Analysis of Fibonacci Numbers Calculations Using Static Programming and Dynamic Programming Algorithms to Get Optimal Time Efficiency

Ventryshia Andiyani, Wirawan Istiono

***Abstract*—Fibonacci numbers are simple numbers that are the sum of two consecutive numbers. There are many methods and many way to get Fibonacci result to solve many problems in real life or programming problem, in this research will determine the optimal time efficiency to solve the problem of calculating Fibonacci numbers, either by using static programming algorithm, such as recursive algorithm or with the dynamic programming algorithm, such as top-down approach algorithm and the bottom-up approach algorithm method. The problem-solving strategy is performed by calculating the execution time required by the three algorithms to get accurate results using C language with many various counts of inputs. From the comparison of Fibonacci search methods, it was found that by using the Dynamic programming method with a bottom-up approach the algorithm has a more optimal efficiency than the top-down approach from dynamic programming or from static programming with a recursive algorithm**

***Keywords*—Fibonacci numbers, Recursive algorithm, Top-down approach algorithm, bottom-up approach algorithm.**

I. INTRODUCTION

The development of information and communication technology in the age of globalization is growing. Many innovations, one of them is in the field of algorithms and programming, which produce big impact to solve problems quickly and precisely. So far, the programming algorithm is divided into two types, namely static programming algorithm and dynamic programming algorithm . The application of both static programming and dynamic programming algorithms are quite extensive, for example, in the calculation of Fibonacci numbers. Fibonacci numbers were first explained in 1150 by Indian mathematicians named Gopala and Hemachandra. Leonardo of Pisa studied Fibonacci numbers around 1200 when he discussed the ideal growth of the rabbit population and then known as Fibonacci [1]. Fibonacci numbers result from the sum of two consecutive numbers. Therefore, the Fibonacci numbers contain looping or recursive elements.

Solving the problem of calculating Fibonacci numbers can use static programming or dynamic programming algorithms. Static programming, such as C and C ++, is a program that must be compiled before running the program so the program will stop when there is an error in the program. Unlike the case with dynamic programming such as Javascript, Python, PHP, etc. do not require compilation before running the program. The program will be compiled when the program is executed, so when there is an error, the program will stop right then. Based on the statement above, this research will discuss about the calculation of Fibonacci numbers using static programming algorithm, such as the recursive algorithm method and by using dynamic programming, namely the algorithm of the top-down approach and the bottom-up approach to achieve the most optimal time efficiency point by calculating the execution time needed by the program from using the three algorithms [2][3].

II. RESEARCH METHOD

A series of numbers can be defined as a sequence consisting of numbers arranged according to certain rules and patterns. In the sequence of numbers, there is an element called a term. The word of 'Fibonacci' is taken from the name of its discoverer. The Fibonacci number series was first discovered by an Italian mathematician named Leonardo Fibonacci in the 30th century [4]. Fibonacci is a sequence of numbers that has the first term that is 1, the second term is also 1, and then the third term is the sum of the previous terms (the first term and the second term). A series of Fibonacci numbers will produce a ratio when it divided. The series of numbers provided on the Fibonacci sequence contained are 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, etc. Based on the sequence number, the Fibonacci number series can be formulated as shown in Equation 1 from Poulos [5].

$$Fn = Fn\text{-}1 + Fn\text{-}2 \text{ with condition } n \geq 3 \quad (1)$$

$$F0 = 0 \text{ and } F1 = 1$$

Comparison between Fn+1 and Fn is almost always the same for any value, this ratio is fixed or called Golden Ratio, whose the value is close to 1.618 [6]. The implementation of the Fibonacci numbers will be seen in Fig.1.

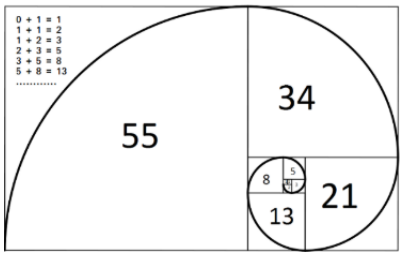


Fig. 1. Implementation of the Fibonacci numbers

An algorithm is the sequences or steps that used to solve problems arranged sequentially or systematically. The arranged algorithms must consider the results produced (the results must be as close as possible to the actual values) and the maximum of efficiency of time and memory. Therefore, a good algorithm is an algorithm that can produce the right results with the minimum of time and memory. The algorithm is arranged of various machine languages. The algorithm is divided into two types based on the programming language used, namely the static programming algorithm and the dynamic programming algorithm [7][8].

.

An algorithm is often called a static programming algorithm because users do not communicate with computers. Therefore, when the program is compiled, the program can run without waiting or require any input from the user [9]. Static Programming has a separate compilation and execution phase, the compiler runs longer and produces optimal machine code [10]. In doing some work, a static programming algorithm optimizes hardware efficiency, so that it can be several times faster than a dynamic programming algorithm. Examples of static programming languages: C, C ++, etc. Dynamic programming was first developed by Richard E. Bellman in 1957. In this technique, the problem solving method can be seen as the result of interrelated decisions. According to Munir (2008) in a dynamic program, an optimal set of decisions is made using the Optimality Principle [11].

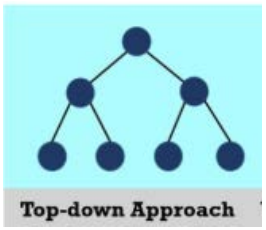
The algorithm is often called dynamic programming algorithm because the user communicates with the computer. This is indicated by the need for input from the user when compiled. Dynamic Programming does not separate the compilation phase and the execution phase, the execution of the code is done at compile-time and compiles the code at execution [12]. Using dynamic programming algorithms designed to optimize programmer efficiency, so that it can implement functionality with less code than a static programming algorithm but in other words compromising the speed of execution. Dynamic programming algorithm has two approaches, top-down approach algorithm, and bottom-up approach algorithm.

The preparation of Fibonacci numbers calculation using static programming algorithm, one of them will be used the recursive algorithms [13]. The pseudocode algorithm static programming will look like as shown in **Error! Reference source not found.**.

```
function fib(n)
   if n = 0 or n = 1
        return 1
   else
        return fib(n-1) + fib(n-2)
```

Fig. 2. Fibonancci pseudocode algorithm static programming

In a static program, several 'fib' that have the same results are counted repeatedly, compilation as needed, so this algorithm has exponential time complexity [14]. The dynamic programming algorithm has two methods of approach algorithm, called the Top-down approach and Bottom-up approach. Top-down Approach algorithm, this method solving the problem with broken down the problem into sub-problems and each sub-problem is solved first then the solution saved if sometimes the solution is needed again then just need to be called it again [15][16]. This method uses recursion (repetition) and memorization (storing calculated values) combined into one so that the program does not need to recalculate the previously calculated value when it is needed. [17]. Top-down approach algorithm can be seen in **Error! Reference source not found.**.



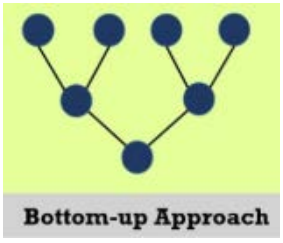
(a)

```
if n = 0 then
   return 0
else if n = 1 or n = 2 then
   return 1
else
   fib[1] = 1
   fib[2] = 1
   for i = 3 to n do
        fib[i] = fib[i-1] + fib[i-2]
   endfor
   return fib[i]
endif
```

(b)

Fig. 3. Top-down approach conception and pseudocode

As can be seen in Figure 3, there is using of arrays to store calculated the values. When the loop stops, the function will return the last element of the array which is the nth fib number calculated. This Bottom-Up Approach method [18], all sub-problems that may be needed to solve first and then used to build solutions to larger problems, the conception and pseudocode its shown in **Error! Reference source not found.**.



(a)

```
prevF ← 0
currF ← 1
for i←2 to n do
    newF ← prevF + currF
    prevF ← currF
    currF ← newF
endfor
return currF
```

(b)

Fig. 4. Bottom-up approach conception and pseudocode

If the top-down approach algorithm uses memorization, the bottom-up approach algorithm uses the tabulation so that arrays are no longer needed so that the required memory can be minimized. The bottom-up approach algorithm can look like in Figure 4b. With this method, starting from calculating the smaller 'fib' value then calculating the larger value than

the value previously obtained. It can be seen in Figure 4 the calculation of fib(s) only done once, then use it to calculate Fibonacci (4) and Fibonacci (3), so that both approaches have the same complexity is polynomial [19].

III. Result And Discussion

Based on the research method about Fibonacci and all methods to calculate the Fibonacci problem, Fibonacci numbers calculation algorithms in dynamic programming are estimated to be able to increase efficiency to be more optimal than recursive algorithms in static programming. How much increase in time efficiency that occurs can be proven through real testing [10]. This test is done by making three programs based on three Fibonacci algorithms, both static programming algorithm and dynamic programming algorithm in C ++ language by using the CodeBlock application and the GCC compiler. The program will be executed and take into account the time needed to complete the calculation of Fibonacci numbers. Static programming testing is carried out with a recursive algorithm in accordance with the pseudocode above. Source Code Fibonacci calculation program uses static programming, and the results of the execution of the static programming using the recursive algorithm program when the variable value inputted are 5, 10, 15, 20 and 25, as can see in **Error! Reference source not found.**.

Table 1. Result table Fibonacci numbers and execution time static programming

Fib to-	Fibonacci Summary Result	Execution Time (ms)
5	5	897.00
10	55	918.00
15	610	967.00
20	6765	1041.00
25	75025	1091.00
Average		982.80

Dynamic programming testing is carried out using two algorithms in accordance with the pseudocode above. The testing is done by making two programs based on this algorithm which consists of Top-Down Approach, with the following Source Code is a Fibonacci calculation program using the Top-Down Approach dynamic programming algorithm, the results of the execution of the dynamic programming using the top-down approach algorithm program when the variable value inputted are 5, 10, 15, 20 and 25, as can see in **Error! Reference source not found.**.

Table 2. Result table Fibonacci numbers and execution time dynamic programming Top-down approach

Fib to-	Fibonacci Summary Result	Execution Time (ms)
15	610	658.00
20	6765	697.00
25	75025	784.00
Average		654.40

The following Source Code is a Fibonacci calculation program using the Bottom-Up Approach dynamic programming algorithm, and The results of the execution of the static programming using the recursive algorithm program when the variable value inputted are 5, 10, 15, 20 and 25, as can see in **Error! Reference source not found.**.

Table 3. Result table Fibonacci numbers and execution time dynamic programming Bottom-up approach

Fib to-	Fibonacci Summary Result	Execution Time (ms)
5	5	451.00
10	55	472.00
15	610	546.00
20	6765	564.00
25	75025	668.00
Average		540.20

From the results of the execution time of testing the three methods obtained, the Recursive method, Top-down approach method, and Bottom-up Approach method, get the graph as can see in **Error! Reference source not found.**.

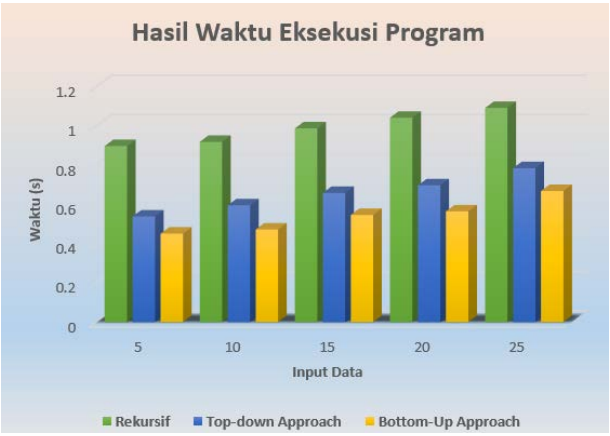


Fig. 5. Result graph comparison, between Recursive method, Top-down approach and Bottop-up approach

IV. Conclusion

From the graph result that shown in **Error! Reference source not found.**, it can be seen the results of tests on the Fibonacci number calculation algorithm, both static programming and dynamic programming algorithm. Using dynamic programming has a more optimal time efficiency than static programming, because dynamic programming is able to avoid repetitive calculations. However, can be seen too in dynamic programming, using a top-down approach or

bottom-up approach algorithm having different efficiencies in calculating Fibonacci numbers. Even though both of these algorithms have the same complexity, namely, O(n). Dynamic programming with a bottom-up approach algorithm has a more optimal efficiency than the top-down approach. This efficiency difference can be caused by the computer's ability to perform Fibonacci calculations in a traversal manner. By using the traversal methods the calculation is more quickly completed than the recursive method. This is due to the recursive method, and the processor works harder with context switching. Therefore, it can be concluded that in order to achieve the most optimal time efficiency in calculating Fibonacci numbers is to utilize dynamic programming as a solution to solving problems.

ACKNOWLEDGMENT

Thank you to the Universitas Multimedia Nusantara, Indonesia which has become a place for researchers to develop this journal research. Hopefully, this research can make a major contribution to the advancement of technology in Indonesia.

REFERENCES

[1] T. C. Scott and P. Marketos, "On the origin of the Fibonacci Sequence," *Research Gate*, no. March 2014, 2019.

[2] L. Overview and F. Numbers, "Dynamic Programming I : Memoization , Fibonacci , Crazy Eights , Guessing," *Fall 2009*, vol. 6, no. 1, pp. 1–5, 2008.

[3] R. A. Pambudi, W. Lubis, F. R. Saputra, H. P. Maulidina, and V. N. Wijayaningrum, "Genetic Algorithm for Teaching Distribution based on Lecturers' Expertise," *Kinetik: Game Technology, Information System, Computer Network, Computing, Electronics, and Control*, vol. 4, no. 4, p. 297, Oct. 2019.

[4] I. L. Alfred S. Posamentier, *The Fabulous Fibonacci Numbers*. Prometheus; F First Edition edition, 2007.

[5] R. Fisher and J. Fisher, *Candlesticks, fibonacci, and chart pattern trading tools*. Canada: JOHN WILEY & SONS, INC., 2003.

[6] T. Omotehinwa and S. Ramon, "Fibonacci Numbers and Golden Ratio in Mathematics and Science," *International Journal of Computer and Information Technology (ISSN:*, vol. 02, no. 04, pp. 630–638, 2013.

[7] H. I. Flower, *Introduction to the second edition*, First edit., vol. 7. Massachusetts London, England: McGraw-Hill Book Company, 2010.

[8] G. Viko and F. N. Ferdinand, "DESIGN OF SIMULATION TECHNIQUES FOR DATA PREDICTION IN PUBLIC TRANSPORTATION," *International Journal of Computer Science Engineering and Information Technology Research*, vol. 7, no. 3, pp. 33–38, 2017.

[9] S. Sinha, "The Fibonacci Numbers and Its Amazing Applications," *International Journal of Engineering Science Invention*, vol. 6, no. 9, pp. 7–14, 2017.

[10] S. R. Ellis Horowitz, Sartaj Sahni, *Computer Algorithms*, vol. 53, no. 9. New York: Computer Science Press, 2013.

[11] I. Stojmenovic, "Recursive algorithms in computer science courses: Fibonacci numbers and binomial coefficients," *IEEE Transactions on Education*, vol. 43, no. 3, pp. 273–276, 2000.

[12] A. Albano, G. Ghelli, and R. Orsini, "Fibonacci: A programming language for object databases," *The VLDB Journal*, vol. 4, no. 3, pp. 403–444, 1995.

[13] S. Falcon, "Generalized (k , r) – Fibonacci Numbers," *Gen. Math. Notes*, vol. 25, no. 2, pp. 148–158, 2014.

[14] J. Oliver, "The Truth About Fibonanci Trading," *Journal of Chemical Information and Modeling*, vol. 53, no. 9, pp. 1689–1699, 2013.

[15] Á. Erdosné Németh and L. Zsakó, "The place of the dynamic programming concept in the progression of contestants' thinking," *Olympiads in Informatics*, vol. 10, pp. 61–72, 2016.

[16] Andrias Rusli & Friska Natalia Ferdinand, "Model and Simulation To Minimize Empty Space Left With Laff Method in the Loading of Cargo System," *International Journal of Computer Science Engineering and Information Technology Research (IJCSEITR)*, vol. 7, no. 3, pp. 11–18, 2017.

[17] L. Overview and F. Numbers, "Dynamic Programming I : Memoization , Fibonacci , Shortest Paths , Guessing," *Fall 2011*, vol. 6, no. 1, pp. 1–6, 2011.

[18] D. Mohammad. Alalaya, "Fibonacci Retracement and Elliot Waves to Predict Stock Market Prices: Evidence from Amman Stock Exchange Market," *International Journal of Applied Science and Technology*, vol. 8, no. 3, pp. 69–86, 2018.

[19] C. Canaan, "All about Fibonacci : A python approach," *World Applied Programming journal*, no. April, pp. 72–76, 2011.

Ventryshia Andiyani, was born in Jambi, Indonesia, in 1999. She received the bachelor's degree in Informatics from Universitas Multimedia Nusantara, in 2020. Her research interests include a user and learning based web and mobile programming

She now works as Software Engineer in Sirclo, Tangerang Indonesia from June 2020 until present. And before that, she has worked as Web Developer in Universitas Multimedia Nusantara, Tangerang Indonesia.

Her last publication entitled is "Penerapan Konsep Non-Deterministic Finite Automata (NFA) pada Aplikasi Simulasi Mesin Kopi Vending" in 2019. Email: ventryshia.andiyani@student.umn.ac.id

Wirawan Istiono, was born in Jakarta Indonesia, he received his Masters of Computer Science degree in Budi Luhur University, Indonesia, in 2018, focusing in the Software Engineering field.

He is currently a lecturer and researcher in Universitas Multimedia Nusantara and also serving as the head coordinator of the Game Development Laboratory. His research interests include requirements engineering in software application development, computer engineering, and human computer interaction.

His publication entitled "Saving water with water level detection in a smart home bathtub using ultrasonic sensor and Fuzzy logic" in conference Second International on Informatics and Computing (ICIC) in 2017 at IEEE. And last publication entitled "Education Games to Learn Basic Algorithm with Near Isometric Projection Method Unit" in 2020 at IJASCSE. Email: wirawan.istiono@umn.ac.id