MPL-50

To: Multics Performance Log

From: A. Sekino

Subject: Results of Multics Performance Measurements
Effect of Core Size upon System Performance

Date: May 12, 1970

1. Summary of Performance Measurement Results

The measurement runs, MPM96 through MPM103, were made in the following situations. It should be noted that the very important system parameters such as the number of CPU's and the available core size were changed. However, the number of eligible processes allowed in core at a time (= $e$max) has been fixed at three. A fast page-fault handler ($\sim$4 msec/page) and a new command loop were newly installed on the 7.0 system.

Date	Run No.	System	No. of CPU s	Core Size	No. of Users
3/24	MPM96	7.0	1	256 K words	21
3/25	MPM97	6.0	1	256	28
3/27	MPM98	6.0	1	384	28
3/29	MPM99	7.0a	1	256	8$\sim$9
3/30	MPM100	7.0a	1	256	21
4/2	MPM101	7.1b	2	384	23
5/4	MPM102	7.3b	1	384	27
5/7	MPM103	7.3b	1	256	27

The measurement results of the above runs are summarized in the following table.

TABLE I

Time unit = second
Average = per interaction

Run No.	Total CPU Time	Total No. of P.F.	Total Real Time	Average CPU Time	Average No. of P.F.	Average Response Time
MPM96	31.815	3495	2322	.482	52*	6.2
MPM97	48.839	2839	2416	.739	43	9.3
MPM98	37.915	1879	2293	.575	28	5.4
MPM99	23.644	2089	2155	.358	31	2.2
MPM100	28.721	3013	2298	.436	45	5.2
MPM101**	26.859	1868	2216	.407	28	2.9
MPM102	25.749	1703	2248	.390	25	4.1
MPM103	31.340	2771	2425	.474	41	9.6

* The number of missing page faults was found to be larger than usual because of a bug in the supervisor of a newly installed 7.0 system.
** The performance of a 2 CPU, 384 K system measured in MPM101 was analyzed and reported by J.H. Saltzer in MPL-49.
*** The installation of a fast page-fault handler resulted in a shorter mean time between page faults but the number of missing page faults seen by the script did not change significantly.

2. Effect of Core Size upon System Performance

The comparison between MPM97 and MPM98 of the 6.0 system and the comparison between MPM102 and MPM103 of the 7.3b system enable us to see an effect of core size upon the system performance.

6.0 System (28 users, 1 CPU)

TABLE II

Run No.	Core Size (Kwords)	MTBPF (msec) depth 1	2	3	Average No. of P. F.	IO Capacity (%) Drum	DS270	CPU Idle Time (%) 1	2	3	Total
MPM97	256	22	21	16	43	12	30	14	0	0	14
MPM98	384	44	32	24	28	7	22	6	2	3	11

7.3b System (27 users, 1CPU)

TABLE III

MPM103	256	21	18	12	41	12	29	22	0	0	21
MPM102	384	39	30	19	25	6	19	6	8	17	31

The CPU idle time 1, 2, and 3 of the above tables respectively denote the multi-programming idle time, the non-multiprogramming idle time, and the zero idle time. Several observations may be made from the statistics of Table I, II, and III.

1. By increasing the core size from 256Kwords to 384Kwords, the mean time between page faults (MTBPF) has almost doubled (1.5～2 times). The average number of page faults per interaction seen by the Fortran script user has reduced to 61～65%.

2. The resulting decrease in page traffic is observed in the I/O rate; the decrease in drum traffic is 40～50% and that in disk traffic is about 30%.

3. The multi-programming idle time, which can be almost accurately predicted from the MTBPF's by a mathematical model, has drastically decreased. Furthermore, it is seen that the 384K system is no longer under a full load when there are 27~28 simultaneous users.

4. The effect on CPU usage may be easily predicted assuming a constant page-fault handling time (~8 msec. for the 6.0 system and ~4 msec. for the 7.3b system). The decrease in CPU usage of the Fortran script user was experimentally found to be 22% on the 6.0 system and 18% on the 7.3b system.

5. The effect on the system response time is particularly immense, as shown in Table I. However, this 50% decrease in the average response time mainly comes from the decrease in the exceptionally long response time of the "fortran" command. The 256K system under a 27~28 user load spent a 45~86 second response time with about 260 missing page faults for a simple Fortran compilation, while the 384K system under the same load spent a 6~27 second response time with 118~214 missing page faults for the same compilation. The big difference between these two response times is due to the difference in the degree of satulation of each system and in the number of induced missing page faults on each system. This observation suggests that the pre-paging and post-purging technique could drastically improve the system response time of the heavier commands if this technique will be properly installed.

3. Effect of the Other Processor

It seems that a very drastical performance improvement such as the one discussed in 2 cannot be expected by simply adding another processor to the system because of the interference for access to core memory, as pointed out also in MPL-49. This means that each user's CPU usage will increase because of the core interference and the increased paging traffic but that the system response time to each user will be slightly improved because of the increased computational capability of the dual-processor system. This tendency was actually found in the results of MPM102 and MPM101.