Three Tasks Allocation for Multitasks Processing (TTAMP)

Amnat sawatnatee[1], Somchai Prakancharoen[2]
Faculty of Science, Chandrakasem Rajabhat University
Bangkok Thailand
Amnats.cru@chandra.ac.th[1],somchai.p@chandra.ac.th[2]

บทคัดย่อ - วัตถุประสงค์ของงานวิจัยนี้คือ การกำหนดลำดับการมอบหมายงานเข้าทำการประมวลในหน่วยประมวลผลกลางของงานสามงาน โดยการพิจารณาจากประวัติการใช้งานหน่วยประมวลผลกลางของแต่ละงาน ผู้ดูแลจะทำการกำหนดเงื่อนไขการมอบในการหมายงาน การวิเคราะห์หาเวลาเริ่มต้นการทำงานของแต่ละงานได้รับการคำนวณโดยวิธีการวิเคราะห์ค่าความเหมาะสม แบบมีเงื่อนไข หลังจากนั้นได้ทำการทดลองแนวทางดังกล่าวกับการใช้งานจริงรวมห้าวัน โดยเก็บข้อมูลวันละสามสิบหกข้อมูล จากการตรวจสอบเวลาประมวลผลที่ต้องรอคอยผลลัพธ์พบว่า วิธีการที่นำเสนอให้ค่าเวลาในการรอคอยลดลงสิบสองจุดหกศูนย์เปอร์เซ็นต์ เมื่อเทียบกับการมอบหมายงานวิธีการเดิม ที่ให้บริการแก่งานที่ร้องขอตามลำดับก่อนหลัง

คำสำคัญ *การมอบหมายงาน การประมวลผลหลายงาน*

***Abstract* - The objective of this paper is to design a scheduler of task allocation time of simultaneous running three majority tasks. Prior CPU usage of each task was constructed from historical CPU usage data. Objective of task allocation scheduler is to minimize three tasks CPU usage while they are both processing. Constraint of starting time and finish time of all tasks were pre assigned by system administrator. Lagrange function was used to find out the minimum value of total CPU usage. Feasible result variable, starting time of each task, were used to be as a posterior starting time of each task. For performance testing, this scheduler was test on 5 working day on ordinary transaction processing. There were 36 collected data on a working day. The result of average turnaround time of three major tasks allocation under designed TTAMP scheduler are decrease about 12.60% from ordinary FIFO scheduler. TTAMP gave a better solution about when to start task processing for a given task than FIFO multi task scheduler.**

Index Terms – Three task allocation scheduler, Multitasking.

I. INTRODUCTION

This paper presents a practical tasks allocation for single computer processing unit. There are many operations, application programs, simultaneous running on department computing server. There are many times that these applications consume computer processing unit performance above the critical limitation CPU usage. In order to keeping all operation in order, system administrator has to solve this problem. This paper suggests a solution to overcome this problem by rearrange running time of each program. Prior CPU usage pattern of each application were studied. System administrator considers the constraint of early start time of each application program. Minimization optimization of CPU usage, under defined constraints, was calculated by using single objective optimization method based on Lagrang's transformation function. All applications or tasks are then be assigned suitable starting time to begin their working. Three applications are start running in different time in order to minimization use of CPU performance. Therefore, the situation of CPU over load using is then be decreased. More ever, all application program could gain a good performance. In this research, the number of application program, that were running in parallel, was limited in three application programs

II. RELATED THEORY AND RESEARCH

A. Multitasking [1]

Multitasking refers to having multiple processes or tasks running at the same time. There is only one task running in a central processor unit while other are waiting until running process or task is finished or time quantum is met. Running process and data are then transferred to be kept in memory. Unloaded process's state is changed to waiting state in queues, in case of just not finish processing. After that, another waiting tasks data and process is allocated into memory. This situation takes amount of time on data and process in-out portage. Delay time is a major cause of performance decline.

B. Constrained optimization [2]

Mathematic optimization is a mathematic method that is used to calculate of feasible solution of some objective function especial in maximum or minimum value. Sometime there are constraints about some variable.

There are many mathematic techniques that are used to solve this problem such as linear programming that suitable on linear objective and constraint function. On the other hand, linear or nonlinear objective function is common solved by

Lagrangian function technique. Let $f(x)$ is an objective function and $g_i(x) = b, i = 1, 2, ..n$.

$x^* = (x_1^*, x_2^*, ...x_n^*)$ is vector of variables that maximizes or minimizes objective function if there exists a vector $\lambda^* = (\lambda_1^*, \lambda_2^*, ...\lambda_m^*)$ such that $\nabla L(x^*, \lambda^*) = 0$. While, L is Lagrangian function and λ is multiplier.

C. *Related research*

Pınar Muyan[3] has implement a CPU scheduler algorithm of queue management that assigned less spectrum time than high priority task. CPU is designed to handle more small kernels which could load a small task and running concurrency. These strategies increase performance of CPU and significantly decrease processing time of multitask processing.

Samih M. [4] has design a scheduler of multithread based on sibling thread in order to reduce undesirable event. This kind of thread will be adjusted. Performance evaluation results shown that using designed scheduler is effectively reduce turnaround time of program processing.

III. TASK ALLOCATION DESIGN

A. *CPU usage of main organization programs (Tasks)*

This research was experimented of task allocation on PC desktop Pentium CPU and Window 7 platform.

Five working days CPU usage data of each task were gathered as shown in Table I. For each working day, data were captured every ten minutes so that there were 36 time periods per a working day.

TABLE I
Partial average CPU percent usage data observations of task1, task2 and task3

Time period	X_1	X_2	X_3
1	0	5	0
2	10	12	0
3	10	18	0
4	10	22	0
5	20	25	0
6	20	28	1
7	20	34	1
8	20	36	2
9	30	36	2
10	30	35	3

Each series of each task's data was transformed to a best fitting curve. Vertical line represents CPU percentage usage while horizontal line represents time period.

Best fitted linear polynomial equation degree 2 of x_1, x_2 and x_3 are shown in figure 1 and 2.

R^2 value of y_1 and y_2 functions were 0.949 and 0.792.

Best fitted linear polynomial equation degree 3 of x_3 was shown in figure 3.

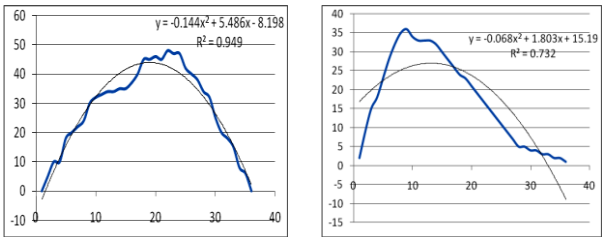

FIG. 1 OBSERVED DATA AND CURVE FITTING OF X_1 (Y_1) AND X_2 (Y_2)

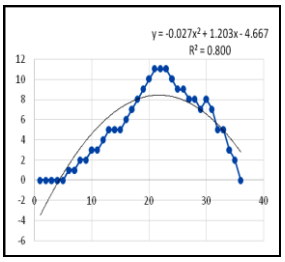

FIG. 2 OBSERVED DATA AND CURVE FITTING OF X_3(Y_3)

R^2 value of y_3 functions were 0.80.

Task 1, 2 and 3 CPU usage pattern functions as shown in equations (1), (2) and (3) respectively.

$$y_1 = -0.14x_1^2 + 5.49x_1 - 8.20 \quad (1)$$

$$y_2 = -0.07x_2^2 + 1.80x_2 + 15.19 \quad (2)$$

$$y_3 = -0.027x_3^2 + 1.20x_3^1 - 4.67 \quad (3)$$

B. *Minimization objective function*

Total CPU percentage usage is "100" as maximum value. Therefore, maximum CPU percentage usage of three tasks that running simultaneously should not greater than "100".

These should be presented as equation (3) and (4).

$$TotalCPUusage = y_1 + y_2 + y_3 \quad (3)$$

Task allocation, scheduler, must manage total CPU usage of all tasks that running together so that total CPU usage is minimum value and not exceed 100.

$$Min_{x^*} TotalCPUusage \leq 100 \quad (4)$$

C. *Constraints*

Assume that task1, task2 and task3 are not necessarily start processing in the same time. In many departments there may have some application that process in the morning of each day while some application has to be processed in afternoon of all working day.

In this experiment, task1 was assumed to start early and task2 was start later while task3 should start latest.

Task2 was start later than task1 about 12 units of times, as shown in equation (5).

$$x_1 + 3 \le x_2 \tag{5}$$

Task3 should start after task2 starting about 5 unit of times, constraint as shown in equation (6).

$$x_2 + 2 \le x_3 \tag{6}$$

Nevertheless, total time of both tasks should not greater than 36 unit of time as shown in equation (7).

$$x_1 + x_2 + x_3 \le 36 \tag{7}$$

D. Feasibilities of objective function

Therefore, Function of minimize total CPU percentage usage could presented in short in equation (8).

$$Min_{x^*}(100 - ((-0.14x_1^{\ 2} + 5.49x_1 - 8.20) + (-0.07x_2^{\ 2} + 1.80x_2 + 15.19)) + (-0.027x_3^{\ 2} + 1.20x_3^{\ 1} - 4.67) \tag{8}$$

Subject to

$$x_1 + 3 \le x_2 \tag{9}$$

$$x_2 + 2 \le x_3 \tag{10}$$

$$x_1 + x_2 + x_3 \le 36 \tag{11}$$

$$x_1, x_2, x_3 \ge 0 \tag{12}$$

Lagrangian function as shown in equation (10).

$$L = (100 - ((-0.14x_1^{\ 2} + 5.49x_1 - 8.20) + (-0.07x_2^{\ 2} + 1.80x_2 + 15.19) + (-0.027x_3^{\ 2} + 1.20x_3^{\ 1} - 4.67)) + \lambda_1(36 - x_1 - x_2 - x_3) + \lambda_2(x_2 - x_1 - 6) + \lambda_3(x_3 - x_2 - 3) \tag{13}$$

Perform $\nabla L(x^*, \lambda^*) = 0$. Thus,

$$\frac{\partial L}{\partial x_1} = 0 - (-0.28x_1 + 5.49) - \lambda_1 - \lambda_2 = 0 \tag{14}$$

$$\frac{\partial L}{\partial x_2} = 0 - (-0.14x_2 + 1.80) - \lambda_1 + \lambda_2 - \lambda_3 = 0 \tag{15}$$

$$\frac{\partial L}{\partial x_3} = 0 - (-0.054x_2 + 1.20) - \lambda_1 + \lambda_3 = 0 \tag{16}$$

$$\frac{\partial L}{\partial \lambda_1} = \lambda_1(36 - x_1 - x_2 - x_3) = 0 \tag{17}$$

$$\frac{\partial L}{\partial \lambda_2} = \lambda_2(x_2 - x_1 - 6) = 0 \tag{18}$$

$$\frac{\partial L}{\partial \lambda_3} = \lambda_3(x_3 - x_2 - 3) = 0 \tag{19}$$

First criteria: $\lambda_1 = 0 \rightarrow \lambda_2 = -2.98$

Second criteria: $\lambda_2 = 0 \rightarrow \lambda_1 = -0.226$

Third criteria: $\lambda_3 = 0 \rightarrow x_3 = -21.08$

Third criteria: $\lambda_1, \lambda_2 = 0 \rightarrow x_1 = 19.61,\ x_2 = 22.00,$ $x_3 = 24.00$ and $\lambda_3 = 1.28\ y = 75.108$

Fourth criteria: $\lambda_1, \lambda_3 = 0 \rightarrow \lambda_2 = -4.73$

Fifth criteria: $\lambda_2, \lambda_3 = 0 \rightarrow \lambda_1 = -0.66$

Sixth criteria: $\lambda_1, \lambda_2, \lambda_3 = 0 \rightarrow x_1 = 19.61,\ x_2 = 12.86,$ $x_3 = 24.00$ and $y = 80.961$

In summary third and sexth criteria is gain feasible solution but third criteria is a good one since y is less value than y in criteria sixth.

E. Availability test

The time value x_1, x_2 and x_3 are then assigned as a starting time of task1, task2 and task3 respectively. The third constrain (10) was valid while (9) and (11) were not valid.

This pattern was experimental used in order to check if it was work properly on tasks scheduling. Five day, 36 units of time per day or 6 working hours and 10 minutes data were gathered, CPU percentage usage were collected in two conditions. First was an average turnaround time of two tasks running simultaneously under designed scheduler while the second was running on ordinary FIFO scheduler.

Result of measurement, the average turnaround time of three major tasks allocation under designed scheduler is less than ordinary FIFO scheduler about 12.60%.

Three tasks were evaluated by its users about satisfaction on waiting time on transaction processing. Each task was evaluated, by their users, every ten minutes per one hour. The average waiting time for 5 working days were presented by 36 observations as shown in table II. Waiting time was reported by user on waiting time when using FIFO and TTAMP.

TABLE II
Partial average waiting time on transaction processing of task1, task2 and task3

	Task1		Task2		Task3	
#	FIFO	TTAMP	FIFO	TTAMP	FIFO	TTAMP
1	2	1.5	3	2.4	5	5.1
2	2	1.4	2	2	4	3.8
3	3	2.1	3	2.2	4	3.6
4	3	2.3	3	2.1	6	5.5
5	2.5	2.1	4	3.2	3	2.9
6	3	2.2	5	3.8	3	2.5
7	3	2.4	2	1.1	2	1.8
8	3.5	3.1	4	3.5	5	3.8

Paired t-test statistics and some statistics of each task were summarized as shown in table IV.

TABLE IV
Partial average waiting time on transaction processing of task1, task2 and task3

	Task1		Task2		Task3	
	FIFO	TTAMP	FIFO	TTAMP	FIFO	TTAMP
Average turn around time	3.44	2.75	3.81	3.39	4.22	3.94
t- value, p-value	13.69	0	7.166	0	4.27	0
μFiIFO>μTTAMP	Yes		Yes		Yes	
(TTAMP/FIFO)*100%,% Reduction	79.94%	(-20.06%)	88.98%	(-11.10%)	93.36%	(-6.63%)
Average reduction percentage of all task	12.60%					

IV. RESEARCH SUMMARY

The TTAMP scheduler can applied to manage simultaneous running tasks not only three tasks but un limited number of tasks. More ever, TTAMP scheduler is easy to implement and user do not need to re-configure or modify any operating system program.

V. SUGGESTION

Limitation of this research is that it used prior CPU percentage usage. There may be some event that cause the prior CPU percentage usage could not fitted to present situation. Likelihood of occurring observations of studying day is used to adjust prior knowledge to a posterior CPU percentage usage. The posterior knowledge is suddenly substituted the prior knowledge.

Another problem, studied data observation of two tasks prior CPU percentage usage are considered as a concave function. Many tasks may be nonmonotonic characteristic function. Therefore, TTAMP scheduler must try to optimization calculation under condition that both tasks are all monotonic function.

REFERENCES

[1] Z-World Corporate, Multitasking and Multiprocessing, California 95616, USA, 2018.
[2] D Nagesh Kumar, Optimization Methods, NPTEL, India, 2018.
[3] Pınar Muyan, Methods for multitasking among real-time embedded compute tasks running on the GPU, Wiley online library, 05 June 2017 available on:https://doi.org/10.1002/cpe.4118.
[4] Samih M. Mostafa,Effect of Thread Weight Readjustment Scheduler on Scheduling Criteria, Information Engineering Express International Institute of Applied Informatics 2015, Vol. 1, No. 2, 1 – 10.