COMP9334
Capacity Planning for Computer Systems and Networks

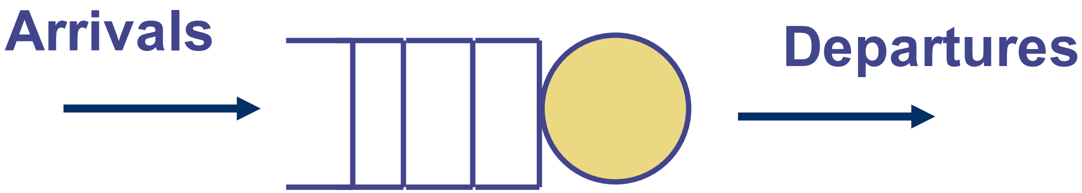
Week 9: Mean Value Analysis

Classification of queues

- Single server queue versus a network of queues
- Open queueing networks versus closed queueing networks

Weeks 3 & 5: Open queues

- Single-server M/M/1



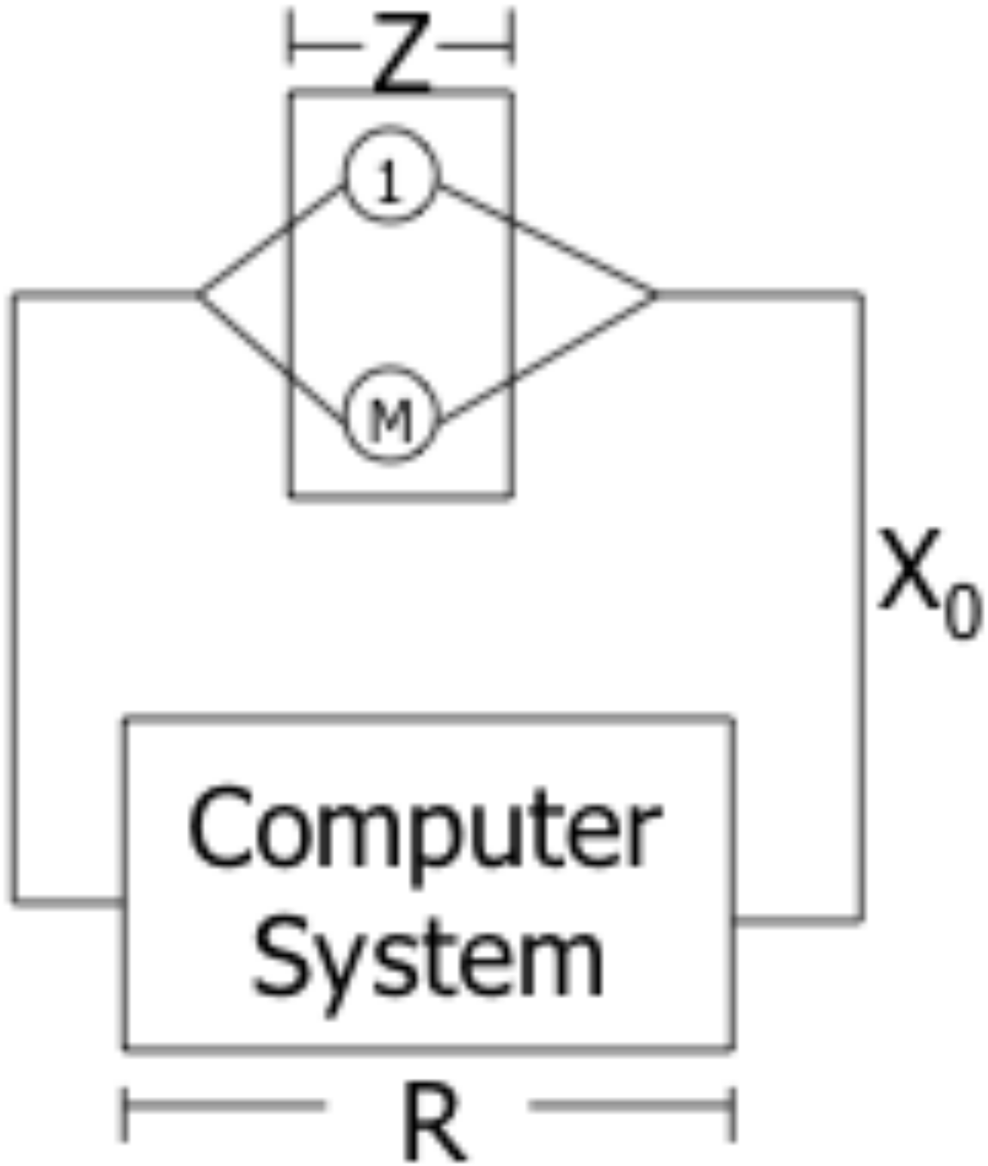
Exponential inter-arrivals (λ)

Exponential service time (μ)

- Also M/G/1, G/G/1, M/G/1 with priority
- Characteristics of open queueing networks
  - Have external arrivals and departures
  - Customers will finally depart from the system
  - Workload intensity specified by inter-arrival and service time distributions

Weeks 2 & 4: Closed queueing networks

- Closed queueing networks
  - Have no external arrivals nor departures
  - Can be classified into *Batch Systems* and *Interactive Systems*
- Examples of interactive systems
  - Interactive terminals
  - Machine reliability analysis (Week 4) can be modelled as an interactive system

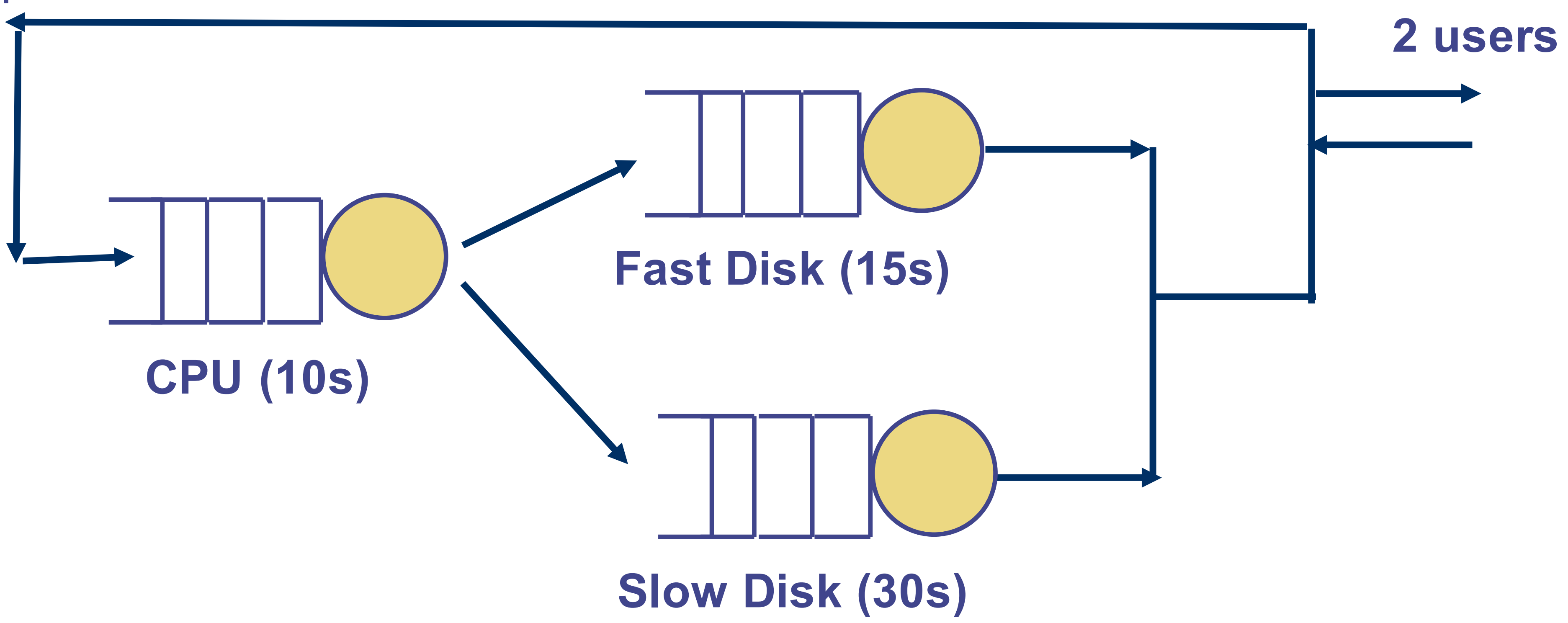


This lecture

- Methods to *efficiently* analyse a closed queueing network
- Motivation
  - You have learnt how to analyse a closed queueing network in Week 4 using Markov chain
  - However, the method can only be used for a small number of users
- This week we will study a method that can be used for a large number of users

- Let us begin by revisiting the database server example in Week 4

DB server example

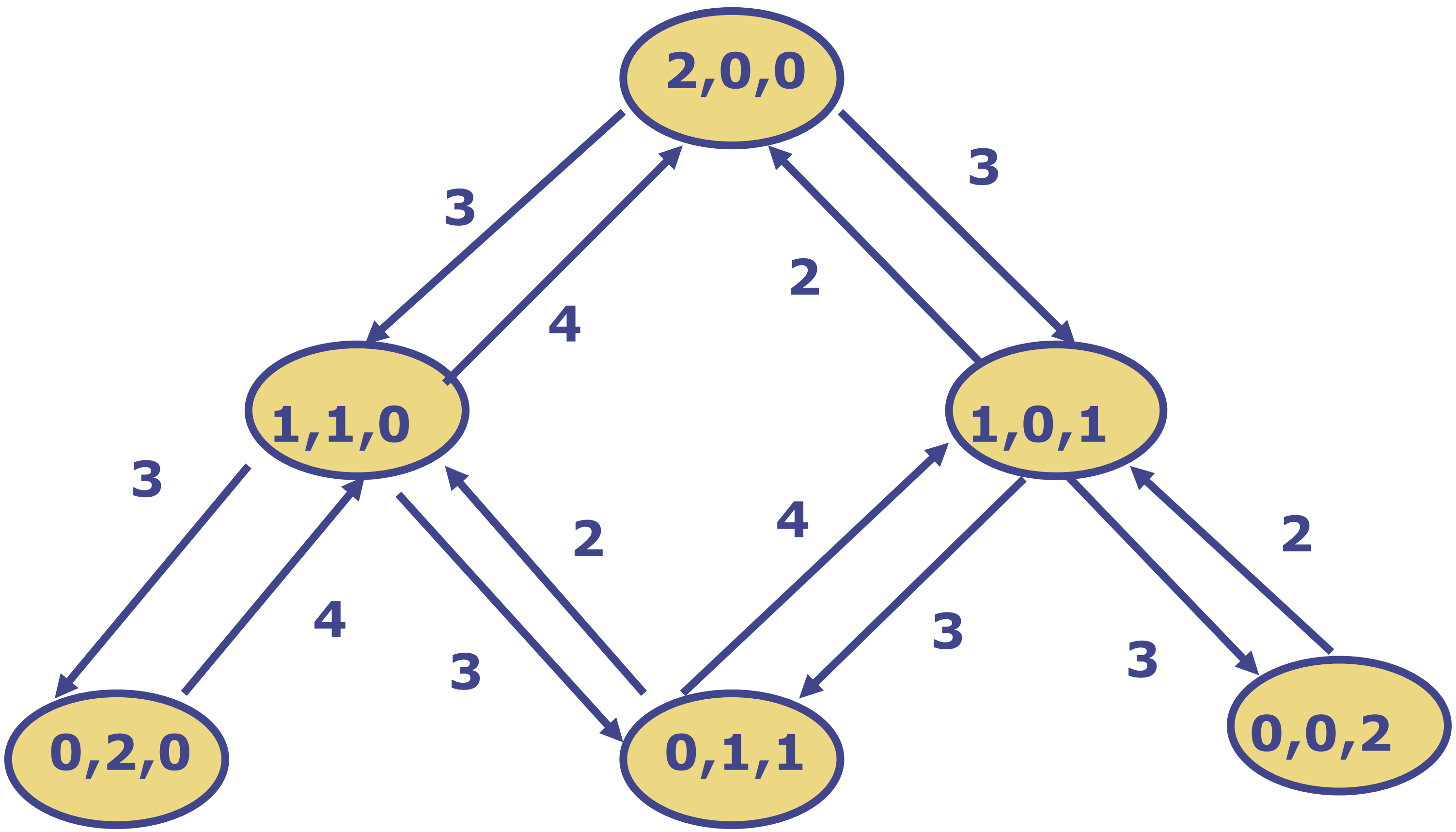


- 1 CPU, 1 fast disk, 1 slow disk.
- Peak demand = 2 users in the system all the time.
- Transactions alternate between CPU and disks.
- The transactions will equally likely find files on either disk
- Service time are exponentially distributed with mean showed in parentheses.

Markov chain solution to the DB server problem

- In Week 4, we used Markov chain to solve this problem
- We use a 3-tuple (X,Y,Z) as the state
  - X is # users at CPU
  - Y is # users at fast disk
  - Z is # users at slow disk
- Examples
  - (2,0,0): both users at CPU
  - (1,0,1): one user at CPU and one user at slow disk
- Six possible states
  - (2,0,0) (1,1,0) (1,0,1) (0,2,0) (0,1,1) (0,0,2)

Markov model for the database server with 2 users



Solving the model

- Solve for the probability in each state P(2,0,0), P(1,1,0), etc.
  - There are 6 states so we need 6 equations
- After solving for P(2,0,0), P(1,1,0) etc. we can find
  - Utilisation
  - Throughput,
  - Response time,
  - Average number of users in each component etc.

What if we have 3 users instead?

- What if we have 3 users in the database example instead of only 2 users?
- We continue to use (X,Y,Z) as the state
  - X is the # users at CPU
  - Y is the # users at the fast disk
  - Z is the # users at the slow disk
- How many states will you need?
- We need 10 states:
  - (3,0,0),
  - (2,1,0),(2,0,1)
  - (1,2,0),(1,1,1),(1,0,2)
  - (0,3,0),(0,2,1),(0,1,2),(0,0,3)

What if there are *n* users?

- You can show that if there are *n* users in the database server, the number of states *m* required will be

$$\frac{(n+1)(n+2)}{2}$$

- For *n* = 100, *m* (= #states) ~ 50000
- You can automate the computational process but where is the computational bottleneck?
  - Solving a system of m linear equations in *m* unknowns has a complexity of $O(m^3)$
- For our database server with *n* users, the computational complexity is about $O(n^6)$

Weaknesses of Markov model

- The Markov model for a practical system will require many states due to
  - Large number of users
  - Large number of components

- Large # states
  - More transitions to identify
    - Though this can be automated
  - If you've *m* states, you need to solve a set of *m* equations. A larger set of equation to solve.
    - The complexity of solving a set of *m* linear equations in *m* unknowns is $O(m^3)$

Mean value analysis (MVA)

- An iterative method to find the
  - Utilisation
  - Mean throughput
  - Mean response time
  - Mean number of users

- The complexity is approximately $O(nk)$ where
  - n is the number of users
  - k is the number of devices

- The complexity of MVA makes it a very practical method

MVA - overview

- MVA analysis has been derived for
  - Closed model
    - Single-class
    - Multi-class
  - Open model
  - Mixed model with both open and closed queueing

- This lecture discusses MVA for single-class closed model

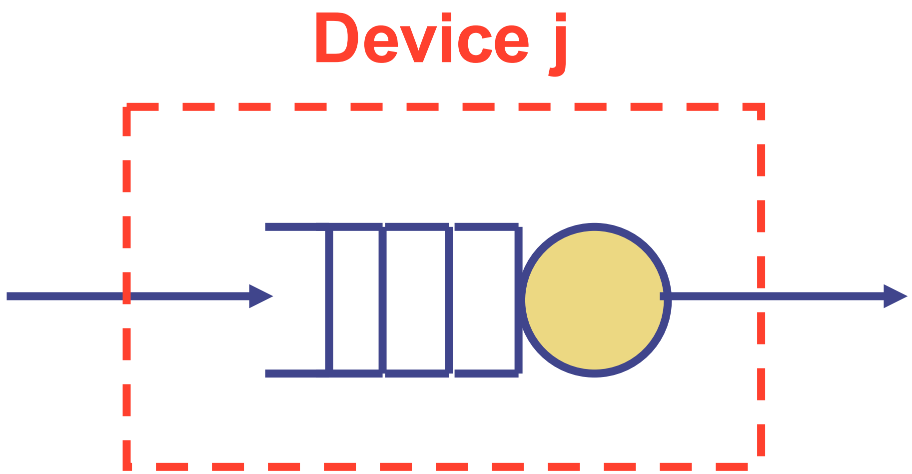
MVA for closed system

- Consider a closed queueing network with a single-class of customers
- You are given a system with *K* devices
- You are given that each customer
  - Visits device *j* on average *V(j)* times
  - Requires a mean service time of *S(j)* from device *j*
    - *Note: The service time required is assumed to be exponentially distributed*
- From the information given, we can deduce that the service demand *D(j)* for device *j* is *V(j) S(j)*
- How do we obtain *D(j)* for a practical system?

Key idea behind MVA

- Key idea behind MVA is *iteration*
  - If you know the solution to the problem when there are *n* customers in the system, you can find the solution when there are *(n+1)* customers

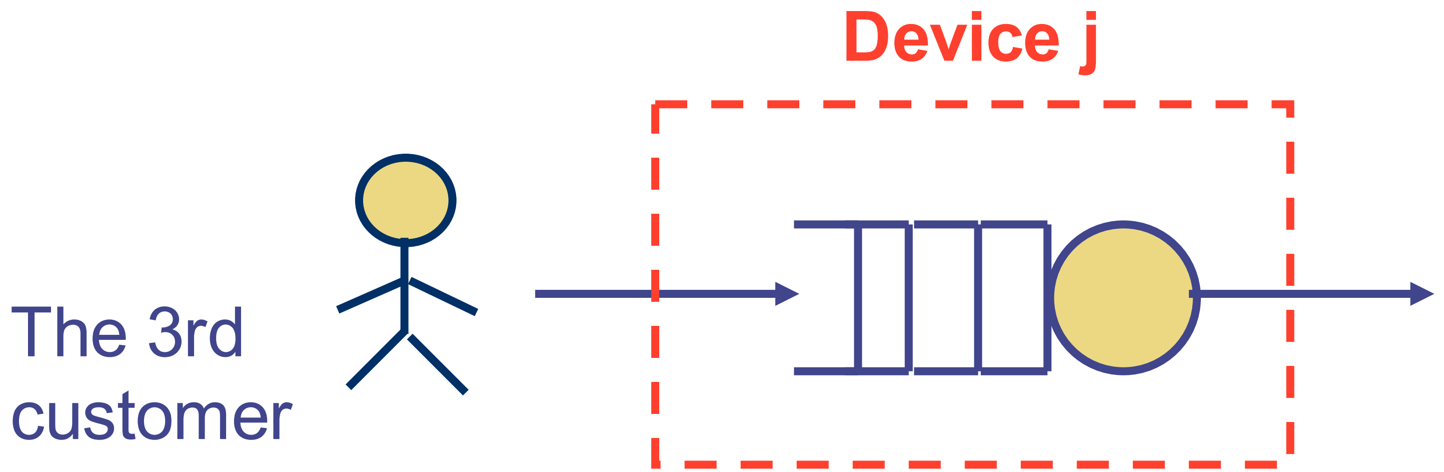
Let us consider a simple example to motivate the iteration in MVA. Consider single device j of a queueing network.



Assume that we know when there are 2 customers in the system, the average number of users in device j is 0.6 (say).

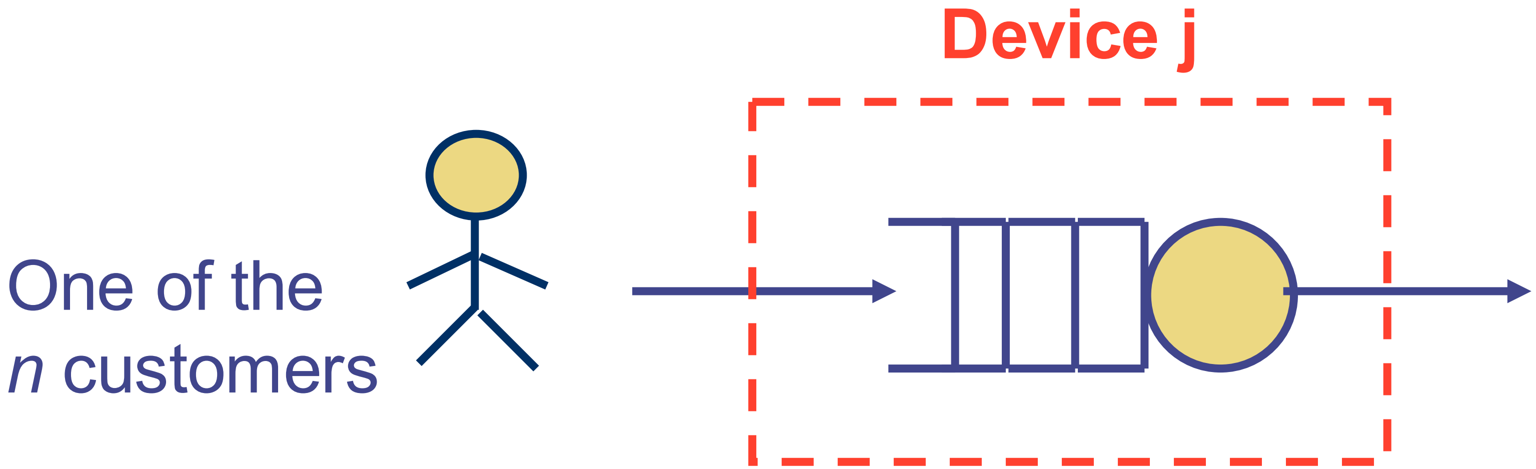
What happens when there are 3 customers?

What happens when there are 3 customers?



- Let us assume the 3rd customer is arriving at device *j*.
- Where will the other 2 customers be? We cannot tell exactly but we know that there is on average of 0.6 customers in device *j* when there are 2 customers.
- The 3rd customer will see on average 0.6 customers when it arrives at device *j*.

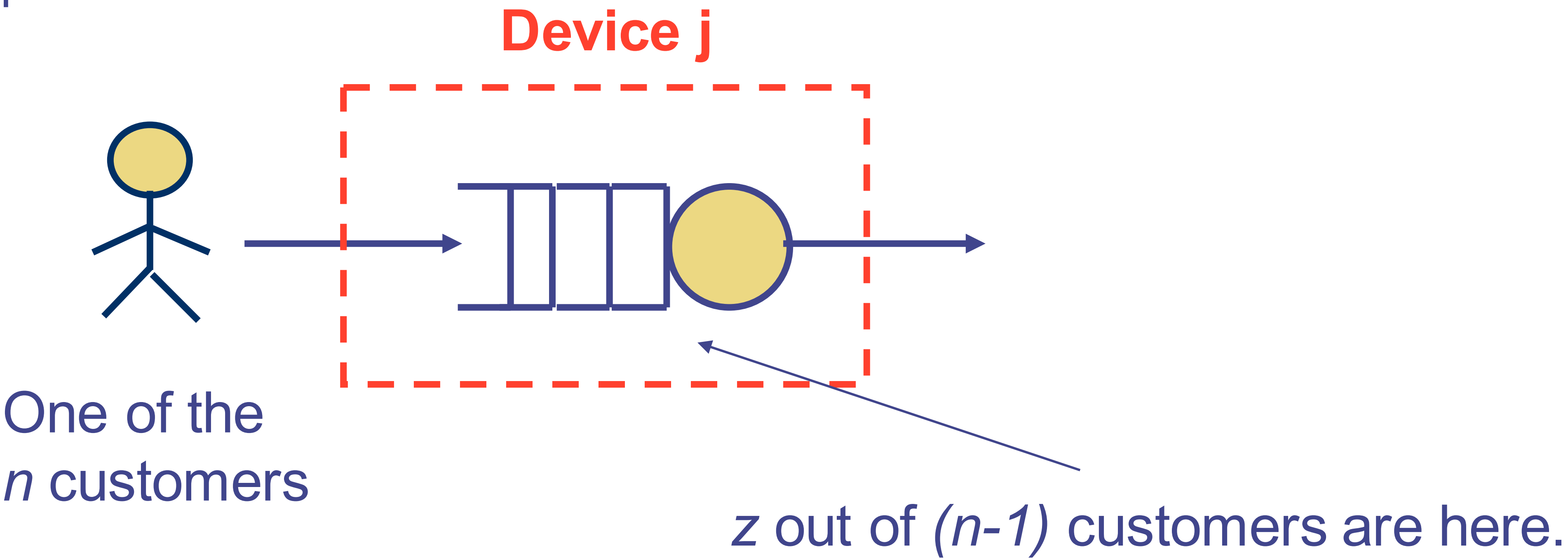
When there are n customers …



Arrival Theorem

- If there are *(n-1)* customers in the system, the mean number of customers in device *j* is *z* customers,
- Then, when there are *n* customers, each customer arriving at device *j* will see on average *z* customers ahead of itself in device *j*.

How can Arrival Theorem help?



Let $S(j)$ = mean service time at device j.
When there are n customers,
The mean waiting time at device $j = z\ S(j)$
The mean response time at device $j = (z+1)\ S(j)$

Iterations of MVA:

Mean number of customers in each device

#customers = n-1

Mean response time for each device

…..

…..

Mean number of customers in each device

#customers = n

Mean response time for each device

….

#customers = n+1

Some notation

Note "(n)" means there are n customers in the system

$$\bar{n}_i(n) = \text{Mean \# of customers in device i}$$

$$R_i(n) = \text{Mean response time in device i}$$

$$R_0(n) = \text{Mean response time of the system}$$

$$X_i(n) = \text{Throughput of device i}$$

$$X_0(n) = \text{Throughput of the system}$$

Mean response time of each device $R_i(n)$

$$R_0(n) = \sum_{i=1}^{K} V_i \times R_i(n)$$

System response time $R_0(n)$

$$X_0(n) = \frac{n}{R_0(n)}$$

Throughput of the system $X_0(n)$

$$X_i(n) = V_i \times X_0(n)$$

Throughput of each device $X_i(n)$

$$\bar{n}_i(n) = R_i(n) \times X_i(n)$$

Mean # customers in each device $\bar{n}_i(n)$

Initialisation of MVA:

Mean number of customers in each device

#customers = 0

$$\bar{n}_i(0) = 0$$

Mean response time for each device

…..

…..

Mean number of customers in each device

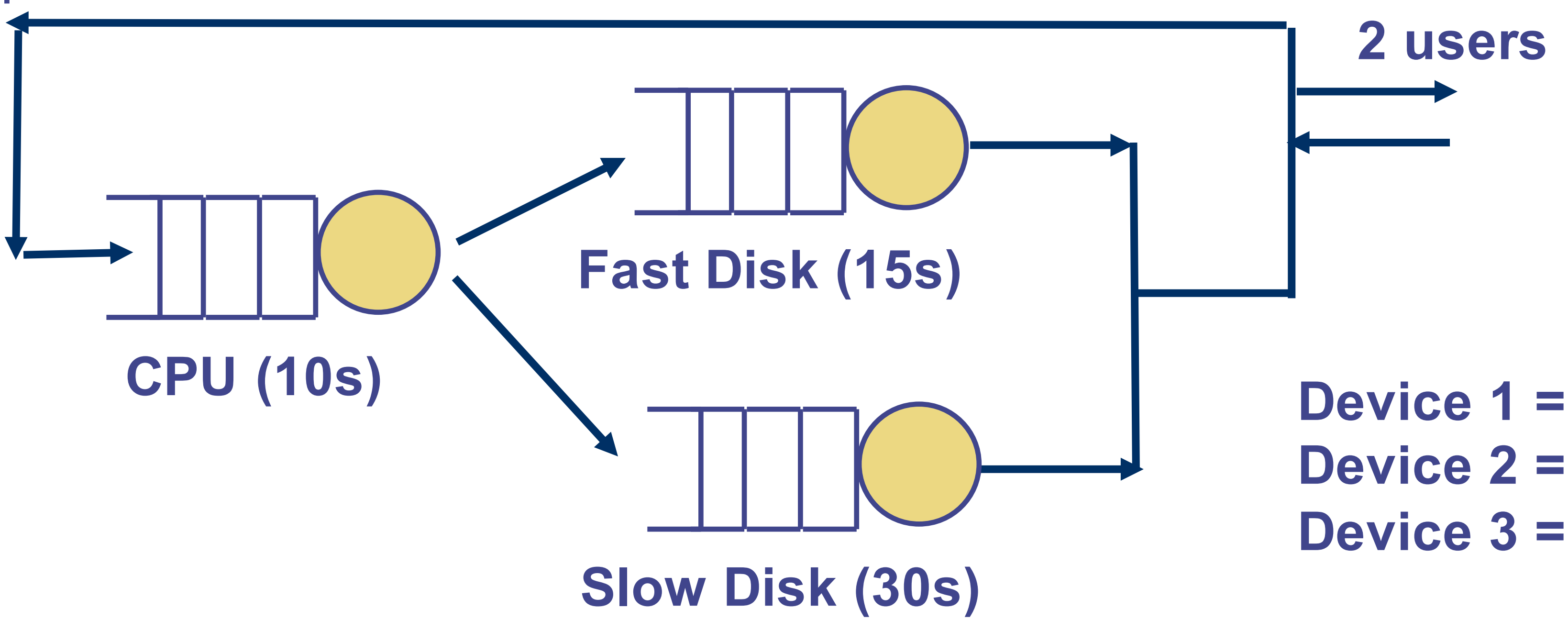
#customers = 1

Mean response time for each device

….

#customers = 2

Let us apply MVA to the database example



Device 1 = CPU;
Device 2 = Fast disk;
Device 3 = Slow disk

$$S_1 = 10; S_2 = 15; S_3 = 30;$$
$$V_1 = 1; V_2 = \frac{1}{2}; V_3 = \frac{1}{2};$$

- Determine the performance when there are 2 users in the system
- And how about 3 users?

Limitation of MVA

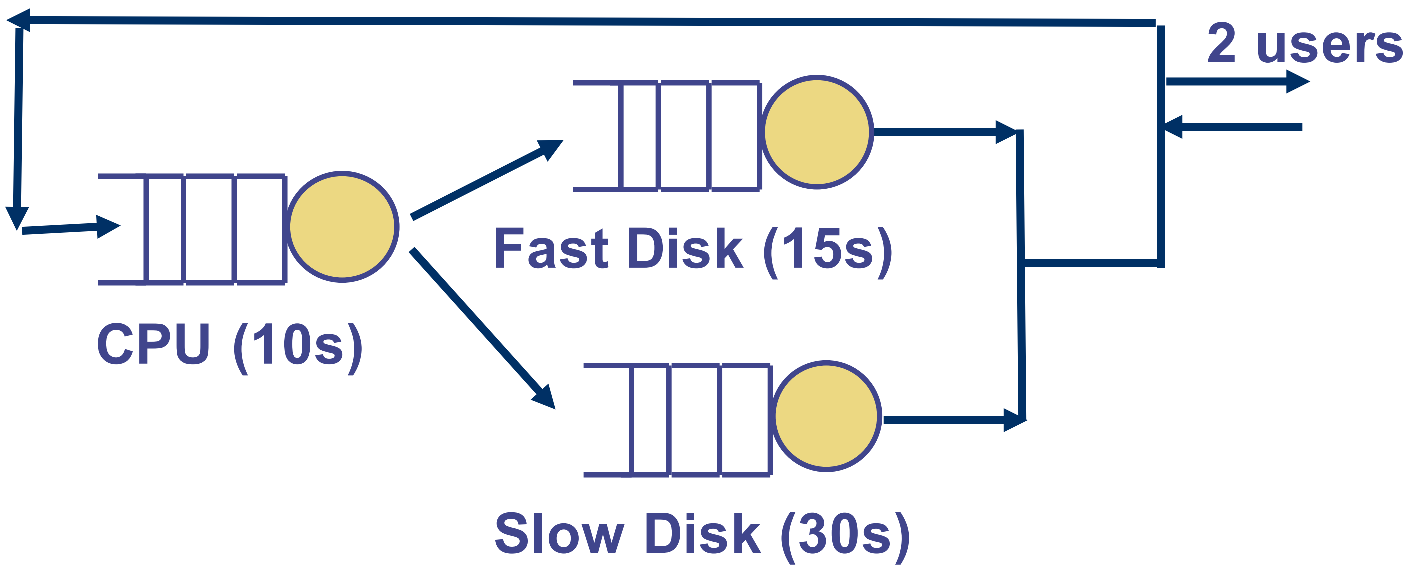
- MVA allows you to find the mean value of throughput, response time etc.
- However, if you are interested to find the probability that the system is in a certain state. MVA cannot give you the answer. You will need to resort to Markov model.

Extensions of MVA

- Closed queueing networks with multiple classes of customers
  - Example: Database servers with 2 classes of customers
    - One class of customers require mean service time of 0.02s, 0.03s and 0.05s from the CPU, fast and slow disk
    - Another class of customers require mean service time of 0.04s, 0.01s and 0.1s from the CPU, fast and slow disk
- Open queueing networks
- Mixed queueing networks

Assumptions behind MVA

- The service time is exponentially distributed
- The service time required at each component is independent
  - For example, MVA assumes that the service time required at CPU is independent of the service time at the disk

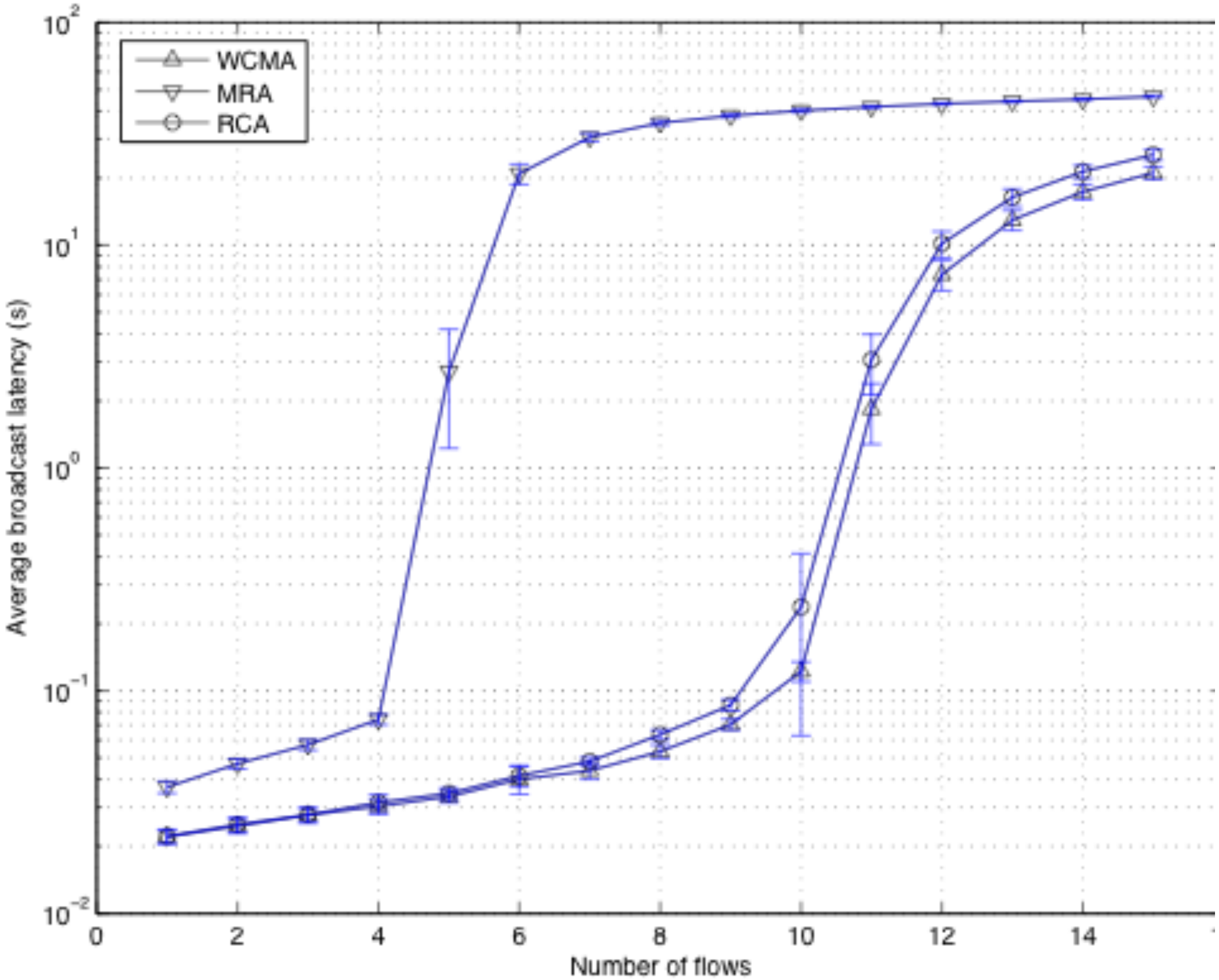


Solution to network of queues

- You have seen two possible methods to solve a network of queues
  - Analytical solution
  - Simulation
- For closed queueing networks with exponentially distributed service time
  - Markov chain
  - MVA
- Commercial simulation tools can deal with hundred of nodes

Multicast in wireless mesh networks

- In my research on designing multicast protocol for wireless mesh networks, we use simulation package *Qualnet* to investigate which of the multicast protocols that we have designed is better
- The network has 400 wireless mesh routers (= 400 queues)



- You can find out more on my research from my web site: http://www.cse.unsw.edu.au/~ctchou/

Analytical solution versus simulation

- Analytical solution
  - Limited to specific cases
    - E.g. Exponential assumptions
  - Efficient computation algorithm exists for certain cases
    - MVA for closed queueing networks with exponential service time

- Simulation
  - Can apply to general settings
    - Difference classes of traffic, protocols etc.
  - Can apply to reasonably large networks too

References

- The primary reference for MVA for closed queueing networks with one class of customer is:
  - Chapter 12, Menasce et al., “Performance by design”
- An alternative reference for MVA is Chapter 6 of Edward Lazowska et al, Quantitative System Performance, Prentice Hall, 1984. (Now out of print but can be download from http://www.cs.washington.edu/homes/lazowska/qsp/)
  - Note that Chapter 6 has a wider coverage. It talks about open queueing network as well as approximation method too.
- For a formal mathematical proof of Arrival Theorem, see Bertsekas and Gallager, “Data networks”, Section 3.8.3