PROJECT MAC

December 1, 1975

Computer Systems Research Division

Request for Comments No. 103

MEASUREMENTS OF USL MULTICS

by J.H. Saltzer and D.D. Clark

On November 17, 1975, we visited the Multics installation at the University of Southwestern Louisiana (USL) at Lafayette, Louisiana. The performance of the system installed there is of particular interest because its configuration consists of 1 processor, 1 million words of primary (LSI) memory, and no "bulk store". During the day there we took a variety of measurements, which are reported here.

I. Processor and memory speed calibration

The program mip_test (see RFC-39) was used to measure the processor speed to allow comparison with benchmarks run at M.I.T. and CISL. The instructions "ada", "spr", "epp", "epp indirect", and a "Multics mix" were measured. When all operands and intructions were in the cache, the USL processor timings were within 2% of the corresponding instructions on the two M.I.T. processors (which, incidentally, currently differ in speed by about 2%). The following table summarizes the result:

instruction	speed, with cache
ada	.67 μs
spr	.80 μs
epp	.90 μs
epp, indirect	1.71 μs
mix	.92 mips

When the cache was disabled (by making the test program a shared, writeable segment) the relative speeds of the USL and M.I.T systems were quite different, apparently reflecting the difference between core memory (at M.I.T.) and LSI memory (at USL). The difference is shown in the following table:

---

instruction	USL speed, no cache	M.I.T. speed, no cache
ada	1.33 μs	1.0 μs
spr	1.61 μs	1.44 μs*
epp	1.22 μs	1.07 μs
epp, indirect	2.87 μs	2.45 μs
mix	.584 mips	.73 mips

The difference between the times of the epp and the epp, indirect, can be taken to be two memory reference times, since one reference time is required to fetch the indirect word and one reference time is lost through disabling of address preparation lookahead. This gives us apparent memory reference times of

	USL	M.I.T.
memory reference time	0.83 μsec	0.69 μsec

The 140 nanosecond difference is too great to be explained by differences in cable lengths (70 feet of cable difference would be required) so it apparently represents the difference between the magnetic core memory installed at M.I.T. and the LSI memory installed at USL. Actually, since LSI memory should differ with core memory non-uniformly in all of read time, write time, and cycle time, the cycle time difference as measured above does not tell the whole story. This lack is evident in the speed of the Multics mix, in which the M.I.T. machine runs 25% faster, even though its memory cycle time is only 19% faster.

The performance impact of the memory difference depends on the cache hit ratio. If a hit ratio of 75% is assumed, then the overall performance of the machines should be roughly

	USL	M.I.T.	$\frac{\text{M.I.T.}}{\text{USL}}$
speed of mix with 75% hit ratio	.80 mips	.86 mips	1.08

* In January, 1975, this instruction was clocked at 2.18 μsec on M.I.T. processor A. At that time it was suggested that its slow timing might be due to disabling of address preparation overlap. Since it is now faster by 0.74 μsec, or about one operand fetch time, it would appear that some field change since January repaired the problem. All the other instructions currently measure within a few percent of their January, 1975 values.

In other words, the USL machine should be taken to be about 8% less effective than the M.I.T. machine, in raw speed.

II. Disk Driver Timings

The lack of a "bulk store" paging device on the USL system means that all missing page faults that require reading of data are directed to the disk driving software, known as the "Disk DIM". On the M.I.T. system, only a small fraction of read requests are directed to the Disk DIM--most go to the "bulk store DIM". Measurements of the average times to handle a page fault on the two systems can thus be used to compare the performance of these two DIMs and to apportion the total time in paging to the two levels. If we denote the average time to handle a bulk store page fault as t_b and the average time to handle a disk page fault as t_d, we have

$$f \cdot t_d + (1-f)t_b = T_{av}$$

where f is the fraction of page faults directed to the disk and T_{av} is the observed average page fault time (the value "AVE page faults" printed by ttm). Two measurements with widely different values for f will allow reasonably accurate estimation of t_d and t_b. For USL, f = 1.0, by definition, while for M.I.T., a value of f = 0.1 is more typical. The following values were observed:

	USL#1	USL#2	M.I.T. #1	M.I.T. #2
metering time	3 h 22 m	1 h 44 m	7 h 23 m	5 h 30 m
AVE page fault time (T_{av})	7.05 ms	5.76 ms	1.99 ms	2.45 ms
$\frac{\text{disk reads}}{\text{total reads}}$	1.0	1.0	.059	.12

Using any pair of these four columns (except 1 and 2, which produce a singular result for t_b) we can estimate t_d and t_b:

	using Col 1 + Col 3	using Col 1 + Col 4	using Col 2 + Col 3	using Col 2 + Col 4	using Col 3 + Col 4
t_d	7.1 ms	7.1 ms	5.8 ms	5.8 ms	6.3 ms
t_b	1.7 ms	1.8 ms	1.8 ms	2.0 ms	1.6 ms

These various combinations (including the combination of two runs at M.I.T.) are fairly consistent, and suggest that the disk DIM produces page fault times averaging about 4.5 ms longer than the bulk store DIM. This difference in time suggests that there is a substantial difference in behavior between the two DIM's*.

---

* See part III for a discussion of this difference.

We can apply this observation to analyze the 5 h 30 m M.I.T. measurement run of column 4 above. The following measurements, rounded to 1%, were reported.

Page fault time	28%
Getwork	3%
Interrupts	6%
Segment faults	3%
MP idle	2%
Other idle	13%
Useful	45%
	100%

(1)

Knowing that the fraction of disk page faults was 0.12, and taking estimates of $t_d = 7.1$ ms and $t_b = 1.8$ ms from our earlier intersystem measurement, we can divide the page fault time into

$$\text{Disk Page fault time} = f \cdot (t_d/T_{av}) \times 28\% = 10\%$$
$$\text{Bulk Store Page fault time} = (1-f)(t_b/T_{av}) \times 28\% = 18\% \qquad (2)$$

We should, for a complete picture, attribute most of the Getwork, Interrupt, and MP idle time to disk paging, and attribute most of the segment fault time to the presence of the bulk store.* This attribution leads to the following breakdown of system utilization:

	percentage of total time	percentage of non-idle time
Bulk store paging	20%	23%
Disk paging	21%	24%
Idle	13%	
Useful	46%	53%
	100%	100%

(3)

The right-most column is the one of interest--it tells directly the cost of having a bulk store paging device at M.I.T., and the possible benefit that could result from replacing the bulk store with the same number of words of directly addressable primary memory. We would anticipate that the time currently spent in bulk store paging would be distributed proportionately among useful work and disk paging, leading to a prediction of

Disk paging	31%
Useful work	69%
	100%

(4)

---

* This attribution is based on the observation that if the bulk store were replaced with primary memory, a larger Active Segment Table could be used. The USL system, with a larger AST, exhibits less than 1% time in segment faults.

These predictions of performance of the M.I.T. system with 2 processors and 2 million words of primary memory are corroborated by observation of the USL system with 1 processor and 1 million words of primary memory: Disk paging under full load used from 30% to 36% of system capacity. Two conclusions can be drawn:

1) Replacement of the bulk store with an equal-sized primary memory at M.I.T. would increase useful work from about 53% to about 69%, a performance increase of 30%.
2) The time spent handling disk page faults is just as significant as the cost of having a bulk store. If strategies can be found to reduce that time, the performance effect could be significant.

III. Potential effect of a Disk DIM performance bug

Initial exploration of the code for the bulk store DIM and the Disk DIM suggests that their page fault times should not be terribly different. As a result, there is reason to suspect that there may be some performance bug in the Disk DIM, perhaps relating to frequent filling of Disk Driver queues. Further examination and analysis of the Disk DIM certainly seems warrented, and we may also analyze the potential effect of discovery and fixing a performance bug. Suppose that some way were found to lower t_d so that it is similar in magnitude to t_b. In that case, the page fault time attribution of equations (2) above would be revised by reducing the disk page fault time by a factor of t_b/t_d, or 0.25. We would then have a

$$\begin{aligned} \text{Disk Page Fault Time} &= 2.5\% \\ \text{Bulk Store Page Fault Time} &= 18\ \% \\ \text{Additional Idle Time} &= 7.5\% \end{aligned} \qquad (5)$$

assuming that the time released by the shorter Disk DIM path length would emerge as additional idle time for the same load.

Following a similar reattribution of Getwork, Interrupt,* and MP idle time to disk paging and segment fault time to bulk store paging, we would have

---

* We assume, conservatively, that any problem repaired in the Disk DIM would not shorten the interrupt time.

	Percentage of total time	Percentage of non-idle time
Bulk store paging	20%	25%
Disk paging	14%	18%
Idle	20%	
Useful	46%	57%
	100%	100%

(6)

By comparing (5) and (6) we can conclude that improvement in the Disk DIM could be expected to raise useful work from 53% to 57%, on 8% improvement in system performance.

Finally, replacement of the bulk store paging device with main memory would yield a breakdown of

Disk paging	24%
Useful Work	76%
	100%

(7)

Comparing (7) with (6), useful work rises from 57% to 76%, on increase of 33%. Thus, if there <u>is</u> a problem in the disk DIM, after it is fixed the payoff for replacement of the bulk store by primary memory would be even greater than before. These results are normalized and summarized below:

Performance of M.I.T. system at present	100%
Projected performance of M.I.T. system with bulk store replaced	130%
Projected performance of M.I.T. system with Disk DIM repaired	108%
Projected performance of M.I.T. system with both	143%
Performance of USL system at present	100%
Projected performance of USL system with Disk DIM repaired	110%

<u>IV</u>. <u>Effective system performance comparison</u>

Putting together the observations of part

effective performance of the two systems by calculating the processing capacity actually delivered to the customer under full load. At USL, the delivered capacity is about (0.8 mips) x .70, while at M.I.T. the delivered capacity is about (0.86 mips) x .53 x 2, since there are two processors, giving

	USL	M.I.T.	M.I.T./USL
delivered processing capacity	0.56 mips	.91 mips	1.63

In other words, the 2 CPU, 384K, 2000K M.I.T. system is about 60% more effective than the 1 CPU, 1000K USL system.

V. System response to large demands for memory

Two metering programs, named plow and oof, attempt to measure the system's ability to deliver a large amount of real memory to a program that needs it. They measure this by repeatedly touching a specified number of pages in a short time, and counting the number of page faults taken while doing so. If a small number of pages are specified, the number of page faults will be about 1 per page, but if the number of pages specified is larger than the number that one process can keep in core, a large number of page faults will result as they displace each other during the repeated references. Running these programs for various number of pages allows one to determine a rough upper limit on the number of private pages that may be in the in-core working set of one process. Typical numbers for the M.I.T. system are listed below.

number of users	max. working set, pages
24	200
48	160

The numbers vary with the number of users logged in and competing for memory at the time of the experiment.

The following numbers were observed at USL:

number of users	max. working set, pages
37	475
20	625
8	725

Since the M.I.T. system with its two processors can support a larger number of users than the USL machine, direct comparison of the two systems supporting the same number of users could be misleading. Even so, it is generally clear that the USL machine is capable of delivering a large amount of memory

to a single large user. This suggests that large LISP programs, such as MACSYMA, would run well at USL. It is not certain from these experiments, however, what the effect would be if more than one large program attempted to run at once.

VI. Miscellaneous observations

In this section we mention several miscellaneous minor observations, some quite subjective.

1) "Memory units" for typical jobs were grossly smaller on the USL machine. This observation suggests either that the memory unit measuring principle is grossly wrong, or else that the mechanisms of measurement are grossly in error.

2) Overload of the USL machine seemed to occur with much more grace than it does at M.I.T. During the day, as system load climbed to the point that idle time vanished, response time grew slowly worse, as expected. But, as the load (measured by number of users) increased well above this point, the only subjective effect was a gradual lengthening of response times. The USL computation center finds that it can allow the number of users to climb as high as 50% above the number that just use up the machine, before the response delays become really painful. This observation suggests that the USL system does not exhibit the non-linear thrashing effect noticed near saturation at M.I.T. This reduction of the non-linear thrashing may well be the most important result of replacement of the bulk store with directly addressable memory.

3) The average multiprogramming level of the USL machine was small--the first three processes in the multiprogramming queue absorbed most of the processor time. This observation suggests that careful control of the eligibility level is no longer a very important issue, and that part of the scheduling algorithm may be simplifiable and not need tuning any more.