COMP9334 Capacity Planning for Computer Systems and Networks

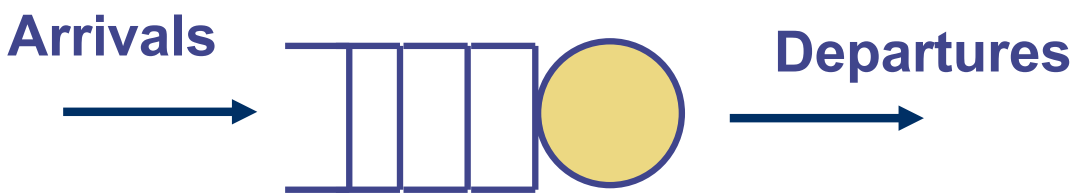
Week 5: Non-markovian queueing models and queueing disciplines

Week 3: Queues with Poisson arrivals (1)

- Single-server M/M/1

Exponential inter-arrivals (λ)

Exponential service time (μ)

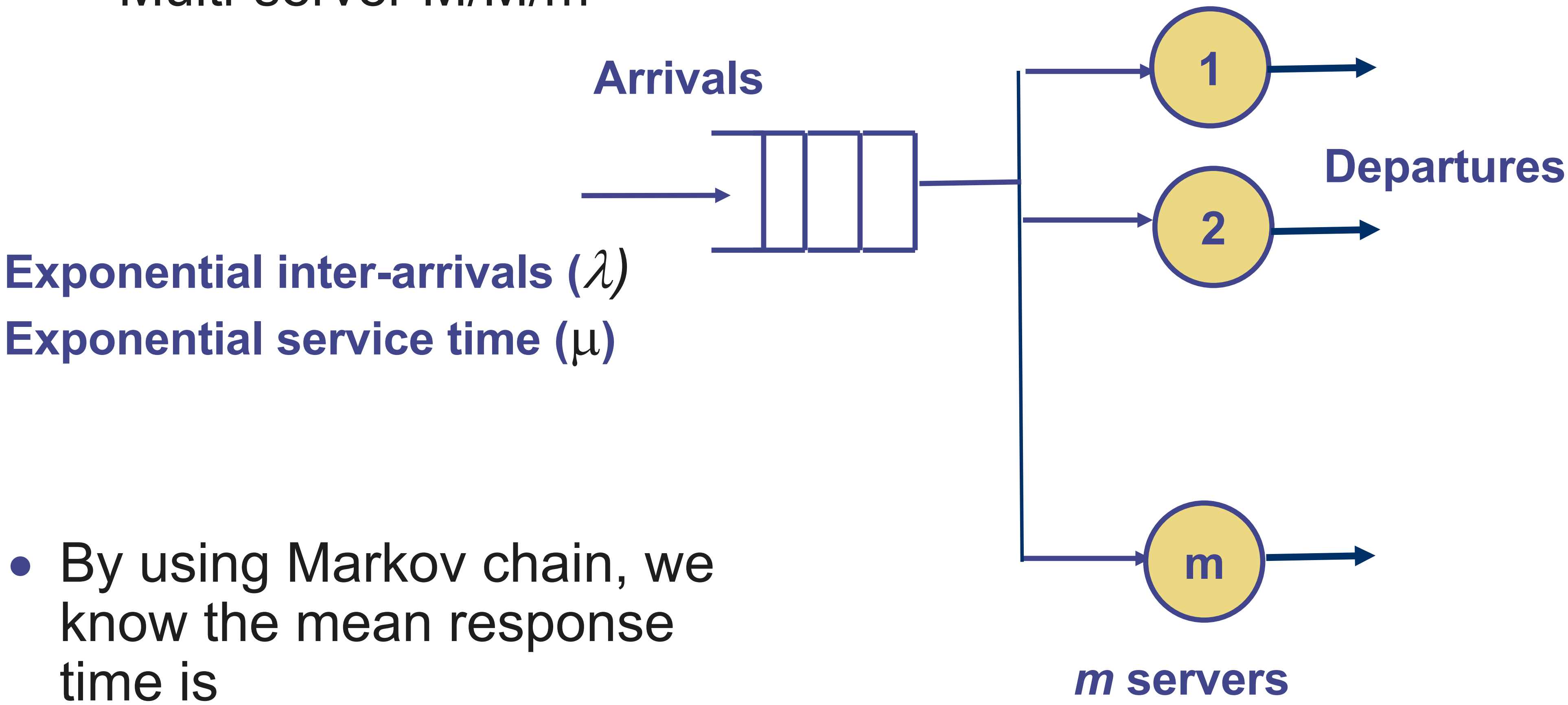


- By using a Markov chain, we can show that the mean response time is:

$$= \frac{1}{\mu - \lambda}$$

Week 3: Queues with Poisson arrivals

- Multi-server M/M/m



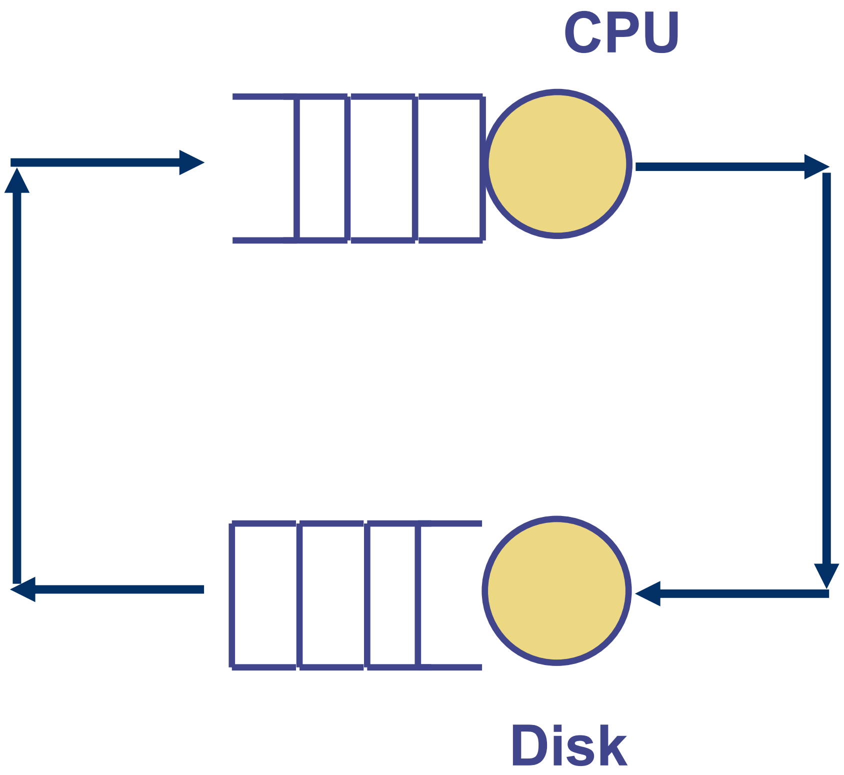
- By using Markov chain, we know the mean response time is

$$T = \frac{C(\rho, m)}{m\mu(1-\rho)} + \frac{1}{\mu}$$

$$\rho = \frac{\lambda}{m\mu} \qquad C(\rho, m) = \frac{\frac{(m\rho)^m}{m!}}{(1-\rho)\sum_{k=0}^{m-1}\frac{(m\rho)^k}{k!} + \frac{(m\rho)^m}{m!}}$$

Week 4: Closed-queueing networks

- Analyse closed-queueing network with Markov chain
  - The transition between states is caused by an arrival or a departure according to exponential distribution

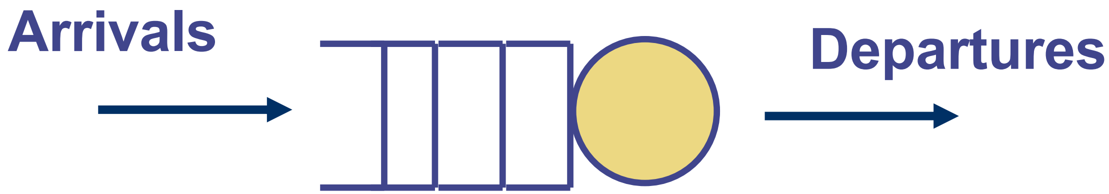


- General procedure
  - Identify the states
  - Find the state transition rates
  - Set up the balance equations
  - Solve for the steady state probabilities
  - Find the response time etc.

This lecture: Road Map

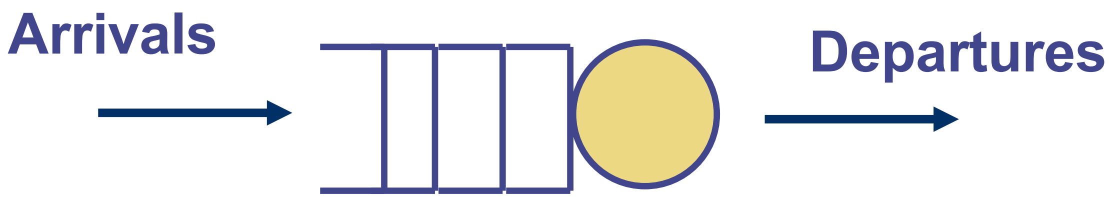
- Single-server queues
  - What if the arrival rate and/or the service rate is not exponentially distributed
- Multi-server queues
  - What if the arrival rate and/or the service rate is not exponentially distributed
- Queuing disciplines
- Processor sharing

General single-server queues



- Need to specify the
  - Inter-arrival time probability distribution
  - Service time probability distribution
- Independence assumptions
  - All inter-arrival times are independent
  - All service times are independent
    - The amount of service of customer A needs is independent of the amount of time customer B needs
  - The inter-arrival time and service time are independent of each other
- Under the independence assumption, we can analyse a number of types of single server queues
  - Without the independence assumption, queueing problems are very difficult to solve!

Classification of single-server queues



- Recall Kendall's notation: "M/M/1" means
  - "M" in the 1st place means inter-arrival time is exponentially distributed
  - "M" in the 2nd place means service time probability is exponentially distributed
  - "1" in 3rd position means 1 server
- We use a "G" to denote a general probability distribution
  - Meaning any probability distribution
- Classification of single-server queues:

		Service time Distribution:	
		Exponential	General
Inter-arrival time distribution:	Exponential	M/M/1	M/G/1
	General	G/M/1	G/G/1

Example M/G/1 queue problem

- Consider an e-mailer server
- E-mails arrive at the mail server with a Poisson distribution with mean arrival rate of 1.2 messages/s
- The service time distribution of the emails are:
  - 30% of messages processed in 0.1 s, 50% in 0.3 s, 20% in 2 s
- What is
  - Average waiting time for a message?
  - Average response time for a message?
  - Average number of messages in the mail system?
- This is an M/G/1 queue problem
  - Arrival is Poisson
  - Service time is not exponential
- In order to solve an M/G/1 queue, we need to understand what the moment of a probability distribution is.

Revision: moment of a probability distribution (1)

- Consider a discrete probability distribution
  - There are *n* possible outcomes: $x_1, x_2, \ldots, x_n$
  - The probability that x_i occurs is p_i
- Example: For a fair dice
  - The possible outcomes are 1,2,…, 6
  - The probability that each outcome occurs is 1/6

- The first moment (also known as the mean or expected value) is

$$E[X] = \sum_{i=1}^{n} x_i p_i$$

- For a fair dice, the first moment is

= 1 * 1/6 + 2 * 1/6 + … + 6 * 1/6 = 3.5

Revision: moment of a probability distribution (2)

- The second moment of a discrete probability distribution is

$$E[X^2] = \sum_{i=1}^{n} x_i^2 p_i$$

- For a fair dice, the second moment is

$= 1^2 * 1/6 + 2^2 * 1/6 + \ldots + 6^2 * 1/6$

- You can prove that
  - Second moment of X = $(E[X])^2$ + Variance of X

- Note: The above definitions are for discrete probability distribution. We will look at continuous probability distribution a moment later

Solution to M/G/1 queue

- M/G/1 analysis is still tractable
- M/G/1 is no longer a Markov chain
- For a M/G/1 queue with the characteristics
  - Arrival is Poisson with rate λ
  - Service time S has
    - Mean = $1/\mu$ = E[S] = First moment
    - Second moment = $E[S^2]$
- The mean waiting time W of a M/G/1 queue is given by the Pollaczek-Khinchin (P-K) formula:

$$W = \frac{\lambda E[S^2]}{2(1-\rho)} \quad \text{where} \quad \rho = \frac{\lambda}{\mu}$$

Back to our example queueing problem (1)

- Consider an e-mailer server
- E-mails arrive at the mail server with a Poisson distribution with mean arrival rate of 1.2 messages/s
- The service time distribution of the emails are:
  - 30% of messages processed in 0.1 s, 50% in 0.3 s, 20% in 2 s

- *Exercise*: In order to find the mean waiting time using the P-K formula, we need to know
  - Mean arrival rate,
  - Mean service time, and,
  - Second moment of service time.
- Can you find them?

Back to our example queueing problem (2)

- Consider an e-mailer server
- E-mails arrive at the mail server with a Poisson distribution with mean arrival rate of 1.2 messages/s
- The service time distribution of the emails are:
  - 30% of messages processed in 0.1 s, 50% in 0.3 s, 20% in 2 s

- Solution
  - Mean arrival rate = 1.2 messages/s
  - Mean service time
    = 0.1*0.3+0.3*0.5+2*0.2 = 0.58s
  - Second moment of the service time
    = $0.1^2 * 0.3 + 0.3^2 * 0.5 + 2^2 * 0.2 = 0.848\ s^2$
- You now have everything you need to compute the mean waiting time using the P-K formula

Back to our example queueing problem (3)

- Since
  - Mean arrival rate λ = 1.2 messages/s
  - Mean service time (E[S] or 1 / μ) = 0.58s
  - Second moment of mean service time $E[S^2] = 0.848\ s^2$
- Utilisation $\rho = \lambda / \mu = \lambda E[S] = 1.2 * 0.58 = 0.696$
- Substituting these values in the P-K formula

$$W = \frac{\lambda E[S^2]}{2(1-\rho)}$$

W = 1.673s.

- How about:
  - Average response time for a message
  - Average number of messages in the mail system

Back to our example queueing problem (4)

Since the mean waiting time W = 1.673s.

The mean response time T is

$$T = W + E[S] = 1.673 + 0.58 = 2.253$$

By Little's Law,
Average # messages in the system
= Throughput x mean response time
$= \lambda T = 1.2 * 2.253 = 2.704$ messages

Exercise: Can you use mean waiting time and Little's Law to determine the mean number of messages in the queue?

Understanding the P-K formula

- Since the Second moment of S = $E[S]^2$ + Variance of S
- We can write the P-K formula as
  - Meaning waiting time =

$$W = \frac{\lambda(E[S]^2 + \sigma_S^2)}{2(1-\rho)}$$

- Smaller variance in service time ➔ smaller waiting time
- M/D/1 is a special case of M/G/1
  - "D" stands for deterministic: Constant service time E[S] and Variance of S = 0
  - For the same value of ρ and E[S], deterministic has the smallest mean response time

Moments for continuous probability density

- Exponential function is a continuous probability density
- If a random variable X has continuous probability density function f(x), then its
  - first moment (= mean, expected value) E[X] and
  - second moment $E[X^2]$

  are given by

$$E[X] = \int x f(x) dx$$

$$E[X^2] = \int x^2 f(x) dx$$

- If the service time S is exponential with rate μ, then
  - $E[S] = 1/\mu$
  - $E[S^2] = 2/\mu^2$

M/M/1 as a special case of M/G/1

- Let us apply the result of the M/G/1 queue to exponential service time
  - Let us put E[S] = 1/ μ and E[S^2] = 2 / μ^2 in the P-K formula:

$$W = \frac{\lambda E[S^2]}{2(1-\rho)}$$

- We get

$$W = \frac{\rho}{\mu(1-\rho)}$$

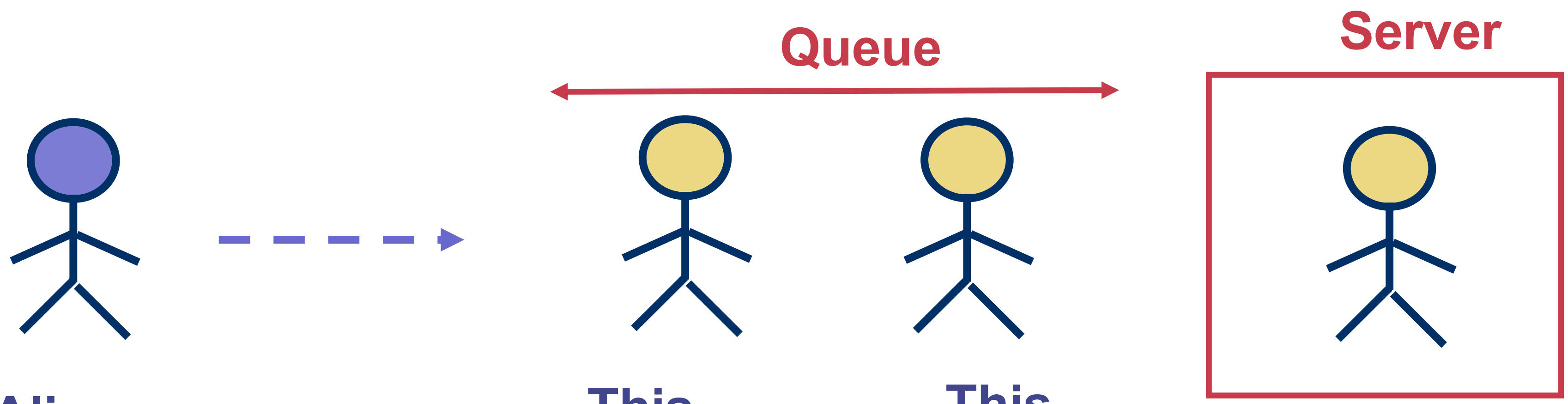
  - Which is the same as the M/M/1 queue waiting time formula that we derive in Week 3

Remark on M/G/1

$$W = \frac{\lambda E[S^2]}{2(1-\rho)}$$

- $\rho \rightarrow 1$, $W \rightarrow \infty$

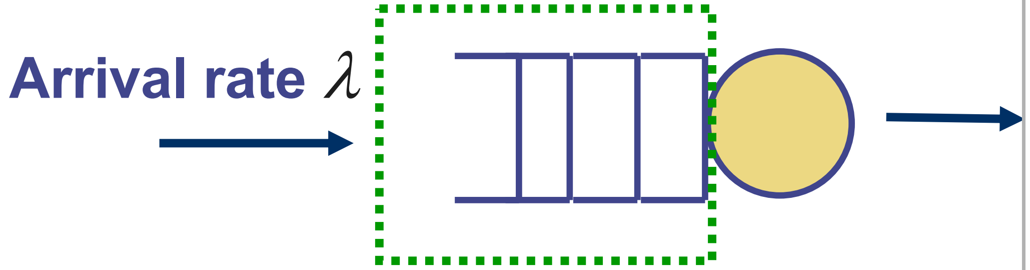
Deriving the P-K formula (1)



- How long does Alice (the arriving customer) need to wait before she gets served?

Deriving the P-K formula (2)

- Let
  - W = Mean waiting time
  - N = Mean number of customers in the queue
  - 1/ μ = Mean service time
  - R = Mean residual service time
- We can prove that
  - W = N * (1/ μ) + R



- Applying Little's Law to the queue
  - N = λ W

Substitution

$$W = \lambda \times W \times \frac{1}{\mu} + R \Rightarrow W = \frac{R}{1-\rho}$$

where $\rho = \frac{\lambda}{\mu}$

Deriving P-K formula (3)

- We have just showed that the mean waiting time in a M/G/1 queue is

$$W = \frac{R}{1-\rho}$$

- The P-K formula says

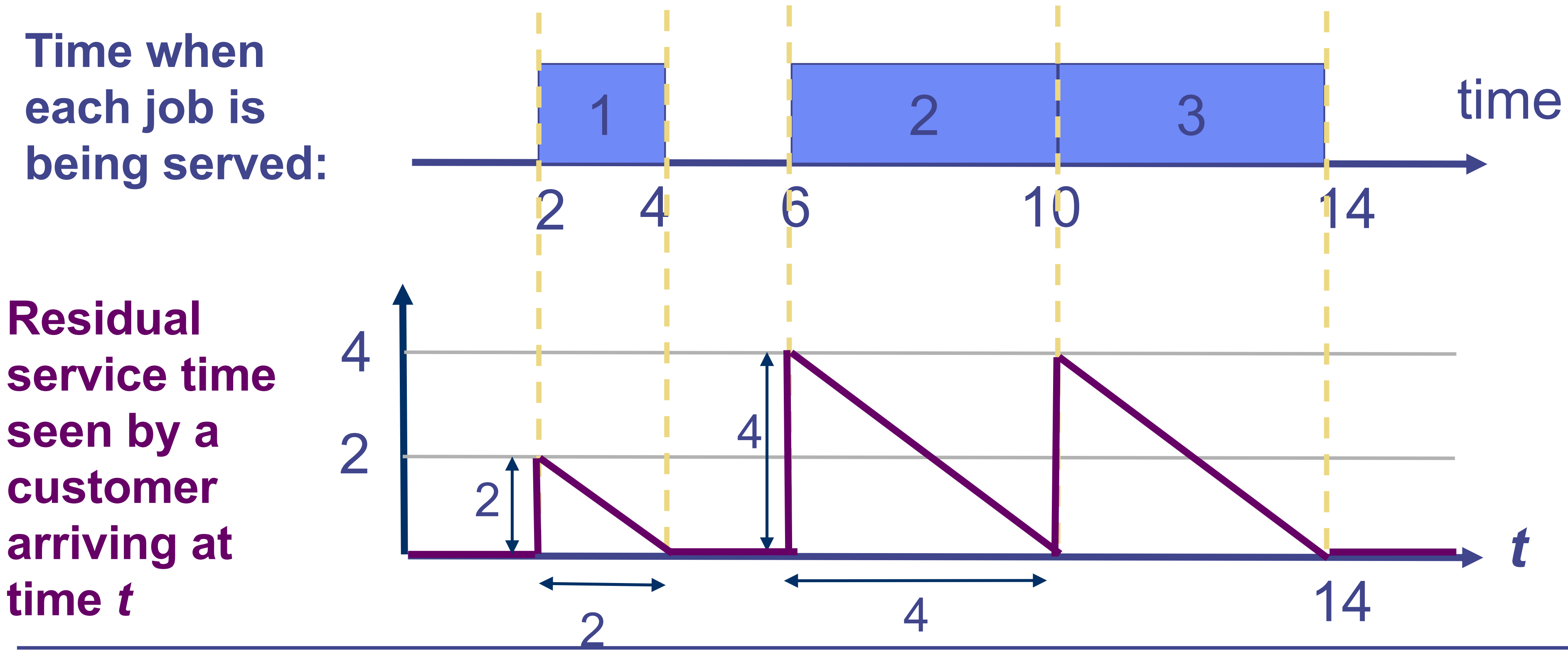
$$W = \frac{\lambda E[S^2]}{2(1-\rho)}$$

- We can prove the P-K formula if we can show that the mean residual time R is

$$R = \frac{1}{2}\lambda E[S^2]$$

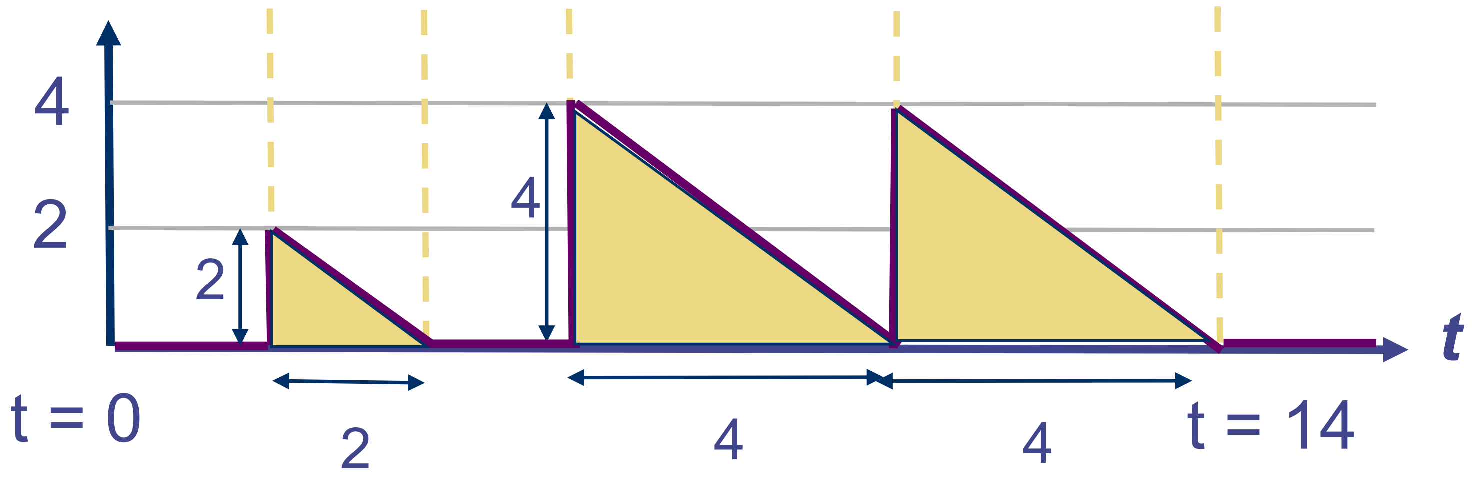
How residual service time changes over time?

Job index	Arrival time	Processing time required
1	2	2
2	6	4
3	8	4



What is the mean residual time …

Residual service time seen by a customer arriving at time *t*



Mean residual time seen by an arriving customer over time [0,14]

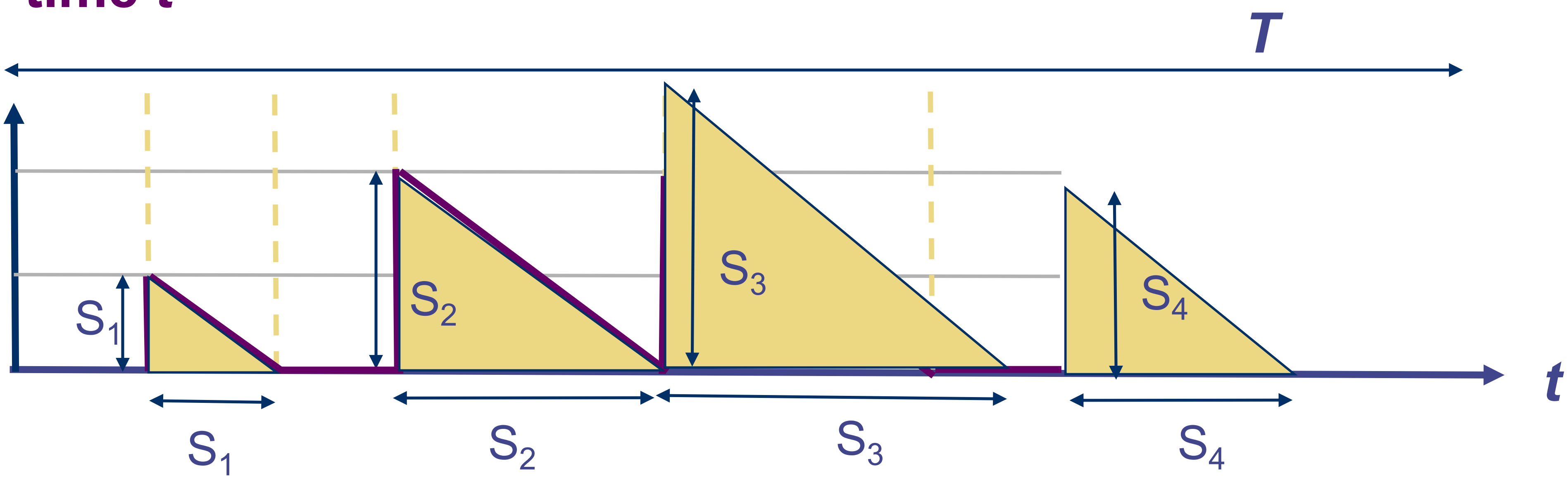
$$= \frac{\text{Area under the curve over } [0{,}14]}{14}$$

Service time!

$$= \frac{\frac{1}{2} \times 2^2 + \frac{1}{2} \times 4^2 + \frac{1}{2} \times 4^2}{14}$$

In general

Residual service time seen by a customer arriving at time *t*



Assuming M jobs are completed in time T
Mean residual time

$$= \frac{\sum_{i=1}^{M} \frac{1}{2} S_i^2}{T} = \frac{1}{2} \underbrace{\frac{\sum_{i=1}^{M} S_i^2}{M}} \underbrace{\frac{M}{T}} = \frac{1}{2} \underbrace{E[S^2]} \underbrace{\lambda}$$

The P-K formula

- Thus, the mean residual time R is

$$R = \frac{1}{2}\lambda E[S^2]$$

- By substituting this into $W = \dfrac{R}{1-\rho}$

- We get the P-K formula

- This derivation also shows that the waiting time is proportional to the residual service time
- The residual service time is proportional to the 2nd moment of service time

G/G/1 queue

- G/G/1 queue are harder to analyse
- Generally, we cannot find an explicit formula for the the waiting time or response time for a G/G/1 queue
- Results on G/G/1 queue include
  - Approximation results
  - Bounds on waiting time

Approximate G/G/1 waiting time

- There are many different methods to find the approximate waiting time for a G/G/1 queue
- Most of the approximation works well when the traffic is heavy, i.e. when the utilisation ρ is high
- Let
  - Mean arrival rate = λ
  - Variance of inter-arrival time = σ_a^2
  - Service time S has mean $1/\mu$ = E[S]
  - Variance of service time = σ_s^2
- The approximate waiting time for a G/G/1 queue is

$$W \approx \frac{\lambda^2(\sigma_a^2 + \sigma_s^2)}{1 + \lambda^2\sigma_s^2} \frac{\lambda(E[S]^2 + \sigma_s^2)}{2(1-\rho)} \quad \text{where} \quad \rho = \frac{\lambda}{\mu}$$

- Note: $\rho \to 1$, $W \to \infty$
- Large variance means large waiting time

Bounds for G/G/1 waiting time

- Let
  - Mean arrival rate = λ
  - Variance of inter-arrival time = σ_a^2
  - Service time S has mean 1/ μ = E[S]
  - Variance of service time = σ_s^2
- A bound for the waiting time for a G/G/1 queue is

$$W \leq \frac{\lambda(\sigma_a^2 + \sigma_s^2)}{2(1-\rho)}$$

- Note that the bound suggests that large variance means large waiting time

Approximation for G/G/m queue

- Only approximate waiting time available for G/G/m
- The waiting time is

$$W_{G/G/m} = W_{M/M/m} \frac{C_a^2 + C_s^2}{2}$$

where

$$W_{M/M/m} = \text{Waiting time of M/M/m queue}$$

$$C_a = \text{Coeff of variation of inter-arrival time}$$

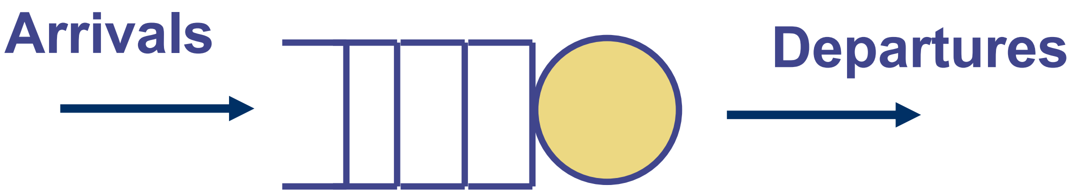
$$C_b = \text{Coeff of variation of service time}$$

- Coefficient of variation of a random variable *X*

= Standard deviation of *X* / mean of *X*

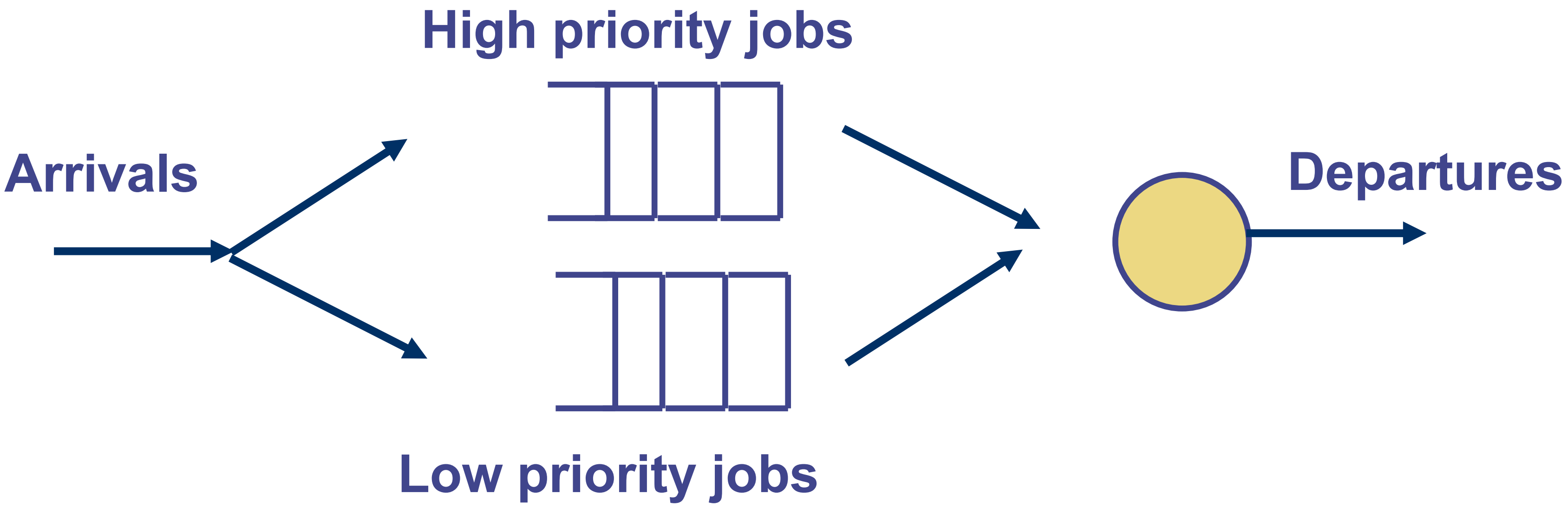
Note: Variance in arrival or service time increases queueing

Queuing disciplines



- We have focused on *first-come first-serve* (FCFS) queues so far
- However, sometimes you may want to give some jobs a higher priority than others
- Priority queues can be classified as
  - Non-preemptive
  - Preemptive resume

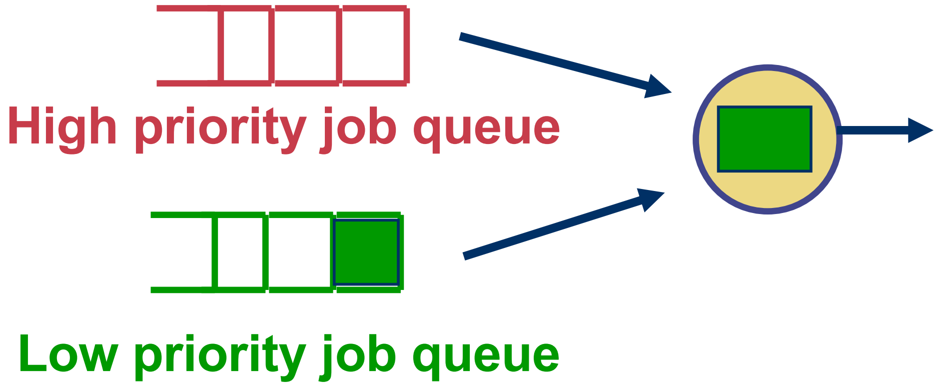
What is priority queueing?

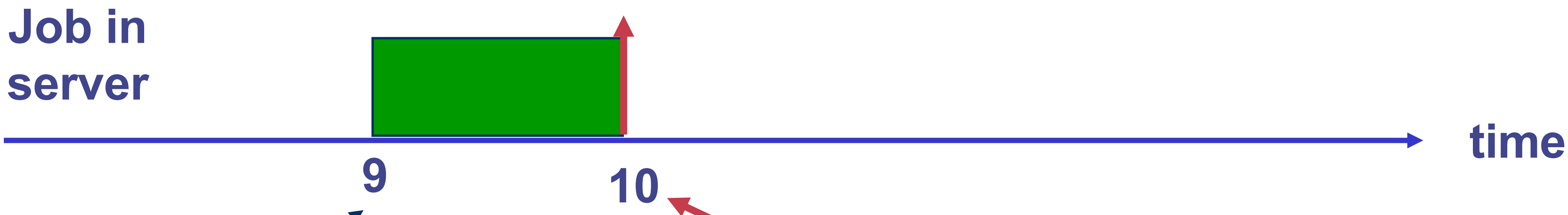


- A job with low priority will only get served if the high priority queue is empty
- Each priority queue is a FCFS queue
- Exercise: If the server has finished a job and finds 1 job in the high priority queue and 3 jobs in the low priority queue, which job will the server start to work on?
  - Repeat the exercise when the high priority queue is empty and there are 3 jobs in the low priority queue.

Preemptive and non-preemptive priority (1)

- Example:





Time t = 9

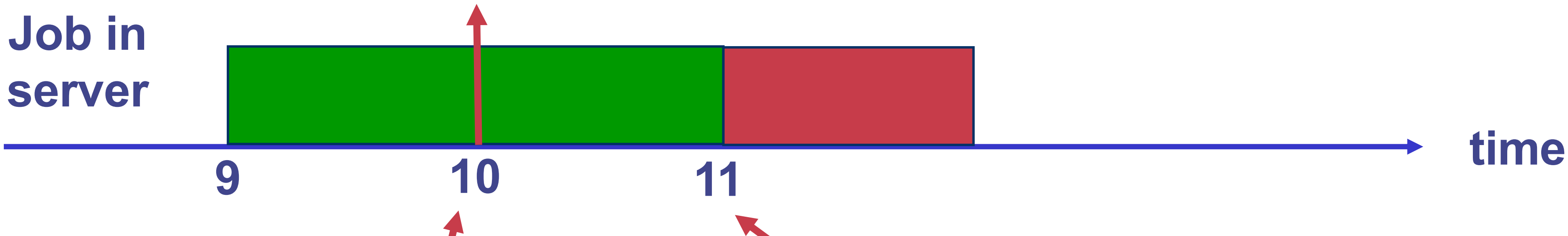
- The high priority job queue is empty
- The server starts serving a low priority job which requires 2s of processing

Time t = 10：A high priority job requiring 1s of processing arrives

Preemptive and non-preemptive priority (2)

- ***Non-preemptive***:
  - A job being served will not be interrupted (even if a higher priority job arrives in the mean time)

- Example: High priority job (red), low priority job (green)



Time t = 10：A high priority job requiring 1s of processing arrives.

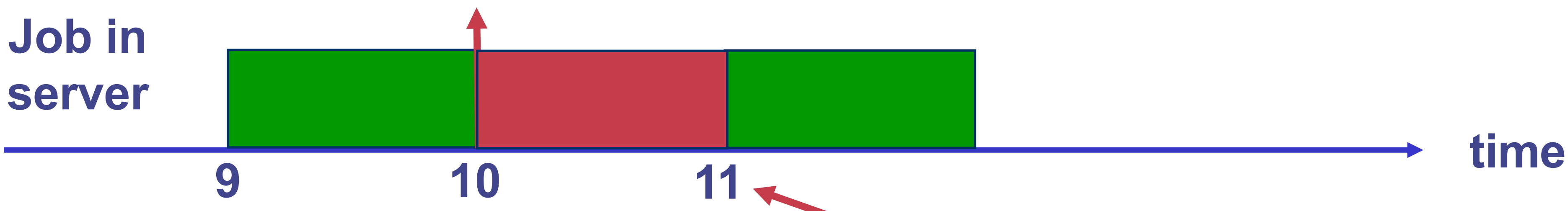
The job joins the high priority queue

Time t = 11：Server finishes processing the low priority job. It takes the high priority job in from the queue

Preemptive and non-preemptive priority (3)

- ***Preemptive resume***:
  - Higher priority job will interrupt a lower priority job under service. Once all higher priorities served, an interrupted lower priority job is resumed.

- Example: High priority job (red), low priority job (green)

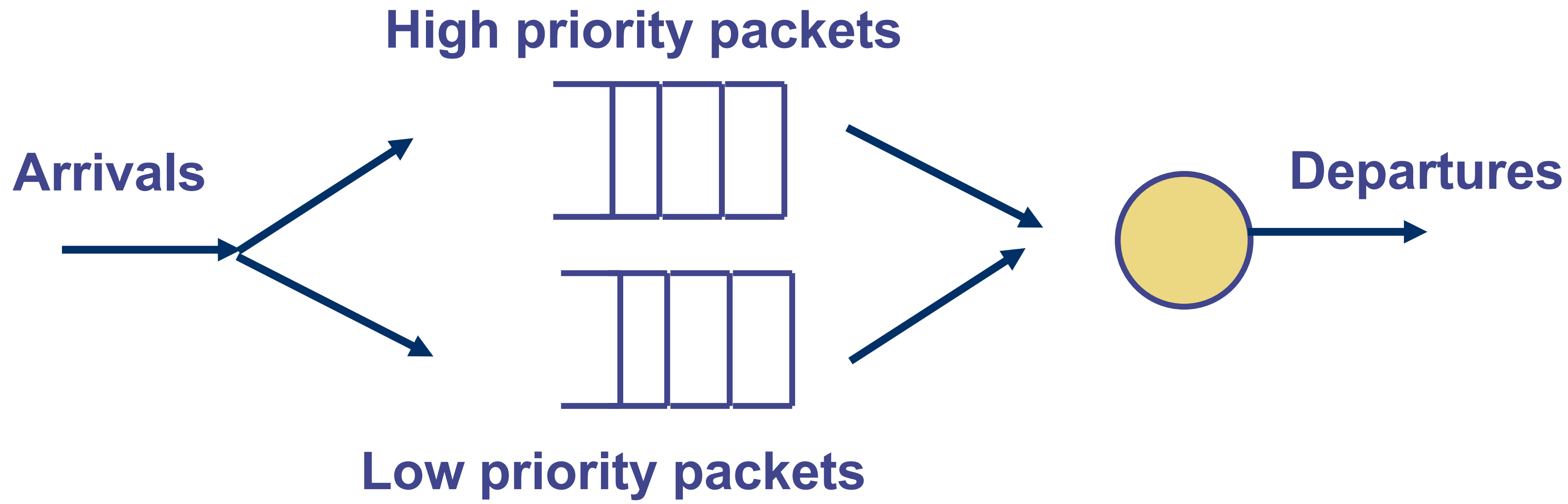


Time t = 10：A high priority job requiring 1s of processing arrives.

The server starts processing the high priority job immediately

Time t = 11：Server finishes processing the high priority job. Since no high priority job arrives in (10,11], the high priority job queue is empty, it resumes processing the low priority job that is pre-empted at time t = 10

Example of non-preemptive priority queueing



- Example: In the output port of a router, you want to give some packets a higher priority
  - In Differentiated Service
    - Real-time voice and video packets are given higher priority because they need a lower end-to-end delay
    - Other packets are given lower priority
- You cannot preempt a packet transmission and resume its transmission later
  - A truncated packet will have a wrong checksum and packet length etc.

Example of preemptive resume priority queueing

- E.g. Modelling multi-tasking of processors
- Can interrupt a job but you need to do context switching (i.e. save the registers for the current job so that it can be resumed later)

M/G/1 with priorities

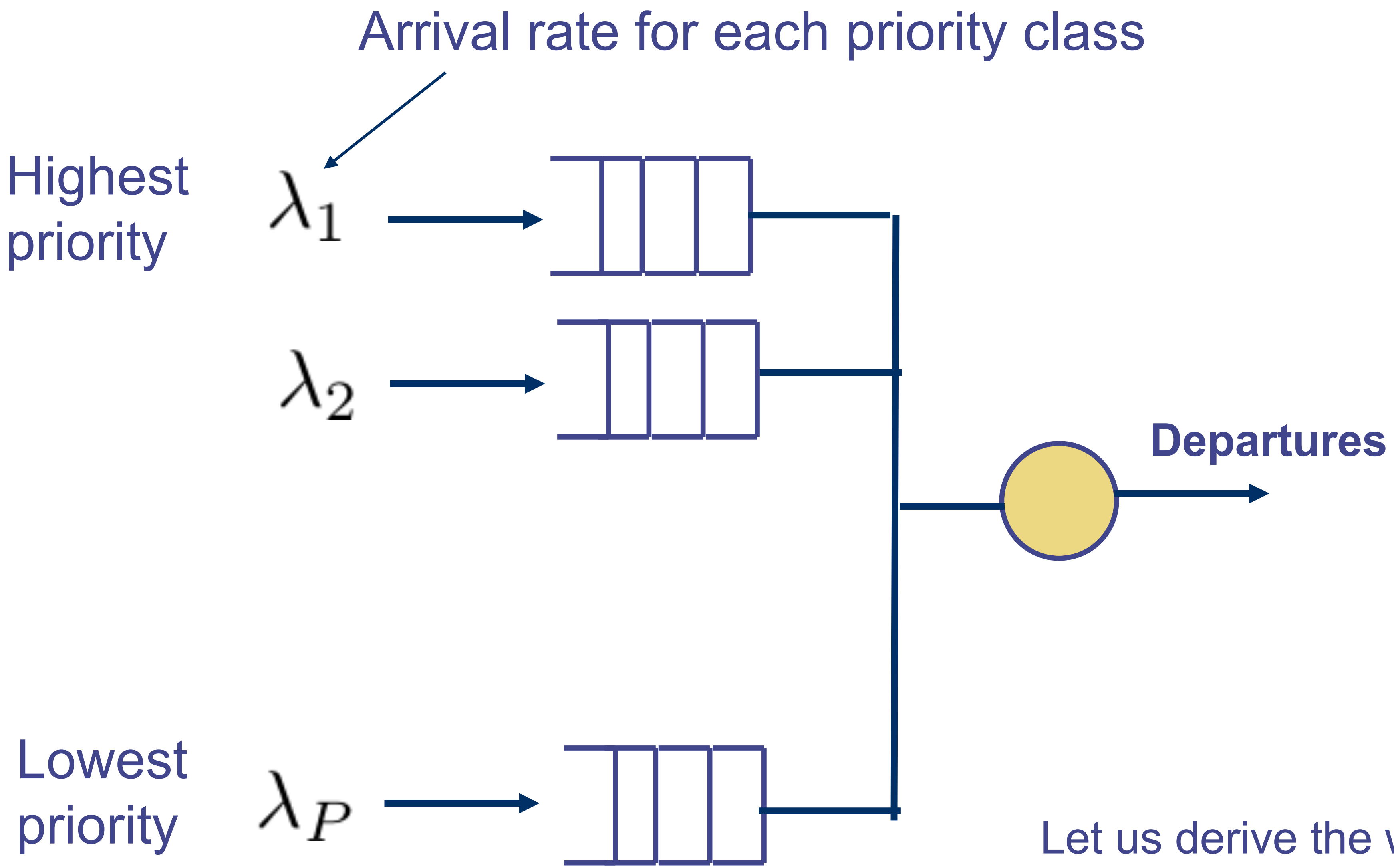
- Separate queue for each priority (see picture next page)
  - Classified into P priorities before entering a queue
  - Priorities numbered 1 to P, Queue 1 being the highest priority
- Arrival rate of priority class p is

$$\lambda_p \text{ where } p = 1, \cdots P$$

- Average service time and second moment of class p requests is given by

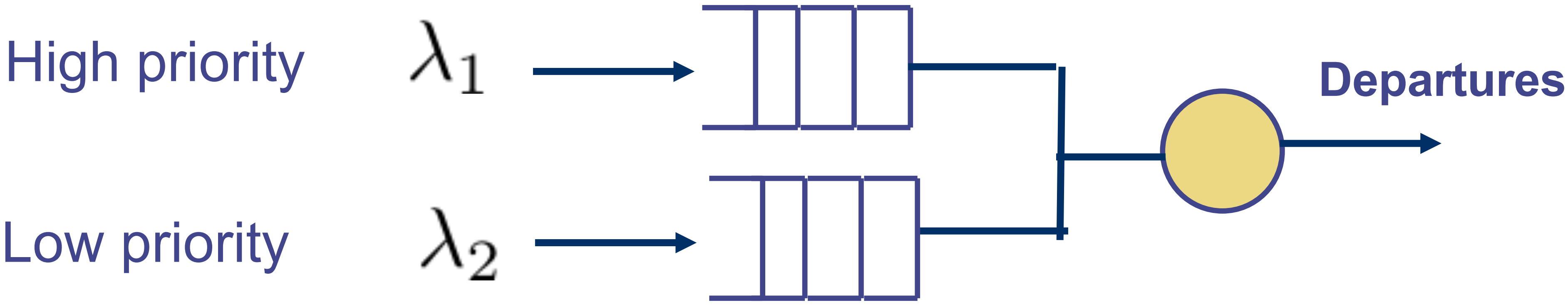
$$E\left[S_p\right] \text{ and } E\left[S_p^2\right]$$

Priority queue



Let us derive the waiting time for P = 2

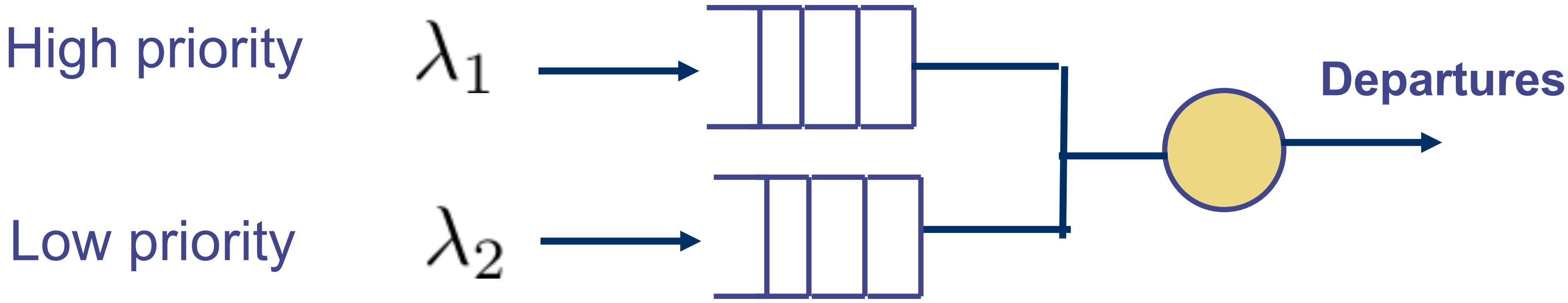
Deriving the non-preemptive queue result (1)



- S_1 - service time for Class 1 with mean $E[S_1]$
- W_1 = mean waiting time for Class 1 customers
- N_1 = number of Class 1 customers in the queue
- R = mean residual service time when a customer arrives
- We have for Class 1: $W_1 = N_1 E[S_1] + R$
- Little's Law: $N_1 = \lambda_1 W_1$

$$W_1 = \frac{R}{1-\rho_1} \quad \text{where} \quad \rho_1 = \lambda_1 E[S_1]$$

Deriving the non-preemptive queue result (2)



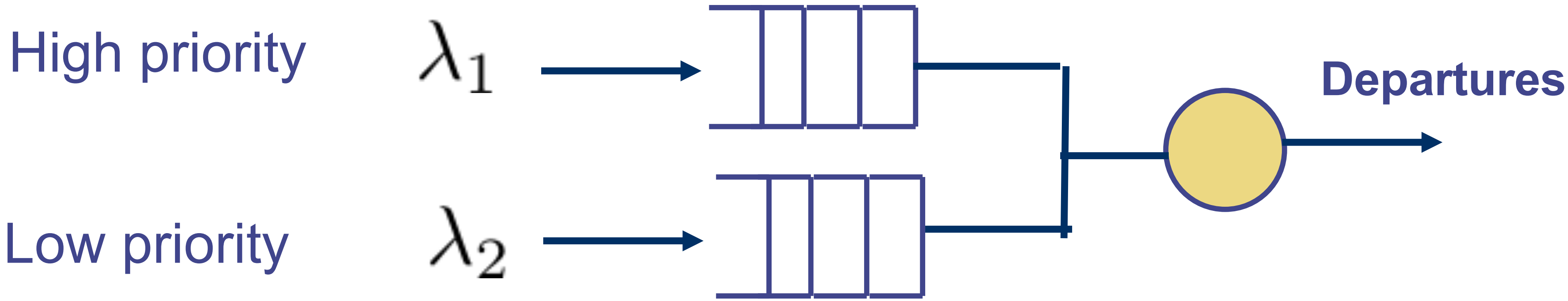
- To find the residual service time R, note that the customer in the server can be a high or low priority customer, we have

$$R = \frac{1}{2}E[S_1^2]\lambda_1 + \frac{1}{2}E[S_2^2]\lambda_2$$

- The waiting time is therefore

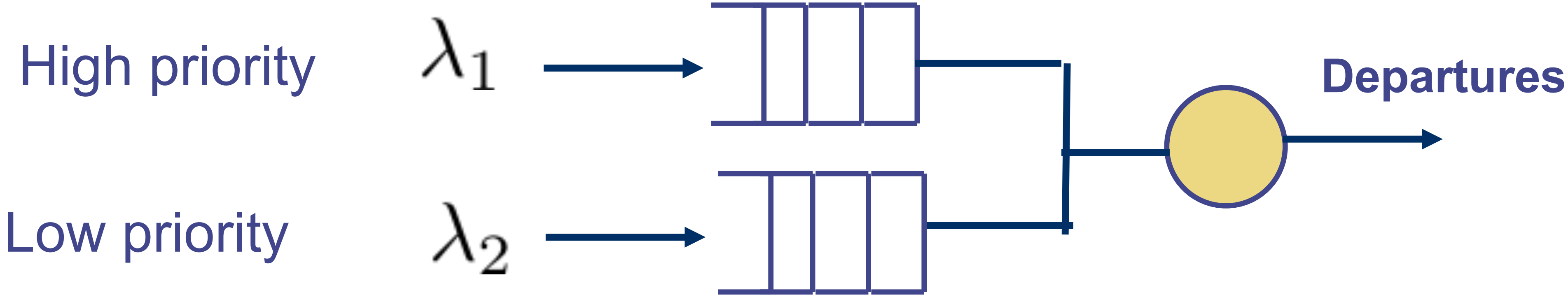
$$W_1 = \frac{R}{1-\rho_1} \quad \text{where} \quad \rho_1 = \lambda_1 E[S_1]$$

Deriving the non-preemptive queue result (3)



- S_2 - service time for Class 2 with mean $E[S_2]$
- W_2 = mean waiting time for Class 2 customers
- N_2 = number of Class 2 customers in the queue
- R = mean residual service time when a customer arrives

Deriving the non-preemptive queue result (4)



- For Class 2 customers:

$$W_2 = R + N_2 E[S_2] + N_1 E[S_1] + \underline{\lambda_1 W_2} E[S_1]$$

Average number of customers already in Queues 1 and 2 when a Class 2 customer arrives

Average number of customers that arrive in Queue 1 after a low priority customer arrives

Deriving the non-preemptive queue result (5)

$$W_2 = R + N_2 E[S_2] + N_1 E[S_1] + \lambda_1 W_2 E[S_1]$$

- Little's Law to Queue 1:

$$N_1 = \lambda_1 W_1$$

- Little's Law to Queue 2:

$$N_2 = \lambda_2 W_2$$

- Combining all of the above

$$W_2 = \frac{R + \rho_1 W_1}{1 - \rho_1 - \rho_2}$$

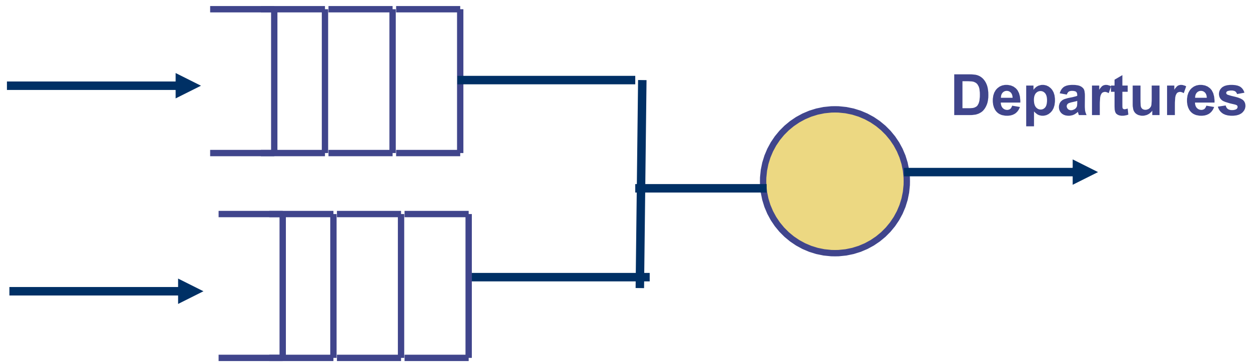
Where

$$\rho_2 = \lambda_2 E[S_2]$$
$$\rho_1 = \lambda_1 E[S_1]$$

Deriving the non-preemptive queue result (6)

High priority λ_1

Low priority λ_2



$$W_2 = \frac{R}{(1-\rho_1)(1-\rho_1-\rho_2)}$$

$$W_1 = \frac{R}{1-\rho_1}$$

where

$$\rho_1 = \lambda_1 E[S_1]$$
$$\rho_2 = \lambda_2 E[S_2]$$
$$R = \frac{1}{2}E[S_1^2]\lambda_1 + \frac{1}{2}E[S_2^2]\lambda_2$$

Non-preemptive Priority with *P* classes

Waiting time of priority class k

$$W_k = \frac{R}{(1-\rho_1-...-\rho_{k-1})(1-\rho_1-...-\rho_k)}$$

where

$$R = \frac{1}{2}\sum_{i=1}^{P} E[S_i^2]\lambda_i$$

$$\rho_i = \lambda_i E[S_i] \text{ for } i = 1,...,P$$

Example

- Router receives packet at 1.2 packets/ms (Poisson), only one outgoing link
- Assume 50% packet of priority1, 30% of priority 2 and 20% of priority 3. Mean and second moment given in the table below.
- What is the average waiting time per class?
- Solution to be discussed in class.

Priority	**Mean (ms)**	**2nd Moment (ms^2)**
1	0.5	0.375
2	0.4	0.400
3	0.3	0.180

Pre-emptive resume priority (1)

- Can be derived using a similar method to that used for non-preemptive priority
- The key issue to note is that a job with priority k can be interrupted by a job of higher priority even when it is in the server
- For *k = 1* (highest priority), the response time T_1 is:

$$T_1 = E[S_1] + \frac{R_1}{(1-\rho_1)}$$

where

$$R_1 = \frac{1}{2}E[S_1^2]\lambda_1$$

$$\rho_1 = E[S_1]\lambda_1$$

A highest priority job only has to wait for the highest priority jobs in front of it.

Preemptive resume priority (2)

- For k ≥ 2, we have response time for a job in Class k:

$$T_k = E[S_k] + \frac{R_k}{\underline{1 - \rho_1 - \ldots - \rho_k}} + \underline{(\sum_{i=1}^{k-1} \rho_i) T_k}$$

An arriving job in Priority Class k needs to wait for all the jobs in Priority Classes 1 to k, that are already in the system when it arrives, to complete.

An arriving job of priority *k* has to wait for all the jobs of higher priorities that arrive during the time that this job is waiting in the queue and in the server.

$$R_k = \frac{1}{2} \sum_{i=1}^{k} E[S_i^2] \lambda_i$$

Note that R_k contains only terms of priority *k* or higher since a job with priority *k* cannot be interrupted by jobs with a lower priority. In other words, a job with priority *k* does not see the residual service time of lower priority classes.

Preemptive resume priority (3)

- Solving these equations, we have the response time of Class k jobs is:

$$T_k = T_{k,1} + T_{k,2}$$

where

$$T_{k,1} = \frac{E[S_k]}{(1 - \rho_1 - ... - \rho_{k-1})}$$

$$T_{k,2} = \frac{R_k}{(1 - \rho_1 - ... - \rho_{k-1})(1 - \rho_1 - ... - \rho_k)}$$

$$R_k = \frac{1}{2}\sum_{i=1}^{k} E[S_i^2]\lambda_i$$

Other queuing disciplines

- There are many other queueing disciplines, examples include
  - Shortest processing time first
  - Shortest remaining processing time first
  - Shortest expected processing time first
- Optional: For an advanced exposition on queueing disciplines, see Kleinrock, “Queueing Systems Volume 2”, Chapter 3.

Processor sharing (PS)

- We have so far assumed that the processor performs work on a first-come-first-serve basis
- However, this is not how CPUs perform tasks
- Consider an example: a CPU has a job queue with three tasks called Tasks 1, 2 and 3 in it
  - CPU works on Task 1 for a certain amount of time (called a quantum) and then returns the task to the job queue if it is not yet finished
  - CPU works on Task 2 for a quantum and returns the task to the job queue if it is not yet finished
  - CPU works on Task 3 for a quantum and returns the task to the job queue if it is not yet finished

Modelling processor sharing

- We assume the context switching time is negligible
- In a duration of time when there are n jobs in the job queue, each job receives 1/n of the service

PS: Example 1

- Example 1:
  - At time 0, there are 2 jobs in the job queue
  - Job 1 still needs 5 seconds of service
  - Job 2 still needs 3 seconds of service

  - Assuming no more jobs will arrive, determine the time at which the jobs will be completed

PS: Example 2

- Example 2:
  - At time 0, there are 2 jobs in the job queue
    - Job 1 still needs 5 seconds of service
    - Job 2 still needs 3 seconds of service
  - Job 3 arrives at time = 1 second and requires 4 seconds of service
  - Job 4 arrives at time = 2 second and requires 1 second of service
  - No more jobs will arrive after Job 4

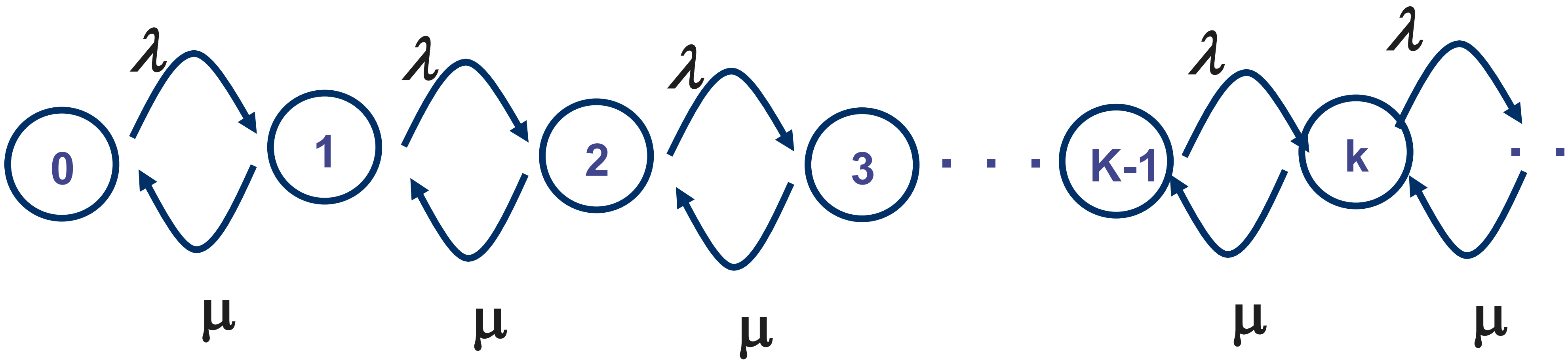
  - Questions:
    - Without computing the finished times for Jobs 1 and 3, are you able to tell which of these two jobs will finish first?

    - Determine the time at which the jobs will be completed

M/M/1/PS queues

- Jobs arrive according to Poisson distribution
- Exponential service time
- One processor using processor sharing

- State *n* = there are *n* jobs in the job queue
- State diagram: same as M/M/1 queue and there is a reason for that



Summary

- We have studied a few types of non-Markovian queues
  - M/G/1, G/G/1, G/G/m
  - M/G/1 with priority
- Key method to derive the M/G/1 waiting time (with and without priority) is via the *residual service time*
- Processor sharing (PS)

References

- Recommended reading
  - Bertsekas and Gallager, “Data Networks”
    - Section 3.5 for M/G/1 queue
    - Section 3.5.3 for priority queuing
    - The result on G/G/1 bound is taken from Section 3.5.4
- Optional reading
  - Harchol-Balter, Chapter 22