Identification

Supervisor Performance Measurement
J. Gintell

Purpose

The Supervisor Performance measurement facility described herein will enable the gathering of information about supervisor performance on a system-wide basis in such a way that minimizes the effect on the performance. It is implemented with a modified FIM to detect certain fault occurrences, an alarm clock handler to handle alarm clock interrupts and a system-wide data base containing gathered data.

Information to be obtained

There are four fault driven supervisory functions in Multics (page fault, segment fault, linkage fault, and wall crossing fault) which are probable system bottlenecks. The scheme developed here will obtain a distribution for the processor time required to handle each of the above faults. The distribution will be an array of entries each containing a total time and number of faults for a given range of processing time. It will take into account the fact that during the processing of one of the above faults (page fault excepted) additional faults of the same or different kind can occur.

Two more types of data pertaining to specific segments describing supervisor behavior can be obtained by this scheme. The first is procedure segment usage for those segments whose numbers are the same for all processes. The alarm clock will be used to interrupt the processor on a regular basis and the saved pbr value used to record which segment was being used by incrementing a usage count for the segment.

The final data obtainable is a count of the number of page and segment faults taken for each segment whose number is constant for all processes.

To make this segment metering more useful the ability to select when this metering is to be done and when not to be done is included by defining process states relating to which of the faults is being currently processed.

Obtaining data relating to specific segments

Any running process can be considered to be in one of five states:

R1. Processing a page fault
R2. Processing a segment fault
R3. Processing a linkage fault
R4. Processing a wall crossing fault
R5. None of the above

At any given time a processor is being used by a running process which is in one of the 5 states hence the processor can be considered to be in one of the five states (R1, R2, R3, R4, R5).

A process switches from one of its five states to another (or back to the same) when one of the four faults occurs, when the dbr is swapped or when the control unit is restored by the FIM upon completion of one of the four fault processings. Each process will maintain in a perprocess data base its current state by updating the state word whenever the process explicitly changes state by receiving a fault of the above type or by completing processing of such a fault.

The processor state can then be determined by looking in the perprocess state word for the currently running process.

There will be a switch in the system-wide data base which states during which of the five processor states segment specific metering should be done. This will allow the obtaining of five metering on a preselected part of the supervisor.

There will be an array in the system-wide data base containing an entry for each segment number; each entry will contain the number of page and segment faults received for each segment and the number of times the alarm clock interrupt occurred during execution of each segment.

Upon reception of an alarm clock interrupt, the handler ands the current processor state with the states_during_which_to_meter switch in the system-wide data base to determine whether to record segment number usage.

When a page or segment fault occurs the handler ands the current state (before switching states) with the above switch to determine whether to record the page or segment faults occurrence in the appropriate segment specific place.

Measuring Time for processing each distinct fault

For the purposes of measuring the time spent processing each type of fault in such a way that disruptions of any one of the faults processing as well as processor switching is accounted for the following scheme will be used.

This is a really spectacular English construction

Associated with each process is a "process clock" which ticks off processor time and by each process. The value of the process clock is obtained by using values in the processor clock and in the active meter table entry which contains the elapsed time used the process and the processor clock value read when the elapsed time was last recorded.

There will be a "stack" of fault meters which will be pushed each time a fault occurs and popped each time a faults processing is completed. The frame will be actually allocated as part of the PDS stack frame used by the FIM which is allocated during entry to the FIM. A pointer to the previous stack frame will be kept.

The contents of the stack frame are:

1. ptr to previous frame
2. state of process associated with frame
3. for any frame other than the last frame the total time used thus far by the process for this fault

   for the last frame, the pseudo start time for the processing of this fault (i.e., the time which the process clock would have read when this fault processing started had there been no interruptions by new faults.

Whenever one of the four faults occurs the following action is taken

1. read the process clock
2. using the pointer to access the previous frame - compute the elapsed time for the fault
3. switch to the new state by changing the state word
4. store in the new frame the ptr to the previous frame
5. store in the new frame the current state
6. store in the new frame the process clock reading

When fault processing is completed the following action is taken:

1. read the process clock
2. using the current frame compute the total time used for this. fault processing = process clock minus pseudo start time
3. record this time in the appropriate array for this fault
4. using the pointer to the previous frame set the previous frame pointer
5. switch to the previous process state
6. compute the pseudo start time = process clock reading minus elapsed time thus far.

There will be one initial frame associated with the process state of processing none of the above faults.

System Data Base

The System Data Base will contain some switches showing the global characteristics of the metering run and some times showing when the run started and stopped.

There will be four arrays contain statistics for page, segment, wall crossing and linkage faults where each array contains 36 entries the nth entry of which contains the number of associated faults which took between 2^{a-1} and 2^n microseconds to occur and the total processor time for all such fault processings.

There will be an array which whose n^{th} entry indicates the number of times segment # n was being executed when the alarm clock was interrupted and another array, whose n^{th} entry indicates the number of segment page faults taken for the n^{th} segment.

There will be a lock associated with the table so that multiprocessor race conditions will not occur.

All of the data bases procedures must be wired down since they may be accesse d during the processing of page faults.

Process Data

For each process there will be a state switch to indicate which of the five states the process is in and a pointer to the current "fault meter".

For each fault occurrence there will be a fault meter which contains a pointer to the previous fault meter, the process state associated with the frame, and the processor time used thus far for this fault or the pseudo start time that the "process clock" was read . These "stack frames will be contained in the stack frame allocated by the FIM upon the occurrence of a fault.

time in this state	# of times state entered	P S W L	Possible
		0 0 0 0	yes
		0 0 0 1	yes
		0 0 1 0	yes
		0 0 1 1	yes
		0 1 0 0	yes
		0 1 0 1	yes
		0 1 1 0	?
		0 1 1 1	?
		1 0 0 0	yes
		1 0 0 1	yes
		1 0 1 0	yes
		1 0 1 1	yes
		1 1 0 0	yes
		1 1 0 1	yes
		1 1 1 0	?
		1 1 1 1	?

How about 16 meters, bumped whenever 1 of the four bits changes.

+ count of # of times in each state