SENG 533 Project - Performance Evaluation of a Web-based Application

Section A: Using Measurement Data to Compute Model Parameters

1. Project Objectives

You are the official performance analysis team in your company. Your company has asked your team to evaluate the performance of an online bookstore application of one of their clients, bookzilla.ca. Specific tasks assigned to you are as follows:

a) Bookzilla.ca has provided you with **some performance data from a test** installation that closely resembles their production bookstore installation. They would like you to analyze this data to obtain some insights on system performance, e.g., determining bottleneck resources limiting performance. You would also like to exploit the data to obtain parameters that can be used for the tasks outlined in b). (Section A of the project)

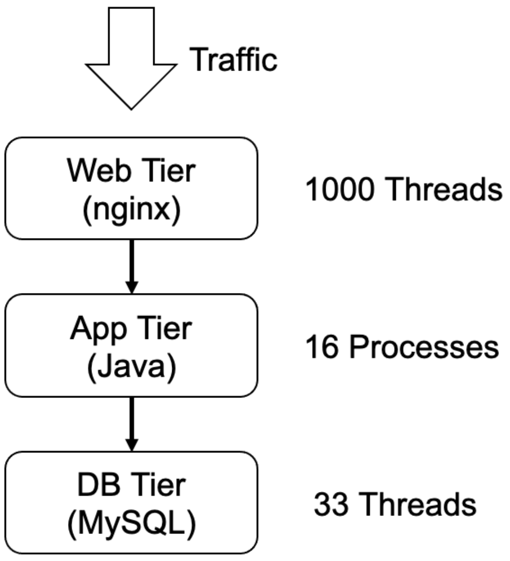
b) Bookzilla.ca is planning a major expansion. The business expects to host a lot more books and serve a lot more user traffic. They would like you to provide them estimates of extra resources needed to handle this increase in workload while satisfying mean response time targets for their customers. As we learned in class, this kind of exercise is known as capacity planning. It typically involves **using queuing models** to predict the performance of a system under alternative system configurations and workloads. (Section B of the project)

You have agreed upon the following general methodology to tackle this project as a team. First, you will analyze performance data from the test installation and determine several systems and workload parameters. Next, you will build a performance model that will take these parameters as input and provide performance predictions. This model will be used to answer various "what-if" capacity planning questions. For example, the model will be used to estimate the number and type of additional servers Bookzilla needs to purchase as part of its expansion so that its customers can continue to experience fast response times.

This document, i.e., Section A, deals with the measurement data analysis. The modeling aspect of your project, i.e., Section B, will be described in a separate document later.

2. Bookstore Testbed

As mentioned previously, you are provided measurements from Bookzilla's internal Windows 3-tier application testbed. The bookstore application has a typical 3-tiered architecture. The **Web tier** consists of a Web server process. The Web server process has *1000 concurrent listener threads* that service customer requests. The **application tier** consists of *16 concurrent application server processes* that service requests from the Web tier. The **database tier** has a *database management system process with 33 threads* handling incoming requests from the application tier.



Bookzilla performance test engineers submitted to the bookstore application a synthetic workload that they believe represents their actual customers' behavior. They used the open-source *httperf* Web workload generator for this purpose. *httperf* generates customer sessions defined in a session file as per a specified mean session inter-arrival time and session inter-arrival time distribution. Session arrivals are "open" – the arrival of a session does not depend on the number of sessions already present in the system. Each session generated behaves as a "closed" user. It submits a series of HTTP 1.1 requests. After receiving a response from the server for a request, each session "thinks" for a certain time before submitting its next request to the server. The think times can be specified in the session file input to *httperf* and can be chosen to match a desired *think time distribution*, e.g., the think time distribution of real Bookzilla customers.

The Web and application servers were running on a dual-processor Windows box (Web/App server machine) with one physical disk. The database server was running on a separate Windows box (DB server machine) with one CPU and one physical disk. *httperf* executed on a separate Linux client machine. The client machine and the server machines were connected by a non-blocking fast Ethernet switch which provided a dedicated 100 Mbps duplex connection between any two machines in the setup. Bookzilla engineers tell you that this ensured that the network was not the bottleneck for the study. Since *networking delays are negligible*, they tell you that the response times measured by *httperf* are likely to be very close to the server response times.

Several tests were performed using the synthetic workload. During each test, performance metrics were collected from the server nodes and *httperf*. The server performance metrics were obtained by running Windows *PerfMon* program. *PerfMon* sampled the system every 1 minute to gather the metrics of interest.

3. Measurement Data

Bookzilla test engineers have given you two sets of data entitled "**Current DB**" and "**Big DB**". Both results were obtained for the same synthetic workload used in performance tests. Current DB represents a configuration where there are 1,000 books in the database. This mirrors the current size of Bookzilla's catalog. Big DB represents a configuration with 100,000 books in the database, which mirrors Bookzilla's expansion plan.

Each set has four files. The "**perfmon-data**" file contains information logged by Windows *PerfMon*. The "**httperf-summary-output**" file contains the summary of statistics, e.g., mean reply time and throughput, provided by *httperf* at the end of a test run. The "**httperf-detailed-output**" file contains the detailed, per-request statistics logged by *httperf* during a performance test run. This file contains information such as the time at which a request was completed, the id of the customer session to which the request belongs, the time taken to establish an HTTP connection with the server, the time elapsed between the submission of a request to receiving the first byte of the response, the time elapsed between receiving the first byte of the response and the last byte of the response, the size of data transferred, and the number of concurrent customer sessions seen by the server when the request was serviced. The "concurrent-sessions-histogram" file provides a histogram for the number of concurrent sessions seen at the server during the performance test run.

Bookzilla test engineers have given you information to interpret the "**perfmon-data**" files. Any counter that begins with "\\isp-01" pertains to the computer which executes the Web and Application server. Other counters pertain to the computer which executes the database server. Counters containing the word "db2syscs" pertain to the database management system process. As mentioned previously, this process has 33 concurrent threads. Counters containing the word "wHTTPg" pertain to the Web server process. As mentioned previously, this process has 1000 concurrent threads. "server#0","server#1"…."server#15" are the 16 instances of the application server process. The first column of a "**perfmon-data**" file represents the timestamp (in seconds since epoch time) at which performance data samples were collected. You may need to consult the documentation of windows *perfmon* (or simply run "*perfmon*" on a Windows computer) to understand the exact meanings of the various counters.

4. Analyzing *perfmon* data

a) What is the difference between "system" and "user" CPU utilization as reported by *perfmon*?

b) Since you are interested in bottleneck identification and model parameter calculation, compute the following: (do this for both Current DB and Big DB)

1) The duration of the performance test
2) The frequency of sampling performance data (in seconds)
3) The mean utilization of the CPUs and disks in the system
4) The mean CPU utilization caused by the Web server process, app server processes, and the database management system process.

c) Bookzilla test engineers have told you that there is very little virtual memory activity in their systems and that you need not worry about this factor during performance evaluation. Based on the *perfmon* data, do you agree with this assessment? Provide concrete reasons for your view.

d) Do you agree with the thread/process concurrency information provided by Bookzilla for the Web, application, and database servers? Provide a justification based on the perfmon data.

e) You will observe a slight discrepancy between what you computed in 4.b.4 and 4.b.3. For example, although the database management system process was the only process using the DB machine, its CPU utilization (computed in 4.b.4) is less than that of the CPU utilization of the DB machine computed in 4.b.3. Provide possible explanations for such mismatches.

f) Based on your analysis in a), which resource is likely to be the bottleneck in Bookzilla's current 1,000 books setup? Will the bottleneck shift with their planned expansion to 100,000 books?

5. Analyzing *httperf* data

You will need metrics marked in bold to parameterize the performance model used in Section B.

a) Let us now focus on application-level metrics such as throughput and response time. Compute the following for both Current DB and Big DB:

1) The per-request mean response time is the sum of the time to establish a connection with the server, wait till the first byte of the response, and ultimately obtain the last byte of the response.

2) The throughput in request completions/second.

3) The mean think time between successive requests from a customer

4) The mean number of concurrent customer sessions in the system. (Hint: You need to use Little's law for this)

b) Bookzilla's test engineers have told you that the network was lightly utilized and that it can be ignored as a factor in your study. Is there any data available to back up this claim?

c) From the analysis in a), discuss the implications of supporting a larger catalog of books on the experience of an end-user of Bookzilla.

6. Computing resource demands for use in a performance model

Do the computations asked for both Current DB and Big DB.

You will need metrics marked in bold to parameterize the performance model used in Section B.

a) As discussed in class, the mean resource demands placed by a customer request on various system resources such as CPUs and disks are essential for building performance models. Apply the utilization law to compute the mean demands placed by request on the following resources:

1) Web/App machine CPUs

2) DB machine CPU

3) Web/App machine disk

4) DB machine disk

b) Although computing the overall resource demand is useful, we can build more detailed models if we know the average demands placed on a resource by the Web, Application, and

Database server processes while they satisfy a request. Using the utilization from 4.b.4, compute the following:

1) Mean per-request demand placed by Web server process on Web/App server machine's CPUs

2) Mean per-request demand placed by App server process (all 16 of them put together) on Web/App server machine's CPUs

3) Mean per-request demand placed by database management system process on DB server machine's CPU.

c) You will observe a slight mismatch between the total demand you calculated for a resource in 6.a and the sum of the demands placed on that resource by processes using that resource (6.b). Explain the reason for this mismatch.

d) Compare the resource demands you computed for the Current DB and Big DB scenarios. Discuss reasons for any significant differences that you observe. Discuss whether these demands provide us any insights on the kind of additional resources needed to satisfy the planned expansion of Bookzilla.

Miscellaneous

1) Ensure that you don't get stuck for an extended period of time. Don't hesitate to seek help from me or the TA during the lab hours: Wednesdays from 8 AM to 9:50 AM.

2) Feel free to write scripts/macros that automate and simplify your analysis.

3) Due on Feb 28. Submit through D2L Dropbox.