Bench4Q: A QoS-Oriented E-Commerce Benchmark Tool

ISCAS

2010-9-25
Contents

- TPC-W
- Bench4Q
- What we have done for OW2 contest
Background

- E-commerce
  - a very popular business model
- Benchmark
  - a typical and impartial way to evaluate the performance of e-commerce system
  - E.g. TPC-W
TPC-W

- An on-line book store
- Emulate Browser
  - 14 kinds of web interaction
  - 3 kinds of mix
  - Think time
- Performance metric
  - WIPS web interactions processed per second
  - WIRT web interaction response time
<table>
<thead>
<tr>
<th>Web Interaction</th>
<th>Browsing mix</th>
<th>Shopping mix</th>
<th>Ordering mix</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Browse-related</strong></td>
<td>95.00</td>
<td>80.00</td>
<td>50.00</td>
</tr>
<tr>
<td>Home</td>
<td>29.00</td>
<td>16.00</td>
<td>9.12</td>
</tr>
<tr>
<td>New products</td>
<td>11.00</td>
<td>5.00</td>
<td>0.46</td>
</tr>
<tr>
<td>Best sellers</td>
<td>11.00</td>
<td>5.00</td>
<td>0.46</td>
</tr>
<tr>
<td>Product detail</td>
<td>21.00</td>
<td>17.00</td>
<td>12.35</td>
</tr>
<tr>
<td>Search request</td>
<td>12.00</td>
<td>20.00</td>
<td>14.53</td>
</tr>
<tr>
<td>Search result</td>
<td>11.00</td>
<td>17.00</td>
<td>13.08</td>
</tr>
<tr>
<td><strong>Order-related</strong></td>
<td><strong>5.00</strong></td>
<td><strong>20.00</strong></td>
<td><strong>50.00</strong></td>
</tr>
<tr>
<td>Shopping cart</td>
<td>2.00</td>
<td>11.60</td>
<td>13.53</td>
</tr>
<tr>
<td>Registration</td>
<td>0.82</td>
<td>3.00</td>
<td>12.86</td>
</tr>
<tr>
<td>Buy request</td>
<td>0.75</td>
<td>2.60</td>
<td>12.73</td>
</tr>
<tr>
<td>Buy confirm</td>
<td>0.69</td>
<td>1.20</td>
<td>10.18</td>
</tr>
<tr>
<td>Order inquiry</td>
<td>0.30</td>
<td>0.75</td>
<td>0.25</td>
</tr>
<tr>
<td>Order display</td>
<td>0.25</td>
<td>0.66</td>
<td>0.22</td>
</tr>
<tr>
<td>Admin request</td>
<td>0.10</td>
<td>0.10</td>
<td>0.12</td>
</tr>
<tr>
<td>Admin confirm</td>
<td>0.09</td>
<td>0.09</td>
<td>0.11</td>
</tr>
</tbody>
</table>
Bench4Q vs TPC-W

- Bench4Q implements and extends TPC-W benchmark to a QoS-oriented benchmark
  - Open load simulation
  - QoS-aware users
  - More business metrics
Open mode
Latency Tolerance

- Latency Tolerance measures the time a customer will wait for a response before change his behavior.

- E.g. for some important interactions, customers usually would like to wait longer than some unimportant interactions.
Latency tolerance and tenacity

### Definition of tenacity for online behavior

<table>
<thead>
<tr>
<th>Class</th>
<th>$\mu$</th>
<th>Interactions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Important</td>
<td>12</td>
<td>Buy Request, Buy Confirm, Admin Request, Admin Confirm</td>
</tr>
<tr>
<td>Less important</td>
<td>10</td>
<td>Shopping cart, Registration</td>
</tr>
<tr>
<td>Unimportant</td>
<td>8</td>
<td>Others</td>
</tr>
</tbody>
</table>
Business metrics

- Metrics of TPC-W
  - WIPS
  - WIRT
- Is that enough to describe performances?
- let’s do an experiment!
Preliminaries

- In the same conditions, including hardware and OS, WIPS can be improved by optimizing parameters below in Tomcat.
  - sessionTimeout
  - connectionTimeout
  - acceptCount

- Environment
  - DBMSs: IBM DB2 V95
  - OSs: Microsoft Windows 2003 server
  - HTTP Servers: Tomcat 6.x
  - CPU: P4 2.8G * 2
  - Memory: 1G
  - EB: 500 and no think time
Normal VS Optimized

A: normal

B: unrealistic optimized
Illusion

- Obviously, B is better than A.
- Vendor could improve the performance by modifying the parameters when the other conditions are the same.
- And users may choose B because of high performance.
Seeing is believing?

A: normal

B: unrealistic optimized
Real phenomenon

- It’s surprised that A, may be, is better than B, because the completed sessions in A almost 11000, but which in B just 10000.
QoS metrics
# Bench4Q vs TPC-W

<table>
<thead>
<tr>
<th>Load Simulation</th>
<th>TPC-W</th>
<th>Bench4Q</th>
</tr>
</thead>
<tbody>
<tr>
<td>simulation mode</td>
<td>Close Request-based</td>
<td>Open Session-based</td>
</tr>
<tr>
<td>Load Fluctuation Control</td>
<td>Uncontrolled</td>
<td>Muti–agent distributed typical load fluctuations simulated</td>
</tr>
<tr>
<td>User behavior</td>
<td>Think time</td>
<td>Think time &amp; tolerance</td>
</tr>
</tbody>
</table>

## Metrics Analysis

<table>
<thead>
<tr>
<th>performance metrics</th>
<th>WIPS</th>
<th>WIRT</th>
</tr>
</thead>
<tbody>
<tr>
<td>QoS metrics</td>
<td>none</td>
<td>SPS(Session per second) Profit session</td>
</tr>
</tbody>
</table>

- **Break session dependency**
- **More flexible workload simulation**
- **More realistic workload**
- **More effective metrics**
Bench4Q tool
Architecture of Bench4Q Tool for large scale workload
What we have done for OW2 contest

- Monitor the resource of servers
  - CPU usage
  - Free memory
  - Disk write & read
  - Network sent & received
- Workload recording
- Result comparison
Resource Monitor

- **ServerMon**
  - Server monitor
  - One node

- **ClusterMon**
  - Cluster monitor
  - One leader several nodes
ServerMon

Diagram:

Client Host
- Bench4Q Tool (Java)
- RMI

Server Host
- Adapter (Java)
- Native Code (C++)
- Native Code (C++)
- PDH API
- /proc filesystem
- Windows
- Linux
- Hardware
ClusterMon

Client

Get Monitoring Data via RMI

Server Cluster

Leader

Subordinates

Notification via IP Multicast

Monitoring Data via UDP
Workload recording

- Workload are generated randomly
- Compare the performance of different servers

Replay the workload
Result comparison
Welcome

http://forge.ow2.org/projects/jaspte

http://www.trustie.com/projects/project/show/Bench4Q
Questions?
Thank you!