Journal of Sisfotek Global
ISSN (Online): 2721 - 3161, ISSN (Print): 2088 – 1762
http://dx.doi.org/10.38101/sisfotek.v12i1.449
Vol. 12, No. 1, March 2022, pp. 12-17

Reliability Comparison of High Performance Computing between Single Thread Loop and Multiple Thread Loop using Java-Based Programming at Fingerprint Data Processing

Arief Ginanjar[1] and Kusmaya[2]

[1,2] Langlangbuana University, Bandung, Indonesia, 40261.
E-mail: [1]arief.ginanjar@unla.ac.id, [2]kusmaya7165@gmail.com

ARTICLE HISTORY

Received : 5 February, 2022
Revised : 5 March, 2022
Accepted : 21 March, 2022

KEYWORDS

High Performance Computing
Thread
Java Programming
Multiple Thread
Fingerprint

ABSTRACT

High Performance Computing is one of the mechanisms in the programming family with a focus on increasing high performance in any programming environment, especially programming languages that use virtual machine environments such as C/C++, Java and Python. several sub-clusters of information technology such as Big Data, Data Warehouse, Business Intelligence and Artificial Intelligence, the use of HPC is widely applied in sophisticated computer machines that have enterprise data processing capabilities. This research was conducted to compare the ability of HPC with algorithms or java programming techniques in data processing that uses a lot of threads when implemented in ordinary computers used in everyday life, the term java programming technique that uses one thread is called Single Thread Loop then when using multiple threads is called Multiple Thread Loop. Due this comparison between sequential and parallel process so this research try to compare between Single Thread Loop and Multiple Thread Loop in Java Programming. The minimum requirement operating system is to use the MS Windows 7 or 10 and a Unix-based OS using an Intel i5 or i7 processor and use a minimum of 16 GB of RAM.

1. Introduction

1.1 Background

Personal identification is one of the most important issues in society related to access control, crime and forensic identification, banking and computer systems. Biometric features that can be used for identification include iris, voice, DNA, and fingerprints [1]. Based on research, the fingerprint is the most widely used biometric feature because of its uniqueness, universality, and stability [2]. From the studies that have been done before, they have done similar research using the C/C++ programming language and no one has done it using Java programming.

Automatic fingerprint identification has become an interesting research topic in the last two decades. Fingerprint recognition tools are very easy to obtain nowadays. On this basis many companies and institutions are using them, along with the increasing number of people who must be identified .

Fingerprint recognition can be grouped into two different forms of problems, namely verification and identification. The problem increases with a large number of fingerprint datasets, which results in more time required for the identification process. However, there is a way to overcome this complexity, namely classification and indexing, and then can also do evolutionary algorithm optimization by combining it with local search methods.

1.2 Research Goal

The objectives of this research are to determine the reliability value of High Performance Computing from combination of the selection programming techniques used and variations in the hardware used. The factors to be studied are the loading time in data processing when using the Single Thread Loop and Multiple Thread Loop programming techniques, The specimen plans to be studied are listed in Table 1.

Table 1 Algorithm Specimen to be studied.

Specimen	Programming Algorithm	Hardware Architecture
1	Single Thread Loop	Stand Alone
2	Multiple Thread Loop	Stand Alone
3	Single Thread Loop	Multiple PC
4	Multiple Thread Loop	Multiple PC

1.3 Research Scoping

In this research, the researcher tries to apply the problem limitation as follows:

- Using the MariaDB database version 10.2.31, Which is placed on a PC or separate in the MS Windows 10 OS.
- Using a network switch port 8 with10/100 Mbps connection.
- Using MS Windows 10 Genuine 64 bit OS for test scripts.
- Using Netbean 8.2 as Integrated Development Editor.

2. Methodology

2.1 Evolutionary Prototyping

The method used to create research reports using experimental quantitative research methods followed by an evolutionary prototyping methodology [3] [4]and adding a data loading performance capability test process with an incremental loop process and system approach to elements and components. The order of the process is carried out is as follows [5]:

- Literature Study, Studying literature sources that will be used as references[6][7]. Literature resources can be books, papers, articles or web pages that discuss High Performance Computing, Java Programming and Performance Tuning.
- Analysis process [8], by studying the architecture of java programming techniques and how to tune the performance coding technique of each programming technique that will be used.
- Software Design [9], In this case the researchers designed the software to be built based on the results obtained from the analysis. As well as designing a test plan that will be carried out.
- Software Implementation, which will be developed based on design and analysis results. The implementation produces a predefined product in the test scenario.
- Testing and Evaluation software products and evaluating each test scenario carried out to find weaknesses in the developed application and fix these weaknesses.
- Iterative is a loop process for the entire software development process that is being developed when the business process does not meet the specifications, and then improve the new design and re-implementation it until meets the requirement.
- Application Performance Testing, the process of testing applications against pressure by running applications repeatedly which is carried out on specimens that have been built into applications uses the testing script method.

The research process refers to the stages of the prototyping evolutionary model[10]; the process can be illustrated as shown in Figure 1. The process consists of input, prototyping process, and output. The user's ability must also meet the system requirements.

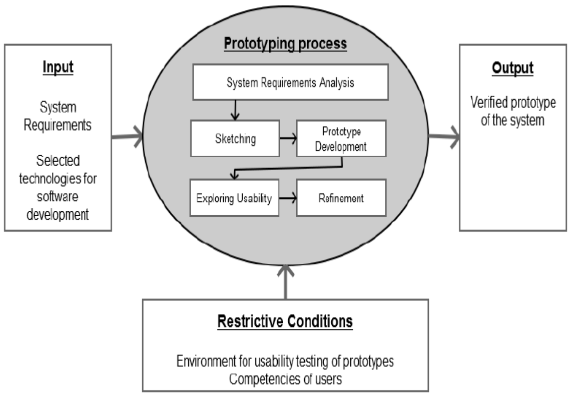

Input
System Requirements
Selected technologies for software development
Prototyping process
System Requirements Analysis
Sketching
Prototype Development
Exploring Usability
Refinement
Output
Verified prototype of the system
Restrictive Conditions
Environment for usability testing of prototypes
Competencies of users

Figure 1 Evolutionary Prototyping.

Code generation begins after studying the workings of MA from previous research, where on average researchers still use C/C++ programming, then by using the reference algorithm, code using Java programming is started to be developed. Then after successfully trying to code MA sequentially using Single Thread Loop, start making code modifications using Multiple Thread Loops so that the algorithm can be run in parallel. Followed by the process described below.

2.2 Multiple Loop Thread Logic Design

This Research need to design a simple application logic architecture as a performance testing media.

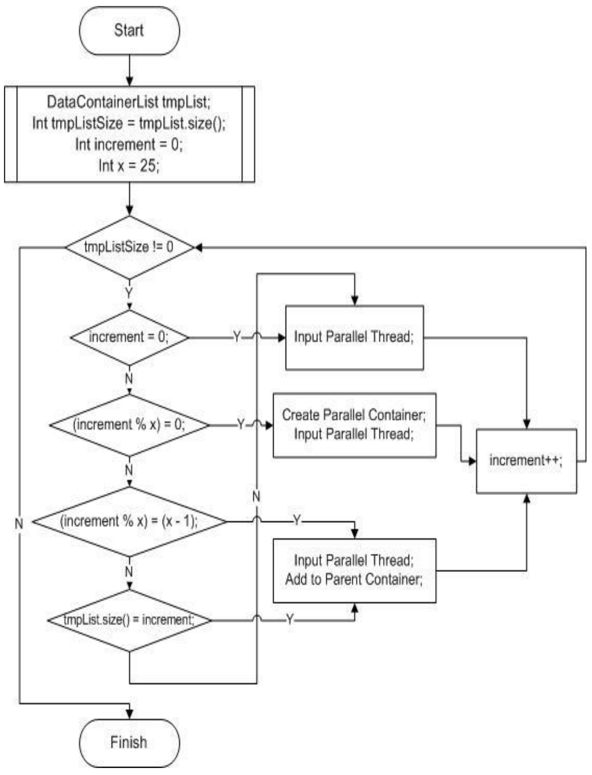


Figure 2 Multiple Loop Thread Logic Design.

Performance testing medium between Single Thread Loop and Multiple Thread Loop uses design as shown in figure 2. In the picture you can see the process of dividing a large data set into several small data groups to minimize data loading so that the process in CPU can be divided into Multiple Thread Loop Data Process. The logic of multiple thread loops is the distinguishing feature between the fingerprint data reading process using queue process and parallel process.

2.3 Application Test Configuration

Each test flow of each specimen involves a hardware architecture that becomes the medium for the application testing process to run. Each hardware architecture testing involves one or two laptops or PCs, where in each of these tools, a MySQL database is installed, which functions as data storage. As well as a small application built using Java programming, which has a function for running logic testing specimens that are run on Netbean IDE version 8.2, this illustration can be seen in Figure 3.

Each specimen listed in Table 1 also requires several other libraries to support the functioning of the main logic processes in the test, namely, MySQL JDBC [11].

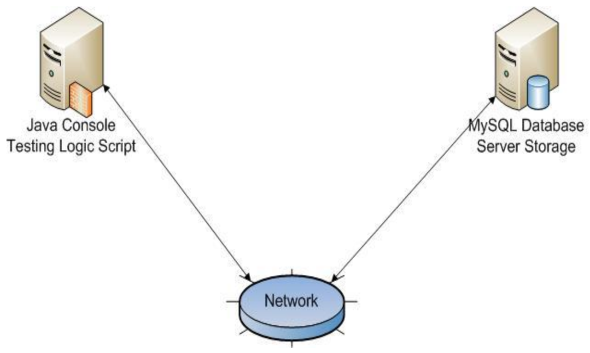
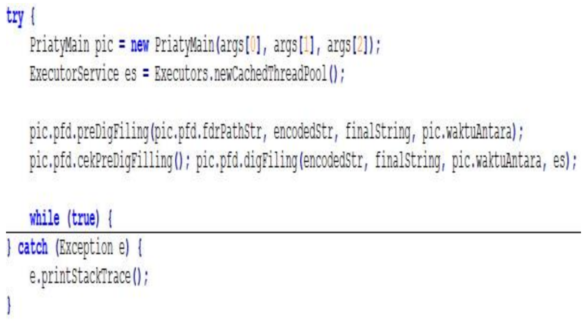


Figure 3 Logic Architecture of Single Thread Loop and Multiple Thread Loop Testing.

2.4 Test Result Data Collection

Generally speaking; the data tested in this study is divided into two data type, namely the source of data originating from the fingerprint image, which is converted into string array data, where the size of each fingerprint data depends on the level of complexity each image. Several of fingerprint data to be tested is 7200 sample data. The process of speed testing between Single Thread Loop and Multiple Thread Loop is done by moving data from the “source_data” table to the “result_data_test” table using the test script shown in Figure 4 and Figure 5, then with the data flow as shown in Figure 6.

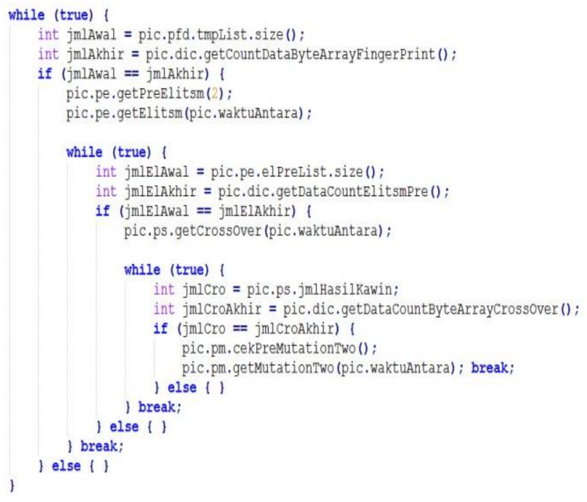


```
try {
    PriatyMain pic = new PriatyMain(args[0], args[1], args[2]);
    ExecutorService es = Executors.newCachedThreadPool();

    pic.pfd.preDigFiling(pic.pfd.fdrPathStr, encodedStr, finalString, pic.waktuAntara);
    pic.pfd.cekPreDigFilling(); pic.pfd.digFiling(encodedStr, finalString, pic.waktuAntara, es);

    while (true) {
} catch (Exception e) {
    e.printStackTrace();
}
```

Figure 4 Highest hierarchical Script for single thread loop and multiple thread loop speed testing.



```
while (true) {
    int jmlAwal = pic.pfd.tmpList.size();
    int jmlAkhir = pic.dic.getCountDataByteArrayFingerPrint();
    if (jmlAwal == jmlAkhir) {
        pic.pe.getPreElitsm(2);
        pic.pe.getElitsm(pic.waktuAntara);

        while (true) {
            int jmlElAwal = pic.pe.elPreList.size();
            int jmlElAkhir = pic.dic.getDataCountElitsmPre();
            if (jmlElAwal == jmlElAkhir) {
                pic.ps.getCrossOver(pic.waktuAntara);

                while (true) {
                    int jmlCro = pic.ps.jmlHasilKawin;
                    int jmlCroAkhir = pic.dic.getDataCountByteArrayCrossOver();
                    if (jmlCro == jmlCroAkhir) {
                        pic.pm.cekPreMutationTwo();
                        pic.pm.getMutationTwo(pic.waktuAntara); break;
                    } else { }
                } break;
            } else { }
        } break;
    } else { }
}
```

Figure 5 Second highest hierarchical script of single thread loop and multiple thread loop speed testing.

Each stage in the test script process are simultaneously measured the speed of the process using the System.currentTimeMillist() command line after process and then subtracted from the result of System.currentTimeMilllist() before process [12], so that the processing time is obtained in milliseconds.

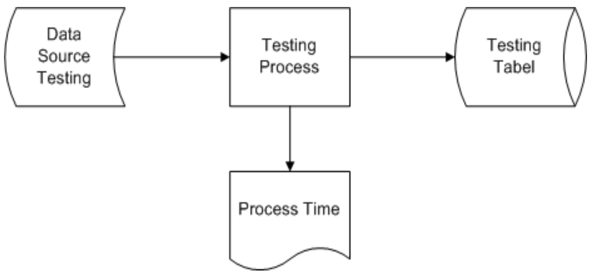
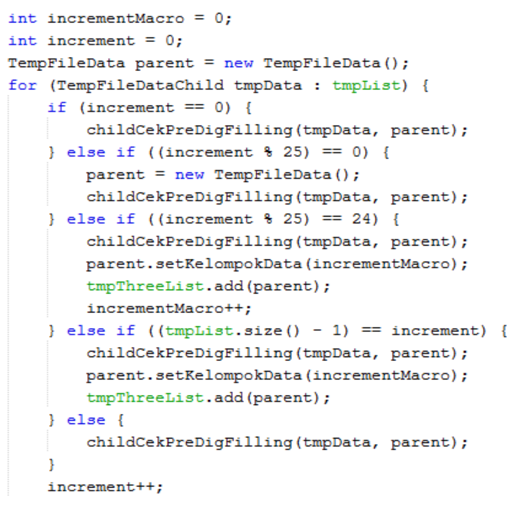


Figure 6 Stages of single thread loop and multiple thread loop speed testing scripts.

The command line script that distinguishes between Single Thread Loop and Multiple Thread Loop processes from the tests that have been carried out can be seen in Figure 7, where the Single Thread Loop command line does not have this script.



```
int incrementMacro = 0;
int increment = 0;
TempFileData parent = new TempFileData();
for (TempFileDataChild tmpData : tmpList) {
    if (increment == 0) {
        childCekPreDigFilling(tmpData, parent);
    } else if ((increment % 25) == 0) {
        parent = new TempFileData();
        childCekPreDigFilling(tmpData, parent);
    } else if ((increment % 25) == 24) {
        childCekPreDigFilling(tmpData, parent);
        parent.setKelompokData(incrementMacro);
        tmpThreeList.add(parent);
        incrementMacro++;
    } else if ((tmpList.size() - 1) == increment) {
        childCekPreDigFilling(tmpData, parent);
        parent.setKelompokData(incrementMacro);
        tmpThreeList.add(parent);
    } else {
        childCekPreDigFilling(tmpData, parent);
    }
    increment++;
```

Figure 7 Difference between single thread loop and multiple thread loops, which only exists in multiple thread loops.

3. Result and Discussion

From the initial 7200 data row then using a memetic algorithm in the process, consisting of elitism, cross over and mutation until it produces 17000 new data lines [13]. Using an elitism value of 2% of the initial population, 9800 additional data are generated from a series of memetic algorithm processes. Between pure Memetic Algorithm process flow with Memetic Algorithm wrapped in High Performance Computing (HPC) which is inserted Multiple Thread Loop algorithm in each testing process, and uses a combination of Single Thread Loop (STL), Multiple Thread Loop (MTL) test algorithms, and hardware testing logic Stand Alone (SA) and Multiple PC (MPC) which can be seen in table 2.

Table 2 Types of tests that have been carried out.

Specimen	Type of test			
	STL - SA	STL - MPC	MTL - SA	MTL - MPC
1	√	√	√	√
2	√	√	√	√
3	√	√	√	√
4	√	√	√	√
5	√	√	√	√
6	√	√	√	√
7	√	√	√	√
8	√	√	√	√
9	√	√	√	√
10	√	√	√	√
11	√	√	√	√
12	√	√	√	√
13	√	√	√	√
14	√	√	√	√
15	√	√	√	√

Then it's also necessary to know that each specimen is the result of grouping a collection of 7200 fingerprint image data, which is divided based on criteria to four factors. Then from these four factors it's varied to form 15 specimens. The four factors are;

- Fingerprint image data focused on the center point of the finger with the image cropped at the edge of the image border.
- Fingerprint image data that is more detailed and has image boundaries.
- More detailed fingerprint image data is broken and has a blank image border.
- Fingerprint image data that focuses on the center point of the finger but the image is damaged or unclear.

From the four factors, when viewed from the side of the fingerprint image display, it can be seen in Figure 8. When in each group of fingerprint images had its own characteristics.

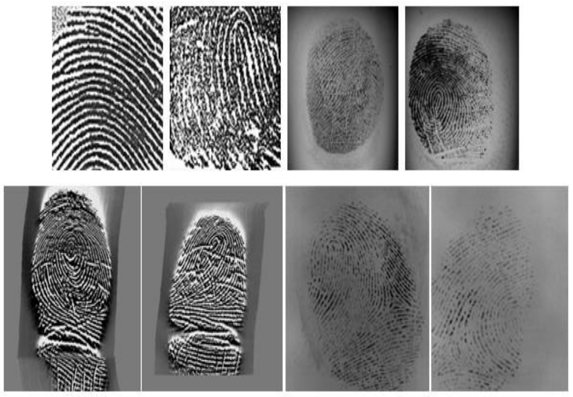


Figure 8 Sample groups of fingerprint images randomly from each factor with a total number 7200 data.

Then from the differences in the four factors of each group of fingerprint images, the authors tried to group the images in various ways, and then they were called specimen groups. Figure 9 below represents eight samples of grouping fingerprint images from a total of 15 specimens.

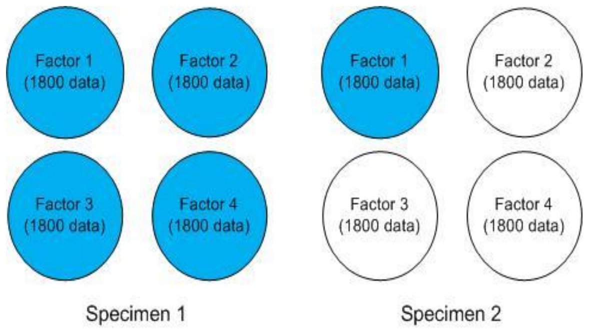


Figure 9 First parts of specimens grouping from total of 15 specimens taken as samples of four specimens

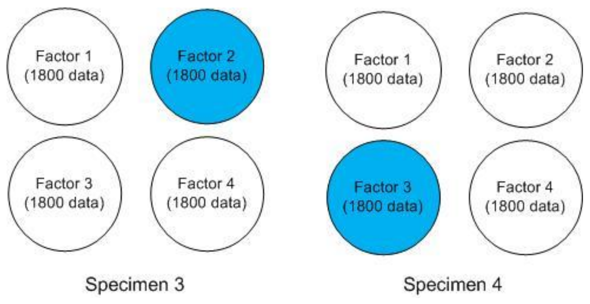


Figure 10 Specimens grouping from total of 15 specimens taken as samples of four specimens.

Figures 9, figure 10 and figure 11 are show how the process of grouping four types of fingerprint images into various specimens. Starting from taking one type of image, which eventually becomes specimens 2, 3, 4, and 5, and then continue to merging two types of fingerprint images such as specimens 6, 7 and 8, eventually combining all four types of images as seen in specimen 1 and so on in a process that creates specimen 9 to 15.

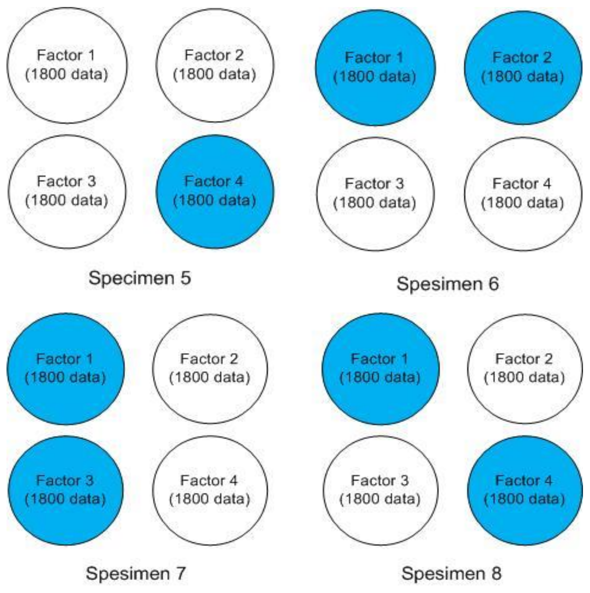


Figure 11 Specimens grouping from four factors into specimens, taken as samples of four specimens.

Then how do we read the data for speed testing? The testing process between Single Thread Loop and Multiple Thread Loop whose testing process uses the test logic architecture that has been illustrated in Figure 3, as well as the flow of the testing process which has also been illustrated in Figure 6. From the test results based on a predetermined scenario, it was found that the specimen number two shows the test results which are quite significantly high [14], In contrast specimen one shows the lowest test results in terms of access speed. Roughly speaking, these conditions show the weight and complexity level of the file to be one indicator of speed or slowness processing data from each of these specimens; it can be seen in the illustration of the comparison of results in figure 12 and also table of test results in table 3.

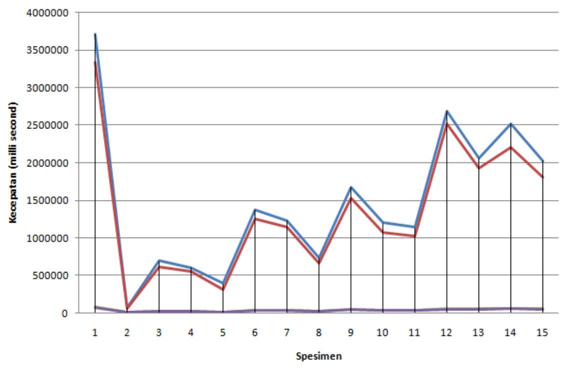


Figure 12 Test results in each specimen are seen from perspective of data processing speed.

The legend of each color line from the graph, are describe below;

- Single Thread Loop tested using Single PC represented by blue.
- Single Thread Loop tested using Multiple PC represented by brown.
- Multiple Thread Loop tested using Single PC represented by green.
- Multiple Thread Loop tested using Multiple PC represented by purple.

From the caption in Figure 11 which is represented by a line diagram with STL - SPC, STL - MPC, MTL - SPC and MTL - MPC data, the researchers took the main data, namely; processing speed of each specimen, all of these data are presented in Table 3.

Table 3 Thread loop speed test result

Specimen	Process Speed (millisecond)			
	STLPC	MTLSPC	STLMPC	MTLMPC
1	3714882	3345766	80982	72904
2	67908	62666	9746	8347
3	690889	618073	22037	20659
4	604551	552432	20919	19130
5	388697	322007	15552	13809
6	1377355	1258207	31988	31650
7	1224936	1144577	32260	28808
8	730945	663853	25545	23517
9	1674012	1530989	44985	41801
10	1197532	1079155	38077	37786
11	1138024	1026498	37573	34830
12	2688017	2524832	59094	48994
13	2065793	1937431	51372	47254
14	2519297	2215919	62838	57098
15	2027109	1812071	54061	44050
Max	3714882	3345766	80982	72904
Min	67908	62666	9746	8347

Specimen	Process Speed (millisecond)			
	STLPC	MTLSPC	STLMPC	MTLMPC
Average	1473996	1339632	39135.27	35375.8

The legend of each colored cell from table 3, are describe below;

- The orange colored cell means the longest processing time.
- The green colored cell means the fastest processing time.
- The blue cell means that the specimen processing time is closest to the average processing time.

From the data that appeared in Table 3, it means that fingerprint data processing result test from 15 specimens combined with four testing methods, resulting in a processing time which then processed and sorted [15]. Then see which time is the fastest in data processing between using a single thread loop and multiple thread loops.

The speed of data access from each specimen is compared with each test scenario based on the data from table 3, known to have the following facts;

- Specimens' number one and number two are specimens with the lowest and highest test results considering that specimen number one is a mixture of all data groups, while specimen number two only contains data from factors to one data group that has the simplest image complexity.
- The specimens in the average processing speed range are specimen number six for the STL - SPC and MTL - SPC tests, while for the MTL - SPC test is specimen number 10 and MTL - MPC is specimen number 11. Specimen number 10 is a mixture data with the second and fourth factors while specimen number 11 is a mixture of data with the third and fourth factors.
- By looking at the fact that the data reading of the two specimen's number 10 is a mixture of data with a high level of image complexity and produces a large binary file size, it can still be in a significant processing speed of 38 seconds using the STL - MPC test.
- As for specimen number 11, which comes from a mixture of complex fingerprint data but the image clarity is not good, it can still be processed at a speed of 34 seconds using the MTL – MPC test.

From the provisional discussion of results from Table 3, the researcher tries to conclude that specimen number two excels in access speed and specimen number one performs the process with the slowest time.

4. Conclusions

The research that has been conducted on the performance investigation testing of Single Thread Loop and Multiple Thread Loop in data processing with the number of fingerprint image data amount of 7200 images, it is known that several results can be known;

- Multiple thread loops is much faster than Single Thread Loop in terms of speed.
- The speed of the Multiple Thread Loop process also depends on data complexity, because fingerprint image data with a high level of image sharpness will produce a large binary file, affecting the processing speed.
- Multiple Thread Loop with a tiny number of threads will affect the consumption of utility processors so that the process runs for a long time and can result in a process crash. Still, if the number of threads is too large, it will affect the efficiency value or will require many processor cores and automatically large costs.

5. Acknowledgement

This research was supported by Langlangbuana University. We thank our colleagues Dr. Priati from Binus University who provided insight and expertise that greatly assisted the research.

References

[1] T. Arifianto, "Penerapan Fingerprint Recognition Dengan Metode Learning Vector Quantization (Lvq) Dalam Automatic Teller Machine (Atm)," *Spirit STMIK Yadika J. Comput. Cybern. Syst.*, vol. 9, no. 2, pp. 8–13, 2017.

[2] A. B. Channegowda and H. N. Prakash, "Multimodal biometrics of fingerprint and signature recognition using multi-level feature fusion and deep learning techniques," *Indones. J. Electr. Eng. Comput. Sci.*, vol. 22, no. 1, p. 187, 2021, doi: 10.11591/ijeecs.v22.i1.pp187-195.

[3] A. Ginanjar and B. Ilman, "Kehandalan Data Reading Cache Engine MyBatis Framework Terhadap Algoritma LRU, FIFO, SOFT dan WEAK dalam Lingkungan Java dan MySQL," *J. Teknol.*, vol. 2019, 2020.

[4] M. B. Firdausa, R. I. Rokhmawati, and S. A. Wicaksono, "Pengembangan Sistem Informasi Manajemen Keuangan Lesehan & Kolam Pancing Kresna Berbasis Website Menggunakan Model Evolutionary Prototyping," vol. 5, no. 12, pp. 5665–5671, 2021.

[5] O. Irnawati *et al.*, "Evolutionary Prototype Dalam Perancangan Sistem," vol. 6, no. 1, pp.

1–8, 2021.

[6] I. Kamlasi, “Descriptive Analyses on English Test Items based on the Application of Revised Bloom’s Taxonomy,” *Metathesis J. English Lang. Lit. Teach.*, vol. 2, no. 2, p. 203, 2018, doi: 10.31002/metathesis.v2i2.847.

[7] A. Tinar, S. H. Wijoyo, and R. I. Rokhmawati, “Evaluasi Usability Tampilan Antarmuka Website Perpustakaan Politeknik Kesehatan Kemenkes Kota Malang menggunakan Metode Usability Testing dan Heuristic Evaluation,” *J. Pengemb. Teknol. Inf. dan Ilmu Komput.*, vol. 3, no. 11, pp. 10453–10461, 2019.

[8] P. Pakutharivu and M. V. Srinath, “Analysis of fingerprint image enhancement using gabor filtering with different orientation field values,” *Indones. J. Electr. Eng. Comput. Sci.*, vol. 5, no. 2, pp. 427–432, 2017, doi: 10.11591/ijeecs.v5.i2.pp427-432.

[9] D. Dedi, A. Sidik, M. Raya, and M. B. Ryando, “Perancangan Sistem Informasi Promosi Jasa Foto dan Studio Musik Pada M2N Studio Production,” *J. Sisfotek Glob.*, vol. 11, no. 1, pp. 48–52, 2021, doi: 10.38101/sisfotek.v11i1.344.

[10] Z. Xiaoshuan, F. Zetian, C. Wengui, T. Dong, and Z. Jian, “Applying evolutionary prototyping model in developing FIDSS: An intelligent decision support system for fish disease/health management,” *Expert Syst. Appl.*, vol. 36, no. 2 PART 2, pp. 3901–3913, 2009, doi: 10.1016/j.eswa.2008.02.049.

[11] A. Ginanjar and M. Hendayun, “Spring framework reliability investigation against database bridging layer using Java platform,” *Procedia Comput. Sci.*, vol. 161, no. December, pp. 1036–1045, 2019, doi: 10.1016/j.procs.2019.11.214.

[12] A. Ginanjar, “Perbandingan Kehandalan Operasi CRUD Menggunakan Perpaduan Spring dan MyBatis Framework serta Algoritma Cache Engine .,” vol. 18, no. 1, pp. 11–18, 2021.

[13] M. T. Keimigrasian and P. Imigrasi, “Fingerprint identification with memetic algorithm and high-performance computing memetic algorithm (HPCMA),” *Commun. Math. Biol. Neurosci.*, pp. 1–17, 2021, doi: 10.28919/cmbn/6159.

[14] P. Assiroj, H. L. H. S. Warnars, E. Abdurrachman, A. I. Kistijantoro, and A. Doucet, “Measuring memetic algorithm performance on image fingerprints dataset,” *Telkomnika (Telecommunication Comput. Electron. Control.*, vol. 19, no. 1, pp. 96–104, 2021, doi: 10.12928/TELKOMNIKA.V19I1.16418.

[15] P. Assiroj, H. L. H. S. Warnars, E. Abdurachman, A. I. Kistijantoro, and A. Doucet, “The influence of data size on a high-performance computing memetic algorithm in fingerprint dataset,” *Bull. Electr. Eng. Informatics*, vol. 10, no. 4, pp. 2110–2118, 2021, doi: 10.11591/EEI.V10I4.2760.