Project 1

Chang Li

1 Problem Statement

We are required to analyze the following program/code sample.

int j = 2

while (j < n) {

int k = j

while (k < n) {

sum += a[j]\*b[k]

k = k \* k

}

j = 2 \* j

}

The task is to analyze the time complexity of the given code in term of n. The code contains two nested loops: the outer loop runs over j, and the inner loop runs over k, updating the sum with a product of elements from arrays a and b.

2 Theoretical Analysis

The outer loop starts with j =2 and double j each time, running while j < n. Hence the outer loop will execute approximately log\_2(n) times.

The inner loop, starting at k =j, squares k each time (k = k\*k), significantly reducing the number of iterations. It runs approximately log(log(n)) times for each value of j.

Thus, the time complexity of the algorithm is approximately:

O(log\_2(n) \* log(log(n)))

3 Experimental Analysis

3.1 Program Listing

Below it the Python code used to simulate the algorithm and collect the execution times for various of n. It is important to run the code multiple times for different values of n to observe how the execution time behaves and compare it with the theoretical analysis.

A screenshot of a computer program

Description automatically generated

3.2 Data Normalization Notes

In the experiment, we are normalizing the execution times by a scaling constant derived from the theoretical result to match the experimental results. The theoretical values are unitless and follow the complexity O(log\_2(n) \* log(log(n))). To compare both sets of data, we adjusted the theoretical values using a constant scaling factor.

The scaling factor is calculated based on the ratio of the largest experimental value to the theoretical value.

3.3 Output Numerical Data

The table below shows the experimental results alongside the scaled theoretical results for different values of n.

|  |  |  |  |
| --- | --- | --- | --- |
| n | Experimental Time (ns) | Theoretical Result | Adjusted Theoretical Result |
| 10 | 66995.62 | 2.770596 | 92317.11 |
| 100 | 22888.18 | 10.146362 | 338079.90 |
| 1000 | 24795.53 | 19.260321 | 641759.82 |
| 10000 | 125885.01 | 29.503064 | 983051.19 |
| 100000 | 1352310.18 | 40.585164 | 1352310.18 |

3.4 Graph

A graph with a red line and a blue line

Description automatically generated

This graph plots the experimental results in red and the adjusted theoretical results in blue.

3.5 Graph Observations

The graph shows that the experimental results fluctuate at smaller values of n, likely due to measurement noise and system overhead. However, as n increases, the experimental results begin to follow the adjusted theoretical trend closely. This confirms the theoretical time complexity of O(log\_2(n) \* log(log(n))), especially for larger values of n.

4 Conclusions

In this project, we investigated the time complexity of the algorithm and found that it matched the expected theoretical complexity of O(log\_2(n) \* log(log(n))). Even though there were some small fluctuations for smaller values of n, the experimental results lined up well with the adjusted theoretical ones as n got bigger, backing up our analysis. This show that the theoretical model does a good job of predicting how the algorithm performs.