SHM PUBLISHER
SURYA HIJAU MANFAAT

Journal of Student Research Exploration

https://shmpublisher.com/index.php/josre

p-ISSN 2964-8246 | e-ISSN 2964-1691

Comparative analysis of desktop and laptop performance using fibonacci recursive algorithm with nanosecond time measurement

Sidiq Noor Kharisma[1], Gea Destadia Al-Zahra[2], Much. Aziz Muslim[3]

[1,2,3] Department of Computer Science, State University of Semarang

Article Info

Article history:

Received May 19, 2026
Revised June 1, 2026
Accepted June 1, 2026

Keywords:

Fibonacci algorithm
Recursive algorithm
Benchmarking
Computer performance

ABSTRACT

The purpose of this study was to determine the differences between computing devices, including standard laptops, gaming laptops, and high-performance desktops. We also compared the performance of these computing devices. This research is considered crucial for understanding the effectiveness of processors in executing the Fibonacci algorithm. In this study, we analyzed the performance of three different devices, each with two laptop processors: an Intel Core i5 14450 HX and an Intel Core i5 8350U, and an AMD Ryzen 9 7900X desktop processor, each with the same RAM capacity and operating system. In this study, we used a quantitative experimental method with a processor performance benchmarking approach using a recursive Fibonacci algorithm based on nanosecond precision measurements. The results of our study indicate that the performance of each laptop processor is not significantly different. Meanwhile, the desktop processor performed very well, with execution times twice as fast as the laptop processor. The similarity of operating systems and RAM capacity makes the main difference in performance determined by the processor and clock speed. Our conclusion from this study suggests that a simple recursive algorithm can be used as a benchmark to assess differences in device execution speed when handling workloads. Overall, this research can also be used as a bridge between understanding the theory of computer architecture and organization and its implementation in the real world.

1. Introduction

In this digital era, computers are no longer a foreign object to society, especially students. They have become a crucial part of human life [1]. Computers are sometimes referred to as

[1] ***Corresponding Author:***
Sidiq Noor Kharisma,
Department of Computer Science,
State University of Semarang,
Sekaran, Gunungpati, Semarang City, Central Java 50229, Indonesia.
Email: sidiqnoor07@students.unnes.ac.id
DOI: https://doi.org/10.52465/josre.v4i2.9

the backbone of various human activities, from very light tasks to very heavy tasks, even touching on highly complex computing, such as gaming, data processing, and artificial intelligence. However, despite this progress, many users still don't realize that a device's performance isn't solely determined by its visible specifications (hardware), but rather by how the computer's architecture and organization work internally to execute instructions.

A common phenomenon in the field is that many laypeople choose devices based on labels or trends, such as "gaming laptop" or "high-end PC/desktop," without understanding how differences in internal structure affect performance. In practice, however, differences in processor design, memory management, and efficiency are seen in process execution.

Previous studies have focused more on processor architecture design, memory management, and process execution efficiency. Most of these studies have been theoretical or used simulation approaches, thus not fully reflecting the performance differences of devices used directly by users in real life.

Based on these issues, this study was conducted to provide a more realistic and measurable picture by comparing three types of devices: a standard laptop, a gaming laptop, and a high-performance desktop. This comparison aims to demonstrate how differences in computer architecture and organization impact system performance under similar usage conditions. To obtain sufficiently objective results, we used an experimental approach in this study, running a C++ program based on the Fibonacci algorithm.

The recursive Fibonacci algorithm was chosen because it has an exponential time complexity that requires many repetitive functions [2], [3], [4]. This complexity makes this algorithm sensitive to the performance of single-core processors and suitable for use as a benchmark in computing performance testing. This algorithm is perfect for performance testing of any computer, as it has clear data to assess the performance of various devices to solve their computational load. The Fibonacci equation is written into Equations 1 and 2, and the complexity of the time of the execution of recursion is written into Equation 3 [2], [3].

$$\mathrm{f_n = f_{n-1} + f_{n+2}} \tag{1}$$

$$\mathrm{f_0 = f_1 = f_1} \tag{2}$$

$$\mathrm{O(2^n)} \tag{3}$$

Several previous studies have used Fibonacci algorithms as a learning algorithm complexity [4]. Processor benchmarking research generally uses synthetic software such as Cinebench and Geekbench to measure system performance [5], [6]. However, research using simple recursive algorithm research to compare the effectiveness of desktop and laptop processors is still relatively limited. In addition, there have not been many studies that have reviewed the influence of single-core performance characteristics on the execution of repetitive recursive functions. Therefore, this study was conducted to determine the effectiveness of the processor in handling the complexity of time to complete this algorithm and by using nanosecond unit time. The use of nanosecond time allows the data obtained to be quite accurate with clear numbers and can adjust the desired time measurement conditions [7], [8] This research was carried out with three different computer devices, namely. Desktops with Ryzen 9 7900X processors, laptops with Intel-core i5 14450HX processors, and laptops with Intel-core i5 8350U vPro processors.

This algorithm is executed in a single-threaded manner so that the computation process is more representative of the performance of the single-core processor. Thus, the difference in execution results is more influenced by the clock speed, architectural efficiency, and processing capabilities of the processor cores than the overall number of cores [9], [10].

This study aims to determine the effectiveness of desktop and laptop processors in handling the execution of recursive Fibonacci algorithms using nanosecond precision time measurement.

2. Method

This study uses a quantitative experiment method with a performance benchmarking approach using a Fibonacci recursive algorithm. This test was conducted to determine the effectiveness of the new generation of processors compared to older processors in computing processing. Therefore, an experimental approach with the Fibonacci recursive algorithm was used as a benchmark. This algorithm demands the intensity of function invocation and can represent the workload of the processor in a single-core manner [9], [10].

Research Object

In this study, three devices (two laptops and one desktop PC) with the same operating system (Windows 11) were used with the same treatment, but with different processors. The processor used is as shown in Table 1.

Table 1. Specifications of the processor used as the object of the research

Devices	Processor	Core × Thread	Base Clock	Boost Clock	TDP
Desktop	AMD Ryzen 9 7900X	12 core × 24 threads	4.7 GHz	5.6 GHz	170 watts
Laptop 1	Intel Core i5-14450HX	10 core × 16 threads	1.8 GHz (Efficient-core) / 2.4 GHz (Performance-core)	3.5 GHz (Efficient-core)/ 4.8 GHz (Performance-core)	55 watts
Laptop 2	Intel Core i5-8350U vPro	4 core × 8 threads	1.7 GHz	3.6 GHz	15 watts

Table 1 shows the specifications of the devices used in this study. The desktop device uses an AMD Ryzen 9 7900X processor with 12 cores and 24 threads, a 4.7 GHz base clock, and a 5.6 GHz boost clock [11]. The first laptop uses an Intel Core i5-14450HX processor with a 10-core and 16-thread configuration, a base clock of 1.8 GHz (Efficient-core) and 2.4 GHz (Performance-core), and a boost clock of up to 4.8 GHz [12]. The second laptop uses an Intel Core i5-8350U vPro processor with 4 cores and 8 threads, a base clock of 1.7 GHz, and a boost clock of 3.6 GHz [13].

Algorithms and Implementations

The Fibonacci recursive algorithm is applied using the C++ programming language, with a function structure as shown in Figure 1.

```cpp
long long fibonacci(int n){
    if (n == 0) return 1;
    else if (n == 1) return 1;
    else return fibonacci(n - 1) + fibonacci(n - 2);
}
```

Figure 1. Fibonacci recursive function

When a function is given a value of n = 0 and n = 1, then the function returns a value of 1. For larger n-values, the function will call itself repeatedly with parameters n-1 and n-2, then add up the results of both. This pattern of repetitive invocation reflects the main characteristic of the recursive algorithm, which is to break down large problems into smaller sub-problems until they reach the basic condition.

Testing Procedure

The test was carried out by running a recursive Fibonacci algorithm on various n values. The variation of the values used is 50 to 55. The selection of this test variable aims to test the performance of the processor on a gradually increasing compute load until it approaches a heavier execution limit. Each test is performed, the execution time is recorded using a nanosecond-precision measuring function as shown in Figure 2, and on each test variable is repeated five times in a row to avoid biased results.[14], [15]

```cpp
int main() {
    // mulai timer
    auto start = high_resolution_clock::now();

    int n = 10;

    cout << "Fibonacci " << (n) <<  " = " << fibonacci(n) << endl;

    auto stop = high_resolution_clock::now();
    auto duration = duration_cast<nanoseconds>(stop - start);
    cout << "Executon time: " << duration.count() << " ns" << endl;

    return 0;
}
```

Figure 2. Main functions with nanosecond precision timer

The program is run using the GNU G++ compiler through the Code Runner extension in Visual Studio Code with standard configuration. During the testing process, other applications running in the background are minimized to reduce disruption to resource usage. All devices use the latest version of the Windows 11 operating system and use the same source code to optimize test results. In addition, the device is set to normal power with the best performance mode.

Research Flow

The research was carried out through several stages. The first stage is a literature study related to processor benchmarking and Fibonacci recursive algorithms. The second stage is to determine the device used as the research object, which is then carried out the implementation of the recursive Fibonacci algorithm using the C++ language. After the program is completed, tests are carried out on each device by calling the recursion function (notated "n" in the program) with variations n = 50 to n = 55. Each test is performed five times to prevent possible inconsistent results. The execution time data is then recorded, the average value is calculated and then analysed in the form of tables and graphs to compare the performance of each processor. Figure 3 shows the flow of research that has been carried out.

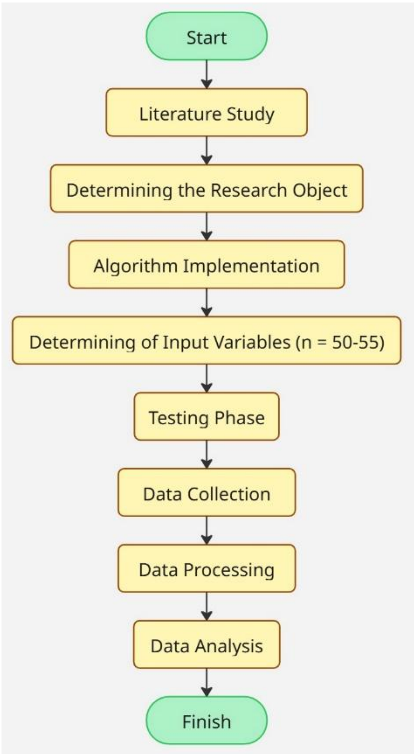

Figure 3. Research flow

Data Collection Techniques

The research data was obtained from the results of testing the execution time of the recursive Fibonacci algorithm program on each device. Measurements are made using the high_resolution_clock function of the chrono library in the C++ programming language. The execution time is recorded in nanoseconds. Each input value is tested 5 times to reduce inconsistent results due to processes running in the background of the operating system. The test data is then averaged to be used in the analysis process.

Data Analysis Techniques

The test results data were analysed using a quantitative descriptive method. The execution time of each device is calculated for its average value and then compared based on the variation in the Fibonacci input value used. The data processing results are presented in the form of tables and line charts to facilitate the analysis of performance comparisons between processors

3. Results and Discussion

From the research conducted, results were obtained in the form of average data from the test results shown in Table 1 represented in Figure 4. The graph shows an increase in execution time with every increase in the value of n. This indicates an increase in workload on the processor core in handling the recursion function.

Table 1. Results of the Fibonacci algorithm execution time measurement test on any PC with nanoseconds (ns)

Value n (Input Fibonacci)	Execution time on individual processors (nanoseconds)		
	Processor AMD Ryzen 9 7900X	Processor i5 14450HX	Processor i5 8350U
n = 50	48336887060	85526016200	87869845600
n = 51	78454825340	139394270400	141575328400
n = 52	127191210740	229769781600	225296957600
n = 53	205138803320	374059401400	366108500600
n = 54	333158774040	382302064000	591354051800
n = 55	538218991780	620048527400	950718977000

The execution time of a recursive Fibonacci algorithm increases significantly as the value of n increases, according to the exponential nature of the algorithm. AMD Ryzen 9 7900X consistently delivers the lowest execution time. This is in line with the [16], [17]clock speed on the processor core, which is the highest among the test processors. Then, the i5 14450HX and i5 8350U show relatively higher execution times with unique variations, where the i5 14450HX executes algorithms slower than the i5 8350U at certain values despite having a higher clock speed. The differences in results in some tests may be influenced by thermal throttling mechanisms and power management in modern laptop processors, especially on Intel's hybrid architecture that uses a combination of Performance-core and Effisent-core.[18], [19]

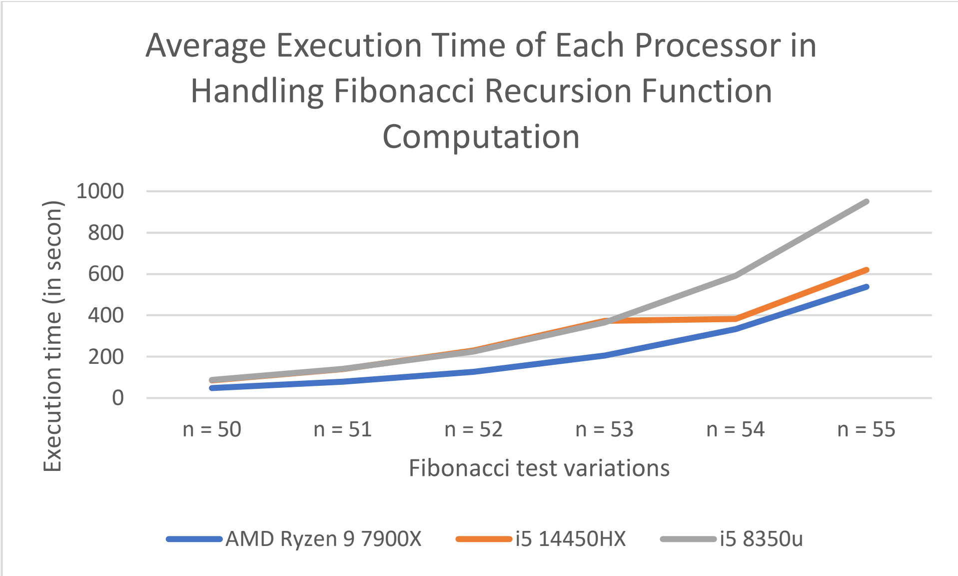

Figure 4. Average execution time for each device from all tests

In the graph shown in Figure 3, it is shown that the AMD Ryzen 9 7900X processor handles the most efficient computing. Meanwhile, devices with Intel Core i5 14450HX processors handle the same computation as the i5 8350U for 50-51 function calls. However, it is still efficient for 54 and 55 function summonses, although it is still less efficient than the AMD Ryzen 9 7900X.

The Ryzen 9 7900X performed approximately 1.77 times faster than the i5-8350U in n = 55 tests and 1.15 times faster than the i5-14450HX in the same test. Meanwhile, the i5-14450HX is 1.53 times faster than the i5-8350U.

4. Conclusion

Based on the test results, AMD Ryzen 9 7900X shows the best perform in handling the execution of the recursive Fibonacci algorithm with the lowest execution time across all input variations. This performance is affected by high CPU clocks and is offset by high TDP as well, as well as the efficiency of desktop processor architectures. This test shows that the Fibonacci recursive algorithm is more representative of single-core performance compared to multi-core because it is executed in a single-threaded manner. Further research can develop tests using iterative Fibonacci approaches, memorization, or parallel algorithms to compare computations more broadly.

REFERENCES

[1] D. Harris-Birtill and R. Harris-Birtill, "Understanding Computation Time: A Critical Discussion of Time as a Computational Performance Metric," in *Time in Variance*, BRILL, 2021, pp. 220–248. doi: 10.1163/9789004470170_014.

[2] Y. Li and W. Wu, "N-factor complexity of the Fibonacci sequence on N and the factor-counting sequences," *Theor. Comput. Sci.*, p. 115880, 2026.

[3] S. F. Ariza *et al.*, "STUDI PERBANDINGAN ALGORITMA PENCARIAN BINARY, JUMP, INTERPOLATION, DAN FIBONACCI : EFISIENSI MEMORI DAN WAKTU EKSEKUSI," 2025.

[4] O. A. Gbadamosi, A. E. Okeyinka, and I. Abdullahi, "Complexity Analysis of Recursive Algorithms with and Without Memoization," in *2024 International Conference on Science, Engineering and Business for Driving*

Sustainable Development Goals (SEB4SDG), IEEE, Apr. 2024, pp. 1–3. doi: 10.1109/SEB4SDG60871.2024.10630224.
[5] I. Zablah, L. Sosa-Díaz, and A. Garcia-Loureiro, “Relevance and Evolution of Benchmarking in Computer Systems: A Comprehensive Historical and Conceptual Review,” *Computers*, vol. 14, no. 12, p. 516, Nov. 2025, doi: 10.3390/computers14120516.
[6] I. Sudija, J. everhard Riwurohi, and M. M. Masad, “Next-Generation CPU Architectures: A Study of the Influence of Nanometer Technology on Computer Performance,” *Eduvest - Journal of Universal Studies*, vol. 4, no. 12, pp. 11669–11675, Dec. 2024, doi: 10.59188/eduvest.v4i12.44711.
[7] M. Y. Garg and M. B. Gupta, “Performance Analysis and Comparison of the Micro Virtual Machines Provided by the Top Cloud Vendors.,” *Journal homepage: www. ijrpr. com ISSN*, vol. 2582, p. 7421, 2023.
[8] W. Lee, J. Lee, B. K. Park, and R. Y. C. Kim, “Microarchitectural Characterization on a Mobile Workload,” *Applied Sciences*, vol. 11, no. 3, p. 1225, Jan. 2021, doi: 10.3390/app11031225.
[9] A. Sahebi, “Reconfigurable Architectures for Accelerating Distributed Applications," A Graph Processing Application Case Study",” 2022.
[10] H. Tran-Ngoc, T. Le-Xuan, S. Khatir, G. De Roeck, T. Bui-Tien, and M. Abdel Wahab, “A promising approach using Fibonacci sequence-based optimization algorithms and advanced computing,” *Sci. Rep.*, vol. 13, no. 1, Dec. 2023, doi: 10.1038/s41598-023-28367-9.
[11] AMD, “AMD Ryzen™ 9 7900X Desktop Processor.” Accessed: Mar. 31, 2026. [Online]. Available: https://www.amd.com/en/products/processors/desktops/ryzen/7000-series/amd-ryzen-9-7900x.html
[12] intel, “Intel® Core™ i5 processor 14450HX.” Accessed: Mar. 31, 2026. [Online]. Available: https://www.intel.com/content/www/us/en/products/sku/235998/intel-core-i5-processor-14450hx-20m-cache-up-to-4-80-ghz/specifications.html
[13] intel, “Intel® Core™ i5-8350U Processor.” Accessed: Mar. 31, 2026. [Online]. Available: https://www.intel.com/content/www/us/en/products/sku/124969/intel-core-i58350u-processor-6m-cache-up-to-3-60-ghz/specifications.html
[14] P. Schober and T. R. Vetter, “Repeated Measures Designs and Analysis of Longitudinal Data: If at First You Do Not Succeed—Try, Try Again,” *Anesth. Analg.*, vol. 127, no. 2, pp. 569–575, Aug. 2018, doi: 10.1213/ANE.0000000000003511.
[15] L. N. Muhammad, “Guidelines for repeated measures statistical analysis approaches with basic science research considerations,” *Journal of Clinical Investigation*, vol. 133, no. 11, Jun. 2023, doi: 10.1172/JCI171058.
[16] P. A. Putra, K. Shandi Dani, and F. Muhammad Aditya, “Analisis Algoritma Nonrekursif pada Angka Fibonacci”.
[17] R. M. Nugraha, “REPRESENTASI BILANGAN FIBONACCI DALAM BENTUK KOMBINATORIAL”.
[18] T. Benoit-Cattin, D. Velasco-Montero, and J. Fernández-Berni, “Impact of Thermal Throttling on Long-Term Visual Inference in a CPU-Based Edge Device,” *Electronics (Basel).*, vol. 9, no. 12, p. 2106, Dec. 2020, doi: 10.3390/electronics9122106.
[19] H. Rafsanjani, A. Marwazi, Z. Januar, T. Sinaga, and D. Sitompul, “Thermal management and power optimization in modern cpu and gpu architectures,” *computing*, vol. 7, no. 13, p. 14, 2025.