# California State University, Fresno

# DEPARTMENT OF COMPUTER SCIENCE

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Class: | **Algorithms & Data Structures** | | | Semester: | **Fall 2020** |
|  | | | | | |
| Points |  | Document author: | **Yet Chun Fong** | | |
|  | Author’s email: | **fongyc7177@mail.fresnostate.edu** | | |
| Laboratory number: | **2** | | |
|  | | | | | |

**1. Statement of Objectives**

The main task in this lab session is to implement function to display Fibonacci Numbers at Nth. However, there are different methods to find the Nth Fibonacci Number, but all methods have different time complexity. Thus, the main objective for this lab session is to differentiate inefficient and efficient methods based on time complexity. The Lab TA introduced 3 methods to find the Nth Fibonacci Number: recursion, arrays, temporary variables. Then, time taken for these 3 methods were recorded and compare to show the differences. The major accomplishments of this lab are differentiating multiple methods to solve the problems and get the same exact result. By then, the reasons of differences in time complexity for different methods are discovered in this lab experiment. ‘No algorithm can perfectly solve every problem’ was proven in this experiment, as recursion and iterative algorithms can provide same output, but they used different amount of time to display the output.

**2. Experimental Procedure**

Before Lab TA introduced 3 different methods to the class, there was a discussion on first natural solution to find Nth Fibonacci number, and all groups agreed on the recursion algorithm. However, recursion is an inefficient method. Then, iterative algorithm is introduced to the class which is known to be efficient method.

For recursion method, function is introduced that take 1 argument, position of N. Then, if then input value of N is either 0 or 1, the same input value will be return to the main function because it is the exact same number in the Fibonacci sequence. Otherwise, the function will return (n-1) + (n+2), recursively until the Nth number is reached. Lastly, the returned value will be print out as output of Nth Fibonacci Number. The main reason why recursion algorithm is an inefficient method to find Nth Fibonacci Number is because each calculated Fibonacci number in the tree is not stored in the memory, so the function have to repeat and calculate the number again and again until it reached Nth position. When the input N gets larger, the tree will become bigger. Thus, the function took a long time to calculate and climb the tree to reach Nth Fibonacci Number. As mentioned, this recursion method consumed too much time when the input number is large because of same value had to be calculated repeatedly.

The first iterative algorithm introduced to the class is using arrays. But this method is not used as the second iterative algorithm, using temporary variables was highly recommended by Lab TA. The function is defined as int variable (int n), which take 1 argument, position of N, same as the recursion method. Then, 3 integer variables were declared: prev = 0, curr = 0, and next. Then, a For loop is implemented, to loop through the elements from 2 to N. In the For loop, we will calculate the value of variable next by adding values of previous and curr, then move the variable forward by reassigning prev to curr and curr to next. Then after the For loop is stopped, the function will return the value of curr which is at position of N. The reason why temporary variable method is much efficient than the recursive method is because it stored the value of previous and current Fibonacci numbers. Therefore, the function does not have to calculate the previous values like recursive method. The Nth Fibonacci number is calculated linearly without any repeated works.

**3. Analysis**

|  |  |  |  |
| --- | --- | --- | --- |
| Input (N) | **Time Taken (In seconds)** | | Output |
| Recursion | Iterative (Temporary variables) |
| 10 | 0.000469 | 0.000098 | 55 |
| 40 | 4.689120 | 0.000262 | 102334155 |
| 120 | No Output | 0.000119 | 428607904 |

From the result, the time taken for recursion increase exponentially, as the input of N increased to 40, the time taken for recursion algorithm was almost 5 seconds already. Then, the time taken for N= 120 using recursion is ‘No Output’ because the growth of execution time is exponential, and it is estimated to take about 4 days to compute the result. The recursion is inefficient as it needed to calculate the value of previous variables repeatedly. In short, recursion is not an efficient method to use for Fibonacci numbers because it needs traverse the tree and recalculate the values of previous variables. In short, we can conclude that the time complexity for recursion algorithm in Fibonacci number’s problem is 2n, increase exponentially.

For iterative algorithm, the time taken for three input N is almost instant in this experiment. From the result, it is notable that the time taken for N =10 and N =120 is very close. Iterative algorithm is much efficient because it stores the values of previous variables which the function does not need to calculate the previous variables over and over again. The time complexity for this algorithm is n, increase linearly according to the input of N. When the input N reached 120, the differences between iterative algorithm and recursion algorithm is more obvious, and iterative algorithm is much efficient in this case as the time complexity is n, linear. Recursion algorithm has exponential growth time which will take up to 4 days to finish compute the result.

A screenshot of a computer screen

Description automatically generated

**4. Encountered Problems**

There were not many issues faced during the lab session, but the time taken to measure N =120, for recursion algorithm took awhile and waited until the end of lab session. Then, Lab TA told the class that it will take about 4 days for the recursion algorithm to compute the result. Therefore, ‘No output’ will be the result of N=120 for recursion algorithm, which saves a lot of time instead of waiting for the recursion algorithm to display the results at N=120. Other than that problem, no significant issue was faced during the process of coding and calculating output because coding guidelines are provided, and it helped the class to understand the problems and solutions easily.

**5. Conclusions**

The iterative method is far more efficient than recursion method in Fibonacci numbers problems. There is no perfect algorithm that can solve all problems. Therefore, different problems require different algorithm that is best fitted in that case or situation. Before this lab, recursion had always been known as the efficient algorithm to solve problems including binary trees, binary search trees, merge sort, etc. After this lab session, it has strengthened the thought that different problems require different algorithm, algorithm is being chosen and implemented based on a case by case basis.

**6. References**

Coding Guidelines provided by Lab TA, Shreeja Miyyar.