Web Applications Engineering:
Performance Analysis: Operational Laws

Service Oriented Computing Group, CSE, UNSW

Week 11

Material in these Lecture Notes is derived from:
- The Art of Computer System Performance Analysis by Raj Jain, Wiley Press
- Performance Testing Guidance for Web Applications: Patterns & Practice by Meier, et. al, Microsoft Corporation
Performance Analysis and Evaluation

Performance analysis as seen by many

- Run testing tools
- ???
- Profit!
Performance Analysis and Evaluation

Performance analysis and evaluation is a continuous process. It must be carried out throughout the lifecycle of a web application including design, development, deployment, maintenance and upgrade.

The steps and activities involved in this process are:

- Defining performance goals and target system
- Selecting appropriate metrics and factors affecting performance
- Creating a model for the workload and the outcomes
- Designing tests to evaluate the actual performance
- Analysing the results of the tests.
- Repeating the above steps.

These activities are collectively known as **Performance Engineering**
Defining goals

There are many questions we need to answer before we deploy a web application, e.g.:

- What is the workload going to be? - number of users, number of requests per second, number of transactions, etc.
- What is the availability for the application? - uptime

Question like these determine the design of the application, the number and type of servers needed to run it, and the bandwidth required to connect it to the Internet.

However, there is no set formula for defining goals as these vary hugely between applications.
Performance Metrics

- **Response time** - the time it takes a system to react to a human request. Also called *Round-trip Time (RTT)*. RTT depends on the processing time at the client, the time taken for the request and the reply to travel through the network, and the time involved in processing the request at the server. A browser incurs at least 3 RTTs: DNS Name Resolution, TCP Connection Setup, and HTTP Request-Response.

- **Throughput** - The rate at which requests are completed from a computer system is called throughput. Measured in operations per unit time - e.g. Requests per second, Sessions per second, Page views per second or transactions per second. Under increasing load, throughput can reach a maximum and plateau or even decrease - *thrashing*.
Performance Metrics

- **Availability** - fraction of time the system is up and available to its customers - also called its *uptime*.
  
  Q: A system with 99.99% would be unavailable for how much time in 30 days?
  
  Availability is affected by failures and overloads.
  
  Availability is also a parameter in system design.

- **Reliability** - is the probability that a system is going to function properly and continuously in a fixed period of time.
  
  Reliability over a large period of time tends to availability.

- **Resource Utilization** - The amount of resources consumed by an application, expressed as a percentage of resources available.
  
  Resources can be of any type: CPU, storage, database, bandwidth, etc.

Other non-functional requirements are: security and extensibility.
Performance Metrics

- Performance objectives are cast in terms of metrics. e.g.
  - 99.999% availability during Nov-Dec
  - Max. response time of 10 ms for the home page of a website
  - Min. throughput of 2 transactions per second.

- Such goals are called *Service Level Objectives (SLOs)*

- Service providers and consumers agree on certain minimum set of SLOs. These are called *Service Level Agreements (SLAs)*

- Violating SLAs can result in penalties for the providers
Modelling

Model - An abstracted view of a real system.

A model needs to contain only those components that determine a particular function under observation. Models are used since building an actual prototype may be infeasible or impossible, given the current knowledge.

The modelling process involves:

- Develop a workload model that approximates the actual workload
- Develop an analytical model that predicts performance of the system
- Run the models under different scenarios to forecast system performance
Queueing Networks - a theoretical framework for performance analysis

Requests wait in a queue the server to be processed. Two key parameters for any queue are the average number of requests that arrive at the queue in a single time period (arrival rate), and the average time taken to process each request (service demand).
Modelling Requests

- **Workload Model** - A representation that mimics the real workload of a web application.

- Useful for constructing a likely scenario for usage of the web application.

- Typical parameters of a workload model: request arrival times, request size, service demands, execution mix

- Each component of the system - database, application server, file system - would be described by different workload models

- All requests are not alike either - different requests may produce different loads on the system.

- Request distributions - Power law model (Zipf’s Law for static content), Bursty, Self-Similar
Modelling Requests

Constructing Workload Models
Some sources...

- Parsing Logs to get metrics such as page views per period, user sessions per period, session duration, page request distribution, interaction speed, user abandonment.
- Observation of utilization through system utilities (Nagios, top, sysperfinfo (MS SQL), etc.)
- Observation of request arrivals at the load balancer
- Experience with similar application
- Probabilities of navigation paths in the application
Modelling Requests

Queueing Classes

Grouping requests into distinct classes based on similarity of quantity of resources used to process them, workload intensity, and/or priority

E.g. Consider your own Music Store web application.

When to use multi-class models:

- Heterogeneous service demands - Requests can be clustered into groups with significantly different service demands.
- Different types of workloads - e.g. online transactions and batch workloads
- Different service level objectives - Different requests have different requirements e.g. response time for 1 class can be 100 ms while for another class can be 500 ms.
Modelling Requests

Classes

Open Class
- Workload intensity specified by an arrival rate
- Unbounded number of customers in the system
- Output is same as input i.e. arrival rate is equal to throughput

Closed Class
- Workload intensity specified by customer population
- Bounded number of customers in the system - number of requests being processed is an input parameter
- Throughput is an output parameter

Mixed Model - Some customer classes can be open and some can be closed.
Modelling Resources

Three types of resources are modeled in queues:

- **Load Independent (LI)** - constant service rate that does not depend on the load (i.e. number of requests in the queue).

- **Load Dependent (LD)** - service rate is a function of the load.

- **Delay** - has no queue, all requests are served immediately

![Diagram of Load Independent, Load Dependent, and Delay](image-url)
Using Modelling for Performance Analysis

**Analytical**

- Using mathematical formulas and/or computational algorithms to describe the behaviour of a system as a function of input workload parameters.

- Workloads are characterized as input parameters to mathematical functions whose outputs indicate the performance measures met.

- Computationally efficient and less expensive to construct.

- Less accurate and may not cover all of the system’s behaviours.
Using Modelling for Performance Analysis

Simulation

- Using programs to emulate different dynamic parts of a system.
- Workloads are derived by observing and measuring similar systems (traces), or are artificially generated by using probability distributions.
- Tests are run repeatedly and statistical measures are used to estimate the value of performance measures.
- Gives a good level of detail for the proposed system.
- Is expensive to build and needs validation so as to be dependable.

In both cases, run models repeatedly to measure average, best case and worst case performances with current assumptions. Validate their outcomes with actual performance evaluation.
Performance Testing

- **Performance Testing**: To determine the response time, scalability or availability of the system.
- **Load testing**: To verify application behaviour under different load conditions (normal and peak).
- **Stress testing**: To observe application behaviour under conditions of extreme load (beyond peak) and failure.
- **Capacity testing**: To determine the number of users that the system can support under current resource allocation. Also, how many additional resources are required to support larger number of users.

Tests are not only used for *validation* but also for *planning* and *comparison*. Testing should be performed in an *unbiased* manner.

The analyst should adopt a *systematic* approach by *framing* the objectives, devising a test plan to meet the objectives, and allowing different perspectives on the results obtained.
Performance Testing

How to Test?

The process behind testing is similar to that for modelling:

- Pick a workload distribution to suit the test.
- Construct actual requests corresponding to the distribution by using a tool (e.g. Apache JMeter)
- Execute the requests and monitor system response using a monitoring tool (e.g. psi-probe)
- Collect and analyse results
Analysis

**Operational Analysis** - Establishing relationships among quantities based on measured or known data about computer systems.

**Measured Quantities:**

- \( \tau \) - the length of observation period
- \( K \) - no. of resources
- \( B_i \) - tot. busy time of resource in \( \tau \)
- \( A_i \) - tot. no. of service requests (arrivals) at resource \( i \) in \( \tau \)
- \( A_0 \) - tot. no. of requests submitted in \( \tau \)
- \( C_i \) - tot. no. of service completions for resource \( i \) in \( \tau \)
- \( C_0 \) - total service completions in \( \tau \)
Operational Analysis

The following measures can be derived:

- Mean service time per completion, $S_i = \frac{B_i}{C_i}$
- Utilization of resource, $U_i = \frac{B_i}{\tau}$
- Throughput (completions per unit time), $X_i = \frac{C_i}{\tau}$
- Arrival rate, $\lambda_i = \frac{A_i}{\tau}$
- System Throughput, $X_0 = \frac{C_0}{\tau}$
- Average number of visits per request to resource, $V_i = \frac{C_i}{C_0}$

These measures are related to each other via a set of laws called operational laws.
Operational Laws

**Utilization Law** - Utilization of a resource is product of its throughput and average service time.

\[ U_i = S_i \times X_i \]

If number of completions in an interval equal the number of arrivals in that interval,

\[ U_i = S_i \times \lambda_i \]

If resource \( i \) has \( m \) servers (e.g. multiprocessor), the

\[ U_i = (S_i \times X_i)/m \]

If there are \( p = 1, 2, \ldots P \) classes of customers, then

\[ U_{i,p} = S_{i,p} \times X_{i,p} \]
**Operational Laws**

**Question 1**

The bandwidth of a communication link is 56,000 bits per sec and it is used to transmit 1500-byte packets that flow through the link at a rate of 3 packets per second. What is the utilization of the link?

\[ X_i = 3 \text{ packets per second} \]
\[ S_i = \text{Time taken for one packet} \]
\[ = \frac{1500\text{bytes/sec} \times 8\text{bits/byte}}{56,000\text{ bits per sec}} \]
\[ = \frac{12,000}{56,000} = 0.214 \text{ sec} \]
\[ U_i = 0.214 \times 3 = 64.2\% \]
Operational Laws

Question 1

The bandwidth of a communication link is 56,000 bits per sec and it is used to transmit 1500-byte packets that flow through the link at a rate of 3 packets per second. What is the utilization of the link?

\( X_i = 3 \) packets per second

\( S_i = \) Time taken for one packet

\( = (1500 \text{ bytes/sec} \times 8 \text{ bits/byte}) / 56,000 \text{ bits per sec} \)

\( = 12,000/56,000 = 0.214 \text{ sec} \)

\( U_i = 0.214 \times 3 = 64.2\% \)
Little’s Law - the average number of requests in a resource (queue length) is equal to average time per request times throughput of the resource i.e.

\[ Q_i = R_i \times X_i \]

where \( Q_i \) is number of requests in resource \( i \) and \( R_i \) is its response time.

The only assumption for Little’s Law is that no requests are created or destroyed inside the system.

Little’s law can be used to examine any “black box” and holds for subsystems as well.
Consider a database system with 1 CPU and 3 disks. Assume all transactions have similar resource demands, and that the transaction load is constant. Measurements give a load of 13,680 transactions in 1 hour and the average number of 16 transactions in execution per sec. What is the response time of database transactions in 1 sec?

\[ X = \frac{13,680}{3,600} = 3.8 \text{ transactions per second} \]

\[ N = 16 \text{ Number of concurrent transactions} \]

\[ R = \frac{N}{X} = 4.2 \text{ sec} \]
Operational Laws

Question 2

Consider a database system with 1 CPU and 3 disks. Assume all transactions have similar resource demands, and that the transaction load is constant. Measurements give a load of 13,680 transaction in 1 hour and the average number of 16 transactions in execution per sec. What is the response time of database transactions in 1 sec?

\[ X = \frac{13,680}{3,600} = 3.8 \text{ transactions per second} \]
\[ N = 16 \text{ Number of concurrent transactions} \]
\[ R = \frac{N}{X} = 4.2 \text{ sec} \]
Operational Laws

**Forced Flow Law** - Throughput of a resource \( (X_i) \) is equal to the average number of visits \( (V_i) \) to that resource multiplied by the system throughput \( (X_0) \)

\[
X_i = V_i \times X_0
\]

E.g. A database system can perform 10 transactions per minute. Each transaction requires on an average 2 I/O operations on a dedicated disk. What is the throughput of the disk in I/O operations per minute?
**Operational Laws**

**Question 3**

For the database server in Question 2, we have measured the following for the disks.

<table>
<thead>
<tr>
<th>Disk</th>
<th>Reads/sec</th>
<th>Writes/sec</th>
<th>Total I/Os/sec</th>
<th>Utilization</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>24</td>
<td>8</td>
<td>32</td>
<td>0.30</td>
</tr>
<tr>
<td>2</td>
<td>28</td>
<td>8</td>
<td>36</td>
<td>0.41</td>
</tr>
<tr>
<td>3</td>
<td>40</td>
<td>10</td>
<td>50</td>
<td>0.54</td>
</tr>
</tbody>
</table>

What is the average number of I/Os on each disk?

\[ V_1 = \frac{X_1}{X_0} = \frac{32}{3.8} = 8.4 \text{ visits to the disk 1} \]

\[ V_2 = \frac{X_2}{X_0} = \frac{36}{3.8} = 9.5 \text{ visits to the disk 2} \]

\[ V_3 = \frac{X_3}{X_0} = \frac{50}{3.8} = 13.2 \text{ visits to the disk 3} \]
Operational Laws

Question 3

For the database server in Question 2, we have measured the following for the disks.

<table>
<thead>
<tr>
<th>Disk</th>
<th>Reads/sec</th>
<th>Writes/sec</th>
<th>Total I/Os/sec</th>
<th>Utilization</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>24</td>
<td>8</td>
<td>32</td>
<td>0.30</td>
</tr>
<tr>
<td>2</td>
<td>28</td>
<td>8</td>
<td>36</td>
<td>0.41</td>
</tr>
<tr>
<td>3</td>
<td>40</td>
<td>10</td>
<td>50</td>
<td>0.54</td>
</tr>
</tbody>
</table>

What is the average number of I/Os on each disk?

\[ V_1 = \frac{X_1}{X_0} = \frac{32}{3.8} = 8.4 \text{ visits to the disk 1} \]
\[ V_2 = \frac{X_2}{X_0} = \frac{36}{3.8} = 9.5 \text{ visits to the disk 2} \]
\[ V_3 = \frac{X_3}{X_0} = \frac{50}{3.8} = 13.2 \text{ visits to the disk 3} \]
Operational Laws

Combining the forced flow law and the utilization law, we get the utilization of the \(i^{th}\) device,

\[
U_i = X_i \times S_i
\]

\[
\implies U_i = V_i \times X_0 \times S_i
\]

\[
U_i = X_0 \times D_i
\]

is known as the **Service Demand Law** where \(D_i = V_i \times S_i\) is called the *service demand* on resource \(i\).

The resource with the highest \(D_i\) has the highest utilization and is the *bottleneck resource* - or the key limiting factor in achieving higher throughput.
Operational Laws

**Question 4**

For the database server in Question 2, what are the service demands for each disk?

\[ D_1 = \frac{U_1}{X_0} = \frac{0.35}{3.8} = 0.092 \text{ sec/transaction} \]

\[ D_2 = \frac{U_2}{X_0} = \frac{0.30}{3.8} = 0.079 \text{ sec/transaction} \]

\[ D_3 = \frac{U_3}{X_0} = \frac{0.54}{3.8} = 0.142 \text{ sec/transaction} \]
Operational Laws

Question 4

For the database server in Question 2, what are the service demands for each disk?

\[ D_1 = \frac{U_1}{X_0} = \frac{0.35}{3.8} = 0.092 \text{ sec/transaction} \]
\[ D_2 = \frac{U_2}{X_0} = \frac{0.30}{3.8} = 0.079 \text{ sec/transaction} \]
\[ D_3 = \frac{U_3}{X_0} = \frac{0.54}{3.8} = 0.142 \text{ sec/transaction} \]
Operational Laws

**General Response Time Law** - total time spent by a request in a resource is the product of time per visit and the number of visits to the server; and the total time in the system is the sum of total times at each resource.

For a balanced system (arrival rate equals departure rate), the total number of requests in the system

\[ Q_0 = Q_1 + Q_2 + \ldots + Q_M \]

where \( M \) is the total number of resources in the system. From Little's Law, it follows that

\[ X_0 . R_0 = X_1 . R_1 + X_2 . R_2 + \ldots + X_M . R_M \]

Dividing both sides of the equation by \( X_0 \) and using the forced flow law, we get

\[ R_0 = R_1 . V_1 + R_2 . V_2 + \ldots + R_M . V_M \]
Operational Laws

Consider an interactive system where $N$ users generate requests that are serviced by one resource and the responses come back to the users. After a think time $Z$, the users submit the next request. Therefore, total cycle time for requests from one user is $R + Z$.

In time period $\tau$, a user generates about $\tau/(R + Z)$ requests. Therefore, system throughput

$$X = \frac{N(\tau/(R + Z))}{\tau}$$

$$= \frac{N}{R + Z}$$

$$R = \frac{N}{X} - Z$$

This relation is known as the interactive response time law

If 7,200 requests are processed during one hour by an interactive computer system with 40 clients and an average think time of 15 sec, the average response time is $R = \frac{40}{(7200/3600)} - 15 = 5$ sec
Operational Laws

Consider an interactive system where $N$ users generate requests that are serviced by one resource and the responses come back to the users. After a think time $Z$, the users submit the next request. Therefore, total cycle time for requests from one user is $R + Z$.

In time period $\tau$, a user generates about $\tau/(R + Z)$ requests. Therefore, system throughput

$$X = \frac{N(\tau/(R + Z))/\tau}{\tau} = \frac{N}{R + Z}$$

$$R = \frac{N}{X} - Z$$

This relation is known as the interactive response time law.

If 7,200 requests are processed during one hour by an interactive computer system with 40 clients and an average think time of 15 sec, the average response time is

$$R = \frac{40}{7200/3600} - 15 = 5 \text{ sec}$$
Bottleneck Analysis

Identifying the bottleneck device is the first step of any performance improvement project.

Assume in a system with $M$ resources, the $i^{th}$ resource is the bottleneck. Therefore,

$$D_i = \max(D_1 + D_2 + \ldots + D_M) = D_{\text{max}}$$

Then, when the utilization of system is highest (i.e. 1), from service demand law,

$$X_0 \leq \min\left\{\frac{1}{D_{\text{max}}}\right\}$$

This is the upper asymptotic bound of throughput under heavy load from Little’s Law, $Q_0 = R_0 \times X_0$. But, when there is no queueing, the response time must be at least equal to the sum of service demands on all resources. Therefore,

$$Q_0 \geq \sum_{i=1}^{M} D_i \times X_0$$
Bottleneck Analysis

when there are \( N \) users concurrently submitting 1 request each, \( Q_0 = N \). Therefore, we can re-arrange the previous equation as

\[
X_0 \leq N / \sum_{i=1}^{M} D_i X_0
\]

This is the *upper asymptotic bound of throughput under light load*. Therefore,

\[
X_0 \leq \min\left\{ \frac{1}{\max(D_i)}, \frac{N}{\sum_{i=1}^{M} D_i} \right\}
\]
A few statistical quantities

- **Mean** - Sum of $n$ observations divided by $n$
- **Median** - Sort $n$ observations from lowest to highest and pick the middle one
- **Percentile** - Sort $n$ observations from lowest to highest. Then, the $k^{th}$ percentile is the value below which $k\%$ of the observations are to be found.
- **Quartile** - The $25^{th}$ percentile is the first Quartile (Q1). The $50^{th}$ percentile or median is the second quartile (Q2). The $75^{th}$ percentile is the third quartile (Q3).