5.2 *Traffic Flow Analysis and Performance Evaluation*

5.2.1 Traffic Flow Analysis Objective

- Estimate:
  - Delay
  - Utilization of resources (links)

- Traffic flow across a network depends on:
  - Topology
  - Routing
  - Traffic workload (from all traffic sources)

- Desirable topology and routing are associated with:
  - Low delays
  - Reasonable link utilization (no bottlenecks)

- Assumptions:
  - Topology is fixed and stable
  - Links and routers are 100% reliable
  - Processing time at the routers is negligible
  - Capacity of all links is given $C = [C_i]$ (in bps [bits per second])
  - Traffic workload is given $\Gamma = [\gamma_{jk}]$ (in pps [packets per second])
  - Routing is given $R = [r_{jk}]$
  - Average packet size is $1/\mu$ bits.

5.2.2 Queuing Analysis

Projections of performance are made on the basis of either:

- The existing load information, or
- The estimated load for the new environment.

Approaches that could be used:

- Do an after-the-fact analysis based on actual values

- Make a simple projection from existing to expected environment

- Develop an analytic model based on queuing theory

- Program and run a simulation tool

5.2.2.1 Queuing Models

- The notation **X/Y/N** is used for queuing models.
  - X = distribution of the interarrival times
  - Y = distribution of service times
  - N = number of servers

- The most common distributions are:
  - G = general independent arrivals or service times
  - M = negative exponential distribution
  - D = deterministic arrivals or fixed length service

- Example: **M/M/1**

- **Single-server queues**

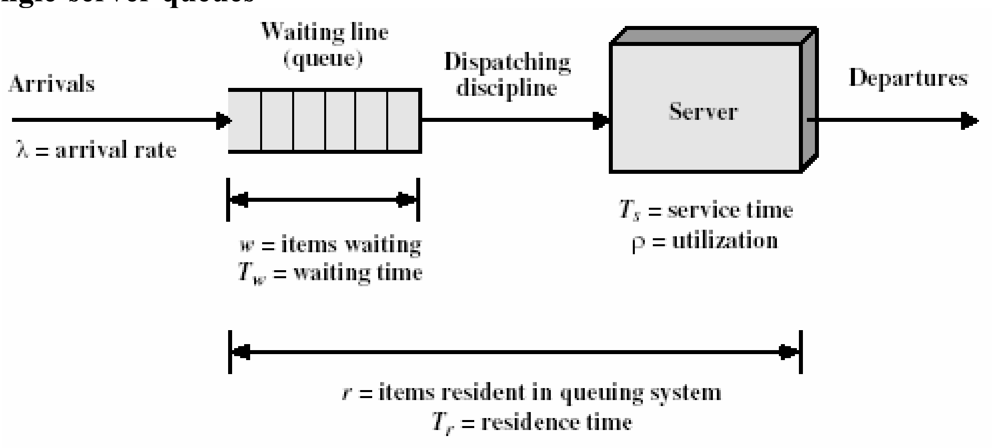

Figure 5.1: Queuing System Structure and Parameters for Single-Server Queue (Taken from "Queuing Analysis" by William Stallings)

<u>Queue parameters:</u>

λ = arrival rate; mean number of arrivals per second

T_s = mean service time for each arrival; amount of time being served, not counting time waiting in the queue

ρ = utilization; fraction of time facility (server or servers) is busy

r = mean number of items in system, waiting and being served (residence time)

T_r = mean time an item spends in system (residence time)

w = mean number of items waiting to be served

T_w = mean waiting time (including items that have to wait and items with waiting time = 0)

Basic Queuing relationship:

- $\boldsymbol{\rho = \lambda * T_s}$
- $r = w + \rho$
- $\lambda_{max} = 1/T_s$
- $\boldsymbol{r = \lambda * T_r}$ **(Little's formula)**
- $w = \lambda * T_w$ (Little's formula)
- $T_r = T_w + T_s$
- $\boldsymbol{r = \rho/(1-\rho)}$

- **Multiserver queue**

  N = number of servers

  ρ = utilization of each server

  $\mathbf{N\rho}$ = utilization of all servers (= $\boldsymbol{\lambda * T_s}$)

5.2.2.2 M/M/1 Queues – Application to Networks

- Each link is seen as a service station servicing packets.

  $\boldsymbol{\lambda_i}$ = arrival rate (in pps); mean number of packets that arrive to link i in one second.

  $\boldsymbol{\mu C_i}$ = average service rate (in pps); mean number of packets that will get out of the link i in one second. (= $\boldsymbol{1/T_s}$)

- Utilization of link i is:

  $\rho_i =$

- Stability condition of a network is:

- The external workload offered to the network is:

$\gamma =$

Where:

γ = total workload in packets per second

γ_{jk} = workload between source *j* and destination *k*

N = total number of sources and destinations

- The internal workload on link *i* is:

$\lambda_i =$

Where:

γ_{jk} = workload between source *j* and destination *k*

Π_{jk} = path followed by packets to go from source *j* and destination *k*

- The total internal workload is:

$\lambda =$

Where:

λ = total load on all of the links in the network

λ_i = load on link *i*

L = total number of links

- The average length for all paths is given by:

E[number of links in a path] = λ/γ

- The average number of items waiting and being served for link i is:

$r_i =$

- The number of packets waiting and being served in the network can be expressed as:

$$\gamma * T =$$

Where:

T = average delay experienced by a packet through the network.

$$T =$$

- T_{ri} is the residence time at each queue. If we assume that each queue can be treated as an independent M/M/1 model (Jackson's Theorem), then:

$$T_{ri} =$$

Where: T_{si} is the service time for link i

$$T_{si} =$$

Where:

- C_i = data rate on the link (in bps)
- **M** = $1/\mu$ = average packet length in bits

Example:

5.3 Network Reliability

5.3.1 Introduction

- A network model is a set of facilities. A facility could be a device or a link.
- A network must contain some slack to allow it to function even if some of its facilities have failed.
- Any network facility is either:
  - Working (**p**)
  - Failing (q = **1-p**)
- **MTBF:** Mean Time Between Failures (**f**).
- **MTTR:** Mean Time To Repair (**r**)

- For any facility i, we'll know from measurements of f_i and r_i:

  P_i = Prob [facility i is working] =

  Therefore:

- We assume that all facilities are independent:

  P(ij) = Prob[facility i and facility j are working] =

  P(i|j) = Prob[facility i or facility j is working] =

- Simplest measure of network reliability:

  $P_c(G)$ = Prob[Network is connected]

  Where: **c** stands for the connectivity of the network, and
  G stands for the graph representing the network

$P_c(G)$ = Prob[All nodes are working and there is a spanning tree of working links]

$P_c(G)$ =

- Since enumerating all trees in G requires an exponential amount of effort, **$P_c(G)$** is very difficult (if not impossible) to compute.

  → We seek simpler measures of network reliability.

5.3.2 Reliability of Tree Networks

- A typical enterprise/campus network includes trees:

- Given a tree T:

$P_c(T)$ = Prob[A tree network, T, being connected]
= Prob[All components (nodes and links) are working]

$P_c(T)$ =

- **$P_c(T)$** can also be computed recursively:

$P_c(T)$ =

Where: **T-i** is the tree T without node i, and
j is the link between node i and the rest of the tree

- Given a particular tree with root r:

  $P_c(i)$ = Prob[node i can communicate with root r]

  $P_c(i)$ =

  Where: **j** is the link between nodes i and k, and
  k is the predecessor of node i

  $P_c(r)$ =

- The expected number of nodes communicating with the root r is:

  E(r) =

- This expression can be computed efficiently for any node as follows:

  E(i) = the expected number of nodes communicating with the node i

  E(i) =

- If node i is a leaf, then:

  E(i) =

Example:

- The expected number of node pairs communicating through the root r is:

EPR(r) =

Example:

- The expected number of node pairs able to communicate in a tree via the shortest path: **EP(r)**

Where:

EP(i) = Expected number of pairs communicating in the tree rooted at i

EP(i) =

5.4 References

1. "Telecommunications Network Design Algorithms" by Aaron Kershenbaum, 1993

2. "Queuing Analysis" by William Stalling, 2000