Measuring IFS AIX/370 File Server Performance in LPAR Mode

Charles J. Antonelli
Steve Burling
Lee Pearson

1. Introduction

Some months ago, other groups within ITD proposed that the hardware configuration of the 3090-600E running at the Computing Center be changed in order to improve UB-MTS performance. At the time, the current configuration ran VM on the entire B-side of the machine, devoting 2 processors to UB-MTS and the remaining processor to VM AIX/370 guests, the MVS library system, and some miscellaneous guests. The proposal involved splitting the B-side of the machine into two separate LPARs, one of which would contain 2 processors running UB-MTS in native mode, and the other running VM and supporting the same melange of guests.

A question arose as to whether the improvement that would accrue to UB-MTS running native instead of under VM would be offset by penalties incurred by running our AIX/370 guests — and thus our IFS AIX fileservers — under VM in an LPAR. In response to our initial queries, we were told (by a SHARE 75 speaker and by multiple sources elsewhere within IBM) that IBM 3090-xx0E class machines, which cannot be retrofitted with the required microcode, cause over a 50% loss in performance when running VM guests in a PR/SM LPAR. We felt that this would result in unacceptable performance for our initial campus deployment effort.

Due to conflicting information as to the precise nature and scope of the performance degradation, and in order to show the effect of this performance degradation on our specific application (AIX/370 file service), we set aside some system time on 23 September 1990 to run several AIX/370 benchmarks on both configurations. This report describes the nature of the benchmarks that were run and the results we obtained.

2. Test Procedure

We ran three classes of benchmarks: raw packet pumping tests, in which the basic performance of the 3090's ethernet interface is determined; AIX/370-centric tests, in which various measurements are run on the file server machine to measure its ability to run workloads; and AFS-centric tests, in which the ability of the file server running on AIX/370 to serve files is measured.

Previous attempts at running benchmarks while the machine was processing normal workloads convinced us of the need to run our tests in an environment free of the interference caused by the activity of other users. Accordingly, we shut down all other activity on the B-side of the machine except for two AIX/370 guests: **bart**, which was the unit under test; and **homer**, which was used to present a standard load to the CPU (more on that in a moment). We also left running a CMS guest that allowed us to collect RTMSF data (essentially % CPU spent running **bart**) when necessary, as well as a number of other guests Chuck Lever assured us we needed to have.

While we needed performance numbers in this stable (and repeatable) environment, we also need to obtain results when the system was not devoting an entire CPU to our file server, since the latter environment is the one in which we will usually serve files. Accordingly, we ran most performance tests in two environments: *idle*, in which **homer** did no additional work, and *loaded*, in which **homer** ran a continuous kernel build, competing for CPU and disk resources with **homer**.

Results comparing VM Basic Mode and LPAR mode are uniformly given; for each mode, idle and loaded performance figures are given. Most tables also indicate the percentage change in performance between the idle and loaded cases.

2.1. Packet Pumping Tests

The following tests were run: exchanging 4- and 1526-byte UDP packets and 1500-byte RX packets between **bart** (an AIX/370 guest running the IFS3.0 version of the file server on the GA release of AIX/370) and an RT/PC running IFS3.0 on AOS. The goal for this class of tests was to measure the effect of the LPAR configuration on raw packet pumping throughput. For all tests we used a Bus-Tech Inc. ELC2 running in native mode. We did not run a test against a loaded system since we were interested primarily in maximum throughput figures.

Table 1: Raw Packet Pumping Performance

Benchmark	Basic Mode		LPAR Mode	
	Elapsed Time	Channel Busy	Elapsed Time	Channel Busy
10000 4-byte UDP packets	8.8s	35%	9.1s	35%
10000 1526-byte UDP packets	45.9	20	45.9	20
1000 1500-byte RX packets	6.25	-	7.3	-

The channel busy times were estimated from the 3090 console display, which showed channel activity at three-second intervals in bar graph form, with each pixel of the bar representing two percent.

We conclude from this test that LPAR mode has a negligible effect on raw UDP packet pumping performance, and a small effect on RX packet pumping performance. We suspect RX packet pumping is affected to a greater degree due to the much larger size and complexity of the RX packet pumping test program.

2.2. AIX/370-Centric Tests

These tests were designed to measure the effect of the LPAR configuration on **bart**'s CPU and disk performance. The following tests were run: Dhrystone (CPU benchmark); Ousterhout microbenchmarks; and a kernel library build benchmark (CPU & disk performance).

Dhrystone

The Dhrystone benchmark [1] attempts to measure pure CPU performance. It was introduced in an attempt to permit gross CPU power comparisons between different UNIX platforms.

Table 2: Dhrystone CPU Benchmark

Benchmark	Basic Mode			LPAR Mode		
	Idle	Load	Change	Idle	Load	Change
Dhrystone	49.8K Dhry/s	50.3	1.00%	50.0	50.9	1.80%

This test indicates that raw CPU performance is not affected by LPAR mode.

Ousterhout Microbenchmarks

The Ousterhout microbenchmarks [2] are a set of programs that each measure a single operating system feature. The test results below reflect the following individual tests in the order indicated: the time to perform a single system call from a process to the kernel; the time to perform a single context switch between processes; the time to open and close a file; and the time to create a file, write the indicated number of bytes, close the file, re-open it, read all the bytes, close the file, and delete the file. The last test is performed for both a small (1 KB) file and a large (10 MB) file. Table 3 shows the results of these tests; it lists the sum of the user and system time (i.e. the total CPU time consumed) and the wall clock time for each benchmark.

Table 3: Ousterhout Microbenchmarks							
Benchmark		Basic Mode			LPAR Mode		
		Idle	Load	Change	Idle	Load	Change
System Call	User+System	16.06us	16.21us	0.93%	17.93us	18.36us	2.40%
	Wall	16.75	32.93	96.60	20.56	47.16	129.38
Context Switch	User+System	.24ms	.48ms	100.00%	.29ms	.63ms	117.24%
	Wall	.25	.56	124.00	.34	.68	100.00
Open/Close	User+System	.30ms	.31ms	3.33%	.32ms	.32ms	0.00%
	Wall	.35	.63	80.00	.37	.83	124.32
Create/Delete 1KB	User+System	2.28ms	2.04ms	-10.53%	1.87ms	2.46ms	31.55%
	Wall	7.23	4.44	-38.59	3.74	6.28	67.91
Create/Delete 10MB	User+System	648.27ms	636.26ms	-1.85%	728.12ms	756.68ms	3.92%
	Wall	690.5	1309.86	89.70	768.29	1892.76	146.36

This table shows that in all cases except one, LPAR performance is poorer than VM Basic Mode*. This trend has been observed in all of the other tables (Table 6 does not entirely follow this pattern) and will not be belabored further; rather, we will compare how performance degrades in both modes when going from an idle to a loaded system.

The most disk-intensive test, Create/Delete 10MB, shows a trend that will reappear in later tables: while Basic Mode performance degradation is about 90% when going from an idle to a loaded system, the equivalent LPAR performance degradation is about 150%. We should expect the wall clock time to roughly double when going from idle to loaded, since we are approximately doubling the workload by giving **homer** work to do, and yet LPAR performance suffers an additional 50% loss.

AIX/370 Kernel Build

The kernel build benchmark simply records the user, system, and wall clock times consumed in building an AIX/370 kernel from its source code, as a measure of combined CPU and disk performance on a large job.

Table 4: Kernel Build Benchmark							
Benchmark		Basic Mode			LPAR Mode		
		Idle	Load	Change	Idle	Load	Change
Kernel Build	User	533.9s	542.1s	1.54%	533.1s	533.1s	0.00%
	System	41.9	53.4	27.45	51.9	67.7	30.44
	Wall	673.4	1327.3	97.10	771.8	1732	124.41

In this test, we observe that the user time remains fairly constant in both modes when going from an idle to a loaded state, system time is up slightly, and wall clock time is up a lot, 24% over what we would expect.

2.3. AFS-Centric Tests

These tests were designed to measure the effect of the LPAR configuration on **bart**'s ability to serve files. The following tests were run: clientload and Connectathon basic tests.

Clientload Benchmark

The clientload test starts fifteen simultaneous client requests (one request per client machine) to read the same 1 MB file; each client flushes the file from its cache before making the request. The elapsed time required to deliver the file to each client as well as the CPU resources consumed on the server are recorded.

The final group of four entries in Table 5 record two sets of values. The first set shows the maximum and average CPU utilization of the fileserver process, expressed as a percentage of the total CPU resources available to **bart**; these utilization statistics were collected by the UNIX `ps` command, run on **bart** every

* The results of this test, Create/Delete 1KB in Basic Mode (idle), should be considered suspect, since these values are greater than the corresponding loaded values.

thirty seconds during the clientload test. The second set shows the maximum and average CPU utilization of **bart** as recorded by VM RTMSF; these statistics were collected by defining a separate CMS guest that collected RTMSF statistics on request, and running a process on **bart** that requested statistics every sixty seconds.

Table 5: Simultaneous 1MB AFS File Read Times

Client	Basic Mode			LPAR Mode		
	Idle	Load	Change	Idle	Load	Change
barnone	109s	102s	-6.42%	87s	155s	78.16%
boston	107	104	-2.80	100	136	36.00
doom	101	103	1.98	101	142	40.59
ebbtide	111	116	4.50	108	152	40.74
eh	76	98	28.95	75	125	66.67
emptys	113	104	-7.96	95	149	56.84
flam	103	103	0.00	92	133	44.57
jackpot	100	100	0.00	96	138	43.75
nunn	99	99	0.00	91	151	65.93
pinhead	93	103	10.75	100	129	29.00
rasta	105	106	0.95	89	151	69.66
rioja	108	109	0.93	106	155	46.23
virgo	109	104	-4.59	105	147	40.00
xtc	108	107	-0.93	102	147	44.12
yale	101	106	4.95	103	146	41.75
averages	103s	104s	1.36%	97s	144s	48.69%
Max FS CPU Util	45%	55%		71%	95%	
Avg FS CPU Util	11	16		21	40	
Max BART CPU Util	48%	45%		52%	35%	
Avg BART CPU Util	16	16		15	15	

This test directly measures our system when it is doing what it is supposed to be doing: serving files. We observe a negligible performance loss when loading the system in Basic Mode, and a 50% loss when loading the system in LPAR mode. We further observe that the file server process running under AIX/370 on the average consumes more of the CPU allocated to **bart** by VM in both the idle and loaded LPAR cases; a maximum value of 95% was observed for the loaded LPAR case. At the same time, the average absolute amount of CPU allocated to **bart** by VM changes only slightly. It thus seems that, in LPAR mode, AIX/370 is getting the same amount of CPU from VM but seems to be spending more of that CPU on the file server. Since the workload should be the same, we conjecture the file server is taking longer in LPAR mode because the instructions are taking longer to execute; if it were just slower I/O performance we would expect things to take longer but we would not expect to see an increase in file server CPU utilization. This corroborates our informal observations of the 3090 console display, which indicated the CPU was spending more of its time in emulation mode when running under LPAR than it did in Basic Mode.

The CPU utilization measurement intervals were pretty coarse: thirty seconds for the FS values and sixty seconds for the BART values. This implies that less trust should be placed on the maximum recorded values than in the averages, although both could be improved by sampling more often. Sampling too often, however, will perturb the results.

Connectathon Basic Tests

The Connectathon basic tests perform a number of repetitive file system operations and record the time required per operation. The test results below reflect the following individual tests in the order indicated: (1) create 155 files in 62 directories 5 levels deep; (2) remove these files and directories; (3) do 500 getwd and stat calls; (4) do 1000 chmods and stats on 10 files; (5a) write a 1 MB file 10 times; (5b) read a 1 MB file 10 times; (6) read 20500 directory entries; (7) do 200 rename and link operations on 10 files; (8) do 400 symlink and readlink operations on 10 files; and (9) do 1500 statfs calls. We ran one copy of this benchmark against the file server and obtained the results shown in Table 6; the final four entries of this table were obtained in the same manner as in Table 5.

Table 6: Connectathon Basic Tests

Benchmark	Basic Mode			LPAR Mode		
	Idle	Load	Change	Idle	Load	Change
Test 1	75.49s	80.39s	6.49%	75.84s	81.9s	7.99%
Test 2	43.77	46.55	6.35	48.5	52.98	9.24
Test 3	43.1	52.6	22.04	45.72	51.69	13.06
Test 4	132.20	134.62	1.83	150.2	141.6	-5.73
Test 5a	137.58	129.31	-6.01	122.36	125.2	2.32
Test 5b	43.84	42.32	-3.47	39.32	39.27	-0.13
Test 6	66.16	68.19	3.07	64.68	77.71	20.15
Test 7	61.85	53.49	-13.52	63.44	69.79	10.01
Test 8	93.51	78.76	-15.77	93.72	96.29	2.74
Test 9	1.33	1.63	22.56	1.52	1.59	4.61
Max FS CPU Util	9%	10%		13%	18%	
Avg FS CPU Util	4	4		5	6	
Max BART CPU Util	15%	14%		19%	20%	
Avg BART CPU Util	8	8		9	10	

The file server seems to be less heavily and more irregularly stressed by this test than the previous one, although the loaded LPAR values are in most cases larger than any of the others.

3. Conclusions

None of the tests we have run has shown any advantage to running the IFS File Server in LPAR mode. On the contrary, a test which exercises the AIX/370 guest in its intended mode of operation — serving files — shows a 50% performance loss at client workstations and a large increase in CPU utilization by the file server. Other tests show a performance loss when jobs involving large amounts of disk I/O are done. Based on these results, we cannot recommend operating the IFS File Server in LPAR mode at this time.

4. Acknowledgements

The effects of the enthusiastic participation of Chuck Lever in our benchmarking activities cannot be overestimated.

1. Reinhold P. Weicker, ‘‘Dhrystone: A Synthetic Systems Programming Benchmark,’’ *CACM* **27**(10), pp. 1013–1030 (October 1984).
2. John K. Ousterhout, ‘‘Why Aren’t Operating Systems Getting Faster as Fast as Hardware?,’’ pp. 247–256 in *Proc. 1990 Summer USENIX Conference* (June 1990).