Data Communication & Networks
G22.2262-001

Session 10 - Main Theme
Performance in Queuing Systems

Dr. Jean-Claude Franchitti

New York University
Computer Science Department
Courant Institute of Mathematical Sciences

Agenda

- Planning for Performance
- Queuing Analysis
- Queuing System Structure
- Queuing System Variables
- Queuing Models and Examples
- References
- Conclusion

Part I

Planning for Performance

Planning for Performance

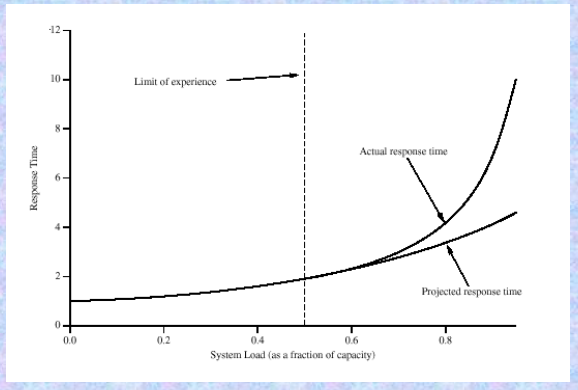
- When building Server apps, what can we do to insure acceptable performance?
- How do we answer questions like these:
  - What happens to response time when utilization goes up?
  - How many concurrent requests can be handled before response time is unacceptable?
  - What is the effect of adding (or removing) another server?

Planning for Performance

- A number of approaches are possible:
  1. Do an after-the-fact analysis based on actual values *("let's build something and see how it works!")*
  2. Make a simple projection by scaling up from existing experience to the expected future environment *("We expect double the number of users, so let's double the processor speed and memory!")*
  3. Develop an analytic model based on queuing theory *(subject of this lecture)*
  4. Program and run a simulation model *(this effort could be as big as the entire development effort!)*

The Problem with Approach 2

- The problem with this approach is that the behavior of most systems under a changing load is not what you might expect!



Part II

Queuing Analysis

Queuing Analysis: Basic Entities

- **Customers** (tasks, requests, etc)
  - Individual requests for service (e.g. a request for I/O, or a request by a customer in a bank, etc.)
- **Queues**
  - Waiting areas where requests for service wait for server(s) (e.g. the ready queue of processes waiting for CPU, or the waiting room at a doctor's office).
- **Servers**
  - Entities or resources that are capable of satisfying the service requests (e.g. CPU, disk, bank teller, etc.)

Queuing Analysis: Characterization

- **Dispatching Discipline**
  - When a server is done serving a customer, it must pick the next customer out of some queue. Algorithm used to do so is called the dispatching discipline. Examples are FCFS, FIFO, SJF, EDF, etc.
- **Distribution of Arrivals**
  - When do customers arrive? We will restrict our analysis to a Poisson process for the arrival of customers to the system.
- **Distribution of Service Time**
  - How long does it take a server to service a customer's request? The service time may be the same for all customers or, more realistically, the service time is likely to be variable.

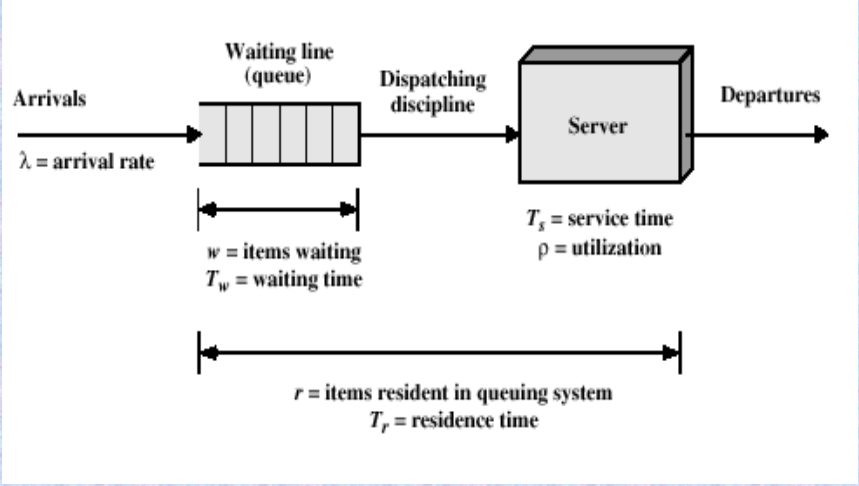
Simplifying Assumptions

- **Population**
  - Assume that Requests for service are generated from an infinite population. WHY? So that the arrival of a request to the system does not influence "future" arrivals
- **Queue Size**
  - Assume that queues have infinite capacity (unrealistic for many applications, but assume this for now)
- **Queue Service discipline** is FCFS

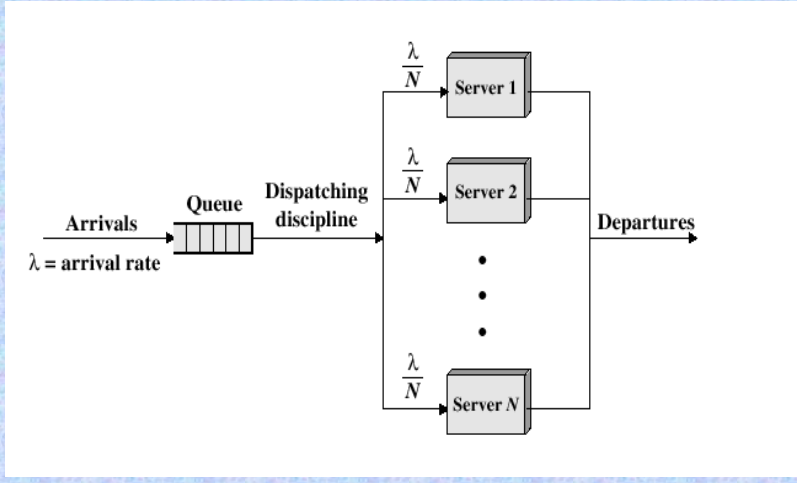
Part III

Queuing System's Structure

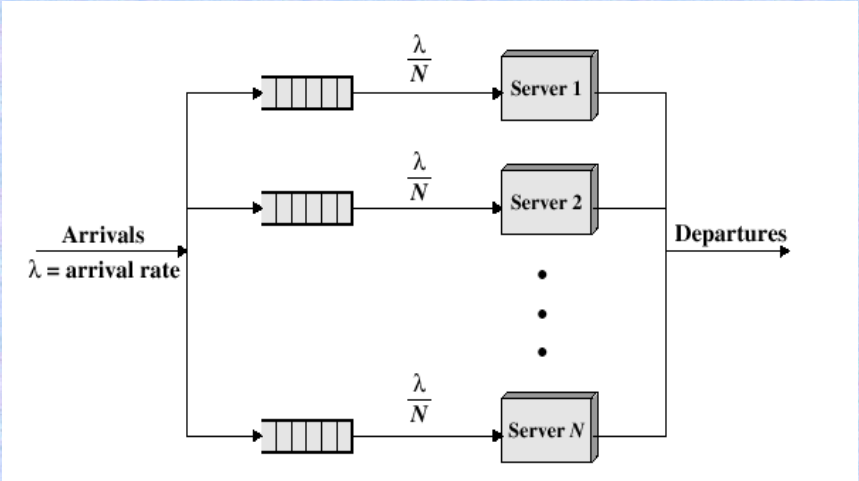
Queuing System Structure: Single Server



Single Queue, Multiple Servers



Multiple Single-Server Queues



Part IV

Queuing System Variables

Queuing System Variables

- If the average time it takes a server to service a request is $\mathbf{T_s}$, then it follows that the average rate of service (if the server has an infinite supply of requests to work on) would be:

$$\mu = 1/T_s$$

- The utilization of the system, which is the ratio between the rate of arrivals and the rate of service is:

$$\rho = \lambda/\mu$$

- Obviously, in the steady state, the rate at which requests are queued cannot exceed the rate at which the server is able to serve them! Thus, we have:

$$\lambda < \mu$$
$$\rho < 1$$

Part V

Queuing Models and Examples

Little's Formulas

The following two relationships are true of any "steady state" queuing system (i.e. a queuing system in equilibrium).

$$\mathbf{r = \lambda\, T_r}$$

$$\mathbf{w = \lambda\, T_w}$$

In a queuing system, a customer's time is spent either waiting for service or getting service.

$$\mathbf{T_r = T_w + T_s}$$

Multiplying the above equation by the arrival rate λ and applying Little's formula, we get:

$$\mathbf{r = w + \lambda\, T_s = w + \lambda / \mu}$$

Remember, $\rho = \lambda / \mu$, so ...

$$\mathbf{r = w + \rho}$$

Notation for Queuing Systems

A/B/c/d

- A = the interarrival time distribution
- B = the service time distribution
- c = the number of servers
- d = the queue size limit

■omitted if infinite

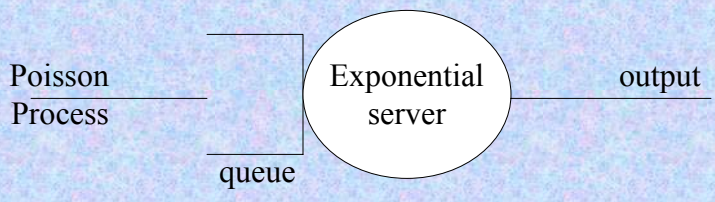
:Where **A** and **B** can be

D for Deterministic distribution

M for Markovian (exponential) distribution

G for General (arbitrary) distribution

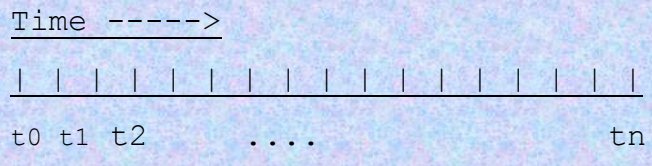
The M/M/1 System



Understanding System Behavior

- To characterize the behavior of the system, we have to understand:
  - The frequency distribution of requests.
    - In any given time period, how many new request enter the system?
  - The distribution of service times
    - Does every request take equal time to process? (probably not).

Understanding Arrival of Requests



In any one time period ti, some number of requests can arrive. The number that arrive is a random variable.

Arrivals Follow a Poisson Process

- a(t) = # of arrivals in time interval [0,t]

$$\Pr(a(t) = n) = e^{-\lambda t} (\lambda t)^n / n!$$

Models for Interarrivals and Service Times

- Customers arrive at times $t_0 < t_1 < \ldots$ - Poisson distributed
- The differences between consecutive arrivals are the **interarrival times** : $\tau_n = t_n - t_{n-1}$
- τ_n in Poisson process with mean **arrival rate** λ, are exponentially distributed,

$$\Pr(\tau_n \leq t) = 1 - e^{-\lambda t}$$

- **Service times** are exponentially distributed, with mean **service rate** μ: $\Pr(S_n \leq s) = 1 - e^{-\mu s}$

M/M/1 Performance

- Average number of customers in a M/M/1 System

$$\mathbf{r} = \rho/(1-\rho)$$

- Average number of customers waiting for service in a M/M/1 system

$$\mathbf{w} = \rho^2/(1-\rho)$$

- Average Time waiting in a M/M/1 queue

$$\mathbf{T_w} = \rho/\mu(1-\rho)$$

M/M/1 Performance

- Probability that the number of customers in the system = **N**

$$\Pr[R=N] = (1-\rho)\,\rho^N$$

- Probability that the number of customers in the system ☯ **N**

$$\Pr[R \text{ ☯ } N] = \sum_{i=0}^{N} (1-\rho)\,\rho^i$$

Examples

- **Given**
  - Arrival rate of 50 requests/sec
  - Service rate of 60 requests/sec
- **Find**
  - Utilization
  - Probability of having 10 requests in the system
  - How big should we make the queue?

$\rho = \lambda/\mu = 50/60 = 0.833$

$Pr[r=12] = (1-\rho)\ \rho^N = (0.166)(0.833)^{10} = 0.026$

$Pr[r=50] = (1-\rho)\ \rho^N = (0.166)(0.833)^{50} = 0.00002$

$Pr[r=100] = (1-\rho)\ \rho^N = (0.166)(0.833)^{100} = 2*10^{-9}$

So, it looks like 100 buffers will suffice (50 will not)

Examples (continued)

- What is the average in-queue wait time?

  $\mathbf{T_w = \rho/\mu(1-\rho)} = 0.833/(60*0.167) = .083 = 83$ msec

And finally, what is the average response time (T_r) ?

$\mathbf{T_r = T_w + T_s}$

$\mathbf{T_s = 1/\mu} = 1/60 = 0.0167 = 17$ msec

$\mathbf{T_r = 83 + 17 = 100}$ **msec**

What happens if we increase service rate to 75/sec?

Examples (continued)

- Service rate is now 75, so

  $\rho = \lambda/\mu = 50/75 = 0.67$

  $T_w = \rho/\mu(1-\rho) = 0.67/(75*0.33) = .027 = 27$ msec

- And finally, what is the average response time (Tr) ?

  $T_s = 1/\mu = 1/75 = 0.013 = 13$ msec
  $T_r = 27 + 13 = 40$ msec!!

- **We increased service rate by only 25% yet response time dropped by 60%!**

Part VI

Conclusion

Assignment & Readings

- Readings:
  - ***Queuing Analysis*** paper by William Stallings (recommended):
    ftp://shell.shore.net/members/w/s/ws/Support/QueuingAnalysis.pdf
  - **Tom Slater's Queuing Theory Tutor**
    http://www.dcs.ed.ac.uk/home/jeh/Simjava/queueing/
  - **Myron Hlynka's Queueing Theory Page**
    http://www2.uwindsor.ca/~hlynka/queue.html

Next Session:

Multimedia Networking
Network Security
Network Management