International Journal of Hybrid Information Technology
Vol.6, No.5 (2013), pp.89-102
http://dx.doi.org/10.14257/ijhit.2013.6.5.09

Optimization Using Multicore-Processor Parallelization: A Case Study of Tax Processing System in Indonesia

Antonius Bima Murti Wijaya
Informathics Engineering, University of Sebelas Maret Surakarta
bimamurti@gmail.com

Abstract

National Ministry of Manpower of Indonesia stated that statistically, the total of civil servants in January 2013 is 4,467,982 [2]. The total of regency or municipality in Indonesia, based on interior ministry of Indonesia, is 497. Therefore, the average number of civil servants in each regency or municipality is about 9909. Every year, the citizens who have been affected by taxes, including these civil servants, need to create incomes and taxes reports. This processing system was made to help the government to process all salary data for taxes income reports.

In improving the speed process of this system can be done in two ways. The first way is by improving computer hardware or algorithm of this system to use effectively the existing hardware such as processor [3]. This research will observe the usage of multithread to process mass data, by considering the bottleneck effect on hard disk. An overlapping method is suggested on this research to exploit the processor resource. The analysis using mathematical method, empirical data, and visualization will be served in this research. The result of this research is that the overlapping method has improved the speed of taxes process system by showing the time reduction until 65.36%.

Keywords: *Optimization, Multicore-Processor, Parallelization*

1. Introduction

Based on Ministry of Indonesian Central Statistic, the population of Indonesia in 2010 was 237,641,326 [1]. Many people work as civil servants to manage them all. National Ministry of Manpower of Indonesia stated that statistically, the total of civil servants in January 2013 is 4,467,982 [2]. As the civil servants, they must pay tax. Tax, as one of national income, need to be observed in a period of time. In this research, tax observation for civil servants will be analyzed for states of province or city. Total of regency or municipality in Indonesia based on Indonesian Ministry of Interior is 497, so the average number of civil servants in each regency or municipality is about 9909. The distribution of these civil servants totally depends on the needs of the regency or municipality. Every year citizens who have been affected by taxes, including this civil servant, need to create income report. It usually every regency or municipality collect all salary data of their civil servants and process it altogether. This processing system was made to help the government to process all salary data for taxes income reports. Some of the states of province collect their civil servants salary data on spreadsheet format for each month salary from January until December, including thirteenth salary. If there are 9000 civil servants in a province state, then they should have 9000 x 13 months of salary that has 117000 data. This processing system will import all of this data from the spreadsheet to this system, clean the bad record of data, and save it to the database, so the system can generate the income reports. The clean activity in this system means repairing the fault records and if cannot be repaired, it will not be processed.

In improving the speed process of this system can be done in two ways. The first way is by improving computer hardware or algorithm of this system to use effectively the existing hardware such as processor [3]. This research will observe the usage of multithread to process mass data, by considering the bottleneck effect on hard disk. In this case, the development of this research will be not for general purpose but specific. The analysis applies mathematical method, empirical data, and visualization and those will be served in this research. The purpose of this optimization is to explain how this optimization works, and compare between all of these optimization methods.

2. Multicore Architecture

Nowadays, the processor was developed in multiple core architecture. Parallelization using multi-thread was made as one of methods to exploit the processor resources. There are three level of parallelism, instruction, data, thread, which can be used to improve the performance by using super scalar execution, multi-threading computation, and streaming Single Instruction Multi Data [4]. Beside of using the multithreading method, *Open MP* is the other method to do parallelization, and the result is not really different with the multi-thread method if we use the method in local parallelization case [5, 6]. The way to make efficient processor performance is by exploiting their queuing methods [7]. Beside *Open MP* and *Multithread* method there is another method that is *CellSs*. *CellSs* is a technology that uses multithread method for multi-core architecture. In this case, the number of threads is less than 4, and this *CellSs* techonolgy is faster than *OpenMP* method [8]. The development of this multi-core technology can affect the data structure such as queue and stack which needs to be modified so can optimize the function of processor multi-core [9].

In order to do high performance computing, it needs to be focused on some characteristic at Multi-core Architecture:

a. Load Balancing: this characteristic is the main condition for high performance computing. This can be an issue when many cores of appearing processor .
b. The Number of Parallelization
c. Effective Scheduling [6].

3. Optimization

Optimization method can be divided into two types that are direct search method and math method [10]. The math method is just like big O notation, big Gamma notation, neural network, and linear programming [11]. The linear programming can be used for multi objective issue [12]. In its development, Linear Programming can be combined with another method such as genetic algorithm or swarm particle [13]. Combining some algorithms is more about completing each other to solve specific problem [14]. On the development of optimization method in a system, sometimes two objects are conflicting, if there is an increasing part in an object, the other part of object become decreasing. This issue can be solved by using multi objective optimization [15, 16].

The optimization method is not only in algorithm but also in database query or in database processes. By using cache memory to temporary save, the query result will be done because of there is the situation where the system send the same query at once or many users at the different time [17, 18]. Besides doing optimization on the query, the accuracy and efficiency are important to be noticed [19]. The query of optimization can be done by using parallelization. The main purpose of the parallel query is for improving the speed and the bigger output [20]. The database of optimization technique cannot continually have good performance in all environments. It is better to develop the adaptive technique for specific context rather than develop it for general purpose [21, 22].

Deep analysis of an old algorithm with a new algorithm is needed for developing an optimization method. There are three types of algorithm analysis that commonly used, there are mathematic, empiric, and algorithm visualization. This kind of analysis has been done by Lin and Ke [23]. The purpose of their algorithms is to explain how the algorithm can reach the maximum point from the existing problem. The other research of this analysis has been done by Sung and Yuen [24]. They did the research by using mathematic equation and empiric data in a chart form. The purpose of their research is to find how fast the algorithm reaching the maximum point and effectiveness of memory usage with execution time.

3. Amdhal's Law

Amdhal's Law characters the maximum velocity (S) that can be reached of *n* processor that collaborates at an application. It is *p* a fraction of a computation which can be executed in parallel. Assume that (normalized) 1 time for a single processor to complete computation. With the *n* related processor, the part of parallel needs *p/n* time, and each of their connection needs *1-p* time. Overall, the parallelism computation needs time: $1 - p + \frac{p}{n}$. The Amdhal's law said increasing speed is the ration between time series at a single processor and parallel time: $= \frac{1}{1-p+\frac{p}{n}}$, in other words *s* is not equivalent with *n*. It is not in every case parallelism part (p) can be always 100%, it can be less than that. It is better to focus on this *100% - p* part [9].

4. The System Process

There are two main processes in this case. The first process is importing the civil servants data, and the second one is importing their payroll data. All of these processes always include process of importing file source from the spreadsheet, adjusting data format, and inserting data to data base. The main problem of this system is making the spreadsheet data saved into the database; because this process need many sub-processes such as importing, adjusting, and saving data. This is how the system works until data is saved on the database. In the process of importing civil servants data, the process will start from the twelfth month and go back to the first month. This is to make the data that got from the last update, this process have inevitability of permutation. In the process of importing the payroll data, the process can start from every month (first month until the thirteenth month) without attending the permutation. This difference will make a different treatment for the parallelization method.

Before the parallelism method appears, the algorithm of each main process should be defocused from secondary memory and focused on primary memory since the access speed in primary memory is much faster than secondary memory. All processes of selecting, repairing, and adjusting the data happen in the primary memory. That is why on process number 5 on Figure 1 is not about saving data to the database and process number 4 on Figure 2 is not about inserting data to the database. All of the unfixed data will be saved on primary memory which in this case, the data is saved to the temporary variable. So when the data is accessed for repairing or selecting or updating process, the process will not touch the secondary memory. After all the data fixed, the data will be saved on the secondary memory which is hard-disk used the export method such as the bulk insert. By using this way, the parallelization can be done for selecting, adjusting, and repairing processes without stuck on bottleneck effect caused by the speed and the queue of secondary memory.

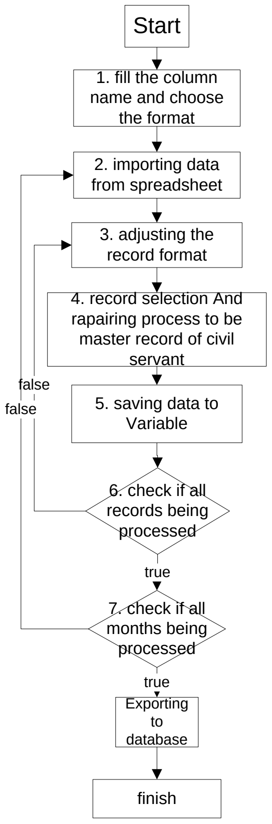

Start
1. fill the column name and choose the format
2. importing data from spreadsheet
3. adjusting the record format
4. record selection And rapairing process to be master record of civil servant
5. saving data to Variable
6. check if all records being processed
true
7. check if all months being processed
true
Exporting to database
finish
false
false

Figure 1. Importing civil servant data

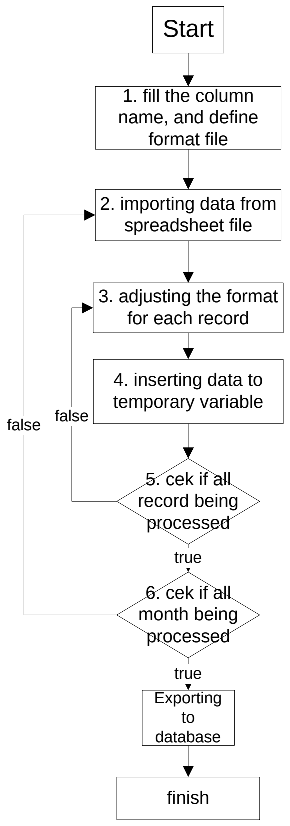

Start
1. fill the column name, and define format file
2. importing data from spreadsheet file
3. adjusting the format for each record
4. inserting data to temporary variable
5. cek if all record being processed
true
6. cek if all month being processed
true
Exporting to database
finish
false
false

Figure 2. Importing the payroll data

There are three types of spreadsheet formats. First is 1 file 13 sheets, means that the data is consisted of 1 file that contains 13 sheets. Second is 13 files means that the data is consisted of 13 files which in each file consist only 1 sheet. The last format is 1 file 1 sheet means that data is collected at one file in the 1 sheet. This is the specification of spreadsheet file that use for empiric and mathematical analysis:

Table 1. file specification

Format	File Size (KB)	Amount of record
1 file 13 sheet	45737	213187
13 file	3645,3620,3621,3608,3598,3668,3640, 3625, 3645,3599,3591,3446,3666	213187
1 file 1 sheet	46024	213187

5. The System Analysis

There are two main processes that will be upgraded with this optimization method, which are input process for employee data and for salary data. This process cannot be described well by using mathematical analysis since this process includes *sql* statement that cannot explore as well. The mathematical problem will describe all variable that affected the speed process. To develop the optimization algorithm, there is the mathematical analysis about time function with another variable from this case that affects the speed process. The time function is:

$t_{total}=nt_n+Lt_l+Kt_k+\sum_{i=1}^{O} t_{o(i)}+\sum_{i=1}^{R} t_{R(i)}$ (1)

where,

t_{total}	: total execution time
n	: amount of data
t_n	: time that needed to adjust the form per one data
L	: amount of query that is inserted to database log
t_l	: time to process 1 query to database log
K	: amount of connection that is used to database
t_k	: time to open 1 connection that is used to database
O	: amount of process to open spreadsheet file
t_o	: time to open 1 spreadsheet file
R	: amount of process to read file
t_R	: time to read 1 spreadsheet file
i	: sequential variable

According to the process happens on importing civil servants data which focused on the primary memory the total execution time is:

Format 1 file 12 sheets:

$t_{total}=213178t_n+1t_l+16884t_k+1t_o+\sum_{i=1}^{12} t_{R(i)}$

Format 13 file :

$t_{total}=213178t_n+1t_l+16884t_k+12t_o+\sum_{i=1}^{12} t_{R(i)}$

Format 1 file 1 sheet :

$t_{total}=213178t_n+1t_l+16884t_k+1t_o+1t_{R(1)}$

According to the process happens on importing payroll data which focused on the primary memory the total execution time is:

Format 1 file 12 sheets :

$t_{total}=213178t_n+13t_l+213179t_k+1t_o+\sum_{i=1}^{13} t_{R(i)}$

Format 13 file :

$t_{total}=213178t_n+13t_l+213179t_k+13t_o+\sum_{i=1}^{13} t_{R(i)}$

Format 1 file 1 sheet :

$t_{total}=213178t_n+13t_l+213179t_k+1t_o+1t_{R(1)}$

By focusing on the system on primary memory, the value of L, K, O, R can be minimized to 1 or 13 depend on the format. This is the performance of the system to finish their process until it is saved into the database:

Table 2. Performance of importing civil servant data algorithm

	experiment (s)										
Format	1(s)	2	3	4	5	6	7	8	9	10	Average
1 file 12 sheet	111.9	110.6	110.5	109.8	111.4	111.3	110.8	111.3	111.3	110.7	90.21
12 file	113.0	113.4	114.7	113.4	114.6	115.3	114.1	113.2	115.1	113.8	85.16
1 file 1 sheet	115.9	115.5	114.8	115.7	115.3	115.3	115.2	115.9	115.8	115.2	85.17

Table 3. Performance of importing payroll data algorithm

	experiment (s)										
Format	1	2	3	4	5	6	7	8	9	10	Average
1 file 12 sheet	53.5	53.5	52.6	53.6	52.8	53.1	53.1	52.9	53.4	53.3	53.18
13 file	58.6	58.8	59.1	58.2	59.2	58.1	59.2	58.4	58.3	58.7	58.66
1 file 1 sheet	85.9	87.1	88.5	89.4	87.4	88.2	89.3	85.9	88.7	85.6	87.6

6. Parallelization Algorithm

In this case, the parallelization would be the local parallelization happens on a computer using multithread technology. It needs to be understood that a work in a thread is not represent a work in a physical core in the processor. While the running system only use 25% source of processor that have two cores of processor, so it should be two times faster, if the condition of all parts can be parallelized. In this case, it is not all parts can be parallelized. There is still a part that can be parallelized, such as importing the spreadsheet process since it communicates with the secondary memory. There are two main steps to develop the parallelization algorithm:

Step 1: Separate the part that can be parallelized by the part which not with a long time execution. In this case, the processes are selecting, repairing and adjusting the record, while the process which cannot be parallelized is the process of importing spreadsheet data.

Step 2: Decide the suit treatment of parallelization for each processes or case.

a. In the process of importing civil servants data, the parallelization appears in the process of selection, adjusting and repairing data in each month, but cannot be parallelized for some months since the process need to be permuted. This is how the parallelization is implemented:

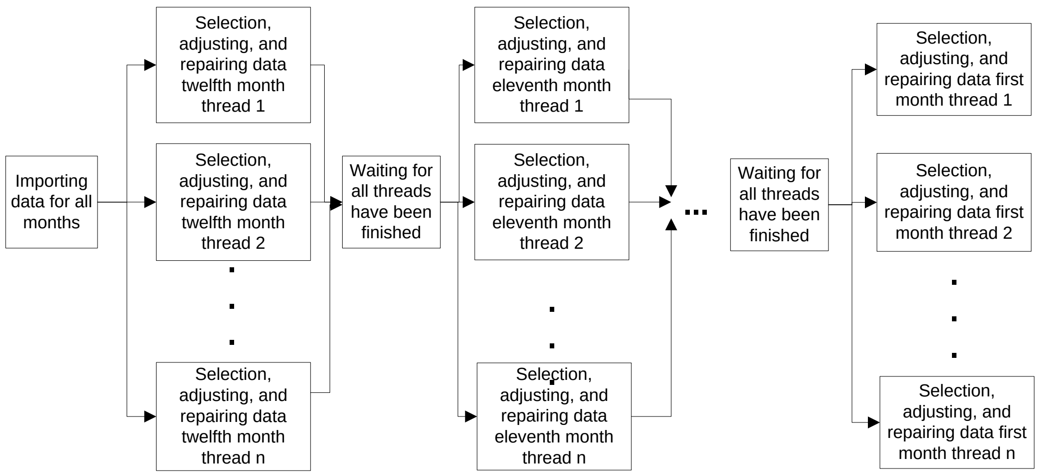

Importing data for all months
Selection, adjusting, and repairing data twelfth month thread 1
Selection, adjusting, and repairing data twelfth month thread 2
Selection, adjusting, and repairing data twelfth month thread n
Waiting for all threads have been finished
Selection, adjusting, and repairing data eleventh month thread 1
Selection, adjusting, and repairing data eleventh month thread 2
Selection, adjusting, and repairing data eleventh month thread n
...
Waiting for all threads have been finished
Selection, adjusting, and repairing data first month thread 1
Selection, adjusting, and repairing data first month thread 2
Selection, adjusting, and repairing data first month thread n

Figure 3. Parallelization on importing payroll data process

The process will start by importing data from the twelfth month to the first month from the spreadsheet, than after importing process, the parallelization appears at selection, adjusting and repairing data from the twelfth month to the first month. After finishing the parallelization process at month twelve, the system will continue to the next month to the first month. The waiting process is a process of waiting all threads in a month to finish their job. This happens since it has inevitability of permutation. This is the viusalization example on 1 file 12 sheets format:

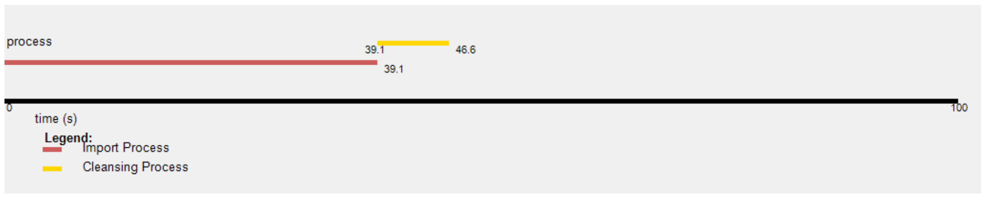


Figure 4. Visualization of importing civil servant data, 1 file 12 sheet format

There are two different colors in the picture. The red one is the import process and the yellow one is the cleansing process (selecting, repairing and adjusting). The parallelization is implemented at yellow line. This is the time execution result of this parallelization:

Table 4. Parallelization on importing civil servants data process

Format	Experiment- (s)										
	1	2	3	4	5	6	7	8	9	10	Average
1 file 12 sheet	46.9	46.5	46.7	46.7	46.5	46.7	46.0	46.6	46.5	46.7	46.58
12 file	43.5	43.2	43.5	43.5	43.4	43.7	43.6	43.6	43.3	43.9	43.52
1 file 1 sheet	52.6	52.4	52.5	52.8	52.8	52.5	52.5	52.3	52.7	52.3	52.54

This is the comparation between Table 4 against Table 2:

Table 5. Performance of parallelization

Format	Average Reduction (%)	Maximum Reduction(s)	Minimum Reduction(s)
1 file 12 sheet	58.02	-65	-63.8
13 file	61.84	-71.4	-69.8
1 file 1 sheet	54.50	-63.1	-62.5

*Average Reduction = 100-((Average 1/ Average 2)*100)*

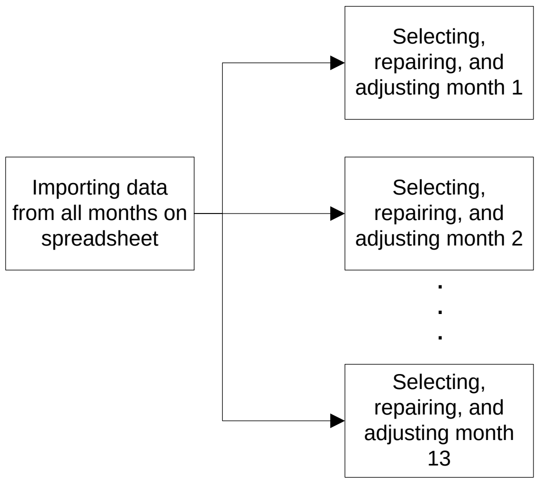
Where, Average 2= average value from table 4

Average 1= average value from table 2

Maximum Reduction = Maximum value from Table 4 - Maximum value from Table 2

Minimum Reduction = Minimum value from Table 4 - Minimum value from Table 2

b. In the process of importing payroll data, there is no inevitability of permutation in this process. The parallelization can be implemented at some months. This is how the parallelization implemented:


Importing data from all months on spreadsheet
Selecting, repairing, and adjusting month 1
Selecting, repairing, and adjusting month 2
Selecting, repairing, and adjusting month 13

Using this way of parallelization, the queueing method and load balancing issue can be passed and let the processor technology handle this issue.

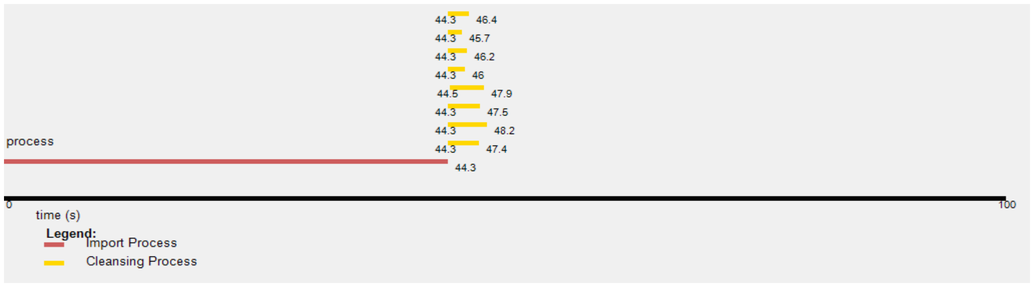


Figure 5. Visualization of importing payrol data, 1 file 13 sheet format.

The import process takes 44.3 seconds and the longest cleansing process takes 3.6 seconds so the total time execution is 47.9 seconds. The time execution result by using this way:

Table 6. Parallelization on importing salary data process

	experiment- (second)										
Format	1	2	3	4	5	6	7	8	9	10	Average
1 file 13 sheet	47.3	47.8	47.8	50	48.2	48.2	47.7	48.8	47.5	47.9	48.12
13 file	50.2	49.1	49	49.4	49.6	49.8	49.6	49.1	48.9	49.1	49.38
1 file 1 sheet	55	55.8	54.9	55.5	55.8	55.3	55.7	55.6	55.3	56	55.49

Table 7. Perfromance of parallelization

Format	Average Reduction (%)	Maximum Reduction(s)	Minimum Reduction(s)
1 file 12 sheet	9.61	-3.6	-5.3
13 file	15.82	-9	-9.2
1 file 1 sheet	36.66	-33.4	-30.7

*Average Reduction = 100-((Average A/ Average B)*100)*

Where, Average A= average value from Table 6

Average B= average value from Table 3

Maximum Reduction = Maximum value from Table 6 - Maximum value from Table 3

Minimum Reduction = Minimum value from Table 6 - Minimum value from Table 3

7. Parallelization using overlapping method

The parallelization algorithm above does the parallelization, after all data have been imported from the spreadsheet. In other words, the parallelization appears after the part that cannot be parallelized has done their processes. The main purpose of this method is to make the parallelization execution of time closer with the execution of the part that cannot be parallelized, by inserting the parallelization process in the process that cannot be parallelized. The process of non-parallelized part has some remaining processor resources, so some of parallelized parts can be processed using the remaining processor resources from parts of process that cannot be parallelized. In this method the time function becomes:

$T_{total}=T_{bottleneck}+(T_{paralel}-T_{overlap})$

where,

T_{total}	: Execution Time Total
$T_{bottleneck}$	: Execution Time from process that cannot be parallelized
$T_{paralel}$	: Execution Time from process that can be parallelized
$T_{overlap}$	: Execution Time from parallelized process that can be overlapped with non parallelized process

$T_{paralel} > T_{overlap}$

From that function, the purpose of this method is to increase the value of $T_{overlap,}$ so the value is closer to the value of $T_{paralel}$. In the Amdhal's law, this method has purpose to maximized the value of process that can be parallelized that is *p*. In this case, the process of selecting, repairing and adjusting will be inserted on each process of the importing data. In the format 1 file 1 sheet, this method cannot be implemented because the importing process only happens once. This overlapping method still uses the last parallelization algorithm.

a. Parallelization using Overlapping method on importing the civil servants data.

Every process of importing, selecting, repairing, and adjusting (cleansing process) in this process must be ordered, so the implementation of this method must not overlap in each month process especially for selecting, repairing and adjusting. The overlapping method appears between process of selecting, repairing, adjusting with process importing the civil servants data from the spreadsheet and will start after the twelfth month have been imported. While the importing process in each month executed the selecting, repairing and adjusting process will process the imported month that have been finished.

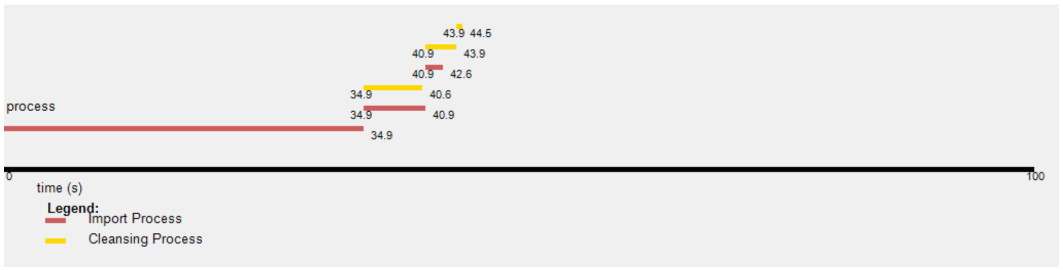


Figure 6. Visualization of overlapping method on importing civil servant data on format 1 file 12 sheets

In the Figure 4, the first red line is the process of importing data from spreadsheet at month 12 until 5, then after importing the month 12-5, the process start to overlap by including cleansing process (selecting, repairing and adjusting process) at month 12-6 and importing the month 5-2, after finish it, the process start cleansing process at month 5-2 together with importing the month 1. In the end, the process starts to clean the month number 1.

Table 8. Result of Importing process on importing civil servant data process

Format	experiment- (second)										
	1	2	3	4	5	6	7	8	9	10	Average
1 file 12 sheet	44.0	44.6	43.9	44.5	44.1	44.1	44.6	44.4	44.1	44.2	44.25
12 file	39.3	39.3	40.3	39.9	39.9	39.4	39.3	39.5	38.8	39.4	39.51

Table 9. Performance of overlapping method on importing civil servant data

Format	Average reduction (%)	Average reduction 2 (%)	Maximum value Reduction (s)	Minimum value Reduction (s)
1 file 12 sheet	60.12	5.00	-2.3	-2.1
13 file	65.36	9.21	-3.6	-4.4

*Average Reduction = 100-((Average A/ Average B)*100)*

*Average Reduction 2 = 100-((Average A/ Average C)*100)*

Where, Average A= average value from Table 8

Average B= average value from Table 2

Average C= average value from Table 4

Maximum value Reduction = Maximum value from Table 8 - Maximum value from Table 4

Minimum value Reduction = Minimum value from Table 8 - Minimum value from Table 4

b. Parallelization using Overlapping method on the importing salary data.

Every process of selecting, repairing, and adjusting have no avitablitiy of permutation so the overlapping method can be implemented by random month, but specifically there is a rule that the salary data in month 13 must be in month 6, too. Therefore, the process will start by importing month 6 first then the month 13 from the spreadsheet file. While the system continue importing spreadsheet file from another month, the system can process the selecting, repairing and adjusting at month 6 and month 13, followed by another month that have been imported from the spreadsheet.

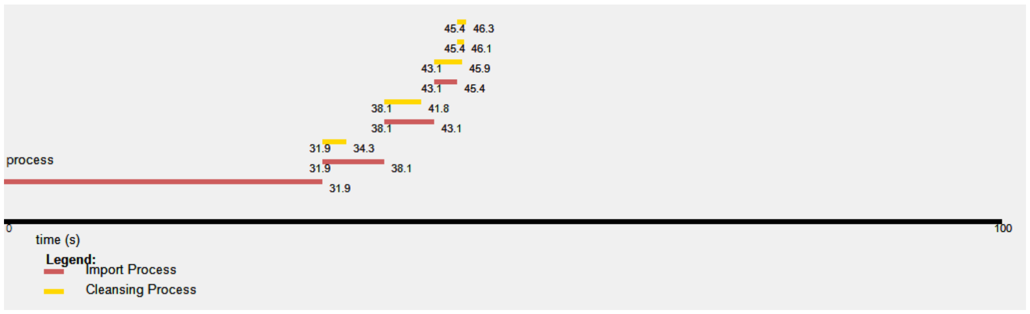


Figure 7. Visualization of overlapping method on importing salary data on format 1 file 13 sheets

In the picture 5, the data on month 6 and 13 are imported at first, after finish importing the data, the system starts the cleansing process form month 6 and 13 together with importing data from month 1 until 5. When the process of importing on month 1 until 5 finish, the process starts to do cleansing process of month 1-5 together with importing data from month 6-10 . Then, the last month is for month 11 and 12. This is the empirical data time execution performance of overlapping method on importing salary:

Table 10. Result of Importing process on importing salary data process

Format	Experiment- (second)										
	1	2	3	4	5	6	7	8	9	10	Average
1 file 13 sheet	46.9	46.2	47	46.1	46.6	46.3	46.8	46.1	46.5	46.8	46.53
13 file	46	45.9	45.6	46.3	46.5	46.5	46.2	46.6	46.2	45.6	46.14

Table 11. Performance of overlapping method on importing salary data

Format	Average reduction (%)	Average reduction 2 (%)	Maximum value reduction (s)	Minimum value reduction (s)
1 file 13 sheet	12.50	3.30	-3	-1.2
13 file	21.34	6.56	-3.6	-3.3

*Average Reduction = 100-((Average A/ Average B)*100)*

*Average Reduction 2 = 100-((Average A/ Average C)*100)*

Where, Average A= average value from table 10

Average B= average value from table 3

Average C= average value from table 6

Maximum value Reduction = Maximum value from table 8 - Maximum value from table 4

Minimum value Reduction = Minimum value from table 8 - Minimum value from table 4

By using this overlapping method, the process of importing data takes longer time. In the picture 4 and 5, the importing process ends at 39.1 and 44.3 (the red line), but in the picture 6 and 7 the importing process (the red line) ends at 42.6 and 45.4. This data indicates that the cleansing process in the overlapping method can steal the execution queue for the importing process in the processor, even though the priority of importing process is high.

8. Conclusion and Future Research

Based on the research experiment, the result of this overlapping method shows that this parallelization method can reduce the execution time until 65.36% for the importing civil servants data process, and 21.34% for the importing salary data process. In the Amdhal's law, this method has purpose to maximize the value of process that can be parallelized that is *p*.

The overlapping method shows that importing process takes longer time for the execution. For further research, it still needs to be focused on this part, to determine the problem and find the solution from this lack, so this overlapping method can reach the purpose more.

References

[1] Annonymous, "Population of Indonesia by Province 1971, 1980, 1990, 1995, 2000 and 2010", Central Statistic Ministry, **(2010)**.
[2] Annonymous, "Number of civil servants by educational level and gender", Ministry of National Employment, **(2013)**.
[3] T. Chen, M. Feng, V. Nagarajan and R. Gupta, "Speculative Parallelization of Sequential Loops on Multicores", Int J Parallel Prog, vol. 37, **(2009)**, pp. 508–535.
[4] M. I. Soliman, "Performance Evaluation of Multi-core Intel Xeon Processors on Basic Linear Algebra Subprograms", World Scientific Publishing Company, vol. 19, no. 1, **(2009)**, pp. 159-174.
[5] P. Ivanov and K. Brandisky, "Parallel implementation of evolution strategy optimization algorithm on multicore processors", The International Journal for Computation and Mathematics in Electrical and Electronic Engineering, vol. 28, no. 5, **(2009)**, pp. 1129-1140
[6] P. E. Hadjidoukas, G. Ch. Philos and V. V. Dimakopoulos, "Exploiting fine-grain thread parallelism on multicore architectures", Scientific Programming, vol. 17, **(2009)**, pp. 309–323.
[7] S. H. Semnani and K. Zamanifar, "New Approach To Multi-Level Processor Scheduling", World Scientific Publishing Company, vol. 19, no. 3, **(2010)**, pp. 335–346.
[8] A. Schiller, G. Sutmann, L. Martinell, P. Bellens and R. Badia, "Particle Methods on multicore architectures: Experiences and Future Plans", Numerical Analysis and Applied Mathematics international conference, American Institute of Physics, vol. 3, **(2010)**, pp. 1797-1800.
[9] N. Shavit, "Data Structures in the Multicore Age", Communications of the ACM, vol. 54, Issue 3, **(2011)**, pp. 76-84.
[10] W. Abo-Hamad and A. Arisha, "Simulation–Optimisation Methods in Supply Chain Applications: A Review", irish journal management, **(2011)**, pp. 97-124.
[11] S. R. McAllister, R. Rajgaria and C. A. Floudas, "A path selection approach to global pairwise sequence alignment using integer linear optimization", Taylor and Francis, vol. 57, no. 1, **(2008)**, pp. 101–111.
[12] M. Maimos, Y. Cherruault, B. O. Konfe and A. S. N. Massamba, "Alienor method to solvemulti-objective linear programming (MOLP)", Kybernetes, vol. 38, no. 5, **(2009)**, pp. 789-799.
[13] M. Cisty, "Hybrid Genetic Algorithm and Linear Programming Method for Least-Cost Design of Water Distribution Systems", Water Resour Manage, no. 24, **(2009)**, pp. 1–24.
[14] Hagh, M. T. Dar and S. Galvani, "Minimization of load shedding by sequential use oflinear programming and particle swarm optimization Turk", J Elec Eng & Comp Sci, vol. 19, no. 4, **(2011)**, pp. 551-563
[15] K. Vergidis, A. Tiwari and B. Majeed, "Business process improvement using multi-objective optimization", BT Technology Journal, vol. 24, no. 2, **(2006)**, pp. 229-235.
[16] M. Luque, F. Ruiz and K. Miettinen, "Global formulation for interactive multiobjective Optimization", vol. 33, **(2011)**, pp. 27–48.
[17] V. Gour1, S. S. Sarangdevot, A. Sharma and V. Choudhary, "Improve Performance of Data Warehouse by Query Cache", International Conference on Methods and Models in Science and Technology, **(2010)**, pp. 198-200.
[18] S. Prabha, A. Kannan and P. Anandhakumar, "An Optimization Query Processor With An Efficient Caching Mechanism For Distributed Databases", International Arab Journal, vol. 3 no. 3, **(2006)**, pp. 231-236.
[19] S. Chaudhuri, G. Das and V. Narasayya, "Optimized Stratified Sampling for Approximate Query Processing", ACM Transactions on Database Systems, vol. 32, no. 2, Article 9, **(2007)**, pp. 1-49.

[20] P. R. Suri and S. Rani, "Mechanisms For Parallel Query Execution", Journal of Theoretical and Applied Information Technology, **(2008)**, pp. 547-553.
[21] A. Deshpande, Z. Ives and V. Raman, "Adaptive Query Processing", Foundations and Trends in Databases, vol. 1, **(2007)**, pp. 1-140.
[22] S. Mahajan and V. P. Jadhav, "A Survey of Issues of Query Optimization in Parallel Databases", International Journal of Computer Applications, vol. 11, **(2010)**, pp. 32-37.
[23] C. -H. Lin and J. -C. Ke, "Optimization Analysis For An Infinite Capacity Queueing System With Multiple Queue-Dependent Servers: Genetic Algorithm", International Journal Of Computer Mathematics, vol. 88, no. 7, **(2011)**, pp. 1430–1442.
[24] C. W. Sung and Y. Y. Shiu, "Analysis Of (1+1) Evolutionary Algorithm And Randomized Local Search With Memory", Evolutionary Computation, vol. 19, no. 2, **(2011)**, pp. 287–323.

Author

Antonius Bima Murti Wijaya, M.T. is a lecturer of Department of Informatics Engineering, Sebelas Maret University Surakarta, Indonesia. He received his Master of Engineer from the Atma Jaya University Yogyakarta, Indonesia in 2012. His research interests in Enterprise Information Systems and the Government Information System.