Delivering Quality in Software Performance and Scalability Testing

Khun Ban, Robert Scott, Kingsum Chow, Huijun Yan
Software and Services Group, Intel Corporation

Pacific Northwest Software Quality Conference 2011
Outline

- Introduction
- Modern Test Setup
- Issues to Avoid
- Summary

This is not a presentation, it is a discussion. We need your participation.
Introduction
Why is Load / Scalability Testing Important?

Twitter is over capacity.
Please wait a moment and try again. For more information, check out Twitter Status »

Screen captured from www.twitter.com
Problem Statement

- Prevent problem from the previous page is HARD
  - This is why we all have seen pages similar to this so often
- Enterprise solutions are designed to support large number of users
  - Hundreds of thousands, some even millions
  - Generating the load to simulate all these users requires many systems
- Problem may be outside of application software
  - May even be outside the application server
Challenges

• Software application needs to handle a large number of users.

• Software testing environment needs to generate the load of a large number of users.
Methodology
Top Down Performance Analysis

Top down performance analysis works well for Java* applications
Software Optimization Flow

1. Receive fixed workload (benchmark, customer use-case)
2. Optimize for the workload
3. Gather Performance Data (CPU, network, disk IO, etc.)
4. Analyze Data Identify Issues
5. Generate Alternatives (different instructions, algorithms, parallelization)
6. Implement Enhancement
7. Test Result
8. Integrate changes into product
Tools

• Load Generator
  • Faban is an open source benchmark harness and a driver development framework.

• System Performance Data
  • Intel® VTune™ Amplifier XE, complete system profiling
  • sar, vmstat, mpstat, iostat, netstat
  • Iperf and Uperf are great and easy tools for testing network infrastructure

• JVM Performance Console such as JRockit Mission Control
  • Execution Profiling, Lock Profiling, Latency Analysis
  • Garbage collection information, Memory/Object Profiling

• JEE Container Management Console
  • Extensive monitoring capabilities can be use for monitoring thread and connection pool sizes, cache hit rates, services deployed, etc.
Collecting Performance Data

• Establish performance baseline
  • Knowing how a healthy system behaves is critical
  • Collect performance data regularly for all the systems involved
    • Overall system performance is only as fast as the weakest link
  • Baseline data helps consultants or support staff to quickly isolate performance problems
    • By having baseline data, it is easier to identify issues

• Data to collect for every experiment
  • CPU usage, Context switches, Disk IO, Network traffic (both incoming and outgoing), Application server log files, JVM output with garbage collection information, AWR report for the database server
Modern Test Setup
Modern Server Layout

- Multiple sockets, many cores, even more threads
- Large caches (>MB), really large main memory (>TB)
  - Memory architecture on latest platform are almost always NUMA
- Greener, much more energy optimized compared to older system
  - Dynamically adjusts operating speed depending on load
Faster Systems Require Faster I/O

Limited HDD Performance results in I/O bottleneck

Normalized CPU Performance
Normalized Media Access Time for 20K Read

CPU performance scaling = 175x

HDD performance scaling = 1.3X since 1/96

How Do You Fill The Gap?

You Could Add Many Spindles!

OR You Could Add A Few SSDs!

SSDs close the Gap between CPUs and HDDs

Source: Intel measurements
Network Choices

- **1 GbE**
  - Cheap
  - Requires many ports to meet higher bandwidth demand of modern workload such as video streaming

- **10 GbE (and faster)**
  - Easy to upgrade
  - Many of hybrid switches available (1/10 GbE)
    - 1 GbE and 10 GbE working together, upgrade only the system requires the additional bandwidth
  - Newer cards support SR-IOV (good for virtualization)
  - Some support RDMA apps over Ethernet (single-digit µs latency)

- **Infiniband**
  - Ultra-low latency (low single-digit µs)
  - Highest cost
  - Require additional work, and hardware such as gateway switches, to integrate with existing Ethernet infrastructure
Large Setup, Many Components

- Easy to make mistakes in the configuration
- Easy to misdiagnose the root cause
- Controlled environment is essential
  - Private network
    - Need to know the real network behavior at all times
    - It is easy to make a mistake in network configuration
      - No one likes to be blamed for bringing down the company network
  - Know exactly the application running on the system, clean system is best
    - Virus Scanner can interfere with testing
    - IT installed software almost always gets in the way
  - We are experimenting with Virtualization to improve setup speed and reproducibility
- Issues may be outside the software
  - Must capture performance data from all components involved
Client-side Issues
Client-tier Must Never be The Bottle-neck

- Monitor critical resources on all client systems
  - CPU usage, user and system (from all processors)
  - Context switches
  - IO-Wait
  - Network usage
- Client threads must not be blocked by resources on the client system
  - Blocked client threads can interfere with load testing of the application server
Issues That Can Block Client Threads

• IO-WAIT due to swapping is always bad
  – Never allocate Java heap larger than memory available on the system
  – If huge pages is allocated on the system, make sure JVM is using them

• Too many threads running on the system
  – Load generator create many threads to simulate different users
  – After some point, the overhead of context switching between threads can become the bottleneck
  – The runqueue length is best to be < 2X number of processor threads
    • Linux sar utility can be used to read
      – runq-sz - Run queue length (number of processes waiting for run time)
Issues That Can Block Client Threads Cont.

• **Network bandwidth**
  
  – Ensure adequate transmit/receive bandwidth
    
    • Monitor usage during the experiments to make sure it is not the bottleneck
    
    • Iperf and Uperf are great and easy to use tools for testing network infrastructure

• **Context Switches** – In general, context switches track throughput
  
  – If injection rate increase, and context switches on the client system remain about the same, then client system may be reaching capacity
Load Generator System Must be Well Tuned

- Help to keep the number of systems required by load generator low.
- Before the tuning
  - Average total CPU usage was less than 50%
  - First few processors are much more saturated
- After the tuning by balancing the interrupt handling
  - Average total CPU usage when up to over 60%
  - CPU usage are more balance
  - Able to generate over 30% more load using the same system, and no changes to the application server
Issues to Avoid
Average is not the complete picture

- **Average CPU Usage for the application server:**
  - User 17.92%
  - System 9.25%
  - Seem healthy

- **CPU 0 is 99% saturated**
  - CPU 0 was the bottle-neck
  - Performance was greatly improvement by dividing the work evenly among all the CPU
### Case Study: Throughput Scaling

<table>
<thead>
<tr>
<th></th>
<th>Baseline</th>
<th>Test Case</th>
</tr>
</thead>
<tbody>
<tr>
<td>Throughput</td>
<td>1.00</td>
<td>1.08</td>
</tr>
<tr>
<td>waitForThinRelease</td>
<td>0.76%</td>
<td>3.78%</td>
</tr>
</tbody>
</table>

- Percentage of time spend (CPU clock cycles) on locks jumps ~5X for an 8% increase in throughput
  - Captured by Intel® VTune™ Amplifier XE
  - Possible solution ➔ Use multiple JVMs ➔ Each JVM performs less work, but more total combined work
    - Load generator generally allows multiple agents to be started

*Watch out for locks*
Case Study: 1-JVM vs. 2-JVM on 32-bit OS

<table>
<thead>
<tr>
<th>System Counters</th>
<th>1-JVM</th>
<th>2-JVM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Performance Metric (Normalized to 1 JVM)</td>
<td>1.00</td>
<td>1.24</td>
</tr>
<tr>
<td>CPU Usage (Normalized to 1 JVM)</td>
<td>1.00</td>
<td>1.06</td>
</tr>
<tr>
<td>Average GC Pause Time (ms)</td>
<td>361</td>
<td>247</td>
</tr>
<tr>
<td>Average Resident Memory Size (MB)</td>
<td>1,169</td>
<td>680</td>
</tr>
<tr>
<td>Average Time between GC (sec)</td>
<td>2.58</td>
<td>5.54</td>
</tr>
<tr>
<td>Percentage of Time Spent on GC</td>
<td>14.01%</td>
<td>4.47%</td>
</tr>
</tbody>
</table>

- 2-JVM setup performs more efficiently
  - 24% increase in throughput with only 6% increase in CPU usage
- 2-JVM setup spends less time doing garbage collection

Using multiple JVMs improves performance

Configuration
- 32-bit JVM on 32-bit OS on 2-socket system with 24 GB of memory
- Java heap size = 2.7GB per JVM
- With 2-JVMs each JVM is bound to a socket using taskset
- Performance counters are averaged over 1 hour during steady state
Case Study: 1-JVM vs. 2-JVM on 64-bit OS

Using multiple JVMs improves performance on 64-bit

### 64-bit JVM on 64-bit Linux (EL 5.3)

<table>
<thead>
<tr>
<th>Performance Metric (Normalized to 1 JVM)</th>
<th>1-JVM</th>
<th>2-JVM</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU Usage (Normalized to 1 JVM)</td>
<td>1.00</td>
<td>1.14</td>
</tr>
<tr>
<td>Percentage of Time Spent on GC</td>
<td>6.3%</td>
<td>4.7%</td>
</tr>
</tbody>
</table>

Configuration

- 64-bit JVM on 64-bit OS on 2-socket system with 24 GB of memory, Java heap size = 20 GB for 1-JVM, heap size = 10 GB for each JVM for 2-JVM setup (total 20 GB)
- With 2-JVMs each JVM is bound to a socket using numactl
- Performance counters are averaged over 1 hour during steady state

Increase in throughput comes from decrease in locks and increase in locality
Summary

• Today’s test environment are large and complicated
• Modern servers are much faster than before
  – The challenges in load and scalability testing have also increased
• Newer technologies like 10GbE and SSD can be used to help keep the environment simple
• Proper tools and best practices are essential to effective load and scalability testing
Optimization Notice

Intel® compilers, associated libraries and associated development tools may include or utilize options that optimize for instruction sets that are available in both Intel® and non-Intel microprocessors (for example SIMD instruction sets), but do not optimize equally for non-Intel microprocessors. In addition, certain compiler options for Intel compilers, including some that are not specific to Intel micro-architecture, are reserved for Intel microprocessors. For a detailed description of Intel compiler options, including the instruction sets and specific microprocessors they implicate, please refer to the “Intel® Compiler User and Reference Guides” under “Compiler Options.” Many library routines that are part of Intel® compiler products are more highly optimized for Intel microprocessors than for other microprocessors. While the compilers and libraries in Intel® compiler products offer optimizations for both Intel and Intel-compatible microprocessors, depending on the options you select, your code and other factors, you likely will get extra performance on Intel microprocessors.

Intel® compilers, associated libraries and associated development tools may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include Intel® Streaming SIMD Extensions 2 (Intel® SSE2), Intel® Streaming SIMD Extensions 3 (Intel® SSE3), and Supplemental Streaming SIMD Extensions 3 (Intel® SSSE3) instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors.

While Intel believes our compilers and libraries are excellent choices to assist in obtaining the best performance on Intel® and non-Intel microprocessors, Intel recommends that you evaluate other compilers and libraries to determine which best meet your requirements. We hope to win your business by striving to offer the best performance of any compiler or library; please let us know if you find we do not.

Notice revision #20101101
Legal Disclaimer

• INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL® PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL’S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER, AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL® PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT. INTEL PRODUCTS ARE NOT INTENDED FOR USE IN MEDICAL, LIFE SAVING, OR LIFE SUSTAINING APPLICATIONS.

• Intel may make changes to specifications and product descriptions at any time, without notice.
• All products, dates, and figures specified are preliminary based on current expectations, and are subject to change without notice.
• Intel, processors, chipsets, and desktop boards may contain design defects or errors known as errata, which may cause the product to deviate from published specifications. Current characterized errata are available on request.
• Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark* and MobileMark*, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products.
• Intel processor numbers are not a measure of performance. Processor numbers differentiate features within each processor family, not across different processor families. Go to: http://www.intel.com/products/processor_number
• Intel product plans in this presentation do not constitute Intel plan of record product roadmaps. Please contact your Intel representative to obtain Intel’s current plan of record product roadmaps.
• Intel, Xeon, Sponsors of Tomorrow and the Intel logo are trademarks of Intel Corporation in the United States and other countries.
• *Other names and brands may be claimed as the property of others.
• Copyright ©2011 Intel Corporation.