

# “A little Queuing Theory”

## A Very Short Course in Capacity Planning and Queuing Theory

Tom Farwell  
[www.tomfarwellconsulting.com](http://www.tomfarwellconsulting.com)

# Introduction

- This presentation introduces some key concepts on planning/sizing of computers and how queuing theory is the tool of choice for accurate analysis.
- Also presented are illustrations on why “linear thinking” is not correct in computer analysis.

# Part I

## A Little Capacity Planning

# Objectives

- Explain Capacity Planning
- Explain Device Saturation
- Discuss the Linear Thinking Fallacy
- Auto Rental Example
- Discuss the What-ifs
- Describe Bottlenecks
- Discuss Service Demands
- Explain why Capacity Planning is Required

# Capacity Planning

- Capacity planning is the process of predicting when future load levels will *saturate* a system.
- It is also used to determine the most cost-effective way of delaying system saturation as much as possible.

# Device Saturation

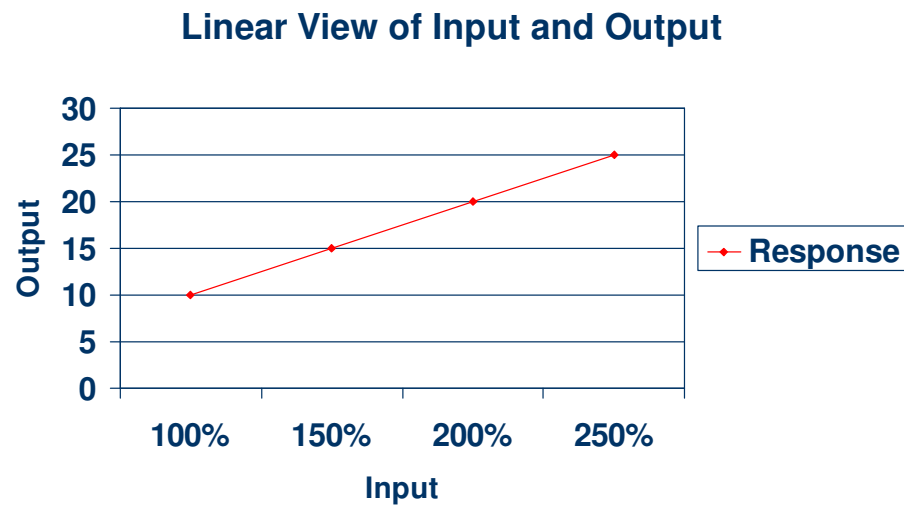
- The term saturation should not be inferred to indicate that a device (e.g., CPU, disk) reached 100 percent utilization, because service level agreements can be violated well before the 100 percent utilization mark.

# Linear Thinking Doesn't Apply

- Linear thinking is a common human process. This notion implies that if an input increases by 20 percent the output of the system changes by 20 percent.

# Linear Thinking Doesn't Apply

- Linear View of Input and Output



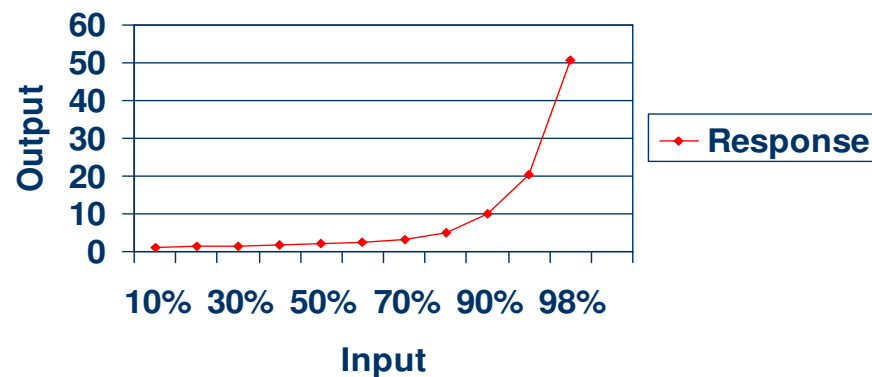
# Computer Response

- Computer response does not follow a linear curve. It is entirely possible that a change in computer input by 20 percent results in a change in output of hundreds of percent.

# Computer Response (continued)

- Computer response to increasing Input

Input versus Output (Utilization vs. Response Time)



# Auto Rental Example

- A car rental agency wants to grow the customer base of a C/S database system, whereby regional services will handle transactions.
- The average reservation rate was 6 transactions per second, during peak hours.
- 1800 customer service reps have client workstations to enter data.

## Example (continued)

- Reservations and road side assistance are the two types of transactions.
- Average response time for both types of transactions must not exceed three seconds.
- Management question—What happens to response time if load increases by 5 percent, 10 percent, and 15 percent?

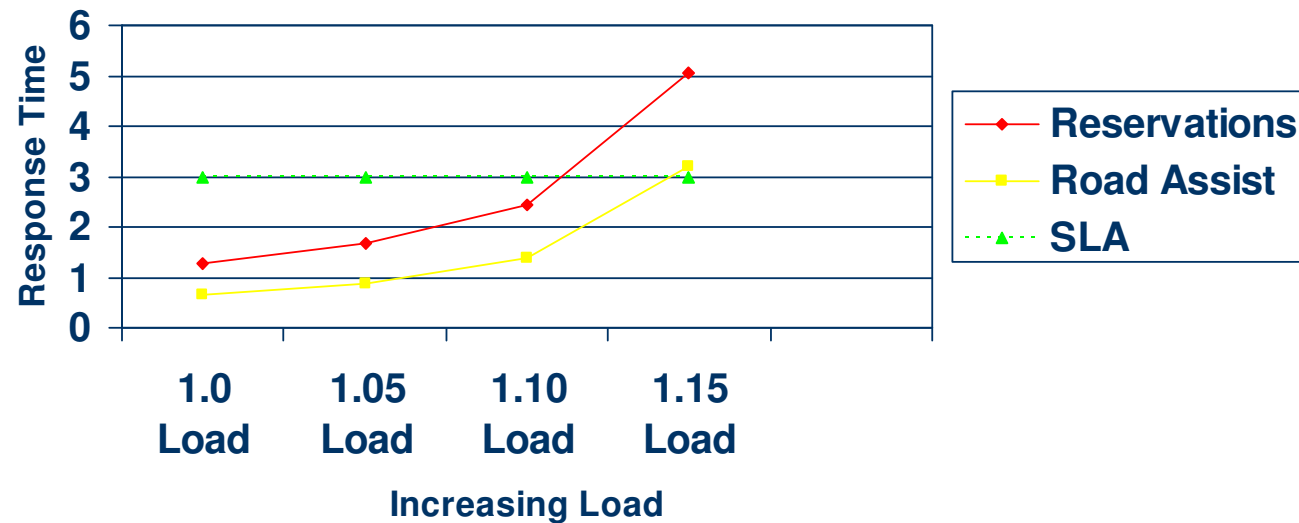
## Example (continued)

- A queuing theory model was used to determine current and future response times.

Transaction	Current Load	105 % Load	110 % Load	115 % Load
Reservation	1.28 sec	1.67 sec	2.45 sec	5.06 sec
Road Assistance	0.64 sec	0.87 sec	1.37 sec	3.20 sec

# Example (continued)

## Response Time vs Increasing Load



## Example (continued)

- So what does this example show?
  - Two things:
    - The 15 percent increase in workload increases the response time for reservations 395 percent and for road assistance a whopping 500 percent.
    - The 3-second SLA is violated for the reservation transaction with about a 111 percent load, and the roadside assistance with about a 114 percent load.

## Say What?

- So how can a 15 percent increase in workload translate into hundreds of percent increases in response time?
  - The answer is a bottleneck—this is the component where a transaction spends most of its time. Response time improvements are limited by the time spent at the bottleneck. Response time includes time to service the request and the time the request had to wait for the device.

# What are the “What ifs”?

- Capacity planning answers the various “what if” questions, some are:
  - What if we get a faster server?
  - What if we add more clients?
  - What if we improve buffer cache hit ratios?
  - What if we get faster disks?
  - What if we get a faster network?

# Bottlenecks are Everywhere

- Mathematically, a bottleneck is the device with the largest service demand.
- A service demand is calculated by the product of the number of times the device is visited in a transaction, times the service time of the device ( $D = V * S$ ). The service time is the amount of time needed for the device to service the request.

# All Systems Have Bottlenecks

- Computer systems have bottlenecks. The key to effective capacity planning is identifying the bottleneck and make decisions based on that information.

# Service Demand Example

- Suppose a database computer has a single CPU and a single disk.
- Also suppose that a database transaction requires 20 ms (0.020 sec) of CPU time and 4 disk I/Os. The average service time for the disk is 10 ms (0.010 sec).

# Service Demand Example

- What are the computed service demands?
  - $D_{\text{CPU}} = 1 * 0.020 = 0.020 \text{ sec}$
  - $D_{\text{disk}} = 4 * 0.010 = 0.040 \text{ sec}$
- So which device is the bottleneck?

# Throughput Saturation

- Throughput of a computer system is limited by the device with the largest service demand.
- The following expression relates throughput to service demand:
  - $X_{\text{sat}} = 1/D_{\text{max}}$

## Throughput Saturation (continued)

- What is the maximum throughput of this system?
  - $X_{\text{sat}} = 1/0.040 \text{ sec} = \underline{25 \text{ transactions per second}}$

# Benchmarks for Scaling

- Can one use benchmarks in an attempt to accurately scale from one computer system to another.
  - NO
    - At best, benchmarks can be used to relate “performance” of one computer to another in the same family, running the same benchmark program. However, erroneous results can occur when trying to scale computer performance using benchmarks, without knowledge of service demands and fundamental queuing theory.

# Capacity Planning is Required

- Some people have the erroneous belief that if their company has large financial resources, capacity issues can be solved by adding new hardware.
- The following are contradictions to this belief:

# Capacity Planning is Required

1. Poor performance may lead to customer dissatisfaction and damage the external image of the company, which results in lost revenue.
2. Solving the performance problem may take some time.
3. The poor performance may stem from a poorly designed architecture, in which case the solution is generally more difficult and time consuming.

## Part II

### A little Queuing Theory

# Objectives

- Explain Queuing Theory Basics
- Discuss the Early Derivations
- Describe Queuing Networks and the Mean Value Analysis

# A Little Queuing Theory

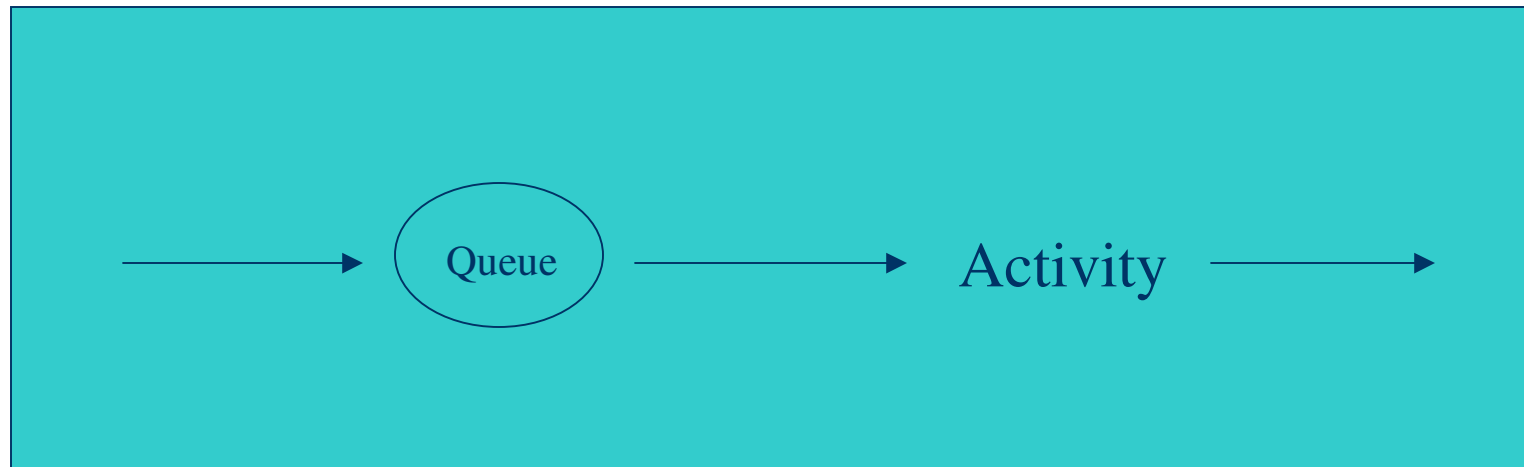
- Queuing theory deals with problems that involve queuing (or waiting). Some examples of queuing include:
  - Banks or supermarkets—customers waiting for service
  - Computers—users waiting for a response
  - Public transportation—riders waiting for a bus/train
  - Failure situations—machinery owners waiting for a failure to occur

## A Little Queuing Theory (cont)

- Queues form because resources are limited. In fact, it makes economic sense to have queues.
- In designing queuing systems, a balance is needed between service to customers (which means short queues and implies many servers) and cost (too many servers waste funds).

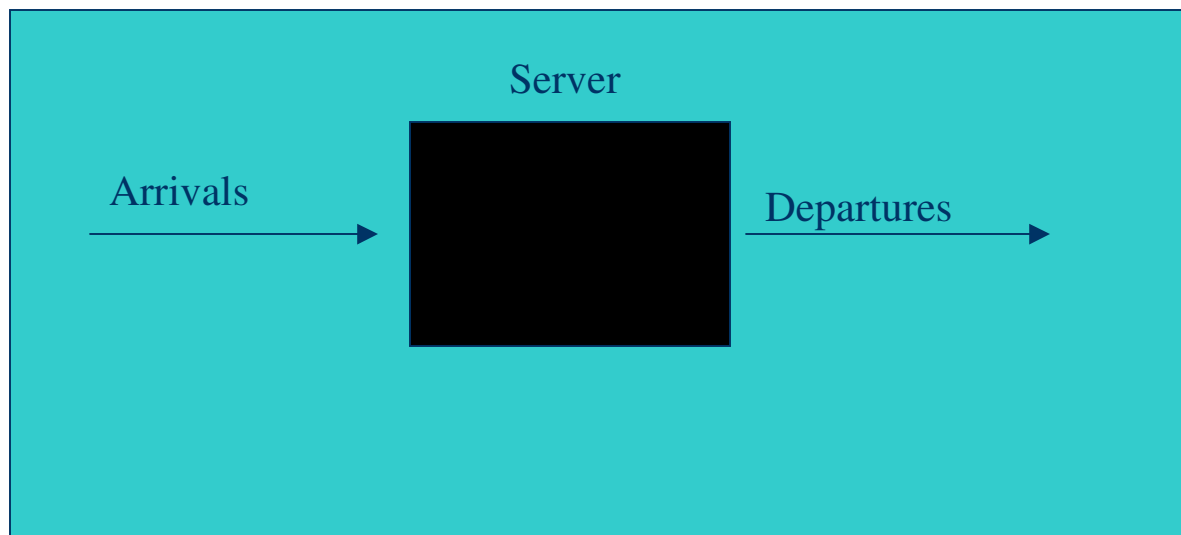
## A Little Queuing Theory (cont)

- Most queuing systems can be divided into individual sub-systems, consisting of *entities* queuing for some *activity*.



# A Little Queuing Theory (cont)

- Queuing theory applies to any system in equilibrium, as long as nothing in the black box is creating or destroying tasks (arrivals=departures).



# Very Complicated Mathematics

- Queuing theory mathematics gets very complicated because it applies probability and statistics to queuing systems.
  - What is the probability that the arriving task will find a device busy?
  - On average, how many tasks are ahead of the task that just entered the system?

# The Early Derivations

- A single server queue is a combination of a servicing facility that accommodates one customer at a time (server) + a waiting area (queue).
- These components together are called a *system*.

## The Early Derivations (continued)

- The early queuing work treated the system as a single homogeneous “server,” without regard to discrete components or types of workloads. These systems were called M/M/1 queues.
- Later, this work was expanded to include multiple homogeneous servers inside the black box.

# Queuing Networks

- After much more work in the queuing theory field (approximately 20 years), a new technique was developed that divided computer system into networks of queues.

# MVA

- This new technique is called Mean Value Analysis. It allowed a computer system to be segregated by workload classes (transactions, arrival rates, numbers of clients) as well as components (CPU, disk, etc.)
- Systems were also delineated as being “open” or “closed.”

## MVA (continued)

- Mean value analysis is an iterative approach of solving three primary equations for class “r” workload at queue “i.”
- The three equations provide solutions for the residence time (response time, per class, per queue), the throughput, and the queue length (number of class “r” tasks at queue “i”).

## MVA (continued)

- Software and hardware contention can be modeled using these techniques.

# Part III

Getting  
the  
Tricky  
Stuff

# Objectives

- Describe how to Obtain Model Parameters
- Discuss Model Calibration
- Discuss Model Prediction
- Methodology

# Obtaining Model Parameters

- Because model parameters, such as arrival rates, transaction rates, and I/Os per class are not easily measured with most analytical tools, inferences have to be made.

# Obtaining Model Parameters

- These inferences, and so called operational laws, allow the analyst to gather data and convert the appropriate values into service demands, workload classes, etc.

# Model Calibration

- Using real data such as current response times, utilization levels, etc., a queuing model to be calibrated to mimic real life data.

# Model Prediction

- After the model is calibrated and gives reasonable accuracy (within 10-30 percent of actual), the true power of the model comes to light. That is, using the analytical model to predict future response times, utilizations, queue lengths, etc.
- The “what if” questions now can be accurately answered.

## Methodology (from test environment)

1. Define workload classes
2. Capture CPU use and I/O per workload class
3. Capture device utilizations
4. Compute Service demands from test data
5. Determine number of concurrent connections allowed in DB
6. Insert data into closed, multiclass MVA analysis
7. Calculate utilizations, response times, etc.
8. Calibrate model
9. Use model for metric prediction

# Part IV

## Return On Investment

## Why Spend the Money for Analysis?

- Using the described capacity planning/queuing methods can save companies substantial monies in at least three areas:

## Why Spend the Money? (cont)

- Get the most out of the servers one already owns.
  - By computing the service demands and determining the bottlenecks, current performance issues may be resolved with little or no additional hardware.
    - Actual examples include spending resources to add hardware only to find out that the bottleneck moved with the new hardware. This results in lots of monies spent with little or no improvement. Know exactly what you need before you buy it.

## Why Spend the Money? (cont)

- Don't over buy when architecting new systems.
  - Quit guessing when designing new architecture. Know exactly where the bottlenecks exist so that hardware purchases only reflect what one actually needs. Why spend a \$1M for a 24-way server, only to find out the \$100K disk subsystem was the bottleneck.

## Why Spend the Money? (cont)

- Consolidate servers that are an inefficient use of the hardware they occupy.
  - Save monies on lease payments, service contracts, floor space, utilities, etc.

# Source Material

- D. Menasce, V. Almeida, *Capacity Planning for Web Services: Metrics, Models, and Methods*, Prentice Hall PTR, Upper Saddle River, NJ (2002).
- D. Menasce, V. Almeida, L. Dowdy, *Capacity Planning and Performance Modeling: From Mainframes to Client-Server Systems*, Prentice Hall PTR, Upper Saddle River, NJ (1994).