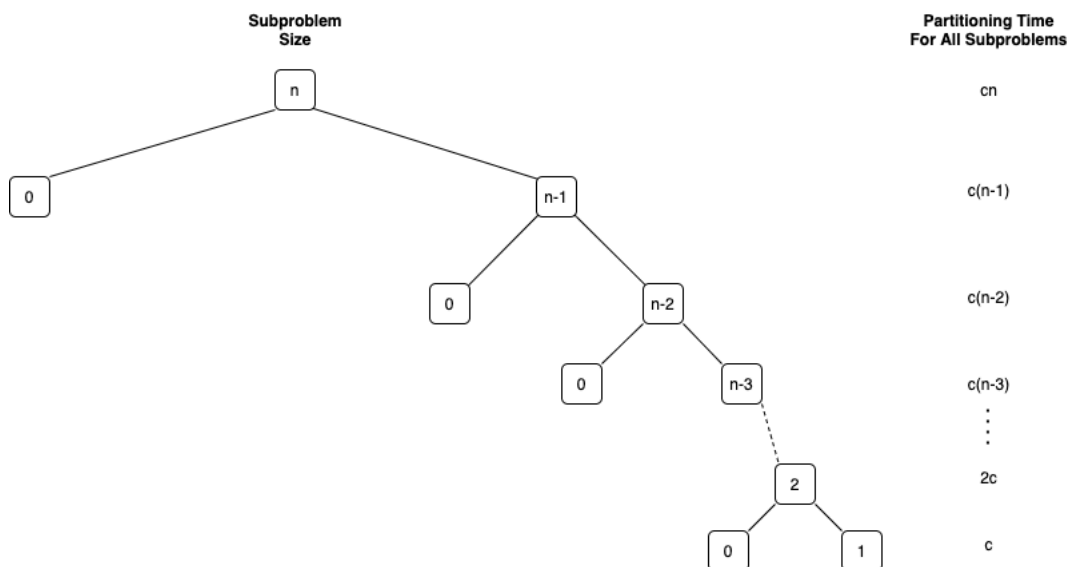


Figure 4. Worst Case of Quick Sort

This means that **worst case** upper bound is $O(N^2)$.

1.1.3. Average Case

In the average case it can be thought as partitioning the subproblem in one third. So left side can be 1/4 and right side can be 3/4 size. So, when problems are partitioned into 3-to-1 split recurrence equation yields to as in Equation 14.

$$T(N) = \begin{cases} 0, & N = 0, 1 \\ T(N/4) + T(3N/4) + N, & N > 1 \end{cases}$$

Equation 14. Average Case Equation

So, equation in simplified form and recurrences can be simply written as in Equation 15.

$$\begin{aligned}
 T(N) &= T(N/4) + T(3N/4) + N \\
 T(N/4) &= T(N/16) + T(3N/16) + N/4 \\
 T(3N/4) &= T(3N/16) + T(9N/16) + 3N/4
 \end{aligned}$$

Equation 15. Recurrences of Average Case

As given it is clear that when $T(N/4)$ and $T(3N/4)$ are summed there becomes another N subproblems so in each sub equation there is N subproblem.

In this problem left side have the lowest and the right side of recurrence equation have longest depth. Now on this point on for the upper bound $T(3N/4)$ can be used. This recurrence equation which will be used now on is at Equation 16.

$$T(3N/4) = T(3N/16) + T(9N/16) + 3N/4$$

Equation 16. Longest Subproblem

As seen here upper bound now depends on $T(9N/16)$. So until $T(1)$ is reached there is simply an amount of $\log_{4/3}^n$ steps which is found by at Equation 17.

$$1 = N \times 3/4^X, \text{ where } X \text{ is the steps that must be used.}$$

Equation 17. Equation which Gives Subproblem Depth

So, it can be said that there are \log_4^n depth in the left side of tree and there are $\log_{4/3}^n$ depth in the right side of recurrence equations. Since question asks the upper bound of the average case $\log_{4/3}^n$ will be used as depth. Since $\log_x^y = \frac{\log_z^y}{\log_z^x}$ it can be said that $\log_{4/3}^n = \frac{\log_2^n}{\log_2^{4/3}}$ so \log_2^n can be used as longest depth. Since there are cn partitioning in each subproblem there are $\log_{4/3}^n \times cn$ operation.

Visualization of worst case of Quick sort is given at Figure 5.

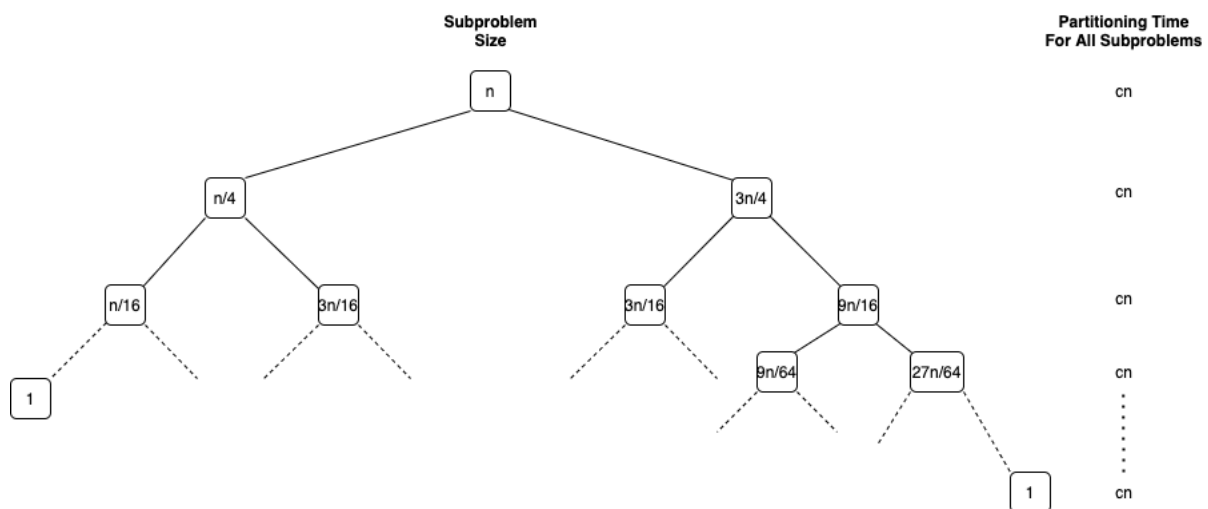


Figure 5. Average Case of Quick Sort

So, **average case** of the Quick Sort can be denoted as $\Theta(n \times \log_2^n)$.

1.2. Different Approach

In this part of the homework it is wanted to sort the dataset initially by descending order of total profit sort the sorted by total profit dataset according to ascending country name. Then it was to whether or not this approach is same as done in the first part.

1.2.1. Explanation and Simulation

In the quick sort algorithm that was used in this homework sorts the country names and if country names are same it sorts according to descending order of total profits. In this algorithm it selects initial location as the first element of array. Then it compares the first element with the pivot. If this element is lower than pivot algorithm says that this number is lower than pivot so do not put pivot before this number; put pivot after this number. Then it swaps the number with the location where it will put the number at. Now it is time to simulate this problem by sorting the dataset, where it was sorted by total profit in descending order, according to country names.

Now suppose the dataset which was sorted according the total profit in descending order is given at Figure 6.

| Sale 1 | Sale 2 | Sale 3 | Sale 4 |
|---------------------------|-----------------------|-----------------------|-----------------------|
| Country Name: Afghanistan | Country Name: Belgium | Country Name: Algeria | Country Name: Belgium |
| Total Profit: 150 | Total Profit: 120 | Total Profit: 90 | Total Profit: 33 |

Figure 6. Initial Array

So, it shows that sales are sorting according to total profits. When Quick Sort algorithm is started, sale 1 will be the initial point and sale 4 will be used as pivot. If the point that is compared is lower than pivot, currently compared sale and the sale that was desired to put pivot at the will be swapped and location of pivot that will be inserted will be incremented by one. In this dataset Sale 1 will be selected as the initial sale which will be compared with sale four. Red backgrounded sale denotes the location which will be swapped with pivot at the end and arrow indicates the sale which is being compared with the pivot. First step is given at Figure 7.

| | | | |
|---------------------------|-----------------------|-----------------------|-----------------------|
| Sale 1 | Sale 2 | Sale 3 | Sale 4 |
| Country Name: Afghanistan | Country Name: Belgium | Country Name: Algeria | Country Name: Belgium |
| Total Profit: 150 | Total Profit: 120 | Total Profit: 90 | Total Profit: 33 |




Figure 7. First Step

In the next part, since first country's name is lower than pivot's name, location which will be used at the end of the process that will be changed with pivot will be incremented by one and currently looking sale will be the next one. This situation's image is given at Figure 8.

| | | | |
|---------------------------|-----------------------|-----------------------|-----------------------|
| Sale 1 | Sale 2 | Sale 3 | Sale 4 |
| Country Name: Afghanistan | Country Name: Belgium | Country Name: Algeria | Country Name: Belgium |
| Total Profit: 150 | Total Profit: 120 | Total Profit: 90 | Total Profit: 33 |




Figure 8. Second Step

Now it is the critical point that currently looking sale has the same country name with the pivot sale's country name. As stated before, since they have same name not lower name end point will not be used and currently looking sale will be the next. So, it is clear that at the end of the process, Sale 2 will be after Sale 4. However, it is time to move on. In the next step it will be looked at the Sale 3 and will be compared with Sale 4. But do not forget that red backgrounded sale will not change. Figure 9 shows what is the next step.

| | | | |
|---------------------------|-----------------------|-----------------------|-----------------------|
| Sale 1 | Sale 2 | Sale 3 | Sale 4 |
| Country Name: Afghanistan | Country Name: Belgium | Country Name: Algeria | Country Name: Belgium |
| Total Profit: 150 | Total Profit: 120 | Total Profit: 90 | Total Profit: 33 |




Figure 9. Third Step

In this part Sale 3 will be compared with the Sale 4 and since Algeria is lower than Belgium Sale 2 and Sale 3 will be swapped. By incrementing the location where will used at the end, Sale 2 keeps its red background. In the last step that won't be executed, array will be look like as given in Figure 10.

| Sale 1 | | Sale 3 | | Sale 2 | | Sale 4 |
|---------------------------|--|-----------------------|--|-----------------------|--|-----------------------|
| Country Name: Afghanistan | | Country Name: Algeria | | Country Name: Belgium | | Country Name: Belgium |
| Total Profit: 150 | | Total Profit: 90 | | Total Profit: 120 | | Total Profit: 33 |




Figure 10. After Changing Sale 2 and Sale 3

So, this is actually the last sorting part. Final array is given at Figure 11

| Sale 1 | | Sale 3 | | Sale 4 | | Sale 2 |
|---------------------------|--|-----------------------|--|-----------------------|--|-----------------------|
| Country Name: Afghanistan | | Country Name: Algeria | | Country Name: Belgium | | Country Name: Belgium |
| Total Profit: 150 | | Total Profit: 90 | | Total Profit: 33 | | Total Profit: 120 |

Figure 11. Final Array

As a result, it is clear that when first dataset is sorted according to total profit by descending order and then sorting the sorted by profit dataset according to country names by ascending order is **not same as** sorting dataset by only country names and if country names are same sorting according to total profit.

If this simulation was done with more than two sales with same name and different total profits, this time lowest profit will be the first one and remaining will be in descending order. Another example with 4 sales that have same country names and different total profits that was sorted in descending order is given at Figure 12. As it is understandable initially it changes the highest value with lowest which is pivot. Than function terminates and it operates on 3 sales which are the right side of dashed line. At the end **first sale becomes the lowest total profit sale and remaining sales are sorted in descending order by total profits.**

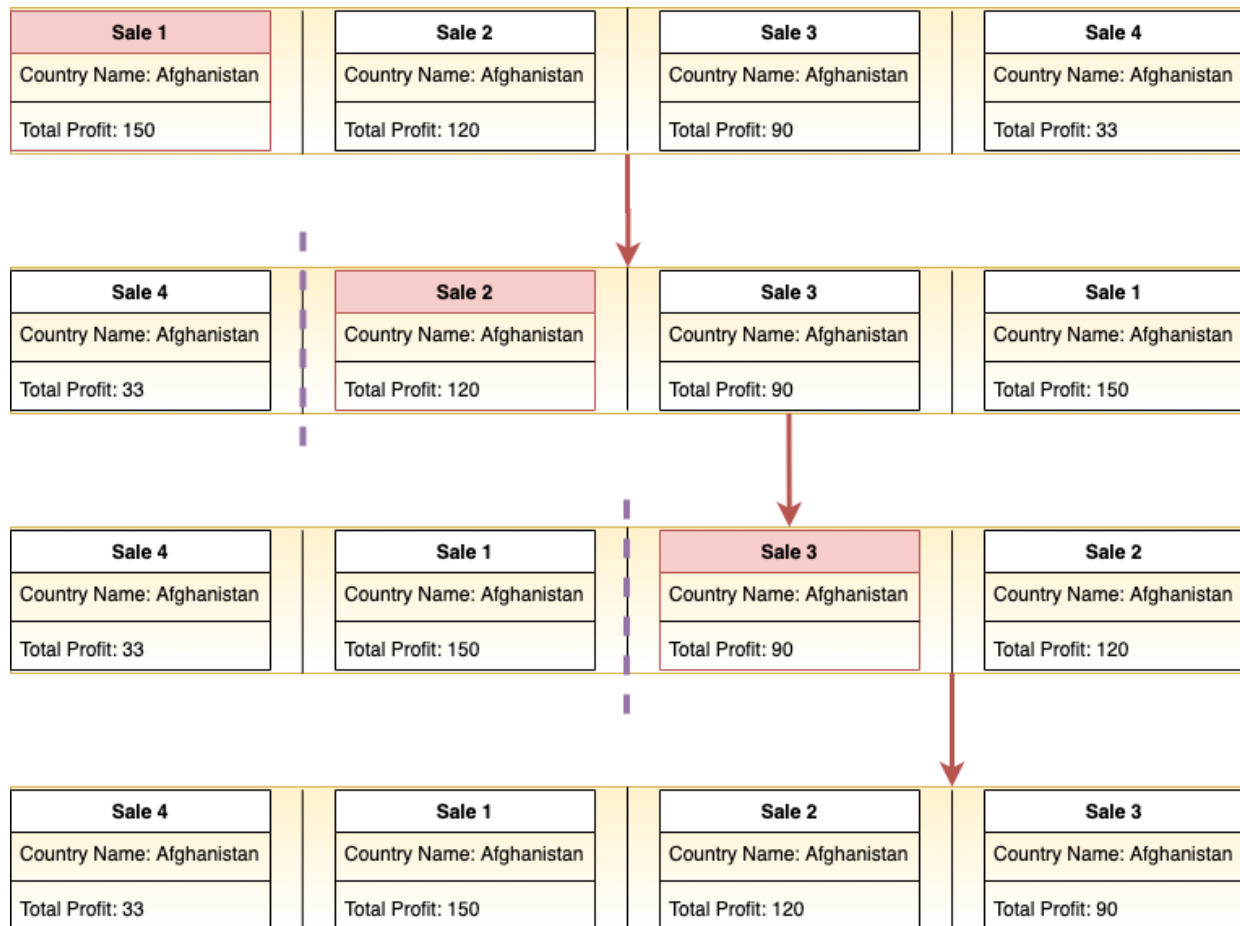


Figure 12. Four Sales of Same Country Name

1.2.2. Sorting Algorithms That Fit These Approach

Insertion Sort, Merge Sort and Bubble Sort fits this approach.

1.3. Running and Commenting Algorithm on Unsorted Dataset

In this part there was tried with 10,100,1000,10K,100K,500K,1M sales 10 times and averages are found. Then there was given a plot which states the Number of Sales - Running Time. After that results are compared with asymptotic bound and commented.

1.3.1. Running on 10 Sales

When algorithm is run on first 10 sales running times are given at Table 1.

| Time | Running Time (s) |
|------|------------------|
| 1 | 0.000012 |
| 2 | 0.000012 |
| 3 | 0.000012 |
| 4 | 0.000012 |
| 5 | 0.000012 |
| 6 | 0.000012 |
| 7 | 0.000012 |
| 8 | 0.000012 |
| 9 | 0.000012 |
| 10 | 0.000013 |

Table 1. Sorting Times of 10 Sales

Average of running times of 10 sales is **0.0000121**.

1.3.2. Running on 100 Sales

When algorithm is run on first 100 sales running times are given at Table 2.

| Time | Running Time (s) |
|------|------------------|
| 1 | 0.000192 |
| 2 | 0.000156 |
| 3 | 0.000155 |
| 4 | 0.000155 |
| 5 | 0.000155 |
| 6 | 0.000155 |
| 7 | 0.000155 |
| 8 | 0.000156 |
| 9 | 0.000156 |
| 10 | 0.000157 |

Table 2. Sorting Times of 100 Sales

Average of running times of 100 sales is **0.0001592**.

1.3.3. Running on 1000 Sales

When algorithm is run on first 1000 sales running times are given at Table 3.

| Time | Running Time (s) |
|------|------------------|
| 1 | 0.002560 |
| 2 | 0.002687 |
| 3 | 0.002734 |
| 4 | 0.002781 |
| 5 | 0.002559 |
| 6 | 0.002557 |
| 7 | 0.002575 |
| 8 | 0.002671 |
| 9 | 0.002644 |
| 10 | 0.002721 |

Table 3. Sorting Times of 1000 Sales

Average of running times of 1000 sales is **0.0026489**.

1.3.4. Running on 10K Sales

When algorithm is run on first 10K sales running times are given at Table 4.

| Time | Running Time (s) |
|------|------------------|
| 1 | 0.031297 |
| 2 | 0.032216 |
| 3 | 0.031893 |
| 4 | 0.032567 |
| 5 | 0.032271 |
| 6 | 0.031584 |
| 7 | 0.032129 |
| 8 | 0.032427 |
| 9 | 0.032725 |
| 10 | 0.032969 |

Table 4. Sorting Times of 10K Sales

Average of running times of 10K sales is **0.032207**.

1.3.5. Running on 100K Sales

When algorithm is run on first 100K sales running times are given at Table 5.

| Time | Running Time (s) |
|------|------------------|
| 1 | 0.387272 |
| 2 | 0.386912 |
| 3 | 0.388561 |
| 4 | 0.387920 |
| 5 | 0.387053 |
| 6 | 0.389713 |
| 7 | 0.389148 |
| 8 | 0.387602 |
| 9 | 0.387219 |
| 10 | 0.388925 |

Table 5. Sorting Times of 100K Sales

Average of running times of 100K sales is **0.388032**.

1.3.6. Running on 500K Sales

When algorithm is run on first 500K sales running times are given at Table 6.

| Time | Running Time (s) |
|------|------------------|
| 1 | 2.253141 |
| 2 | 2.253457 |
| 3 | 2.253905 |
| 4 | 2.251654 |
| 5 | 2.254375 |
| 6 | 2.256148 |
| 7 | 2.259623 |
| 8 | 2.254058 |
| 9 | 2.253603 |
| 10 | 2.263335 |

Table 6. Sorting Times of 500K Sales

Average of running times of 500K sales is **2.2553229**.

1.3.7. Running on 1M Sales

When algorithm is run on first 1M sales running times are given at Table 7.

| Time | Running Time (s) |
|------|------------------|
| 1 | 4.719337 |
| 2 | 4.714015 |
| 3 | 4.718884 |
| 4 | 4.710732 |
| 5 | 4.742990 |
| 6 | 4.712482 |
| 7 | 4.714413 |
| 8 | 4.733261 |
| 9 | 4.727896 |
| 10 | 4.720960 |

Table 7. Sorting Times of 1M Sales

Average of running times of 1M sales is **4.721497**.

1.3.8. Average Running Times of Different Input Sizes and Plotting

So average running times of sorting with different input sizes are given at Table 8.

| Input Size | Running Time (s) |
|------------|------------------|
| 10 | 0.000012 |
| 100 | 0.000159 |
| 1000 | 0.002648 |
| 10K | 0.032207 |
| 100K | 0.388032 |
| 500K | 2.255322 |
| 1M | 4.721497 |

Table 8. Average Running Times of Unsorted Dataset

From these information graph of Number of Sales - Running Time is shown at Figure 13.

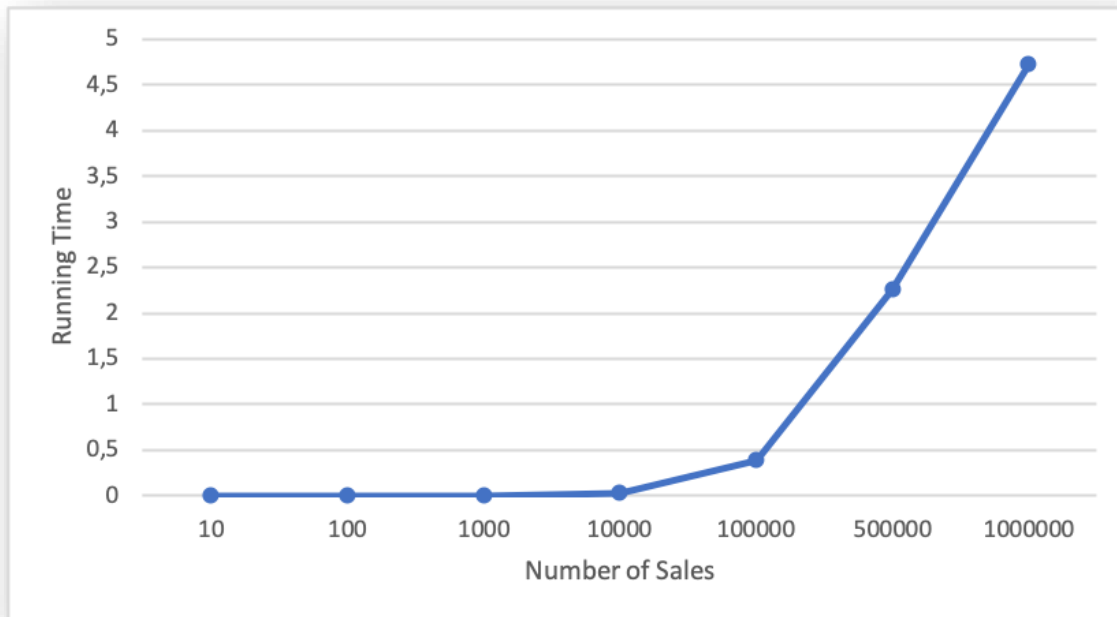


Figure 13. N-Running Time Graph

So now, it is easy to compare the result of running time with asymptotic upper bound where it is average case. In the Table 9, there was given upper bound, running time and division of them.

| Input Size | Average Case Upper Bound = $\Theta(n \times \log_2^n)$ | Running Time (s) | $\frac{\text{Upper Bound}}{\text{Running Time}(s)}$ |
|------------|---|------------------|---|
| 10 | 33.2192 | 0.000012 | 2768266 |
| 100 | 664.3856 | 0.000159 | 4178252 |
| 1000 | 9965.57 | 0.002648 | 3763217 |
| 10K | 132877.12 | 0.032207 | 4126718 |
| 100K | 1660964.04 | 0.388032 | 4280482 |
| 500K | 9465784 | 2.255322 | 4197087 |
| 1M | 19931568.56 | 4.721497 | 4221450 |

Table 9. Relation Between Running Time and Upper Bound

So, from the Table 9, it is clear that except the input size of 10, division of Upper Bound to Running Times are really close to each other. So average case of upper bound is proved in this example.

1.4. Running and Commenting Algorithm on Sorted Dataset

In this section, text file which is produced at the previous parts are used. This text file contains the sorted sales. Pivot will be selected as the maximum named country since at the end of the sales array is the highest named sale. So as explained in the first section of second part, this will be the case of worst case. Because in this time there will be no element in the right side of pivot and all remaining elements will be sorted on the left side. Expectations are going to $\Theta(n^2)$. This section's purpose is to prove whether or not worst case upper bound is satisfied or not by experimentally. In this section, 10 different trials will not be given because it was given for explaining how average running times are handled. The average running times are given at Table 10. There are no 1M and 500K input sizes are calculated since it takes too much time instead 25K and 50K is used.

| Input Size | Running Time (s) |
|------------|------------------|
| 10 | 0.000022 |
| 100 | 0.001381 |
| 1000 | 0.106630 |
| 10K | 7.576012 |
| 25K | 41.92445 |
| 50K | 155.2344 |
| 100K | 591.1225 |

Table 10. Average Running Times of Sorted Text

From these information graph of Number of Sales - Running Time is shown at Figure 14.

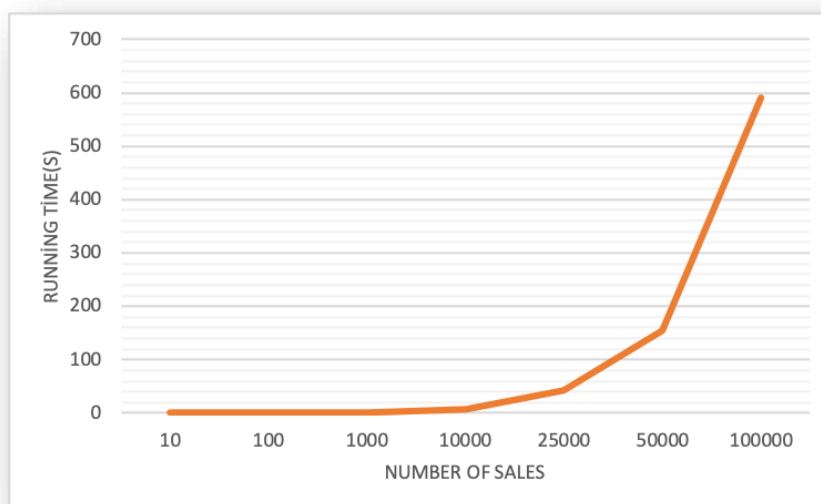


Figure 14. N-Running Time Relation of Sorted Dataset

As explained earlier this situation fits to the worst case which is $\Theta(n^2)$. It will be checked whether or not $\Theta(n^2)$ is related to running time at

| Input Size | Worst Case Upper Bound = $\Theta(n^2)$ | Running Time (s) |
|------------|--|------------------|
| 10 | 100 | 0.000022 |
| 100 | 10000 | 0.001381 |
| 1000 | 10^6 | 0.106630 |
| 10K | 10^8 | 7.576012 |
| 25K | $625 \cdot 10^8$ | 41.92445 |
| 50K | $25 \cdot 10^{10}$ | 155.2344 |
| 100K | 10^{10} | 591.1225 |

Table 11. Worst Case Upper Bound vs Running Time

As shown above between 10K and 25K input sizes there are 2.5 ratio. So, square of 2.5 is 6.25. It can be seen that between their running times there is 6 ratios. And also, between 50K and 100K there is 2 ratios. Square of 2 makes 4 and between their running times there are 3.81 ratios. These results prove that $\Theta(n^2)$ is true.

1.4.1. Comparing Sorted and Unsorted Dataset

When unsorted dataset is used, it is clear that pivot is not biggest or lowest value in the dataset. It might or not be median value so it fits to the average case upper bound. But when sorted dataset is used, pivot is being highest value every time so there becomes all values in the left side of pivot and no elements on the right so this approach fits to worst case which is $\Theta(n^2)$. Because of these situations sorting the sorted dataset with Quick sort is lower and has higher bound than sorting the unsorted dataset.

1.4.2. Other Input Cases that Give the Similar Result

Also, when dataset is sorted in descending order not in ascending order, this time pivot will be the lowest value so there will be no element on the left side of pivot and all elements will be on the right side of pivot so this time will be the same with previous example.

1.4.3. Solution

When there is a sorted dataset it is useful to pick pivot as median since it makes upper bound to fit to the **best case**.

8. REFERENCES

[1] GeeksforGeeks. (2020, April 09). QuickSort. <https://www.geeksforgeeks.org/quick-sort/>

[2] C++ Reference. Clock. <http://www.cplusplus.com/reference/ctime/clock/>