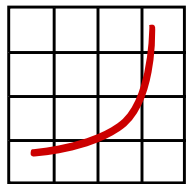


SPECpower_ssj2008** Characterization

Anil Kumar, Larry Gray and Harry Li

Intel® Corporation



spec

SPEC Workshop – January 27, 2008


* Other names and brands may be claimed as the property of others

** SPEC and the benchmark names are trademarks of the Standard Performance Evaluation Corporation

Performance Data as of 30 January 2008.

Agenda

- SPECpower_ssj2008 quick overview
- SPECpower_ssj2008 initial characterization
 - System resources utilization
 - Impact of JVM Optimizations
 - Frequency scaling
 - Processor scaling
 - Platform generation scaling
- General observations
- Summary



SPECpower_ssj2008

Quick overview

SPECpower* - A “Graduated” Workload

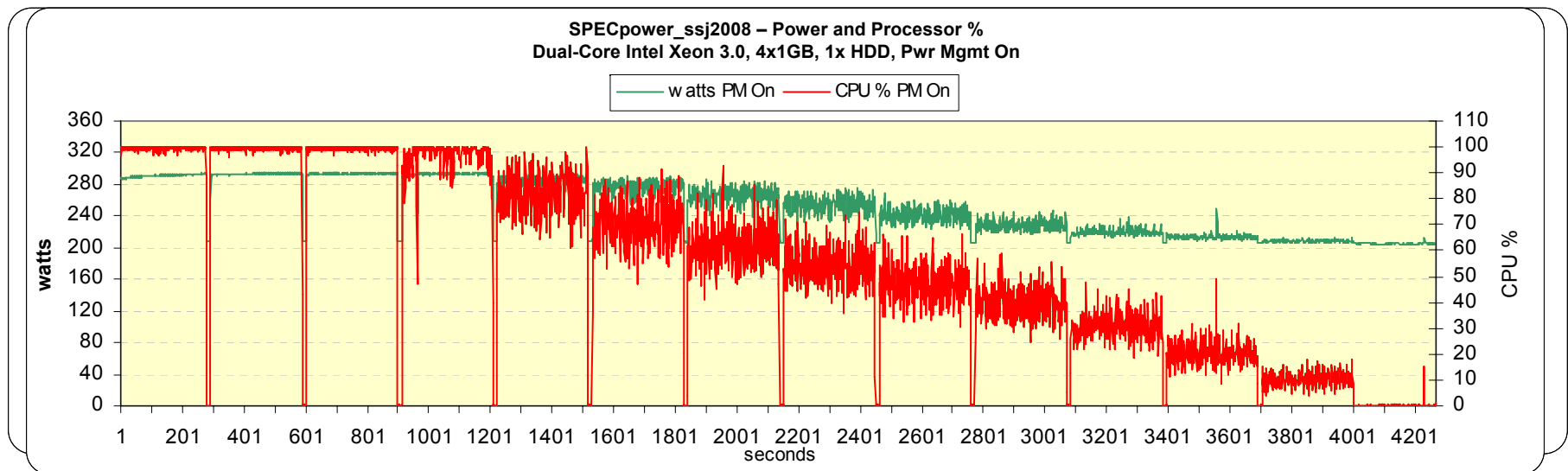
First: A Calibration Phase: Run to Peak Transaction Throughput

- # warehouses or threads = # cores, scheduling is “ungated”

Next: Load Levels: Gradations Based on Calibrated Throughput

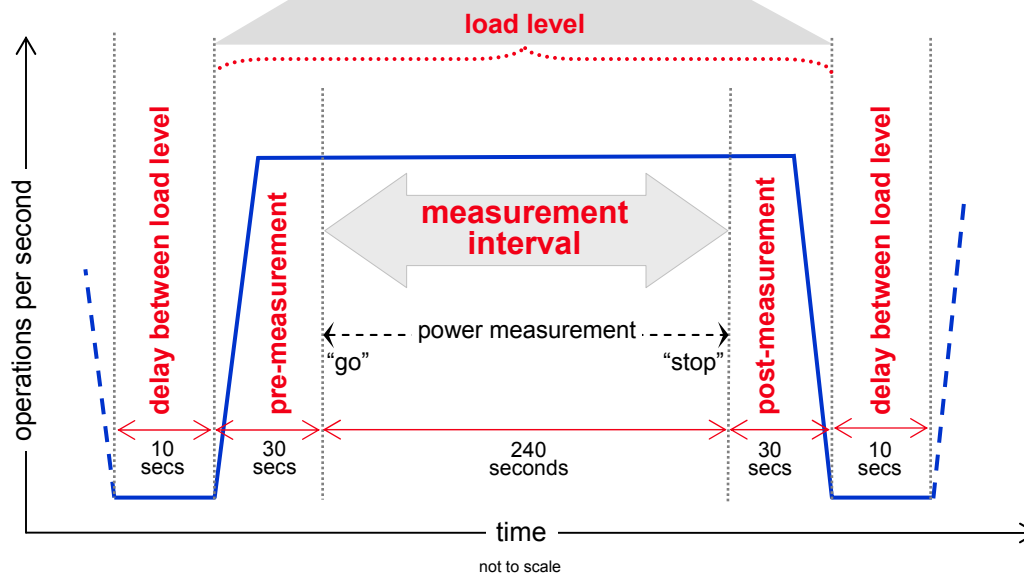
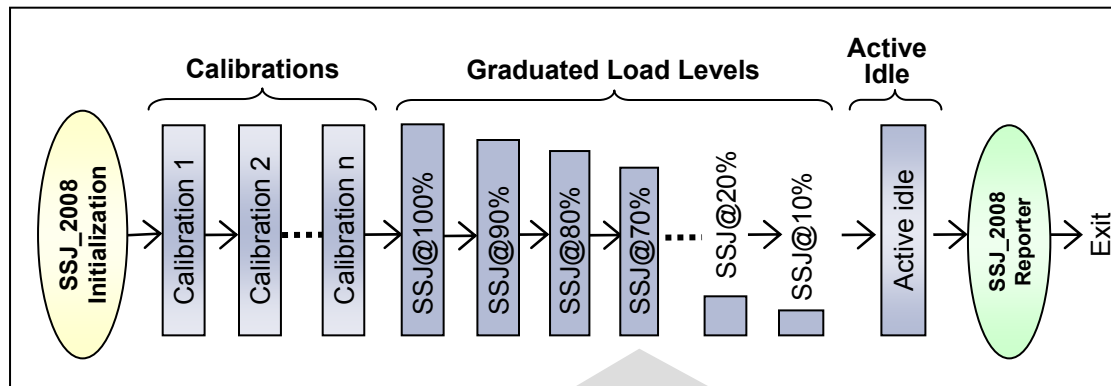
- Average of last two calibration levels = peak calibrated throughput
- Example Below is x10 or 10% increments – the benchmark

Actual Average Per Cent of Calibrated Peak Throughput									
99.8%	90.1%	79.6%	69.7%	60.2%	49.9%	40.1%	30.6%	20.0%	9.9%



Seeks Peak Throughput, Runs and Reports a “Load Line”

Controlling Measurements



- Each load level has a “measurement interval” of 240 seconds, plus,
 - “inter-level” (delay between load levels),
 - ramp up (pre-measurement)
 - ramp down (post-measurement)
- Enables synchronization, power with performance, data capture
- Provides settle time
- Required for Consistent, Repeatable Measurements

SPECjbb2005 vs. SSJ_OPS@100%

SSJ_2008 *derived* from SPECjbb2005 - But different!

- Base code and transaction types are from SPECjbb2005

Substantive changes!

The two are **not comparable**:

- Notable Differences
 - Different transaction mix
 - Transaction scheduling and timing
 - Modified throughput accounting
 - Data collection via network – TCP/IP
 - More logging increases disk I/O
 - Plus others

SPECpower_ssj2008 - Metric Definition

The Primary Metric for SPECpower_ssj2008:

$$\text{overall ssj_ops/watt} = \sum \text{ssj_ops @ 11 pts} / \sum \text{avg watts @11 pts}$$

(includes power at the active idle state)

Table from SPECpower_ssj2008 Full Disclosure Report

Performance			Power	Performance to Power Ratio
Target Load	Actual Load	ssj ops	Average Power (W)	
100%	99.10%	220,306	276	799
90%	90.40%	200,860	269	746
80%	79.50%	176,684	261	677
70%	70.30%	156,344	254	616
60%	59.60%	132,525	245	541
50%	49.60%	110,222	237	465
40%	40.20%	89,388	229	390
30%	30.10%	66,875	221	302
20%	19.90%	44,157	213	207
10%	10.20%	22,649	206	110
Active Idle		0	198	0
$\sum \text{ssj_ops} / \sum \text{power} =$				468

ssj_ops@100%

ssj_ops each level

average power
each level

performance / power
each level

overall ssj_ops/watt

Much more data in SPEC report !

SPECpower_ssj2008 Intel publication #017

http://www.spec.org/power_ssj2008/results/res2007q4/power_ssj2008-20071129-00017.html



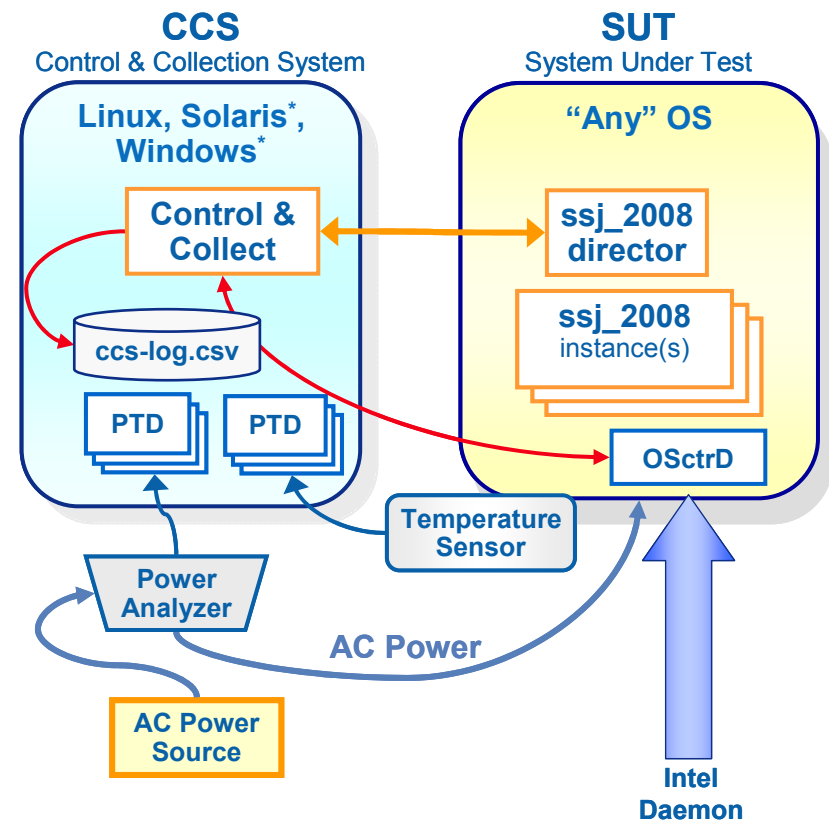
Initial characterization of SPECpower_ssj2008

Hardware and Software

- SUT: Intel® “White Box”
 - Dual and Quad Core Intel® Xeon® 2.0 & 3.0 GHz
 - Supermicro* X7DB8/ Main Board, Super Micro 5000P (Blackford chipset)
 - 4x 2GB FBDIMMs
 - 1x 700W PSU
 - 5U Tower Platform
- Microsoft* Windows Server 2003 64 bit
 - Power Options: Server Balanced Processor Power and Performance
- JVM: BEA* JRockit* P27.4.0 64 bit
 - JVM Command Line similar to published results
- Sampling Rates:
 - Power: 1 second (average from meter)
- SPECpower_ssj2008 setup
 - SSJ Director on SUT
 - load levels 120 seconds

Collecting OS Counters

- Intel Written Daemon, “OSctrD.exe”
 - Counters defined in ccs.props
- Daemon runs on SUT
 - Data to CCS via TCP/IP
 - Can run on CCS
 - CCS logs counters along with watts, trans, etc.
- Windows Only
 - Linux port under consideration
- “Integrated” Log
 - Primary advantage



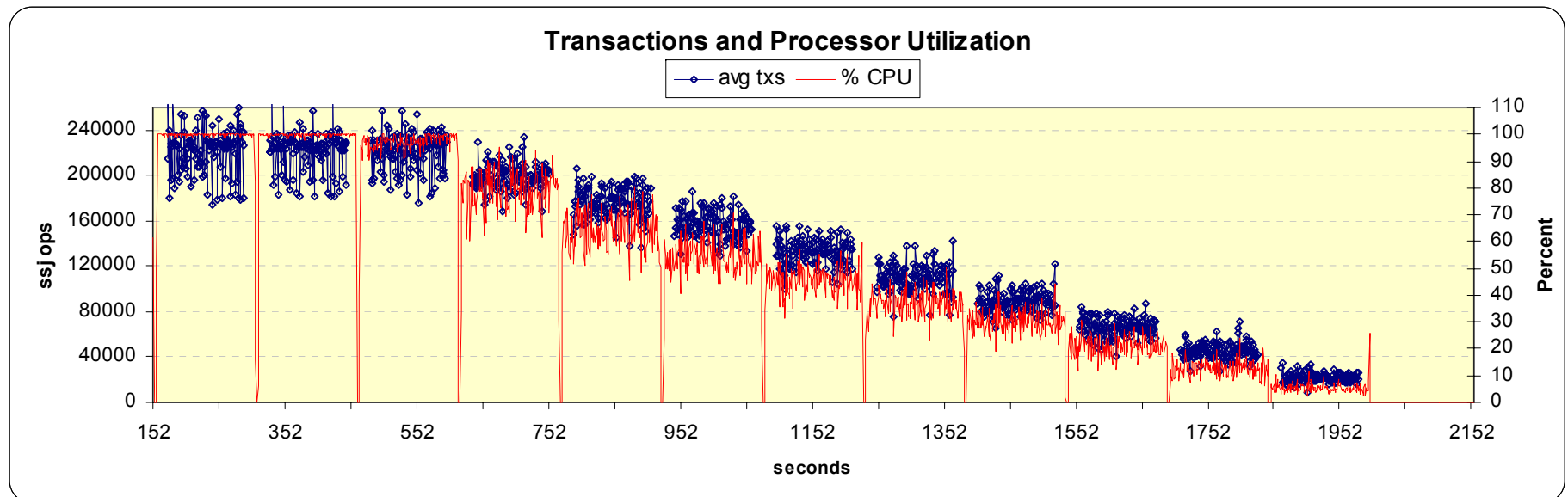
Time	xactions	RT	Watts	amps	PF	Temp	Processor %	...	% C1 time
..

SSJ_2008 Memory Usage

- Code footprint:
 - ~1.5M (total of all methods JIT'ed and optimized)
- Data footprint:
 - ~50MB per warehouse “database” size
 - ~8KB of transient objects per transaction
- JVMs
 - 32 bit JVM - Max. 4GB heap
 - 64 bit JVM - much larger heap (max. 2^{64} Bytes)
 - Multiple instances can/will increase memory footprint
- Optimal memory size is throughput capacity dependent
 - Platform and configuration specific
 - Example: Quad-Core Intel Xeon based Dual Processor system
 - ~8GB optimal for SPECpower_ssj2008
- All above specific to BEA JRockit JVM

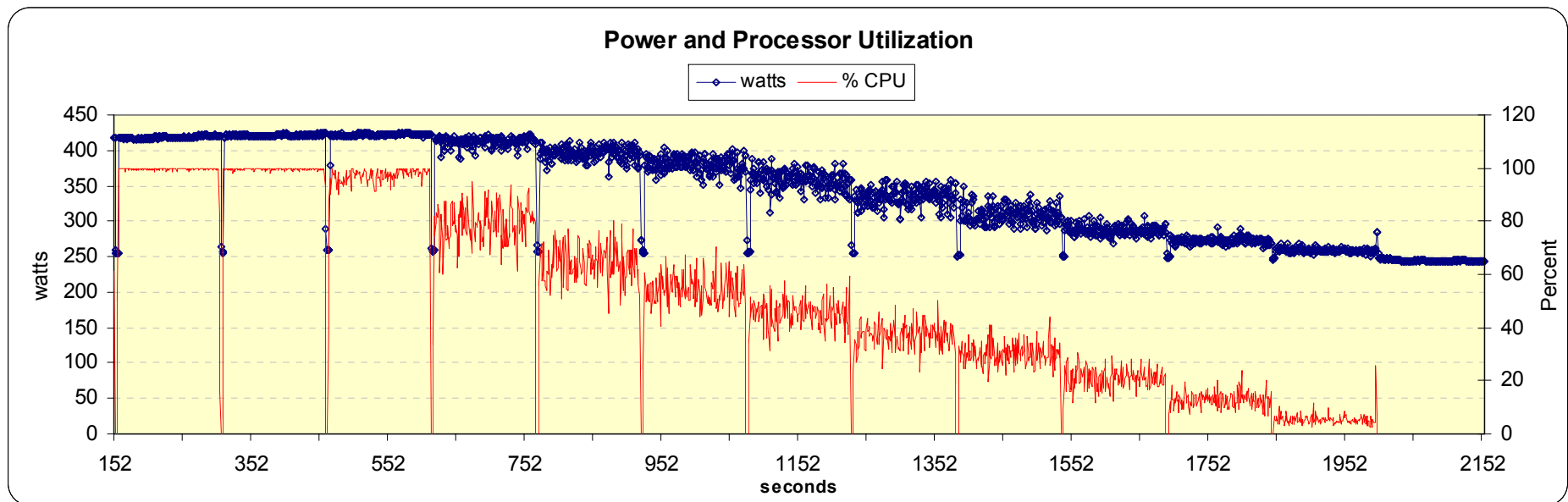
Transactions (SSJ OPS)

- CPU % tracks load
 - As expected on Intel Core 2 architecture
 - Other architectures will vary (SMT etc.)
- Load level targets are % of SSJ_OPS@calibrated
- CPU utilization is *no part* of the benchmark



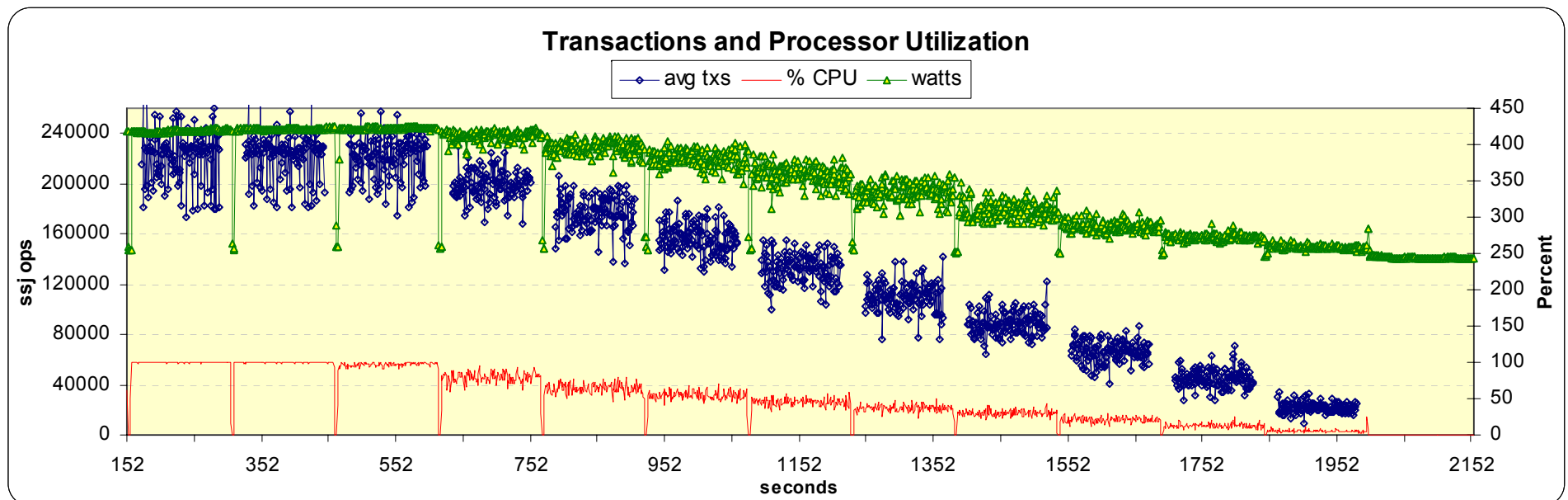
Power and Processor Utilization

- Average SSJ OPS per level tracking as expected
 - Throughput per sec showing desired variability within load level
 - Negative Exponential inter-arrival time batch scheduling
- Power consumption varies with load



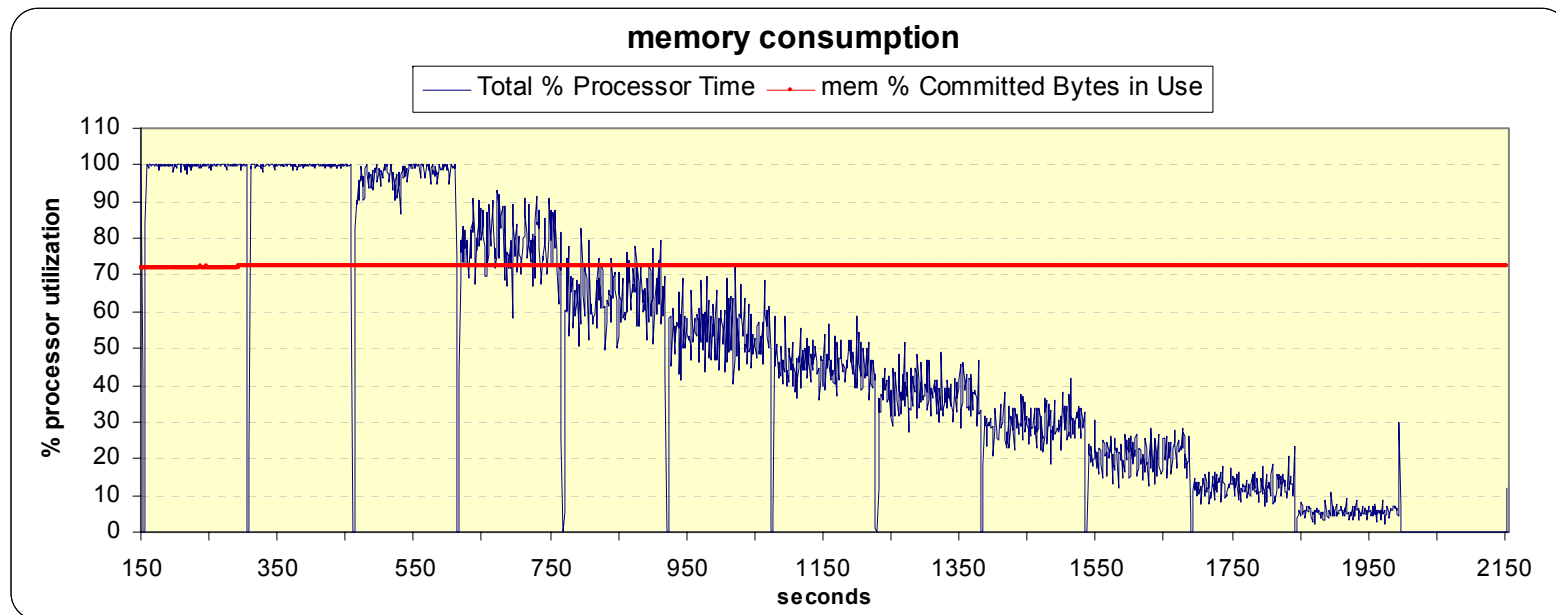
All Three (SSJ OPS, CPU% and Power)

- At all load levels including active idle:
 - All three, SSJ OPS, CPU % utilization and Average Watts
- Three on One Chart – interesting capability



Memory Utilization

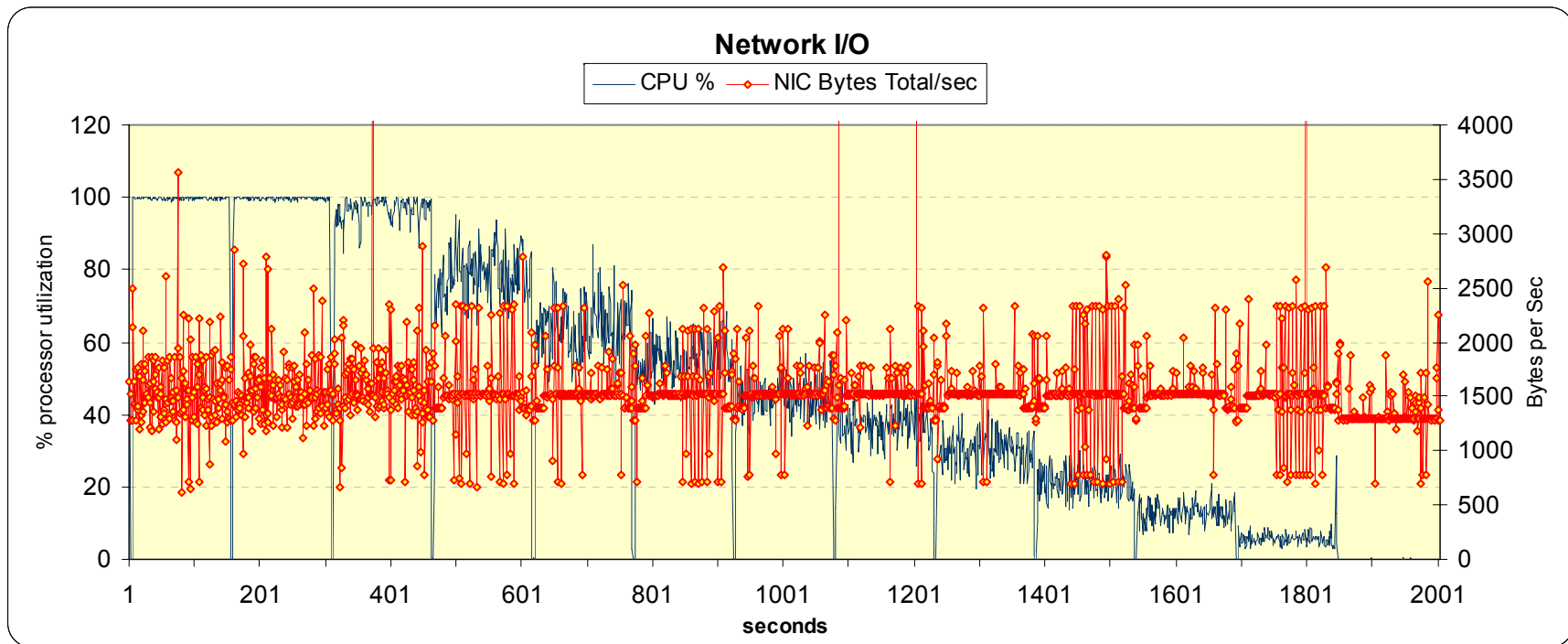
- Java heap size remains same throughout the run
 - With typical tuning; `-Xmx==Xms`
- Constant committed memory in use at all load levels
 - including active idle – JVM(s) still active



Memory usage profile will change with heap settings – especially if “dynamic”

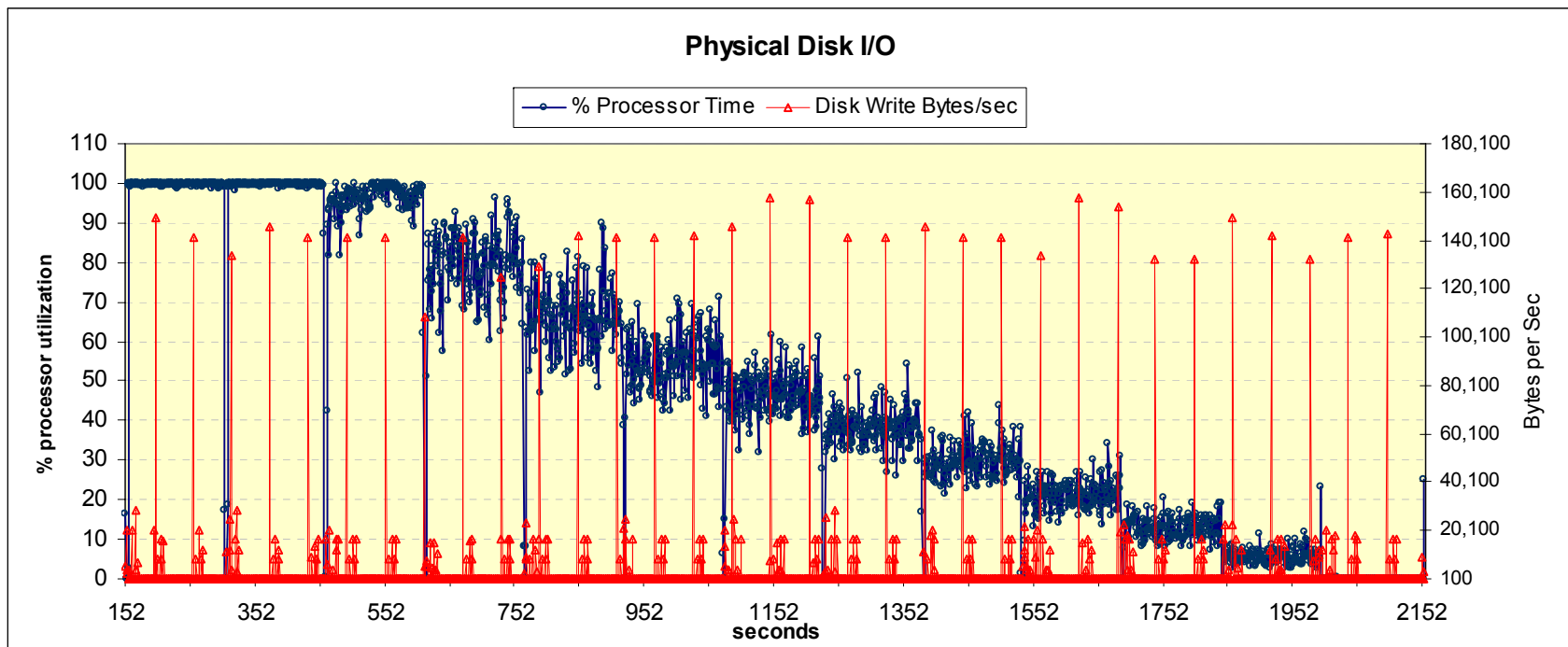
Network I/O

- ~1.5 K Bytes/sec of network I/O at all load levels
 - including active idle:
- Network I/O from per sec request/response
 - between Control & Collect (CCS) and SSJ_2008 Director

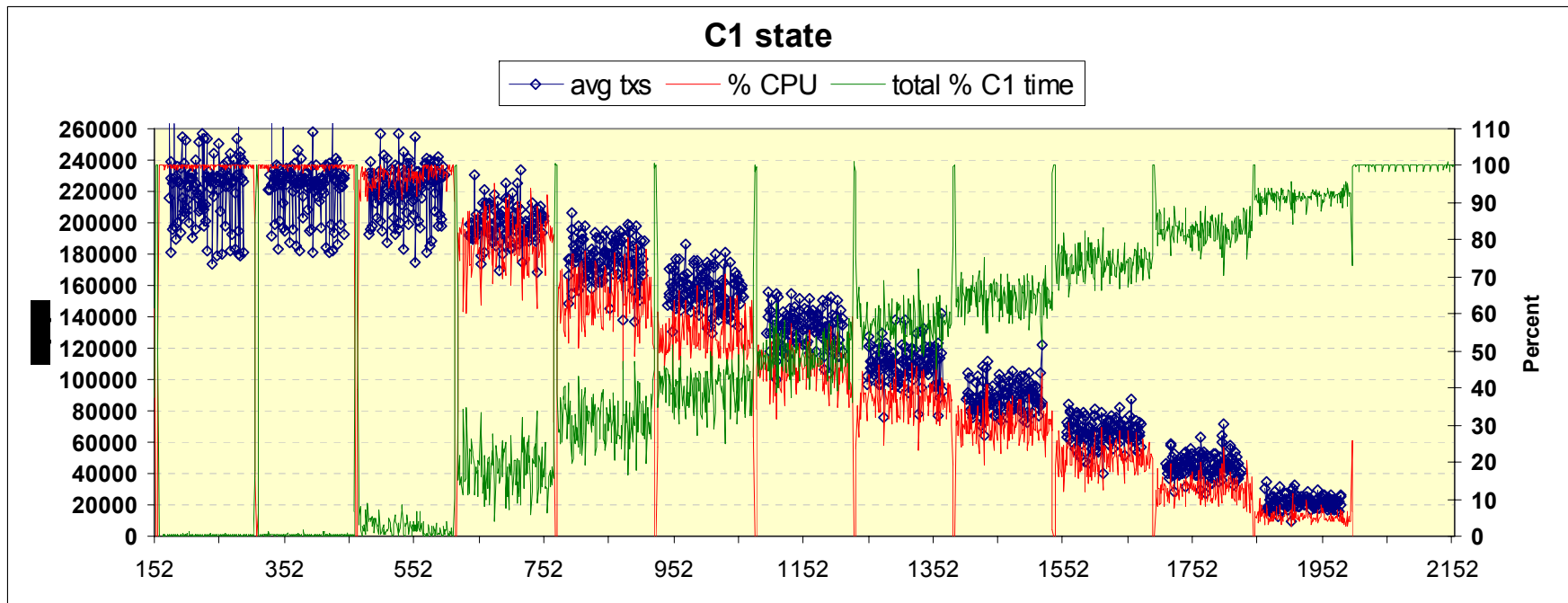


Disk I/O

- Disk I/O – Regular bursts of ~140K Byte writes,
 - ~3.3K Bytes/sec average for all load levels
 - Most disk writes related to SSJ_2008 logging
- Disk reads average zero



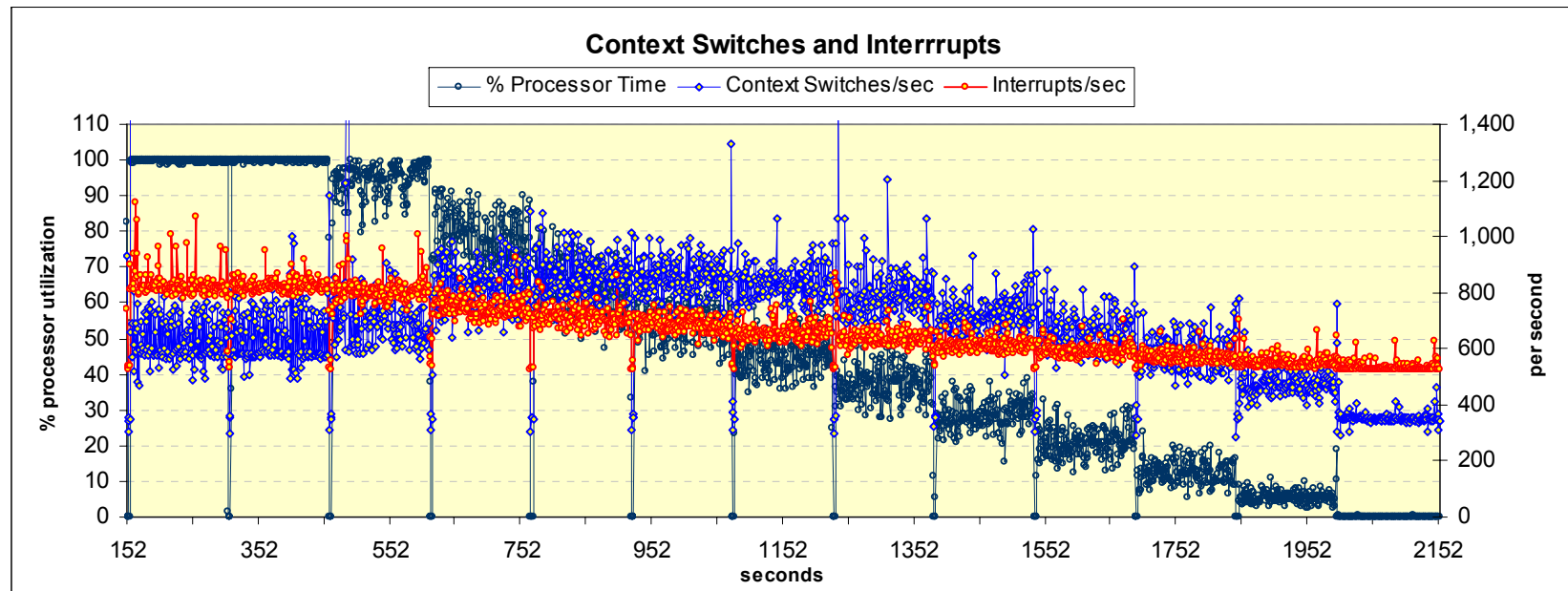
C1 state



- % Time in C1 State – Inverse of CPU %
 - C1/C1E Time contributes to power saving
- Varies with architecture, OS and policies
 - Intel EIST and C1E “enabled” in BIOS

Basic system events

- Interrupts: ~700 /sec at all load levels
- Context switches ~800 /sec
 - Below 50% declining to ~400 at active idle
- Rates are OS and platform dependent
- More Investigation Needed Here



Impact of JVM Optimizations

- Experiment with JVM Options

- JAVAOPTIONS_SSJ="" (None, default heap and optimization)
- JAVAOPTIONS_SSJ="-Xms3000m -Xmx3000m -Xns2400m -XXaggressive -XXlargePages -XXthroughputCompaction -XXcallprofiling -XXlazyUnlocking -Xgc:genpar -XXtlasize:min=12k,preferred=1024k"

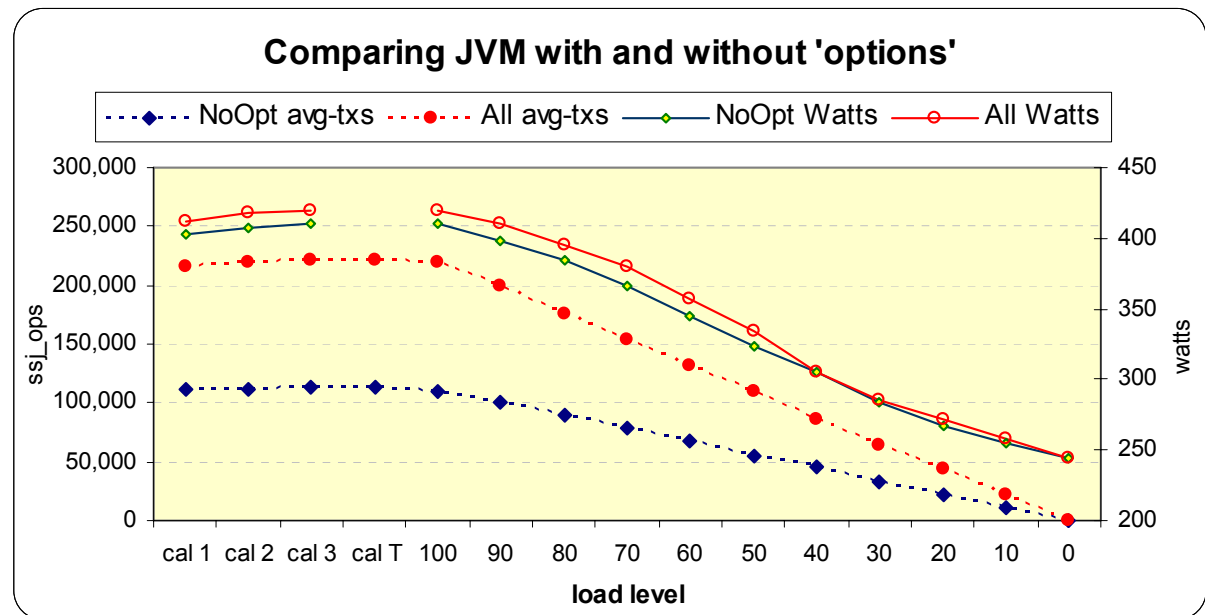
- Performance

Loss ~50%

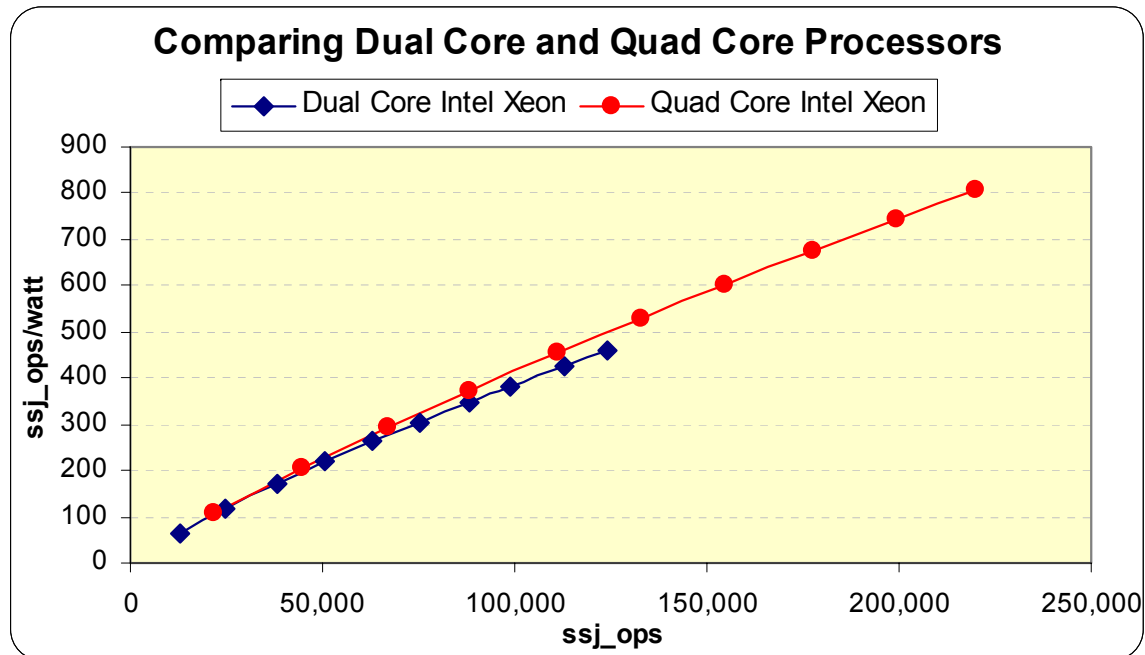
- Power Less by

0 to 3% less

- Your Results dependent on JVM and options



Processor scaling



Dual to Quad Core (Intel Xeon)	% increase
SSJ_OPS@100%	77%
Power@100%	1%
Overall SSJ_OPS/Watt	73%

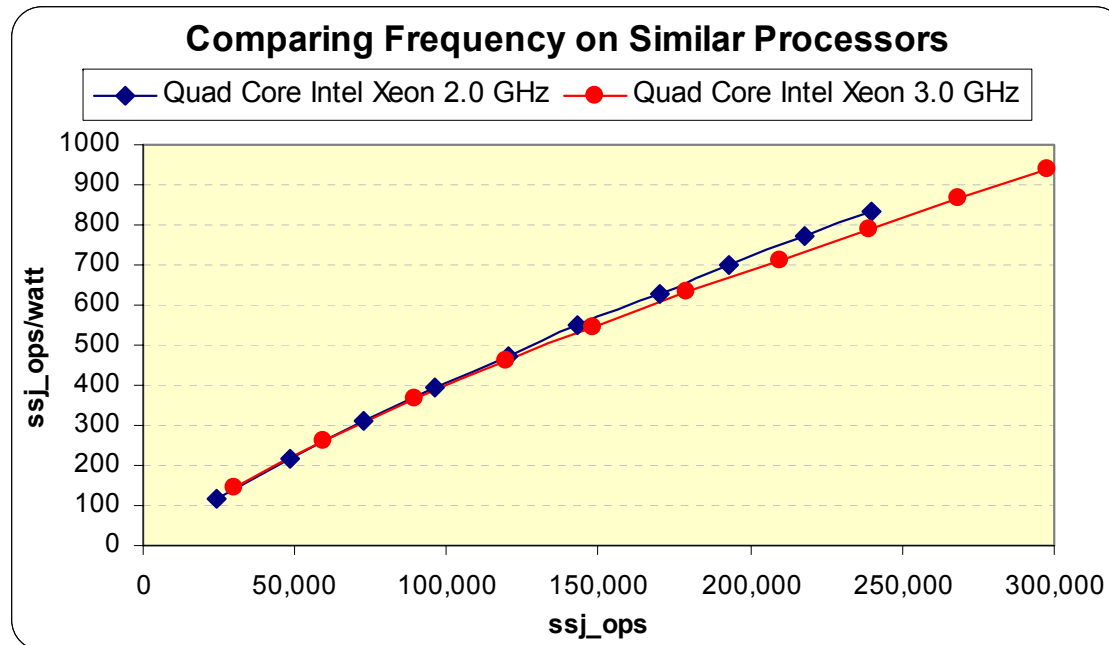
Dual Core Intel Xeon → Quad Core Intel Xeon:

(2.0GHz / 4MBL2)

(2.0GHz 2x4MBL2)

- SSJ_OPS@100% increased by ~77%
- Similar power@100%
- Overall SSJ_OPS/Watt improved by ~73%

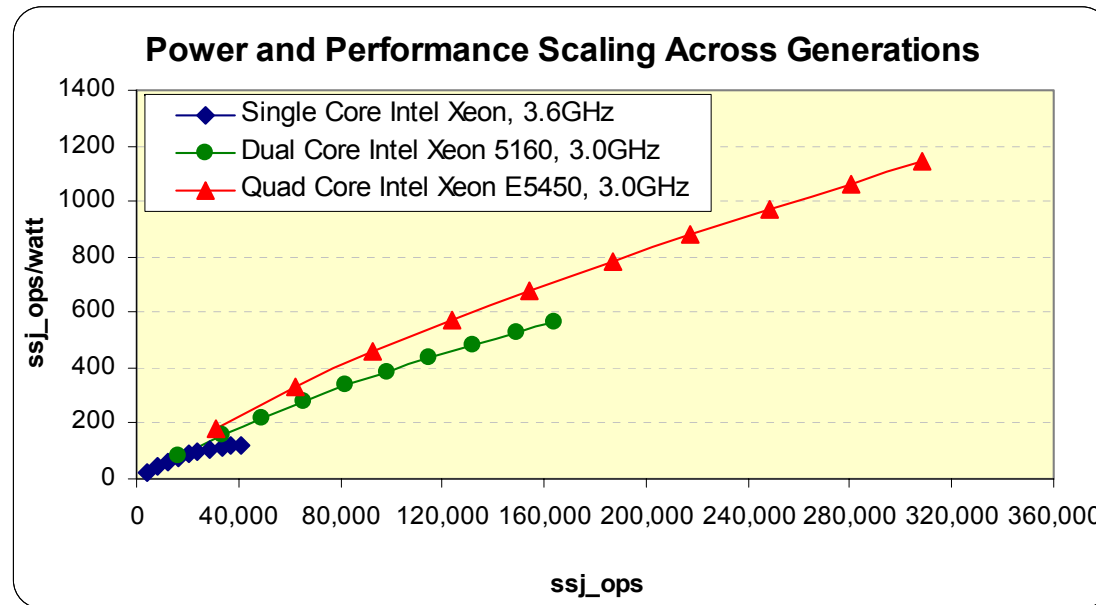
Frequency scaling



Quad Core Intel Xeon	% increase
2GHz-->3GHz	50%
SSJ OPS@100%	24%
Power@100%	10%
Overall SSJ OPS/Watt	16%

- 2.0 to 3.0 GHz Quad Core Intel Xeon (2x6MBL2):
- Frequency increase of 50%
 - SSJ OPS@100% increases by ~24%
 - Power@100% increased by ~10%
 - Overall SSJ OPS/Watt improved by ~16%

Platform generation scaling



- Quad Core Intel Xeon 3.0GHz versus Single Core Intel Xeon 3.6GHz:
 - SSJ_OPS@100% is ~7.5x, Power@100% less by ~20%
 - Overall SSJ_OPS/Watt is ~8.0x

Announced	Processor	Performance ssj_ops@100%	Power watts@100%	Overall ssj_ops/watt
2004	Single Core Intel Xeon, 3.6GHz	40,852	336	87.4
2006	Dual Core Intel Xeon 5160, 3.0GHz	163,768	291	338
2008	Quad Core Intel Xeon E5450, 3.0GHz	308,022	269	698
	Improvement	654%	-20%	702%

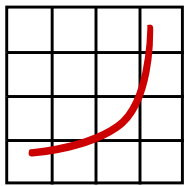
All data as of 30 Jan, 2008 from SPEC published results at: http://www.spec.org/power_ssj2008/results/power_ssj2008.html

General Observations

- CPU Utilization follows the load line (architecture dependent)
- % Time in C1 State – Inverse of CPU %
 - C1 Transitions per second highest at idle
- Memory % Committed – constant across load line
- Disk I/O – Regular bursts of ~140K byte writes,
 - ~3.3K bytes/sec for all load levels
- Network I/O - ~1.5K Bytes/sec, ~constant across load line
- Basic system events require more investigation
- Benchmark metric and other data do effectively show scaling with frequency, cores and across platform generations

Summary

- Results are specific to the platform and OS measured, etc
- SPEC FDR contains unprecedented amount of data
- Some system resources track graduated loads
- Benchmark metric and load level data fairly reflect configuration and OS settings changes
- Next Steps
 - We are just getting started.
 - First look, more refinements required
 - More measurements planned for in-depth characterization



spec