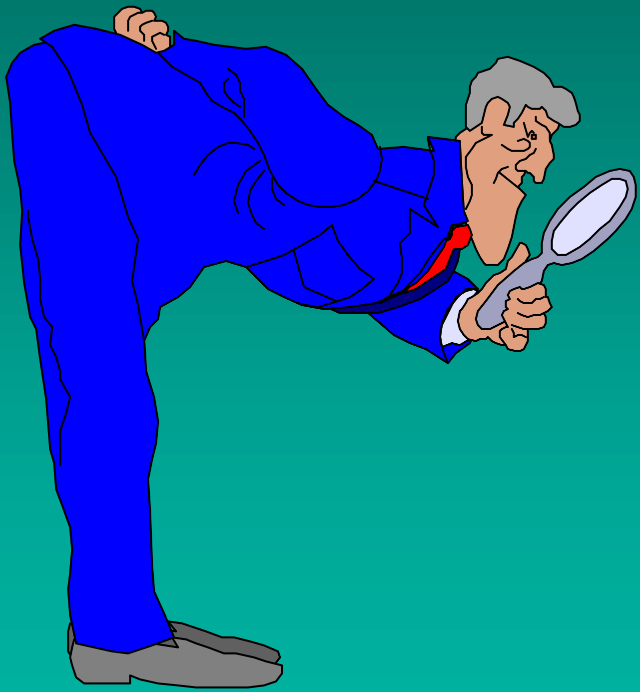
# Queueing Systems
## Modeling and Performance Evaluation with Computer Science

*Spring, 2003*

*Dr. Eric Hsiao-kuang Wu*

*http://wmlab.csie.ncu.edu.tw/course/queueing*

# What is going to be covered? (Queueing System)



# Course Outline

- Probability
  - Discrete/Continuous random variable
  - Conditional Probability
- Queuing Modeling
  - M/M/1/k
  - Bulk Service, Bulk Arrival
  - M/G/1
  - G/G/1
- Case Studies:
  - Computer Applications
  - Wireless Network Applications

# Lecture Progress (February, 2003)

- Queueing Systems
  - System Flow
  - Specification and Measure of Queueing System
- Notation and Structure for Basic Queueing Systems
- Probability Z transform
- Reference (Textbook2)

# Daily Experiences

- Waiting in Line:
  - – Waiting for breakfast
  - – Stopped at a traffic light
  - – Slowed down on the freeways
  - – Delayed at the entrance to parking facility
  - – Queued for access to an elevator
  - – Holding the telephone as it rings..

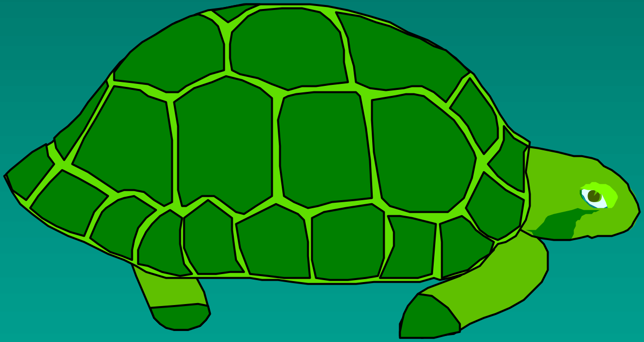
# Systems of Flow

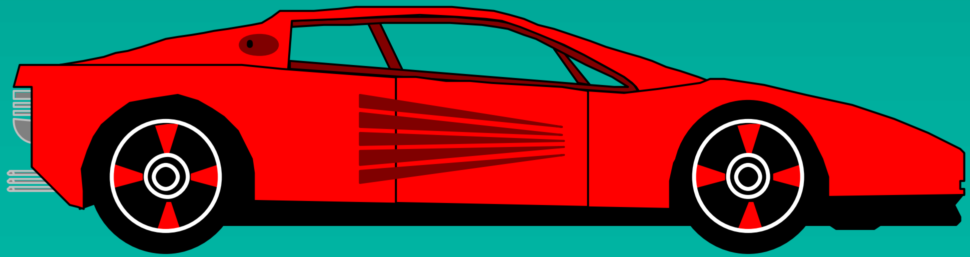
- Queueing Systems
  - Systems of flow
- A flow system is one in which some commodity flows, moves, or is transferred through one or more finite-capacity channels in order to go from one point to another
- Commodity: (produce the demand)
  - Such as packet massage, telephone message, automobiles
- Channel: (provide the service)
  - Such as Internet, telephone network, the highway

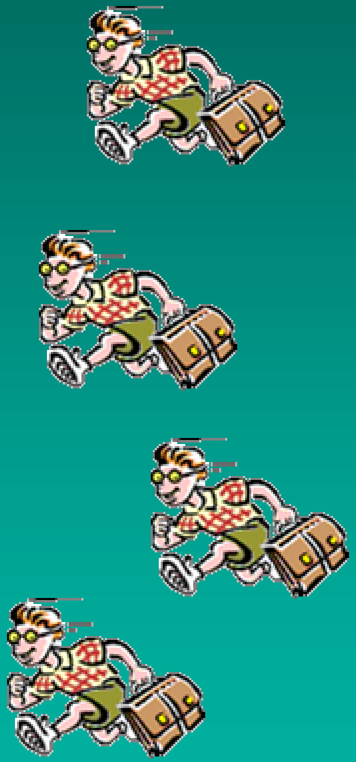
# Service and Demand

the arrival rate R

the service rate (or capacity) C







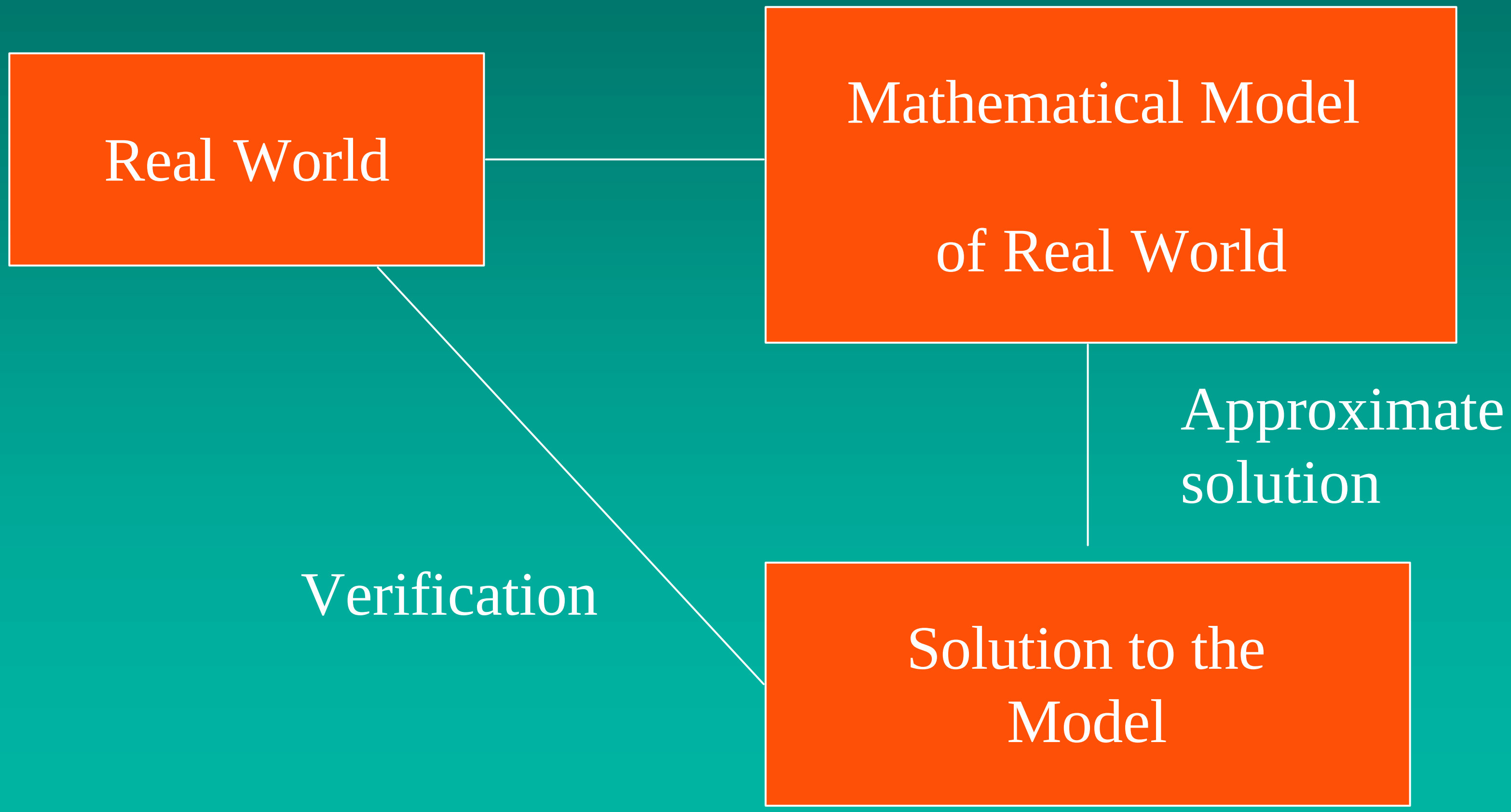
# Steady and Unsteady Flow

- Whether the flow is steady or unsteady?
  - Steady: those systems in which the flow proceeds in a predictable fashion
  - If R<C, a reliable and smooth fashion
  - If R>C, the mean capacity is less than the average flow requirements, chaotic congestion occur

# History of Computer Using

- Single User
- Batch
- Time-Sharing
- Sharing Communication line
- Network (1970's)

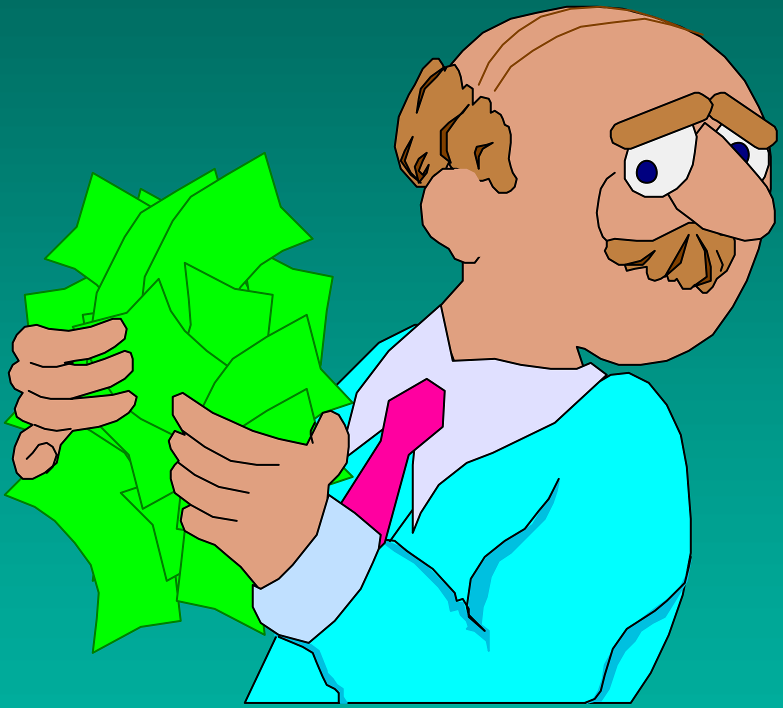
# Modeling


Real World
Mathematical Model
of Real World
Approximate
solution
Verification
Solution to the
Model

# Resource Sharing

- A resource is a device that can do works for you at a finite time
  - e.g. A communication Channel
  - e.g. A computer
- A demand requires work from resource
  - e.g. message
  - e.g. jobs (require processing)

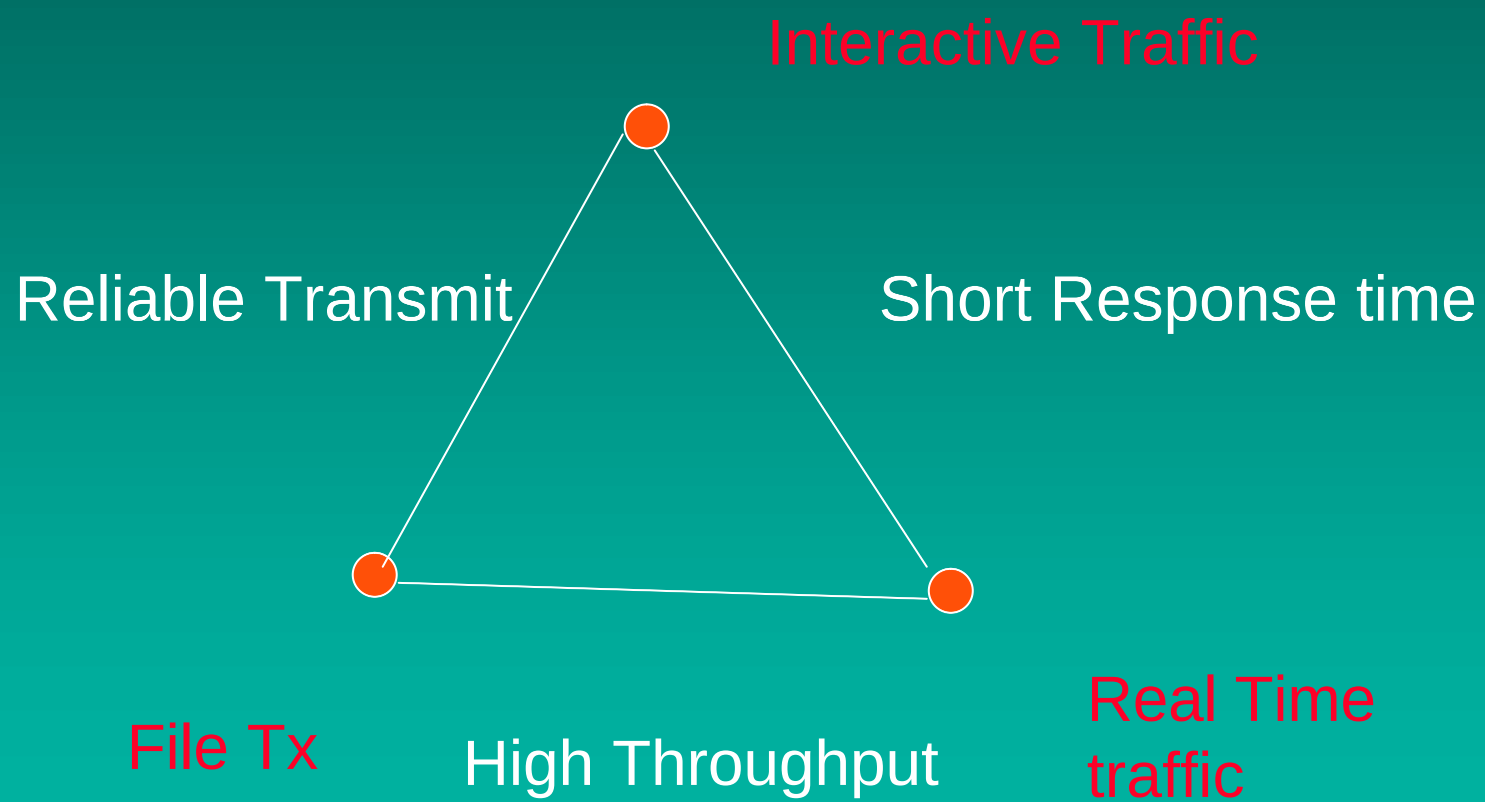
# User Behavior



# Bursty Asynchronous Demands

- You cannot predict exactly when they will demand access
- You cannot predict exactly how much they will demand access
- Most of time they do not need access to resource
- When they ask for it, they want immediate access

# Typical Traffic



# Resource Sharing

- Type1: Everyone use his resource singlely (not efficient).
- Type2: Using Pool of resource sharing those resources (by switching) plus the cost of switch
- Type3: Using a large resource (as an unit).

# Law of Large Number

- The first resource sharing principle
- Although each member of a Large population may behave in a Random fashion, the population as a whole behave in a predictable fashion.
  - This is the “smoothing “ effect of large population
  - The predictable fahsion presents a total demand equal to the sum of the average demands of each member

# Conflict Resolution

- Queueing: one gets severed, others wait
- Splitting: Each get a piece of resource
- Blocking: One get served, all others are refused
- Smashing: Nobody gets served.

# Response Time

- When the throughput and capacity go up, the response time will go down
- Economy of Scale
  - The second resource sharing principle
  - if you scale up throughput and capacity by some factor F, then you reduce response time by the factor

# Economy of Scale


Original: B Block/sec   Cbit/sec

Scale:   NB Block/sec  NC bit/sec

$T(NB,NC) = T(B,C)/N$

# Throughput, Efficiency, Response time

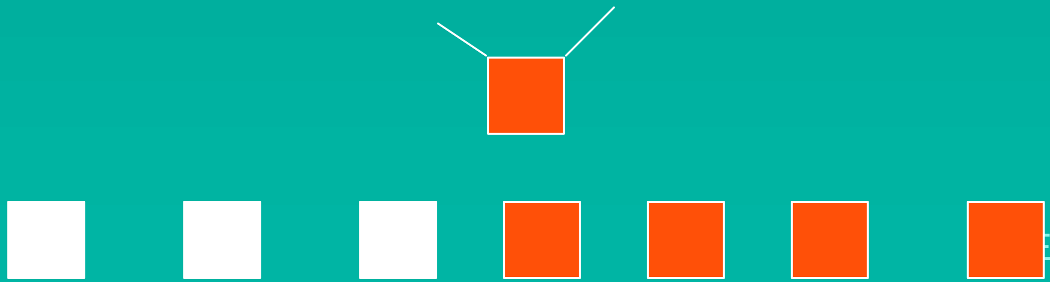
- If you scale the capacity more slowly than throughput while holding response time constant, then efficiency will increase
- Key tradeoff among:
  - Efficiency = Throughput / Capacity

# System of Flow

- Flow of a commodity (demand) through a finite-capacity channel (resource)
  - Steady Flow
  - Unsteady Flow

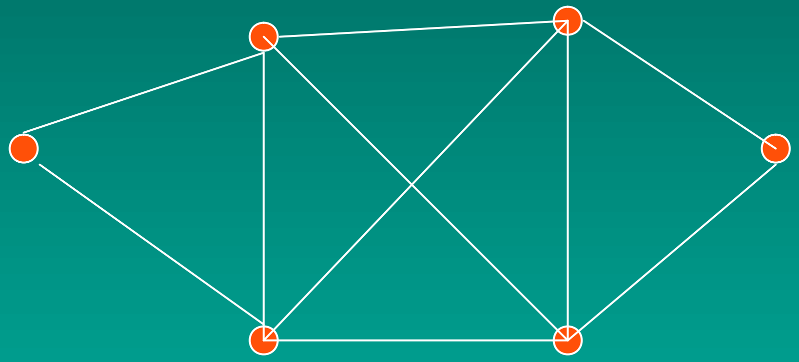
# Steady Flow

- Demand are known, constant smooth: predictable
- Single Channel:
  - R = Arrival Rate (Cans/Sec)
  - C = Capacity (Cans/Sec)
  - if R <= C   Fine
  -    R > C    Chaos



# Network of Channels

- Max-Flow Min-Cut Theorem



ChonLi

Taipei

- R < C for each channel
- Maxmum Flow , label the node, find a path

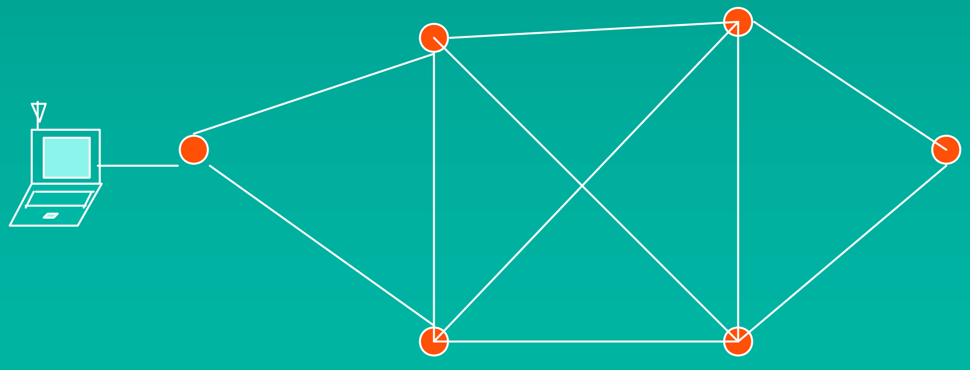
# Unsteady Flow(I)

- Arrival time of Demand: Unpredicatble
- Sise (Service time) of Demand: Unpredictable
- Single Channel:
  - Queue Length
  - Waiting Time
  - Sever Utilization
  - Throughput
  - Probablity kills you

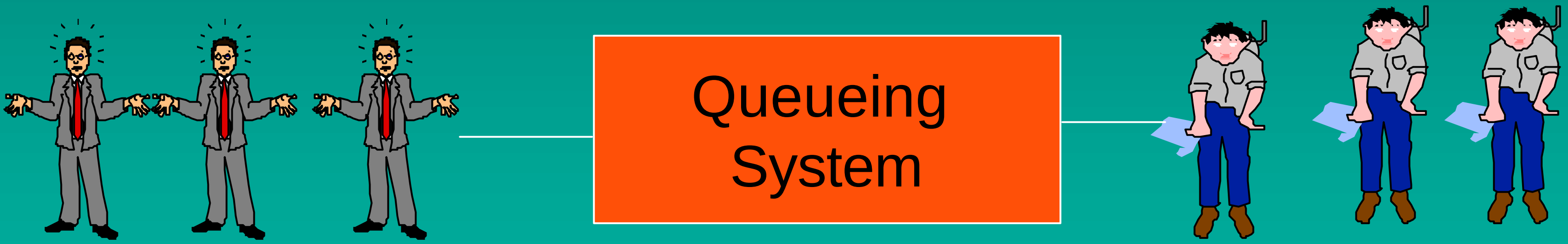
# Unsteady Flow(II)

- Network of Channel
  - capacity
  - throughput
  - Response Time
  - Efficiency
  - design

Combinatonics and probablities kill you



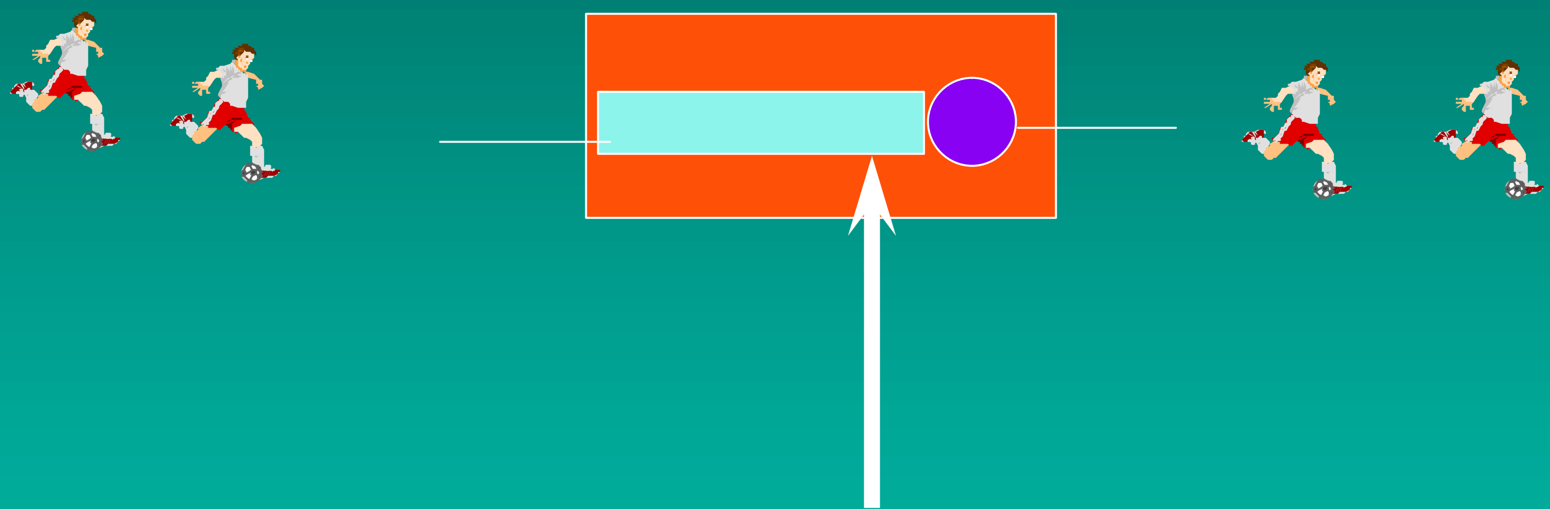
# General Queueing System




- How to improve the system performance
- How to model the system

# Review of Queueing

- Queueing Systems
  - Notation
  - Markovian Queue, Birth-and-death
  - M/M/1 -> M/M/k/m
  - Stage -> Erlangian distribution
  - Parallel
  - Network of Queue
  - M/G/1



limited resouce (fixed number of
queue size buffer N

How often they arrive

how long they will stay
= service time + waiting time

# What we are interested ?

- How long we are going to wait ?
- How big the queue size should be ?

# Observation 1

- Each customer could be characterized as the following:
  - how often the traffic produced ?
  - how many service it may require ?

Arrival Rate    Service Rate

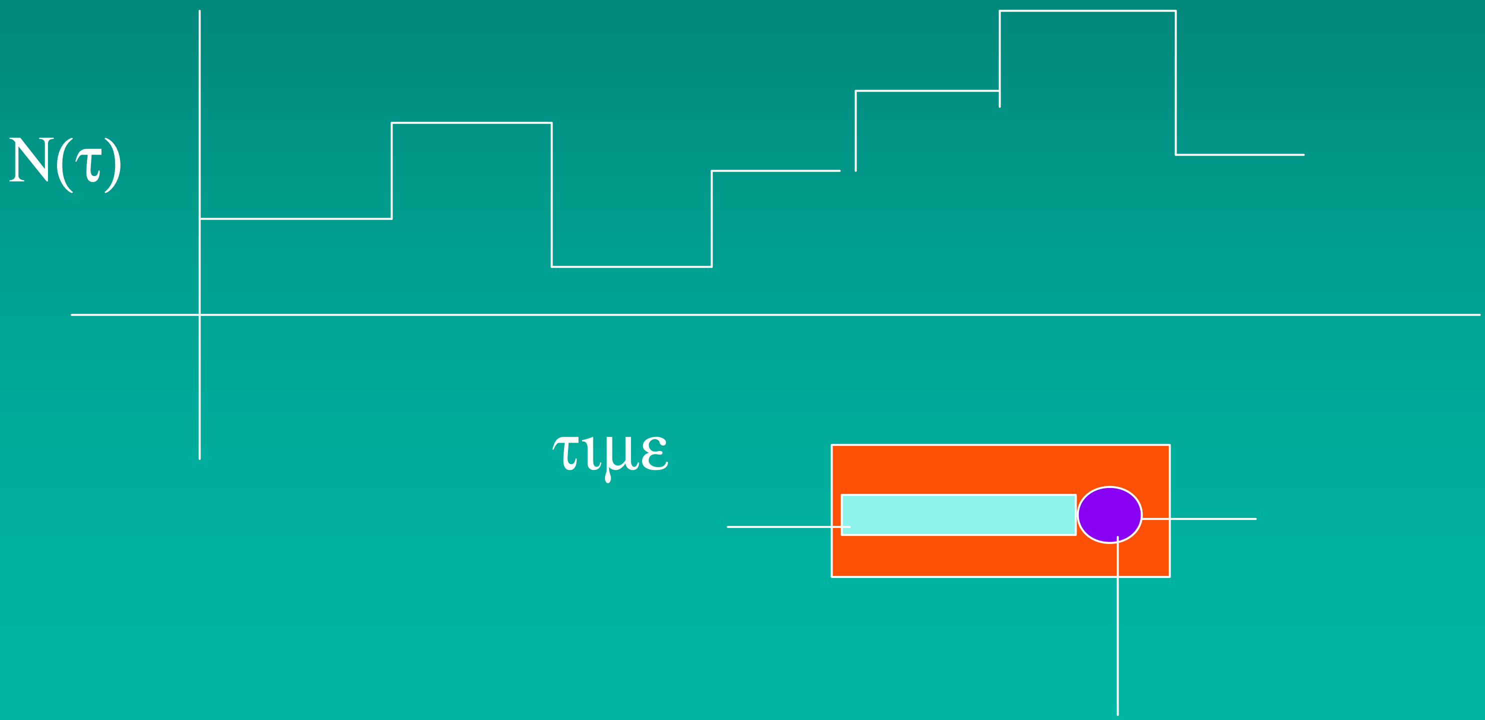
# Observation 2

- Some users might be in the queue ?

Number of users in the system

# Observation

- Current State depends on Previous State



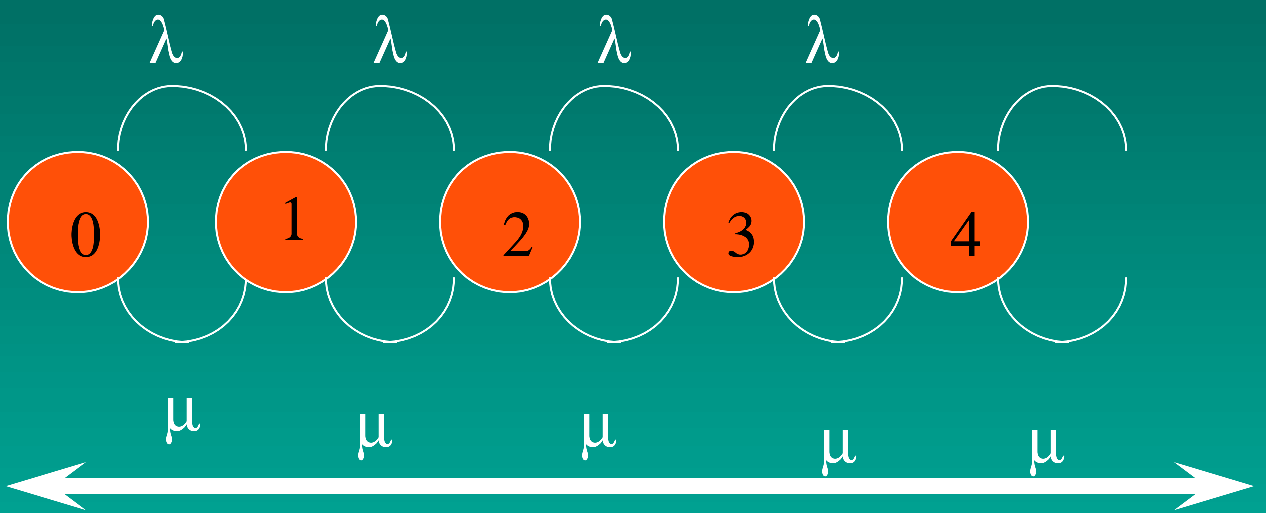
# Computer Queue System

- Markovian Chain:
  - current state depends on previous one state only
  - time domain
    - discrete
    - continuous
  - state domain:
    - dsicrete
    - continuous

# Birth-Death Process

- Transitions are allowd between neighbors:
  - P(k) to P(k+1)
    - birth happen (arrival)
  - P(k) to P(k-1)
    - death happen (death)
- Possion and Exponetial Distributions are memoryless

# M/M/1




Number of buffers <-> Number of Customers
Rate in = Rate out (Flow balance)
Sum of P(k) = 1
Memoryless

# Format

- M / M / 1 / 2

Number of buffer size

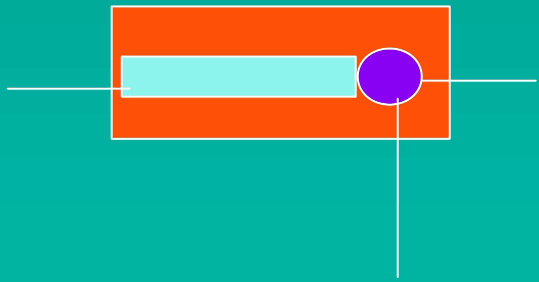
number of server

Amount of service a customer require

$B(x) = P[\text{service time} \le x]$

Arrival Time

$A(t) = P[\text{interarrival time} \le t]$



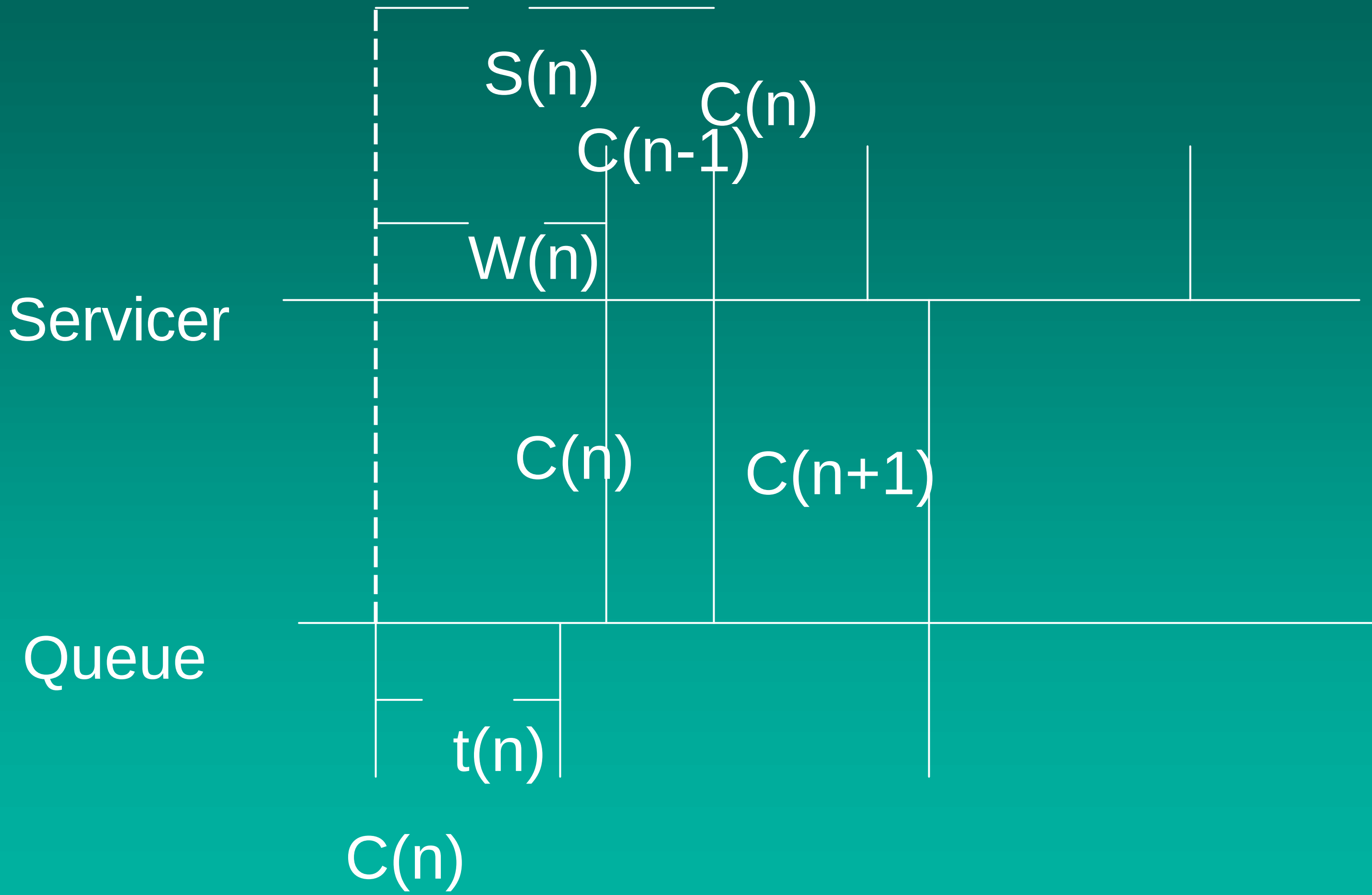
# Probability

- Sum of P(k) = 1
- P(k) <= 1
- E[N] = Sum of K P(k)
- $\rho = \lambda / \mu$

# General Queueing System

- C(n) nth customer to enter the system
- N(t) number of customer in the system at time t
- a(n) arrival time for C(n)
- t(n) interarrval time between C(n-1) and C(n)
- x(n) service time for C(n)
- w(n) waiting time for C(n)
- S(n) system for C(n)

# Time-diagram notation

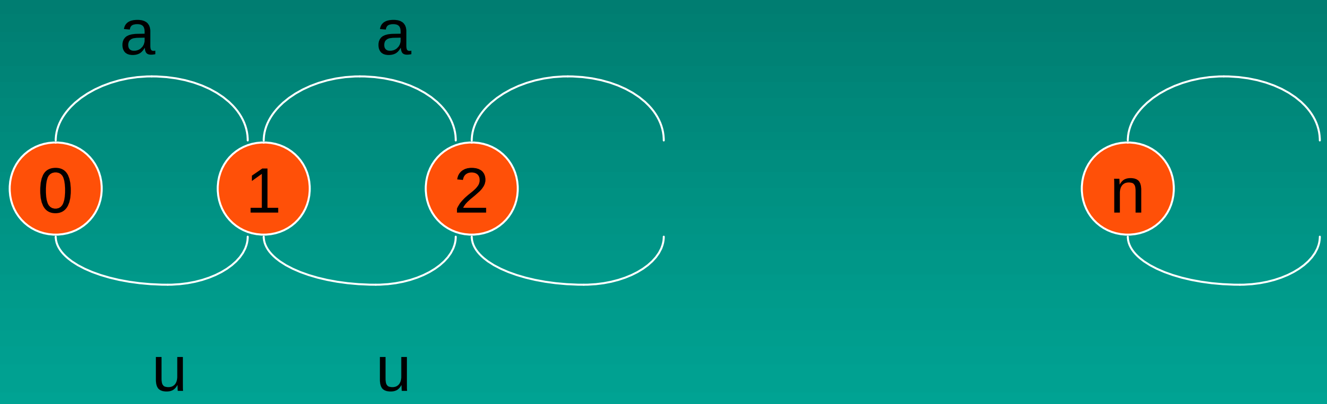


# Classical M/M/1 Queueing

- Single Server Queue
- Poisson Arrival Process
- Exponential Distribution for service time
- M stands for memoryless

# M/M/1 Analysis

- State-transition-rate diagram



# What you should need for Queueing modeling

- Probability (such as arrival rate, service rate)
- Transform (z-transform, Laplace transform)