Tasks Processing Allocation Based on Memory Usage Performance (TPA-MUP)

Amnat Sawatnatee[1]
and Somchai Prakancharoen[2]
Faculty of Science,
Chandrakasem Rajabhat University, Thailand
[1]amnats.cru@gmail.com
[2]somchai.p@chandra.ac.th

Abstract - **The objective of this paper is to design a scheduler of task allocation time of simultaneous running two independence majority tasks. Objective of task allocation scheduler is to minimize two tasks memory usage while they are both processing. Task's data that was kept in memory must be transfer or move out if this task was terminated out of CPU. New allocated task's data must be moved to be kept in memory. Data transfer in / out between memory and disk drive consume too much time. These problems should be solved by arrange both tasks to start in difference time. Each task has its own required physical memory usage. This paper present task allocation for processing in CPU based on their physical memory usage. Prior percentage of physical memory usage equation of each task was constructed from historical memory usage data. Constraint of starting time and finish time of both tasks were assigned. Lagrange function was used to find out the minimum value of memory usage. Feasible result variable, starting time of each task, were used to be a starting time of each task. This scheduler was trained on 5 working day 12 hours working. Percentage physical memory usage was captured every twenty minutes thus there were 36 captured samples per day. TPA-MUP was test in time performance by thirty users, six users per day. The result of average time performance of two major tasks allocation under designed scheduler are decrease about 15% from ordinary FIFO scheduler. Users are appreciated in time performance of two major tasks running under TPA-MUP more than ordinary FIFO at statistical significance α 0.05.**

Keywords - **Task Allocation Protocol, Percentage Memory Usage, Minimization**

I. INTRODUCTION

This paper presents a practical tasks allocation for single computer processing unit (CPU). There are many operations, application programs or task, simultaneous running on department computing server. There are many times that these applications consume percentage of physical memory above critical limitation memory usage. System administrator has to solve this problem. This paper suggests a solution to overcome this problem by rearrange starting time of each program. Prior memory usage pattern of each application was trained or studied. System administrator considers the constraint of early start time of each application program. Minimization optimization percentage of physical memory usage, under defined constraints, was calculated by using Lagrang's transformation function. According to designed protocol, both tasks are then be assigned suitable starting time. Two applications are start running in different time in order to minimization use of physical memory performance (transfer). Therefore, the situation of memory over transferring time is then decreased thus both application program could gain a good performance in CPU processing. In this research, the number of application program, that are simultaneously running, is limited only two application tasks.

A. Multitasking [1]

Multitasking refers to situation that there are many 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 finished processing. After that, another waiting tasks data and process is allocated into memory. This situation took amount of time on data and process in-out portage.

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 a 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. These kinds of thread ought to be adjusted. Performance evaluation results shown that using designed scheduler is effectively reduce turnaround time of program processing.

Inimr [5], memory performance is a major factor of CPU processing throughput time since transfer data between virtual memory, hard disk, and RAM, while task was load or purge in/from CPU, took a large amount of waiting time. Latency time of virtual memory is count on MS, millisecond, while latency time of RAM is counted on NS (Nanosecond). Therefore, increasing memory size or memory latency speed should directly increase CPU processing throughput time.

II. TASK ALLOCATION DESIGN

A. Percentage of Physical Memory Usage of Main Organization Programs (Tasks)

This research was experimented of task allocation on PC desktop Pentium Window 7. Five working days percentage physical memory usage data of each task was gathered as shown in Table I. There are twelve working hours a day. Each hour is sampling for three samples on minute 0^{th}, 20^{th}, and 40^{th}. Therefore, there are 36 samples of percentage physical memory usage in twelve hour working day.

TABLE I
PARTIAL MEMORY USAGE OF TWO MAJOR TASK1 (Y1) AND TASK2 (Y2)

Time Period (x1 & x2)	y1	y2
1	0	2
2	5	9
3	10	15
4	10	18
5	18	23
6	20	28
7	22	32
8	24	36
9	30	37
10	32	37

Each series of each task's data was transformed to a best fitting curve. Vertical line represents percentage of physical memory usage. Horizontal line represents time period.

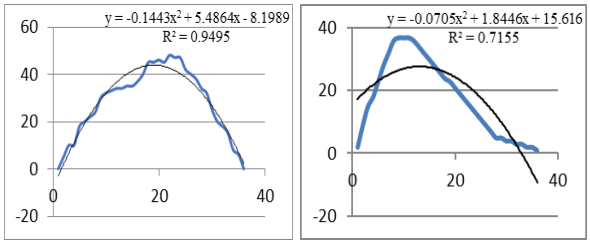

Fig. 1 Observed Data and Curve Fitting of x1 and x2, y1 and y2 is Percentage of Physical Memory Usage of task1 and task2

R^2 value of y1 and y2 function are 0.949 and 0.715. Task# 1, 2 percentage physical of memory usage pattern functions as shown in equation (1) and (2) respectively.

$$y_1 = -0.144x_1^2 + 5.486x_1 - 8.199 \quad (1)$$

$$y_2 = -0.07x_2^2 + 1.844x_2 + 15.616 \quad (2)$$

B. Minimization Objective Function

Total percentage of physical memory usage is "100". Therefore, maximum percentage of physical memory usage of two tasks that running simultaneously should not greater than "100". These should be presented as equation (3) and (4).

$$TotalMemoryUsage = y_1 + y_2 \quad (3)$$

$$Min_{x^*} TotalMemoryusage \quad (4)$$

C. Constraints

Define abbreviations and acronyms the first time they are used in the text, even if they have been defined in the abstract. Abbreviations such as IEEE, SI, MKS, CGS, ac, dc, and rms do not have to be defined. Do not use abbreviations in the title unless they are unavoidable.

In this experiment, task1 was assumed to start early and task2 was start later. Task2 was start later than task1 about 12 unit of times as shown in equation (5). This should consider in general that one task could start twelve times unit after other task is early start.

$$x_1 + 12 \le x_2 \quad (5)$$

Nevertheless, total time of both tasks should not greater than 36 unit of time, in one working day, as shown in equation (6).

$$x_1 + x_2 \le 36 \quad (6)$$

D. Feasibilities of Objective Function

Therefore, Function of minimize total percentage of physical memory percentage usage could presented in short in equation (7).

$$Min_{x^*}(100 - ((-0.144x_1^2 + 5.486x_1 - 8.199) + (-0.07x_2^2 + 1.844x_2 + 15.616)) \quad (7)$$

Subject to four constraints.

$$x_1 + 12 \le x_2 \quad (8)$$

$$x_1 + x_2 \le 36 \quad (9)$$

$$x_1, x_2 \ge 0 \text{ and } \lambda_1, \lambda_2 \ge o \quad (10)$$

Lagrangian function of optimization with inequality constraints is shown in equation (11).

$$L = (100 - ((-0.144x_1^2 + 5.486x_1 - 8.199) + (-0.07x_2^2 + 1.844x_2 + 15.616)) + \lambda_1(36 - x_1 - x_2) + \lambda_2(x_2 - x_1 - 12) \quad (11)$$

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

$$\frac{\partial L}{\partial x_1} = -0.28x_1 + 5.5 - \lambda_1 = 0 \quad (12)$$

$$\frac{\partial L}{\partial x_2} = -0.14x_2 + 1.84 - \lambda_2 = 0 \quad (13)$$

$$\frac{\partial L}{\partial \lambda_1} = \lambda_1(-12 - x_1 + x_2) = 0 \quad (14)$$

$$\frac{\partial L}{\partial \lambda_2} = \lambda_2(36 - x_2 - x_1) = 0 \quad (15)$$

First criteria: $\lambda_1 = 0 \rightarrow x_1 = 19.64\ x_2 = 16.35$ and $\lambda_2 = -0.45 \rightarrow$ not feasible.

Second criteria: $\lambda_2 = 0 \rightarrow x_1 = 1.14\ x_2 = 13.14$ $y = 70.63$ and $\lambda_1 = 5.18 \rightarrow$ feasible.

Third criteria: $\lambda_1, \lambda_2 = 0 \rightarrow x_1 = 19.64\ x_2 = 13.14$ and $y = 26.49 \rightarrow$ feasible.

In summary second and third criteria are gain feasible solutions but second criteria is a good solution since y is less value than y in criteria third. Second criteria met constraint 8 and 9.

E. Turnaround Time Test

The result time value was then assigned as a starting time of task1 and task2 respectively. This protocol was compiled in order to check if it is work properly on tasks scheduling. Five day, 12 samples per day, turnaround time of task processing were collected in two conditions, FIFO and TPA-MUP. These samples were calculated for average value of turnaround time of task processing. In FIFO, the average turnaround time of task processing is about 60 seconds per one transaction data processing. TPA-MUP, the average turnaround time is about 51 seconds per one transaction data processing.

Therefore, the average turnaround time of two major tasks allocation under designed scheduler, TPA-MUP, is less than ordinary FIFO scheduler about 15%.

F. Customer Preference Test

Each task was test of user's preference of time performance on task processing under two tasks allocation protocol. Thirty users, six persons per day and one data collection per one hour, of each task were asked about their preference of task time performance on ordinary FIFO task allocation without memory usage consideration versus TPA-MUP. According to Likert score 5 scales, preference score 5 mean very appreciate in time performance of that task.

TABLE II
PARTIAL PREFERENCE ON TIME PERFORMANCE OF TASK1 (Y1) AND TASK2 (Y2)

Task1	Test #	FIFO	TPA-MUP	Task2	Test #	FIFO	TPA-MUP
	1	3	4		1	1	1
	2	1	4		2	1	2
	3	2	3		3	2	2
	4	1	3		4	3	4
	5	3	5		5	3	3

Paired t-test was used to test hypothesis.

Task1: $Ho: \mu_{task1-fifo} = \mu t_{task1-tpa-mup}$ and
$H1: \mu_{task1-fifo} \neq \mu_{task1-t\ pa-mup}$

Task2: $Ho: \mu_{task2-fifo} = \mu t_{task2-tpa-mup}$ and
$H1: \mu_{task2-fifo} \neq \mu_{task2-t\ pa-mup}$

Descriptive statistics and p-value of paired t-test of user's preference on time performance of each task as shown in table III. The result show that time performance of TPA-MUP is better than ordinary FIFO task allocation. User preference of time performance of task#2 (3.53) is less than task#1 (4.01) since task#2 is assigned to start running after task#1 about 12 minutes, only one task running-no any waiting, after end of 13.14 minutes task#2 is start running cause task#1 has to wait for CPU available. Another reason that treats user give a less preference in task#2 performance is that curve fitting of y2 in equation (2) is not goodness of fit, R^2 value of y2 is 0.715. Task#2's memory usage pattern is too undulation.

TABLE III
DESCRIPTIVE STATISTICS AND P-VALUE OF HYPOTHESIS TEST

Task-1	FIFO	TPA-MUP
Mean	2.53	4.01
SD	0.97	0.91
p-value		0.00
Task-2	**FIFO**	**TPA-MUP**
Mean	2.63	3.53
SD	0.85	0.93
p-value		0.00

III. RESEARCH SUMMARY

The TPA-MUP scheduler can applied to manage simultaneous running tasks not only two tasks but unlimited number of tasks. More ever, TPA-MUP scheduler is easy to implement and user do not need to re-configure or modify any operating system configuration.

IV. RESEARCH SUGGESTION

Limitation of this research is that it used prior physical memory percentage usage. There may be some event that causes the prior memory percentage usage should not fitted to present situation, such as urgent task. Likelihood of occurring observations of studying day is used to adjust prior knowledge to a posterior memory percentage usage. The posterior knowledge is suddenly substitute prior knowledge. Another problem, studied data observation of two tasks prior memory percentage usage is considered as second order Polynomial function. Many tasks may have difference order characteristic function. Therefore, TPA-MUP scheduler should further try to optimization calculation under condition that both tasks are all nonmonotonic function.

V. ACKNOWLEDGMENT

This research experimental data was collected from log data of department of information technology faculty of science Chandrakasem Rajabhat University's server during 1-30 June 2018. I'm grateful to IT department for your kindness.

REFERENCES

(Arranged in the order of citation in the same fashion as the case of Footnotes.)

[1] Z-World Corporate. (2018). "Multitasking and Multiprocessing". California 95616, USA.

[2] Kumar, D.N. (2018). "Optimization Methods". NPTEL, India.

[3] PınarMuyan. (2017). "Methods for multitasking among real-time embedded compute tasks running on the GPU". Wiley online library, <https://doi.org/10.1002/cpe.4118>. Accessed 5 June 2017.

[4] Mostafa, S.M. (2015). "Effect of Thread Weight Readjustment Scheduler on Scheduling Criteria". Information Engineering Express International Institute of Applied Informatics, Vol. 1(2), pp. 1-10.

[5] Inimr. (2018). "Computer's Performance computer-basics-tutorial". Lehigh University, USA.