System Data and Its Evaluation of Time Sharing System with Virtual Memory Concept

Takashi Masuda

Yasufumi Yoshizawa

Toshio Hirosawa

Nobumasa Takahashi

from MEXICO 1971 INTERNATIONAL IEEE CONFERENCE ON SYSTEM, NETWORK AND COMPUTERS

(JANUARY 19-21, 1971)

Central Research Laboratory, Hitachi Ltd.

Kokubunji, Tokyo, Japan

"System Data and Its Evaluation of Time Sharing System with Virtual Memory Concept"

Takashi Masuda

Yasufumi Yoshizawa

Toshio Hirosawa

Nobumasa Takahashi

Central Research Laboratory, Hitachi Ltd.

Kokubunji, Tokyo, Japan

Sept. 14, 1970

Introduction

The HITAC 5020 Time Sharing System has been in operation since April 1958. This system, which we will abbreviate as 5020 TSS in the following, has a segmentation and paging mechanism. At present the maximum number of active terminals is 22, which will become 32 terminals in the near future. The system has adopted almost all merits of virtual machine and virtual memory concepts, i.e. dynamic linking, on-demand paging, treatments of common procedures among processes, a file system which has a direct relation with segmentation and paging mechanism, ring protection mechanism, etc.

A major goal in implementing 5020 TSS has been the collection and quantitative analysis of operating statistics in order to provide an optional system, particularly with respect to segmentation and paging.

We have collected this data using two methods.

First, we implemented in-line software monitoring to continuously collect system statistics.

Second, we developed a Program Address Trace System (PATS) which interpretively executes the system and collects instruction-by-instruction data on magnetic tape for later evaluation.

The first part of the paper briefly describes the hardware and software of 5020 TSS (For a more complete description, see The Hitac 5020 Time Sharing System[1]). The second part describes the collection of system statistics, and points out some of the implications of the data obtained.

Hardware Configuration

(1) Central Processor Unit: HITAC 5020[2)]

HITAC 5020 is a word oriented machine, each word containing 32 bits. Its cycle time is 2 μs and the speed is 13 μs in the Gibson mix.

(2) Hardware for segmentation and paging mechanism: DAT (Dynamic Address Translator)

Segmentation and paging mechanism is implemented on the HITAC 5020 using the DAT. Each address space has 2^{16} segments and each segment consists of 2^{16} words. There are four special base registers by which segments are indicated. Page size is 256 words. Segment Descriptor Table is also paged as each segment is paged. Figure 1 shows the addressing mechanism of HITAC 5020 with DAT.

(3) Memory

The size of core memory is 65 K words of which the resident operating system occupies 20 K words. A drum memory is used for swapping, with an access time of 15 ms, and a transfer speed of 5 msec per page. 8 drums are utilized, attached to a single data channel.

Software System

(1) Scheduling and Swapping Algorithm

Each process may be in one of the four states indicated below.

(a) running: the process is in a running state.

(b) ready: the process is able to be run at any time.

(c) pending: the same as (b)

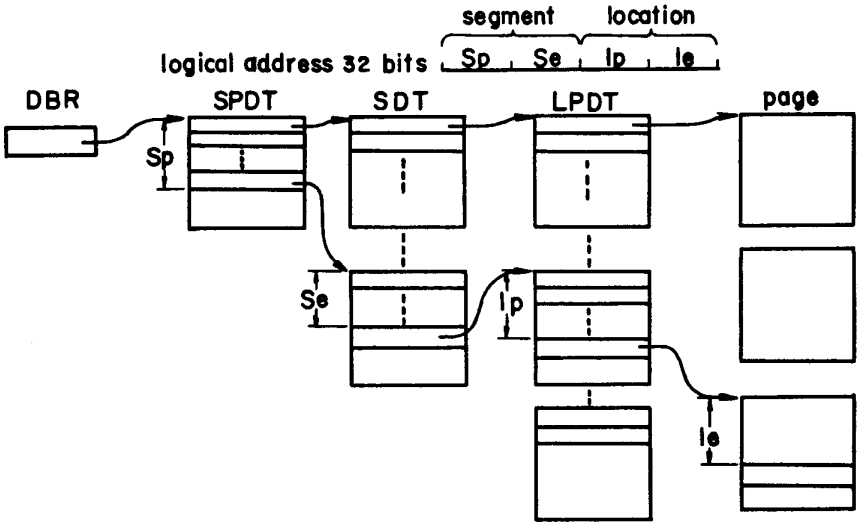
(d) blocked: the process is waiting for the end of input and output.

Corresponding to these states, there are three queues in the system, the ready queue, the pending queue and the blocked queue. The top process in the ready queue is a running process. Figure 2 indicates how the processes change states. The number of processes allowed in a ready queue has been heuristically limited at present to three. If the number of processes in a ready queue becomes less than the allowed limit, the top process in the pending queue is put at the end of the ready queue.

Pages to be used are swapped into main memory on demand. Each pending process or blocked process has a rank number corresponding to the number and type of pages of that process in the main memory. The rank number of pending processes or blocked processes changes when pages belonging to these processes are swapped out.

The pages to be swapped out are determined in the following manner.

(a) Pages to be swapped out are first selected from the blocked



DBR : Descriptor Base Register
SPDT : Segment Page Descriptor Table
SDT : Segment Descriptor Table
LPDT : Location Page Descriptor Table

Figure 1 Address space in HITAC 5020 TSS

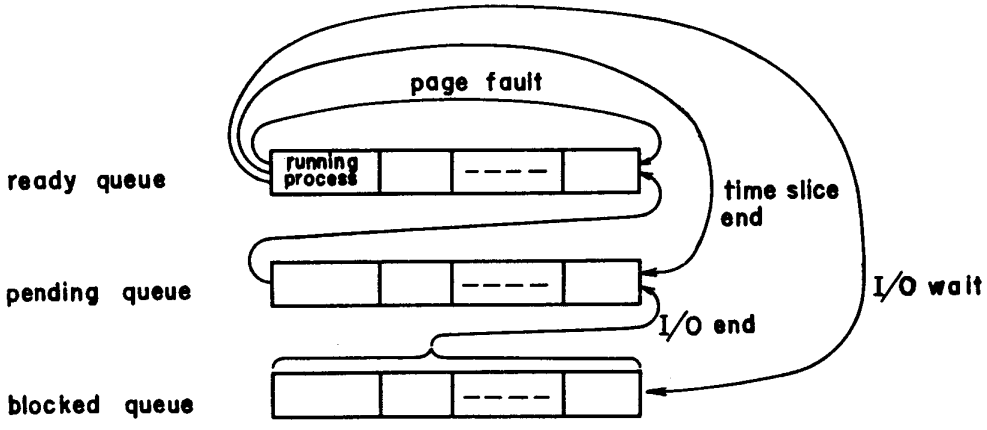


Figure 2 Transition diagram for process status

processes according to their rank number.

(b) If there are no pages to be swapped out in a blocked queue, pages are selected from the pending processes, again based on the rank numbers of those processes.

(c) If there are no pages to be swapped out in a pending queue, the last process in a ready queue is removed and placed at the top of the pending queue.

The selection of pages to be swapped out in a process is followed by the FIFO algorithm (in 5020 TSS, there is no use-bit in a descriptor table). The number of pages to be swapped out concurrently is 3 to 8 pages.

(2) Each program block consists of a pure procedure segment and a data segment (which is called an internal data segment).

(3) Object code produced by a compiler is registered in a file system as a pure procedure file and an internal data file.

(4) As each new file is requested at run time, it is assigned dynamically as a segment in virtual memory (dynamic linking).

Methods of Investigation

System data for the 5020 TSS has been collected by the two methods described below.

(1) Software monitor

The statistics collected by the software monitor are:

(a) the number of active processes

(b) user time, operating system time and time during data transfer between main and secondary memory.

(c) the frequencies of traps and interrupts and the time needed for each type of them.

(d) the frequencies of supervisor calls and the time needed for each type of them.

(e) the amount of virtual memory used

(f) the frequency of missing-page-fault per unit time etc.

(2) PATS

A Program Address Trace System (PATS) was developed for 5020 TSS, which executes interpretively any program instruction by instruction, recording the instruction code, the logical address of the instruction, the logical address of the operands and an indication of whether or not there is indirect addressing for each instruction. The output of PATS is recorded on magnetic-tape, and can be processed later by FORTRAN. Each tape can record about 5×10^5 instructions.

PATS accepts a number of parameters which control the quantity of output. The parameters effect the following actions.

(1) The address patterns of only specific segments are recorded.

(2) The address patterns of only the operating system are recorded.

(3) The address patterns of only the user's program are recorded.

(4) The address patterns of all instruction executed are recorded.

We recorded the address patterns of all trap and interrupt processing modules and all supervisor call processing modules, and analyzed this data by FORTRAN, to find the distributions of the dynamic processing steps of all modules of the operating system.

Results and Discussions

(1) Rate of user time, OS time (Operating System time), I/O rest time.

Figure 3 is a typical example of the rate changes of user time, OS time and I/O rest time during a three hours period. The active number of users ranged between 10 and 20 during this period. The job mix consisted of by BASIC, the PL/1W (a subset of PL/1) and file creation, updating and listing. The PL/1W source programs are compiled serially by establishing a separate queue in the system for them. When the system is crowded, the average rate of user time is 8 to 15%, that of OS time is 35 to 45%.

(2) Analysis of OS time.

As indicated above, the rate of OS time is about 35 to 45% of total time. Figure 4 shows a more precise analysis of OS time, based on an OS time of 100%.

We can see from Figure 4, that the missing-page-fault processing module occupies about half of the total OS time, and the trap from drum channel processing module occupies 22% of the total OS time. This module corresponds to the post-processing of page swapping. Because these two modules occupy about 73% of the total OS time, it is clear that a major problem is the reduction of the time used by these two modules in a paging system, although a significant reason may be the relatively small size of main memory.

Figure 5 shows the distribution of the dynamic steps needed for processing a missing page fault. There are three cases in our

system that cause this processing.

(a) There are enough unused pages to be allotted to.

(b) There are enough unused pages to be swapped in from secondary memory.

(c) Since there is not enough room in main memory, swapping algorithm finds some pages to be swapped out.

An average of 1240 steps are required to process a missing page fault, weighted by the frequency of each type of faults. The results of Figure 5 are very useful for deciding system parameters to improve the system.

(3) Relation between system performance and system load.

Date about the relation between system performance and system load, is important in choosing system parameters, particularly when they may be changed dynamically by the system. We used the rate of user's time as a measure of system performance and the frequency of missing page faults per second as a measure of system load. The results are shown in Figure 6. Each sample point was calculated as the average over a 10 minutes period. Figure 6 shows that the ratio of user's time changes considerably even if the frequencies of missing page fault have the same values.

(4) The rate of change of the page table, pure procedure page and data page in the main memory.

Figure 7 shows the time-variation of the page table, pure procedure page and data page in the main memory. The page table occupies about one-third of the total main memory. Because we assign one page for each page table and one page table for each segment, the average size of each segment in the main memory is two pages. The mean of the actual size of each segment is actually more than two pages since the page table can be swapped out after all program pages corresponding to the segment have been swapped out.

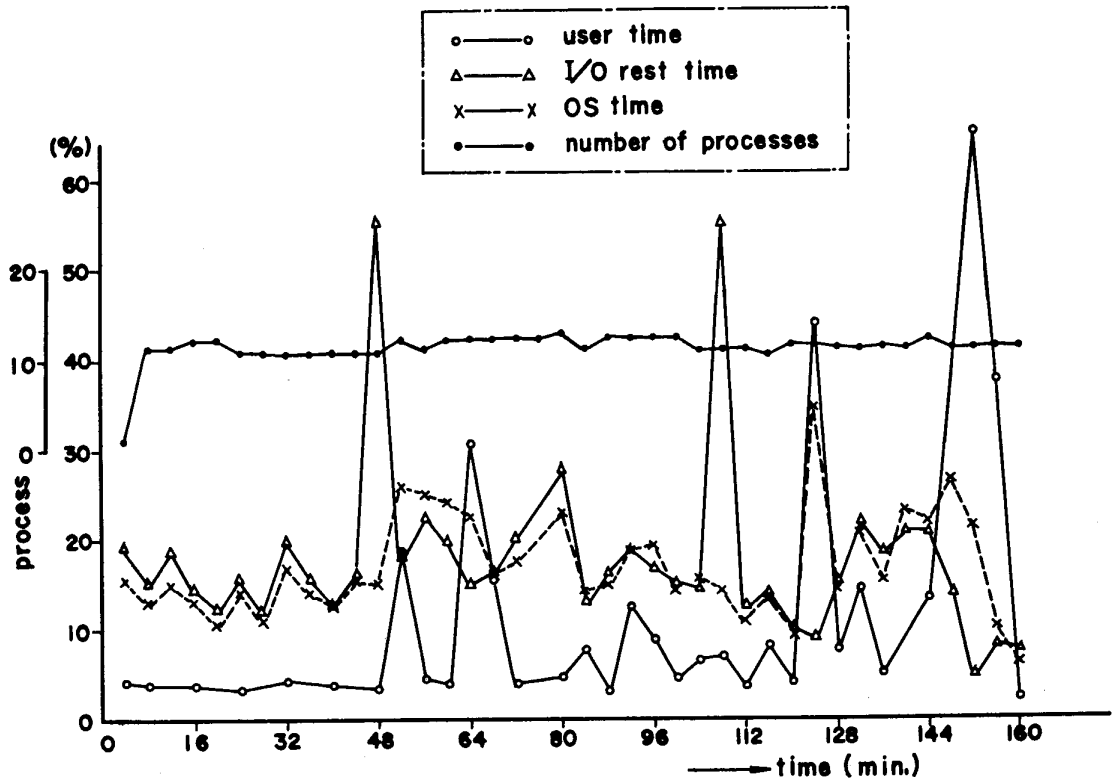


Figure 3 Performance of User time, OS time and Rest time

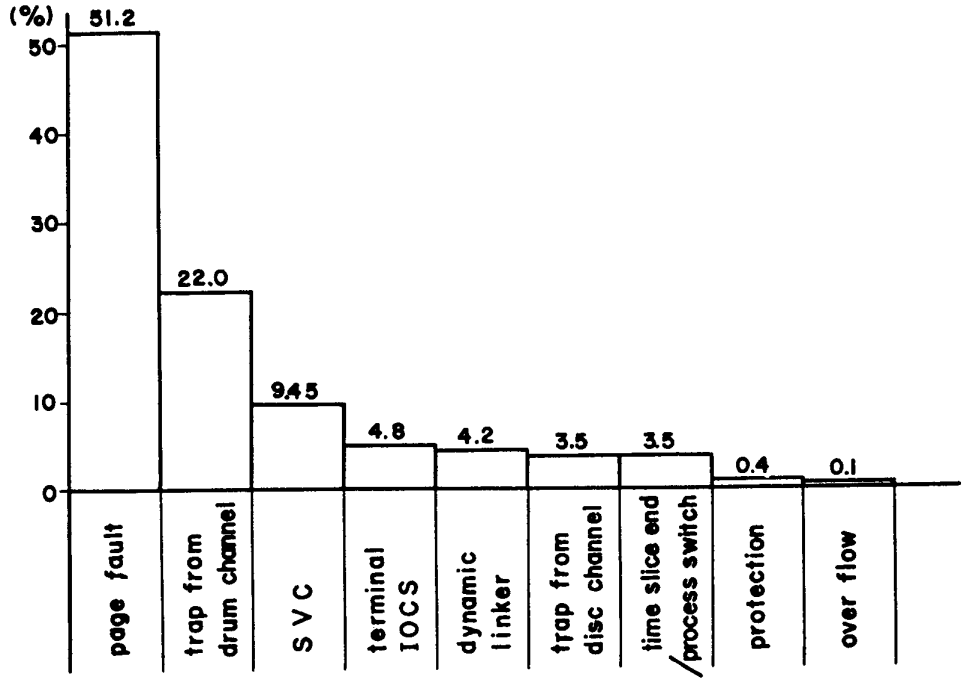


Figure 4 Factor analysis of OS time

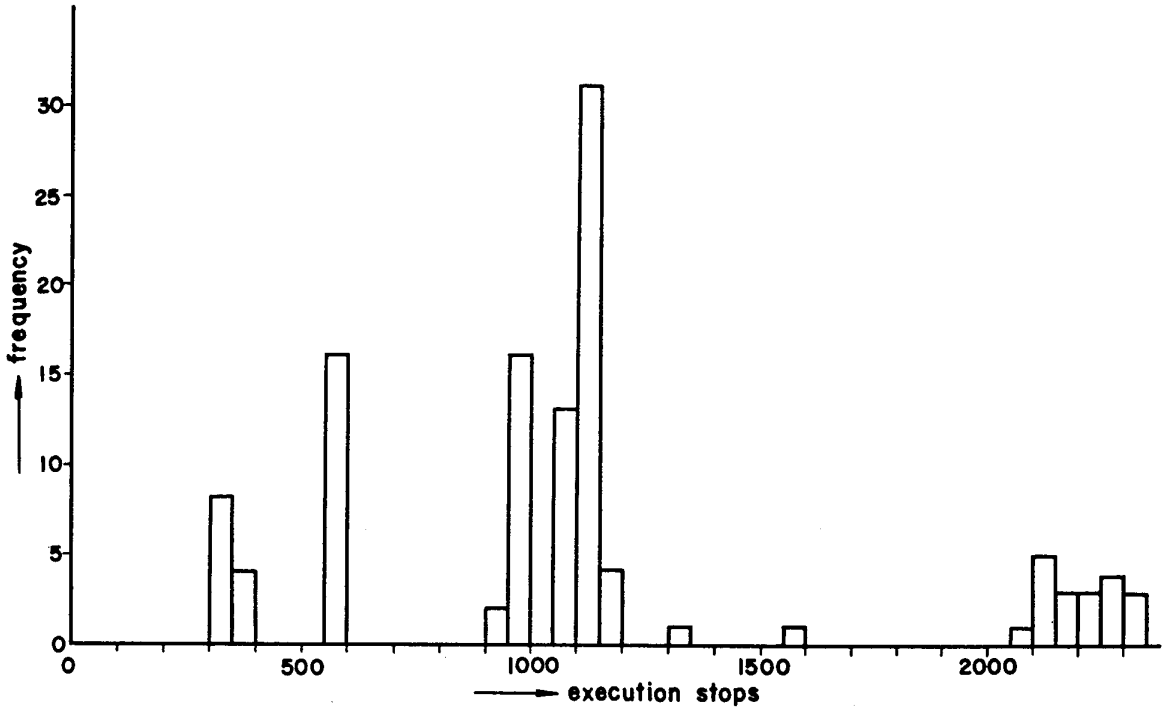


Figure 5 Execution steps required for page-not-in-core fault

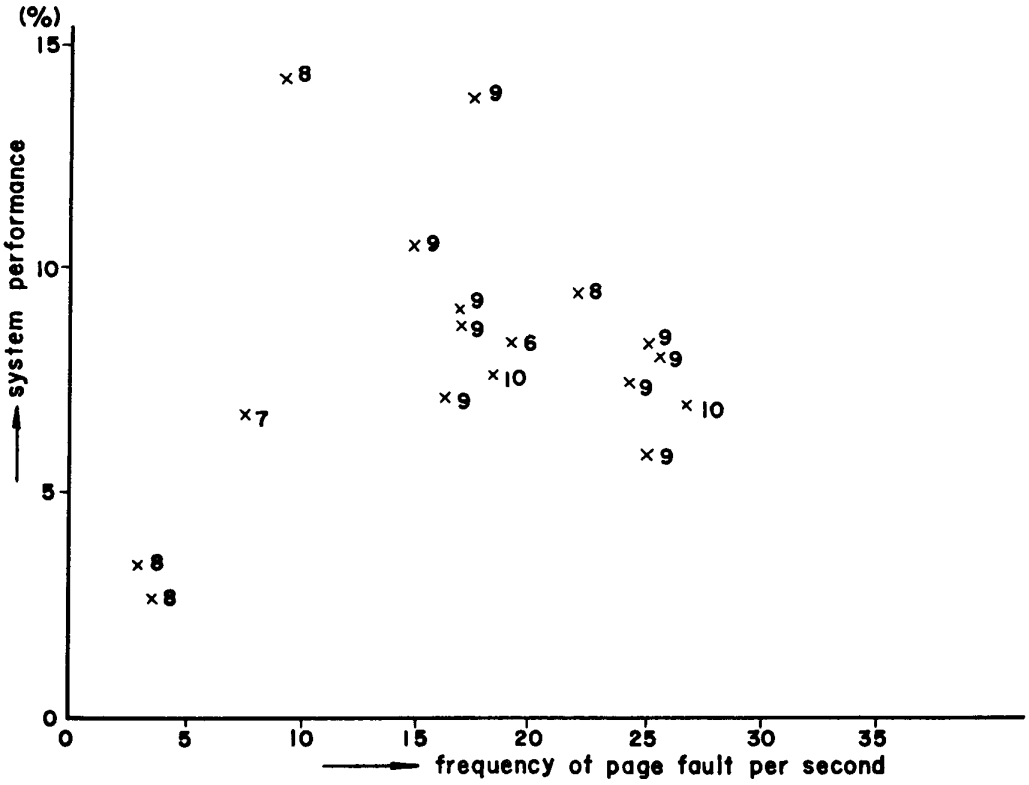


Figure 6 System performance/System load

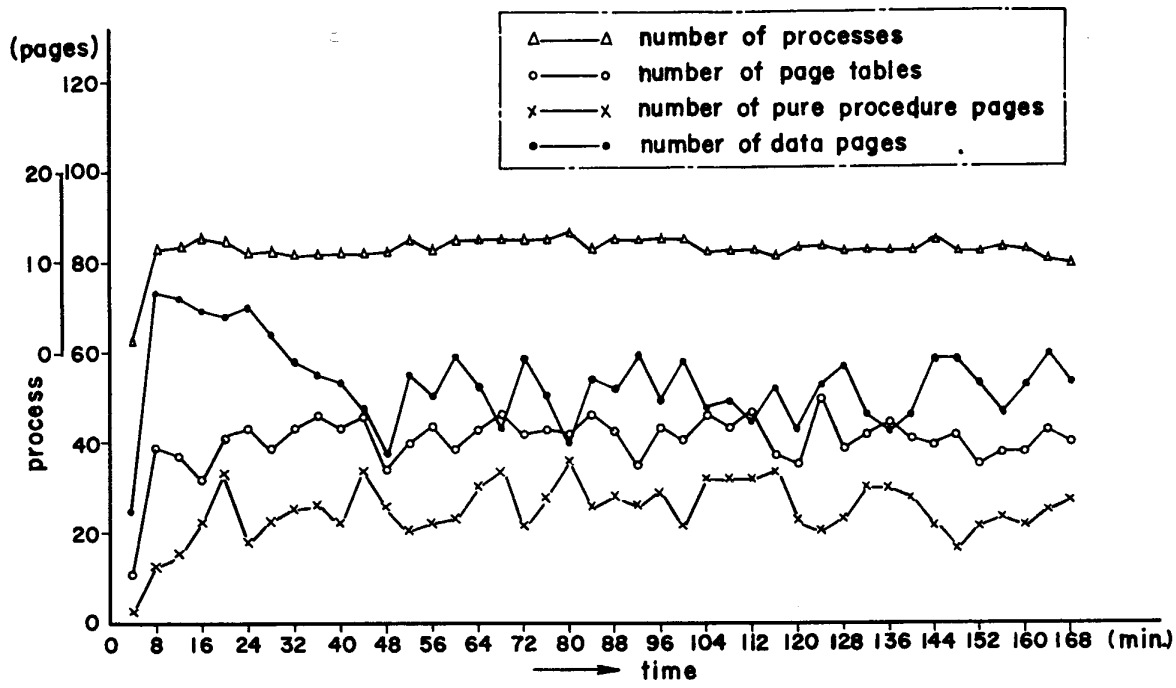


Figure 7 Classification of pages in core memory

Summary

In this paper we have reported the system data of the HITAC 5020 Time Sharing System. Although the results are by their nature machine dependent, we expect that they provide a useful analysis of time-sharing systems utilizing segmentation and paging mechanisms.

References

1) Motobayashi, Masuda, Takahashi, "The HITAC 5020 Time Sharing System" Proc. ACM. 24th, Nat. Conf., 1969 Pt. 419-430

2) Murata & Nakazawa "A very high speed serial and serial-parallel computers HITAC 5020 and 5020E." Proc. AFIPS 1964 FJCC Vol. 26, Pt. 1, 187-203