Bench4Q 1.0

Table of contents

1. Project intruduction ..... - 1 -
    1.1 Bench4Q: A QoS Oriented Benchmark ..... - 1 -
        1.1.1 What is TPC-W? ..... - 1 -
        1.1.2 What is Bench4Q? ..... - 2 -
    1.2 Bench4Q Tool: A implementation of Bench4Q ..... - 3 -
        1.2.1 RBE ..... - 3 -
        1.2.2 SUT ..... - 4 -
        1.2.3 Database ..... - 4 -
        1.2.4 Bench4Q Tool License ..... - 4 -
        1.2.5 Downloading Bench4Q ..... - 5 -
2.Bench4Q tool usage ..... - 5 -
    2.1 Installation ..... - 5 -
    2.2 Requirements ..... - 5 -
    2.3 How do I start The Grinder ..... - 6 -
    2.4 Console and Agent conmmunication Configuration ..... - 6 -
        2.4.1 agent ..... - 6 -
        2.4.2 console ..... - 7 -
    2.5 Test Configurations ..... - 7 -
        2.5.1 RBE Type ..... - 7 -
        2.5.2 Mix ..... - 7 -
        2.5.3 Warmup and Cooldown ..... - 8 -
        2.5.4 Load Fluctuation Control ..... - 8 -
        2.5.5 Think Time ..... - 8 -
        2.5.6 Tolerance Time ..... - 9 -
    2.6 QoS Metrics Analyzer ..... - 10 -

1. Project introductions

1.1 Bench4Q: A QoS Oriented Benchmark

Bench4Q is a QoS oriented benchmark for JEE Middleware. It makes many extensions of TPC-W, especially for load simulation and metrics analysis of a benchmark. We have also implemented a Bench4Q Tool to assist to use Bench4Q.

1.1.1 What is TPC-W?

TPC Benchmark™ W (TPC-W) is a transactional web e-Commerce benchmark, introduced by the Transaction Processing Performance Council. TPC-W specifies an e-Commerce workload that simulates the activities of a retail website which produces heavy load on the backend database.
The workload is performed in a controlled internet commerce environment that simulates the activities of a business oriented transactional web server. The workload exercises a breadth of system components associated with such environments, which are characterized by:

- ✓ Multiple on-line browser sessions.
- ✓ Dynamic page generation with database access and update.
- ✓ Consistent web objects.
- ✓ The simultaneous execution of multiple transaction types that span a breadth of complexity.
- ✓ On-line transaction execution modes.
- ✓ Databases consisting of many tables with a wide variety of sizes, attributes, and relationships.
- ✓ Transaction integrity (ACID properties).
- ✓ Contention on data access and update.

The performance metric reported by TPC-W is the number of web interactions processed per second. Multiple web interactions are used to simulate the activity of a retail store, and each interaction is subject to a response time constraint.
TPC-W simulates three different profiles by varying the ratio of browse to buy: primarily shopping (WIPS), browsing (WIPSb) and web-based ordering (WIPSo). The primary metrics are the WIPS rate, the associated price per WIPS ($/WIPS), and the availability date of the priced configuration.
Users of the TPC-W benchmark can browse and order products from the website. In the case of TPC-W the products are books. The expected introduction or Home page is the first page user will see. It includes the company logo, promotional items and navigation options to the top best selling books, a list of new books, search pages, client's shopping cart, and order status pages. User can browse pages containing a list

of new or best selling books grouped by subject, or perform searches against all books based upon a title, author or subject. A product page will give you detailed information for the book along with a picture of the book's front cover. User may order books by entering the order web pages. If user is a new customer s/he will have to fill out a customer registration page while for returning customers their information will be retrieved from the database and filled in automatically for them. User can change the quantity of the order or delete a book from the shopping cart. When user wishes to buy, s/he enters credit card information and submits the order. The system will obtain credit card authorization from a Payment Gateway Emulator (PGE), and present the user with an order confirmation page. At a later date user can view the status of his/her last order. Two additional web pages are provided for the system administrator to change book's front cover picture and price. This change is reflected in the new product book list. Essentially, the TPC-W benchmark provides basic functionality required of an Internet e-Commerce website.

TPC-W benchmark specifies the application database schema, the set of 14 web interactions (web pages), and how their execution manipulates application data. It also specifies data consistency and transactional requirements, database scaling and population, and various other aspects, e.g., which web interactions must be covered by SSL. However, TPC-W neither provides sample implementation of the application, nor it prescribes any implementation methodic or limits it to a specific technology. Besides the application structure, TPC-W specifies all aspects of the workload profile, performance metrics of interest, and required structure of the official disclosure report.

1.1.2 What is Bench4Q?

1.1.2.1 Overview

Benchmarks have been the effective measures to evaluate performances of middleware products. Such as ECperf(subsequently called SPECjAppServer) has ever been the most famous means to select or tune J2EE application servers. However, the current benchmark specification, which is called TPC-W, is not competent to be the B2C benchmark. The reasons were as follows:

1) TPC-W can’t simulate Internet customers, which are unlimited, unrelated to each other and sensitive to QoS;
2) TPC-W mainly concerns performance metrics such as hit rate and response times, but incapable of analyzing QoS metrics of B2C business, such as session related metrics.

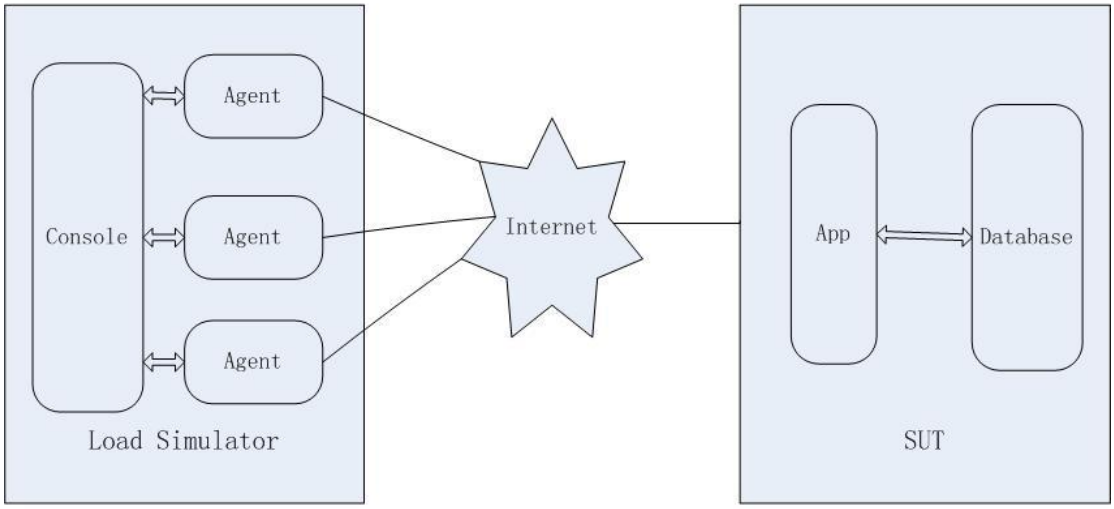
Bench4Q has made many extensions of TPC-W specification for Internetware, especially of load simulation and metrics analysis.

1.1.2.2 Key features

- ✓ It is distributed. Bench4Q Tool makes it easy to coordinate and monitor the activity of agents(which generate load) across a network from a central console.
- ✓ It has two test mode, open and closed. The open mode simulates a new customer when needed and can simulate unlimited customers. While the close mode can only simulate limited customers, a new customer can be simulated only when an old customer has been destroyed.
- ✓ Web tier relies on the session handling provided by the Servlet Container.
- ✓ It is fully J2EE-compliant and packaged; additional application server-specific deployment descriptors (DD) are provided for the Application Server.
- ✓ Database population utility is provided.

1.2 Bench4Q Tool: A implementation of Bench4Q

Bench4Q tool is divided into three parts.



1.2.1 RBE

RBE composed of two parts. The console configures the test, collate and display result. The agent is the real work part. It fallows the console's instrument. It generates load as the console configured.

For heavy duty testing, you can start an agent on each of several load injector machines. The stats of agent can be controlled using the console. There is little reason to run more than one agent on each load injector, but you can if you wish.

1.2.2 SUT

SUT (System Under Test) contains the business logic to simulate the tested scenario.
The SUT comprises all components which are part of the “application” being simulated. This includes network connections, Web Servers, application servers, database servers, etc.
The SUT of TPC-W is an online bookstore, which is a typical B2C application. It simulates 14 business operations, 8 of which involving database interactive.
There is a distribution of the TPC-W Benchmark Java Implementation originated of PHARM at the University of Wisconsin - Madison. Bench4Q Tool’s SUT is based on this distribution.
The main features of this version are listed as follows:
1. This version implements TPC-W spec version 1.8.
2. All VARCHAR database rows are limited to size <=255.
3. In the ITEMS table, I_THUMBNAIL and I_IMAGE are varchars, keeping the name of the image file.
4. There are 1000 item images and thumbnails required. You can download a pregenerated zip file, or generate them yourself. See ImageGen\USAGE.TXT file for further instructions.
5. No Payment Getaway Emulator is used (clause 2.7.3.3), so the CC_XACTS credit card processing record is created right away during the order creation.
6. No SSL is used, all connections are insecure.

1.2.3 Database

DB2 is used in this version.

1.2.4 Bench4Q Tool License

Bench4Q Tool is distributed according to the GNU Lesser General Public License.
Bench4Q Tool is a free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or any later version.
This source code is distributed "as is" in the hope that it will be useful. It comes with no warranty, and no author or distributor accepts any responsibility for the consequences of its use.
This version is a based on the implementation of TPC-W from University of Wisconsin - Madison. See COPYRIGHT file of "COPYRIGHT TPC-W in Java distribution" for license and copyright details.
This version used some source code of The Grinder. See COPYRIGHT file of "COPYRIGHT The Grinder" for license and copyright details.
Redistribution and use in source and binary forms, with or without modification, are

permitted provided that the following conditions are met:

1) Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2) Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3) Neither the names of the copyright holders nor the names of the contributors may be used to endorse or promote products derived from this software without specific prior written permission.

Bench4Q Tool includes PicoContainer (http://picocontainer.codehaus.org/). See the file LICENSE-PicoContainer for license and copyright details.

1.2.5 Downloading Bench4Q

Bench4Q is available on the Internet at http://forge.ow2.org/projects/jaspte. You can find latest version there.
Bench4Q is distributed as three zip files which you should expand using unzip, WinZip (http://www.winzip.com/) or similar. Everything required to run The Bench4Q is in the zip file labelled grinder-version.zip. The remaining files that are needed to build The Grinder are distributed in the zip file labelled grinder-version-src.zip.

2.Bench4Q tool usage

2.1 Requirements

As Bench4Q tool is written in Java, to run Bench4Q tool, a Java Virtual Machine (JVM) is needed.
A IBM DB2 and the jdbc driver are needed.
A JavaEE Application Server is needed.

2.2 Installation

1) Everything required to run The Bench4Q is in the zip file labelled grinder-version.zip. all you need to do is unpack the package.
2) Then you need to install a IBM DB2 and a JavaEE Application Server.
3) Initiate the database using DB2 Generator. First you need to create a database, and then configure the db.properties file. Then run the DBPopulate.jar file. This process may take several hours.
4) Deploy the SUT on your JavaEE Application Server.The web application looks for datasource using JNDI, so you need to configure database.properties file in

WEBAPPROOT\WEB-INF\classes\org\bench4Q\servlet.

2.3 How do I start Bench4Q

It's easy:

1. Create a bench4Q.properties file. This file specifies general control information (how the worker processes should contact the console).
2. Set your CLASSPATH to include the bench4Q.jar file which can be found in the lib directory.

```
java org.bench4Q.Console
```

3. Start the console on one of the test machines:

```
java org.bench4Q. Agent
```

4. For each test machine, do steps 1 and 2 and start an agent process:
5. The agent will look for the bench4Q.properties file in the local directory. If you like, you can also specify an explicit properties file as the first argument. For example:
6. The console does not read the bench4Q.properties file. It has its own options dialog (choose the help/Options menu option) which you should use to set the

```
java org.bench4Q.Agent propertieFile
```

communication addresses and ports to match those in the bench4Q.properties files.

7. The console process controls can be used to trigger agent to start test. Each agent process then generate load as the console configured.
8. After the test done, you can collect result of agent, then some picture will be showed at the console.

2.4 Console and Agent conmmunication Configuration

2.4.1 agent

In bench4Q.properties file, the address of console is configured as follows:

```
bench4Q.consoleHost=133.133.133.123
bench4Q.consolePort=6372
```

The agent will look for the bench4Q.properties file, and connect console using this address.

2.4.2 console

You can choose the help/Options menu option to configure the address of console.

2.5 Test Configurations

2.5.1 RBE Type

There are three RBE types.

- ✓ Closed means test is fully closed.
- ✓ EB Open means starting some EB every interval.
- ✓ Full Open means starting some request every interval. Interval is measured in second.

2.5.2 Mix

The percentage of each web interaction executed during each measured interval must following the mix as defined in following table.

Web Interaction	Browing mix（WIPSb）	Shopping mix（WIPS）	Ordering mix（WIPSo）
Brown	**95.00**	**80.00**	**50.00**
Home	29.00	16.00	9.12
New products	11.00	5.00	0.46
Best sellers	11.00	5.00	0.46
Product detail	21.00	17.00	12.35
Search request	12.00	20.00	14.53
Search result	11.00	17.00	13.08
Order	**5.00**	**20.00**	**50.00**
Shopping cart	2.00	11.60	13.53
Registration	0.82	3.00	12.86
Buy request	0.75	2.60	12.73
Buy confirm	0.69	1.20	10.18
Order inquiry	0.30	0.75	0.25
Order display	0.25	0.66	0.22
Adm.request	0.10	0.10	0.12
Adm.confirm	0.09	0.09	0.11

2.5.3 Warmup and Cooldown

Warm up: Seconds used to warm-up the Agents.
Cool down: Seconds used to cool-down the Agents.

2.5.4 Load Fluctuation Control

The load fluctuation represents the concurrency feature of business and it has great effects on the choice of tuning policy of a B2C server. Bench4Q simulate such load fluctuation by accumulating the simulated loads from multiple Load Workers. Each Load Worker simulates a controlled load by the following parameters:

- ✓ Base Load: Fixed amount of EBs the Load Worker will be generated in each period during the test.
- ✓ Radom Load: Random amount of EBs will be generated with the bLoad.
- ✓ Rate: The rate of base load changes. It can be positive, negative or zero. It's positive means the base load is increasing every second. It's negative means the base load is decreasing every second. While the rate is zero, the load is fixed.
- ✓ Trigger Time: The time for the Load Worker to start generate its load.
- ✓ Duration: The duration for the Load Worker to generate its load.

By composite the simulated loads from multiple Load Works, the following typical load fluctuations of B2C businesses can be simulated.

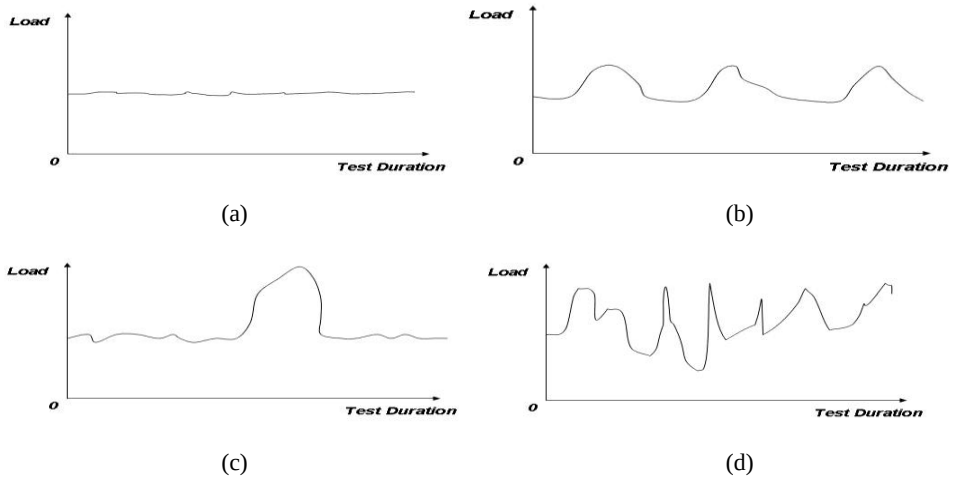


Fig 6. Some typical load fluctuations of B2C

2.5.5 Think Time

Think time multiplication.
Used to increase (>1.0) or decrease (<1.0) think time. In addition to slow-down factor.

2.5.6 Tolerance Time

Patience, customer tenacity and importance of transaction affect the behavior of a B2C customer. Correspondingly, we extend the TPC-W by simulating QoS sensitive behavior of B2C customers as follows:

Latency Tolerance (LT) measures the time a customer will wait for a response before change his behavior (for example, give up the ongoing session). Once the request processing time is over its LT, the Load Simulator will discard the current session. Human behavior across populations tends to follow a given distribution. Statistic data [12] indicates that, for an e-commerce Web site, less and less online customers will tolerate a lengthened LT. Hence, we use the right-skewed distribution to simulate the distribution of LT which characterizes the patience of online customer.

Tenacity (T) measures of determination of a B2C customer in B2C businesses. Customers may have a low latency tolerance, but if their desired business operations are important to them, their LTs will be lengthened accordingly. As shown in Table 1, we categorize the 14 interactions in TPC-W into three classes based on their importance. The profitable and administration related interactions are more important than the others.

The probability density function of the right-skewed distribution for latency tolerance is defined as follows:

$$f(x;\mu,\sigma_1,\sigma_2)=\begin{cases}\dfrac{1}{\sqrt{2\pi}\sigma_1}e^{-\frac{(x-\mu)^2}{2\sigma_1^2}}, & -\infty < x \le \mu \\ \dfrac{1}{\sqrt{2\pi}\sigma_2}e^{-\frac{(x-\mu)^2}{2\sigma_2^2}}, & \mu < x < \infty\end{cases}$$

where is the location parameter, and are scale parameters, is larger than . is the function which generates random LT, according to , before starting new interaction requests. As shown in Figure. 7 and Table1, tenacity is characterized by and correlated to interaction importance. Interaction with higher importance class is specified with a larger , which means the customer will be more patient when waiting for important responses.

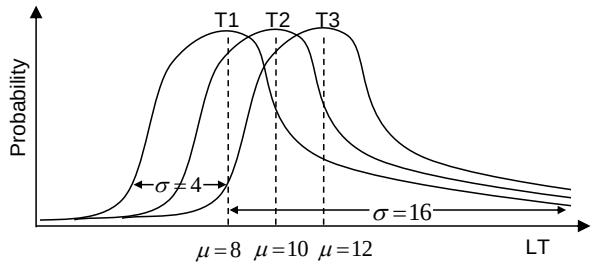


Table 1. Definition of tenacity for online behavior

Class	μ	Interactions
Important	12	Buy Request, Buy Confirm, Admin Request, Admin Confirm

Less important	10	Shopping cart, Registration
Unimportant	8	Others

2.6 QoS Metrics Analyzer

QoS Metrics Analyzer of Bench4Q used to analyze the metrics, especially QoS metrics of the test. The metrics as follows:

- Performance metrics: WIPS, WIRT.
- QoS metrics: Total Sessions, Failure Sessions, Profit Sessions, Failure Profit Sessions, Requests lead to failure sessions.