
Performance Engineering Implications of Hyper-Threading Systems

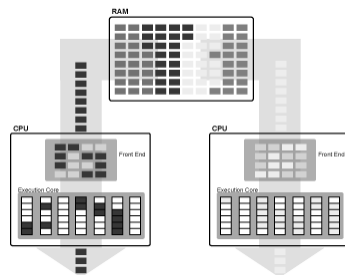
Performance Engineering Implications of Hyper-Threading Systems

- Overview
- Project Plan
- Bench Mark Design
- System Environment
- Service Demand Model Input
- Model Projection for Mars and Zeus
- Results from Mars
- Results from Zeus
- Model Analysis
- Revised Model Projections & Results Comparison
- Lessons Learned / Problems Encountered
- Conclusions

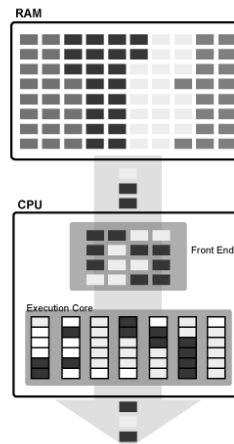
Overview

- Hyper-Threading fools the operating system into thinking there are two physical processors, allowing two threads to be run in parallel.
- The two threads are run on separate 'logical' processors within the same physical processor.

Single-threaded SMP



Hyper-Threaded SMP



Can we model Hyper-Threading?

- Traditional performance models were developed for physical rather than virtual processors.
- Model projections for CPU utilization may not be correct due to incorrect model assumptions.

Project Plan

- Develop an application and system benchmark to obtain inputs to the open queueing network model found in *OpenQN.xls*.
- Validate the model using a traditional dual-CPU SMP system.
- Attempt to validate the model with a Hyper-Threading system.

Benchmark design

- Designed and implemented a common gateway interface (CGI) application using perl with the intention to stress the webserver, database, and disk. The main application components are as follows:
 - connect to database
 - retrieve transaction id
 - close connection
 - write transaction information to disk
 - return transaction information to client

System Environment

SOFTWARE:

- Windows XP
- Linux OS platform
- Apache webserver and MySQL RDBMS
- Perl
- Web Application Stress Tool

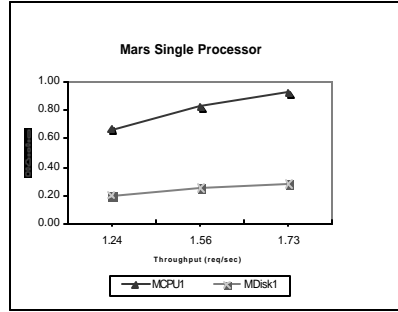
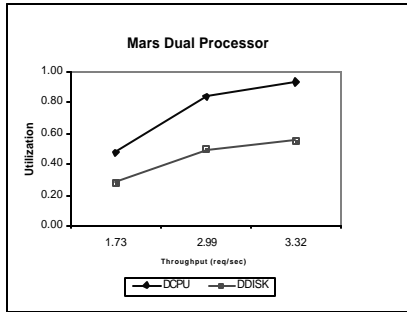
HARDWARE:

<ul style="list-style-type: none">■ Traditional Dual SMP System Hostname: Mars Dual Pentium II 266Mhz CPUs 256 MB RAM Fedora Core 1 w/ Linux 2.4.22 SMP Kernel	<ul style="list-style-type: none">■ Hyper-Threading system Hostname: Zeus Pentium 4 2.6 GHz HT CPUs w/ 256 MB RAM Fedora Core 1 w/ Linux 2.4.22 SMP Kernel
<ul style="list-style-type: none">■ Development Workstation Hostname: Hyper AMD Athlon 2200+ CPU 256 MB RAM Fedora Core 1 w/ Linux 2.4.22 Kernel	<ul style="list-style-type: none">■ Client Workstation Hostname: XP AMD Athlon 1266 MHz CPU 384 MB RAM Windows XP, Service Pack 1

Service Demand Model Input

Traditional SMP System Results - Mars

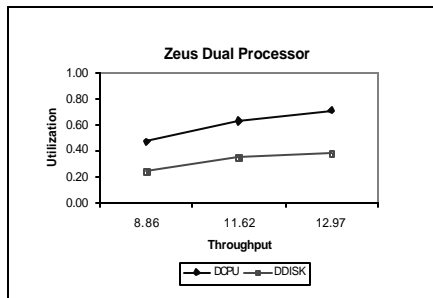
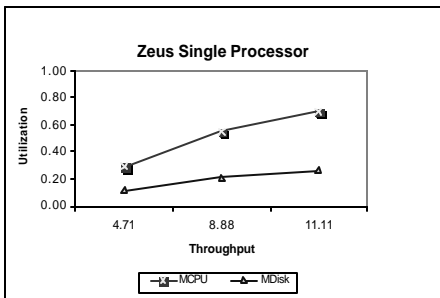
Mars	Dual			Single		
Xo	1.73	2.99	3.32	1.24	1.56	1.73
Avg CPU	0.48	0.84	0.93	0.66	0.83	0.92
Avg Disk	0.28	0.50	0.55	0.20	0.25	0.28
SD CPU	0.2747	0.2806	0.2816	0.5312	0.5297	0.5327
SD Disk	0.1610	0.1656	0.1669	0.1598	0.1608	0.1596



Cont...

Hyper-Threading System Results - Zeus

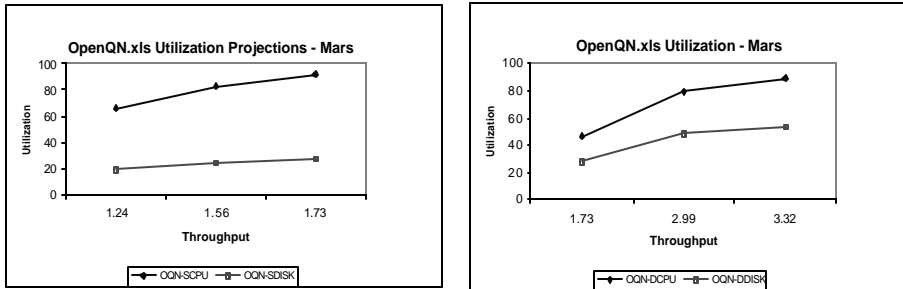
Zeus	Dual			Mono		
Xo	8.86	11.62	12.97	4.71	8.88	11.11
Avg CPU	0.47	0.63	0.71	0.29	0.55	0.70
Avg Disk	0.24	0.35	0.38	0.11	0.21	0.26
SD CPU	0.0532	0.0541	0.0549	0.0623	0.0623	0.0626
SD Disk	0.0272	0.0297	0.0295	0.0241	0.0238	0.0238



Model Projection for Mars

CPU Mode	Service Demand (CPU)	Service Demand (Disk)	Throughput	Projected Utilization (CPU)	Projected Utilization (Disk)
Mono	0.5312	0.1601	1.24	65.87	19.85
Mono	0.5312	0.1601	1.56	82.87	24.98
Mono	0.5312	0.1601	1.73	91.90	27.70
Dual	0.5312	0.1601	1.73	45.95	27.70
Dual	0.5312	0.1601	2.99	79.41	47.87
Dual	0.5312	0.1601	3.32	88.18	53.15

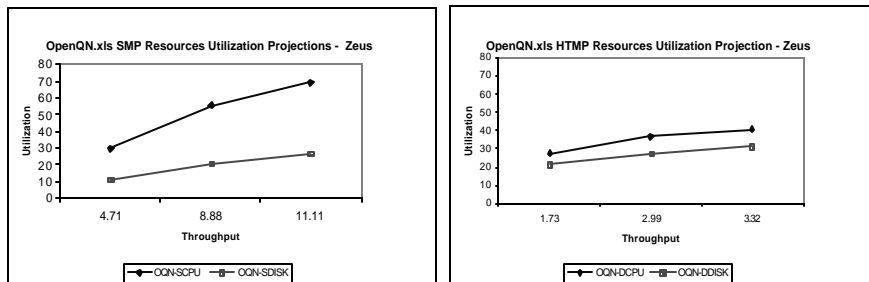
OpenQN.xls Utilization Projections - Mars



Model Projection for Zeus

CPU Mode	Service Demand (CPU)	Service Demand (Disk)	Throughput	Projected Utilization (CPU)	Projected Utilization (Disk)
Mono	0.0624	0.0239	4.71	29.39	11.26
Mono	0.0624	0.0239	8.88	55.41	21.22
Mono	0.0624	0.0239	11.11	69.33	26.55
Dual	0.0624	0.0239	8.86	27.64	21.18
Dual	0.0624	0.0239	11.62	36.25	27.77
Dual	0.0624	0.0239	12.97	40.47	31.00

OpenQN.xls Utilization Projections - Zeus



Revised Model Projections & Results Comparison Mars & Zeus

CPU Mode	Xo	SD (CPU)	SD (Disk)	OpenQN Projected (CPU)	Observed (CPU)	Accuracy	% Off	OpenQN Projected (Disk)	Observed (Disk)	Accuracy	% Off
Mono	1.24	0.5312	0.1601	65.87	65.87	0.00	0.00%	19.85	19.81	-0.04	0.20%
Mono	1.56	0.5312	0.1601	82.87	82.64	-0.23	0.28%	24.98	25.09	0.11	0.44%
Mono	1.73	0.5312	0.1601	91.90	92.15	0.25	0.27%	27.70	27.60	-0.10	0.35%
Dual	1.73	0.5312	0.1601	45.95	47.52	1.57	3.41%	27.70	27.86	0.16	0.57%
Dual	2.99	0.5312	0.1601	79.41	83.90	4.49	5.65%	47.87	49.51	1.64	3.42%
Dual	3.32	0.5312	0.1601	88.18	93.49	5.31	6.02%	53.15	55.42	2.27	4.27%

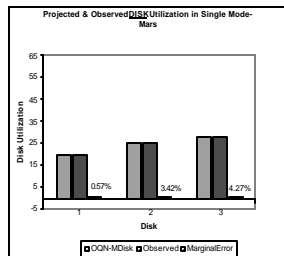
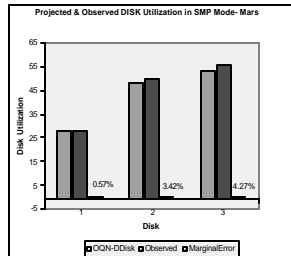
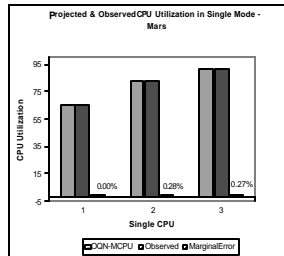
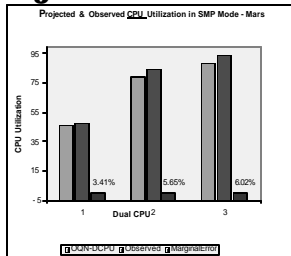
CPU Mode	Xo	SD (CPU)	SD (Disk)	OpenQN Projected (CPU)	Observed (CPU)	Accuracy	% Off	OpenQN Projected (Disk)	Observed (Disk)	Accuracy	% Off
Mono	4.71	0.0624	0.0239	29.39	29.35	-0.04	0.13%	11.26	11.37	0.11	0.99%
Mono	8.88	0.0624	0.0239	55.41	55.36	-0.05	0.09%	21.22	21.10	-0.12	0.57%
Mono	11.11	0.0624	0.0239	69.33	69.54	0.21	0.30%	26.55	26.39	-0.16	0.60%
Dual	8.86	0.0624	0.0239	27.64	47.15	19.51	70.58%	21.18	24.09	2.91	13.75%
Dual	11.62	0.0624	0.0239	36.25	62.81	26.56	73.28%	27.77	34.55	6.78	24.42%
Dual	12.97	0.0624	0.0239	40.47	71.25	30.78	76.06%	31.00	38.30	7.30	23.55%

5/5/2004

Performance Engineering Implications of
Hyper Threading Systems

13

Projected & Observed Mars Utilization

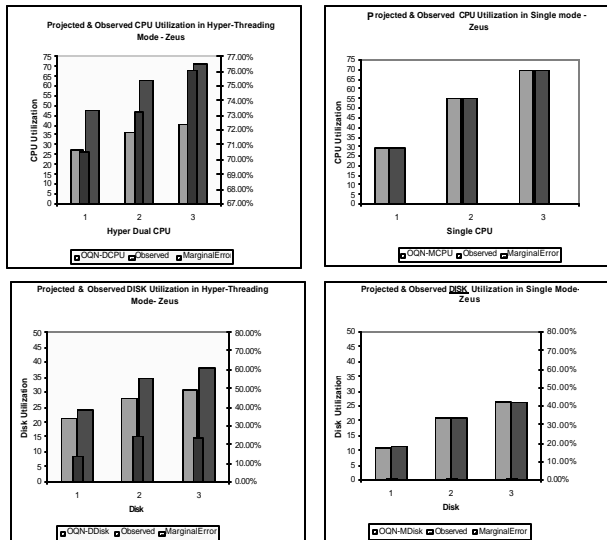


5/5/2004

Performance Engineering Implications of
Hyper Threading Systems

14

Projected & Observed Zeus Utilization



5/5/2004

Performance Engineering Implications of
Hyper Threading Systems

15

Mars Results

- Mars results clearly validate the performance model in *OpenQN.xls*. Both CPU and disk utilization are projected with a high degree of accuracy for uniprocessor and SMP systems.
- Uniprocessor projection is accurate for both device types. SMP projection, while not perfect, is still reasonably accurate.
- It appears that the model is slightly aggressive in its projection of utilization savings projections on SMP systems but overall the high degree of accuracy proves that the model is useful and accurate performance tool.

5/5/2004

Performance Engineering Implications of
Hyper Threading Systems

16

Zeus Results

- Zeus results uniprocessor utilization projection is very good in *OpenQN.xls*. However utilization projections are significantly off for the SMP cases.
- The results in Hyper-Threading mode show a large margin of error for the CPU; this is consistent with our expectations.
- The margin of error for disk utilization projection however is surprising.
- The higher observed disk utilization may be attributed to inherent inefficiencies or penalties in the Hyper-Threading design or our Operating System's implementation of Hyper-Threading support rather than a model error.

Alternative Approaches

- Modify the model?
 - Complicated
 - Needs validation and regression testing
- Modify the inputs?
 - Potentially less complicated
 - Requires only validation

Model Input Modification

■ Assumptions

- Hyper-Threading increases performance 20-30%
- Model assumes 50% CPU Service Demand reduction for SMP

■ Technique

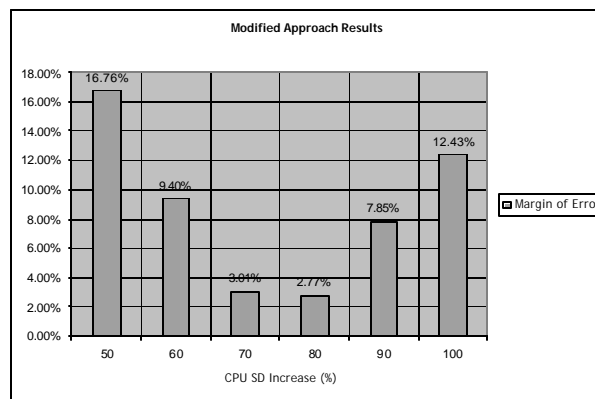
- Determine uniprocessor CPU Service Demand
- Increase Service Demand by 70%

5/5/2004

Performance Engineering Implications of
Hyper Threading Systems

19

Revised Model Projection



5/5/2004

Performance Engineering Implications of
Hyper Threading Systems

20

Lessons Learned / Problems Encountered

- Hyper-Threading performance for single-threaded applications is less clear and in typically provides little performance gain.
- Disk service times increased nearly 20% on Zeus and higher disk utilization was observed when Hyper-Threading was enabled.
- This may be due to an implementation problem at the Operating System disk schedule or at the hardware level.

Conclusions

- Traditional performance models do not natively project Hyper-Threading performance correctly.
- Hyper-Threading processors have predictable performance attributes that can be used to project their performance when usage patterns are known.
- Using the proposed modified approach we were able to increase the accuracy of the projections in *OpenQN.xls*. We found that an assumption of a 20-30% performance gain produced the most accurate results
- Single-threaded applications and applications with a low degree of multi-threading would require a different approach.

References

- [1] Menascé, Daniel A. and Almeida, Virgilio A. F. and Dowdy, Larry W. Performance by Design: Computer Capacity Planning by Example. Prentice Hall, 2004.
- [2] Stokes, Jon "Hannibal". Introduction to Multithreading, Superthreading and Hyper-threading
<http://www.arstechnica.com/aeidia/h/hyperthreading/hyperthreading-1.html>
- [3] Vianney, Duc. Hyper-Threading Speeds Linux
<http://www-106.ibm.com/developerworks/linux/library/l-ht/>
- [4] Intel Hyper-Threading Information
<http://www.intel.com/technology/hyperthread/index.htm>
- [5] Linux Kernel
<http://www.kernel.org>
- [6] Fedora Project
<http://fedora.redhat.com>
- [7] Apache WebServer
<http://www.apache.org>
- [8] MySQL RBMS
<http://www.mysql.org>
- [9] Microsoft Web Application Stress Tool
<http://www.microsoft.com/technet/itsolutions/intranet/downloads/webstres.msp>