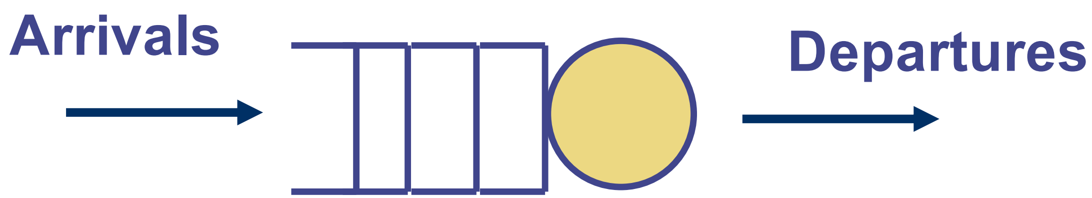
COMP9334
Capacity Planning for Computer Systems and Networks

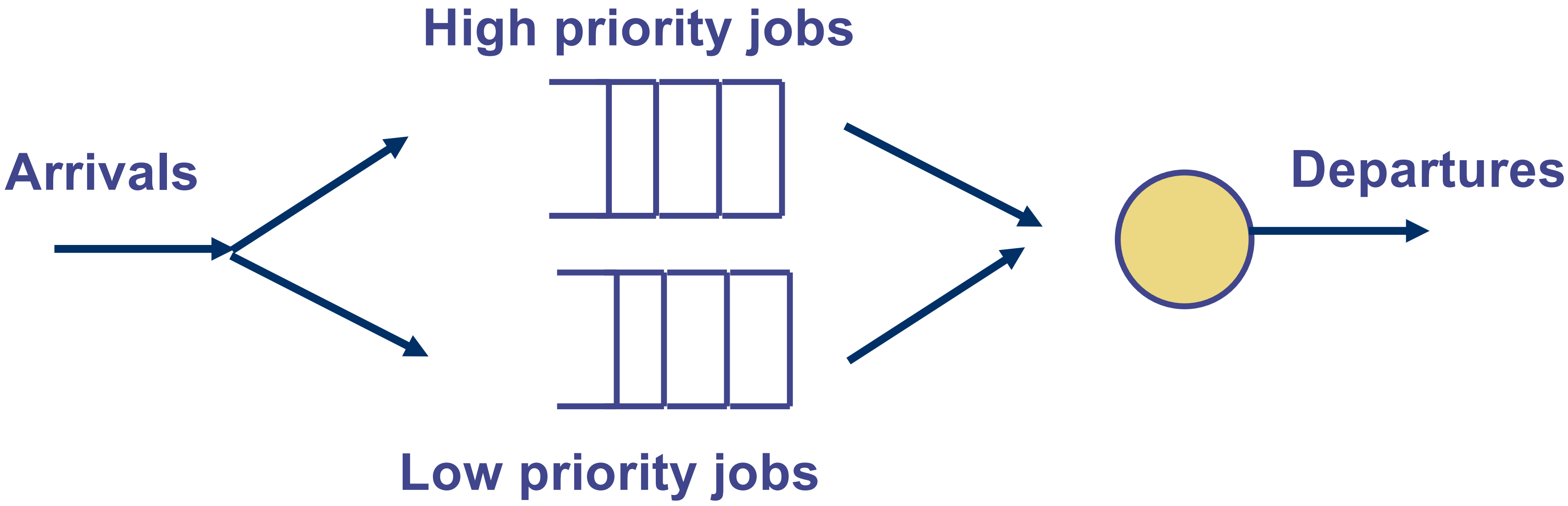
Week 4B_1: Queueing disciplines

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

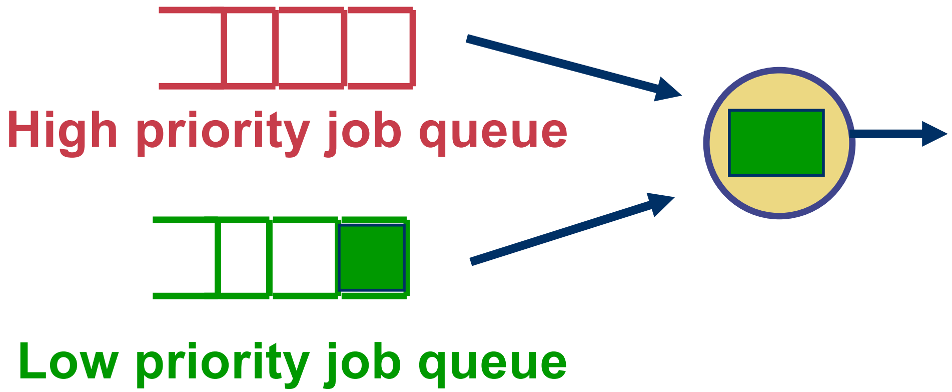
What is priority queueing?



- 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

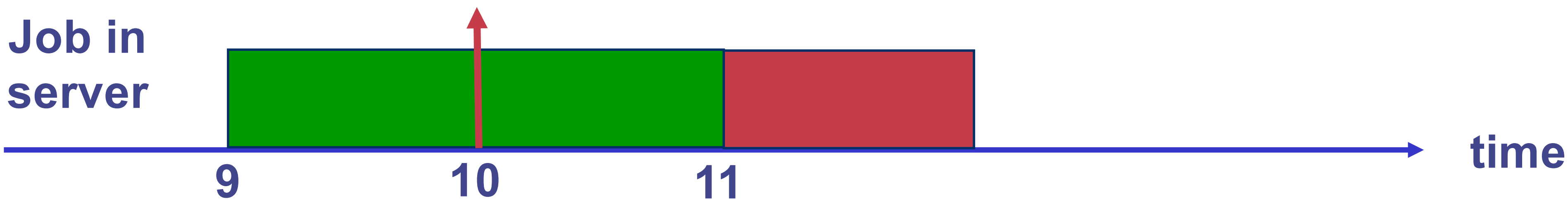
- The high priority job queue is empty
- The server starts serving a low priority job which requires 2s of processing

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.

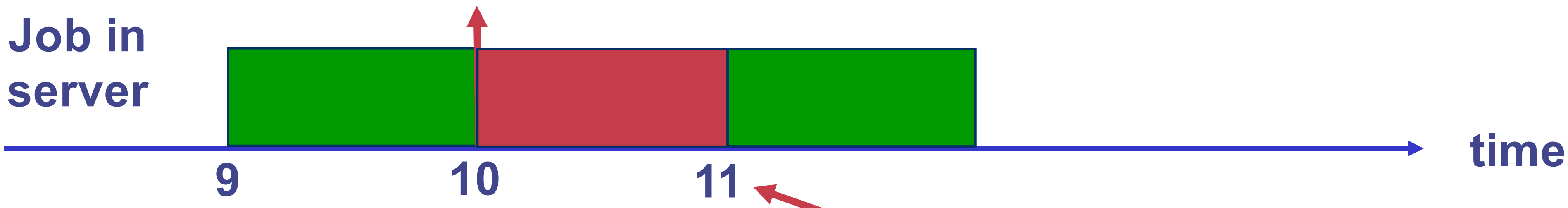
The job joins the high priority queue

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.

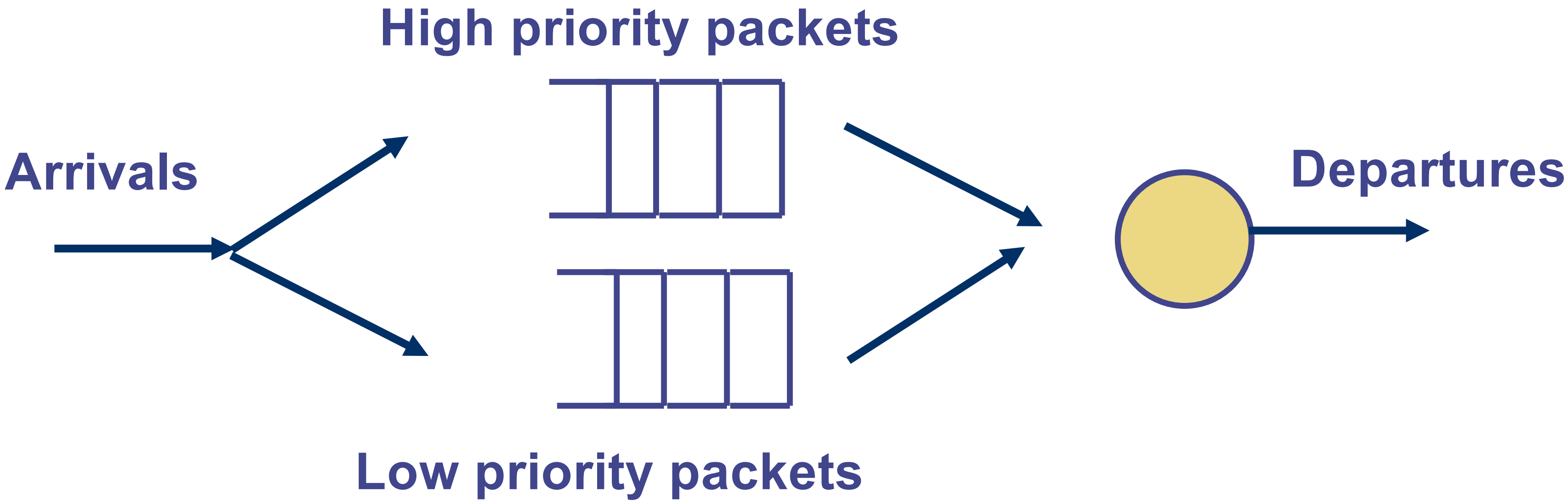
- Example: High priority job (red), low priority job (green)



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

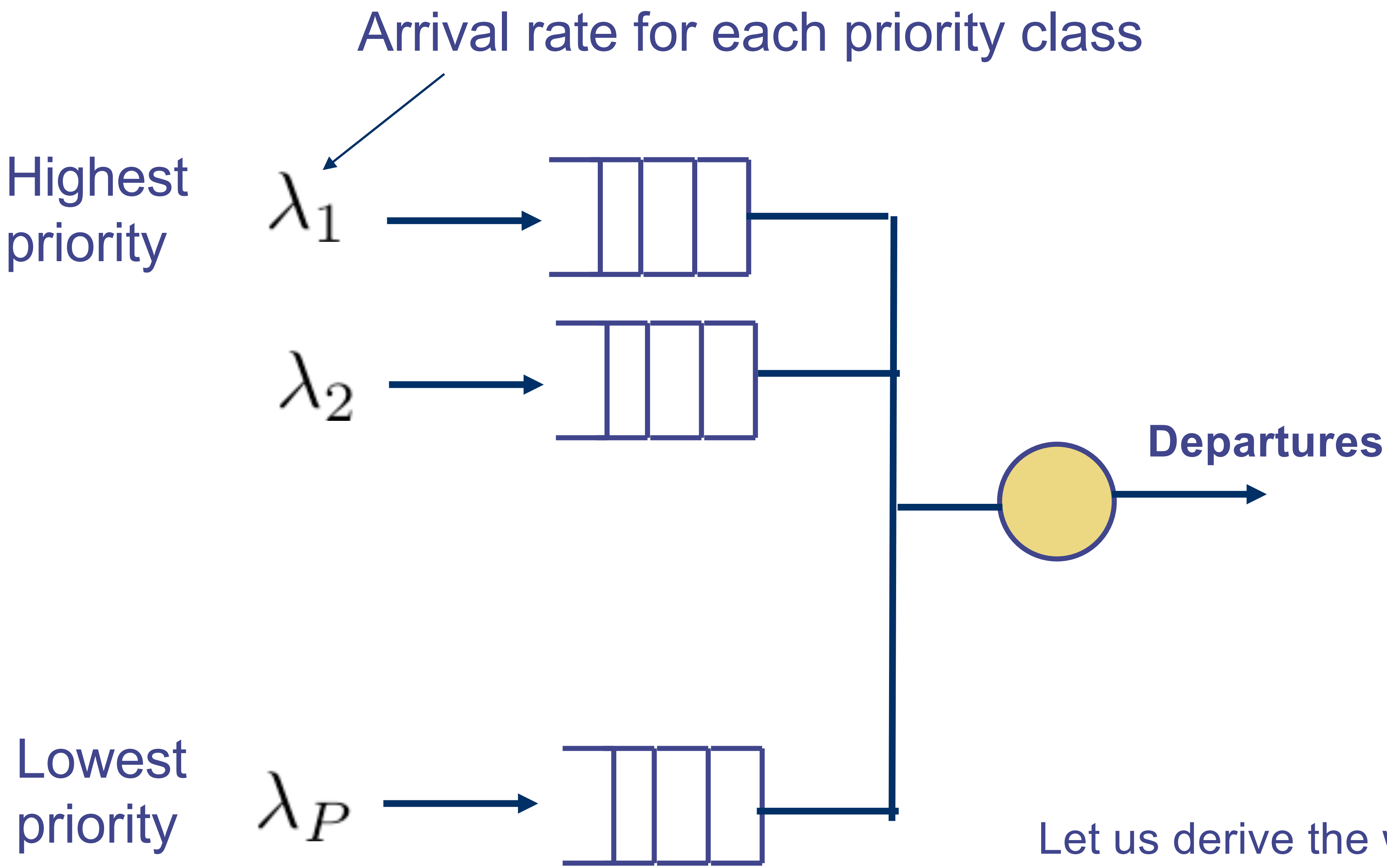
- Separate queue for each priority (see picture next page)
  - Classified into P priorities before entering a queue
  - Priorities numbered 1 to P, Queue 1 being the highest priority
- Arrival rate of priority class p is

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

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

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

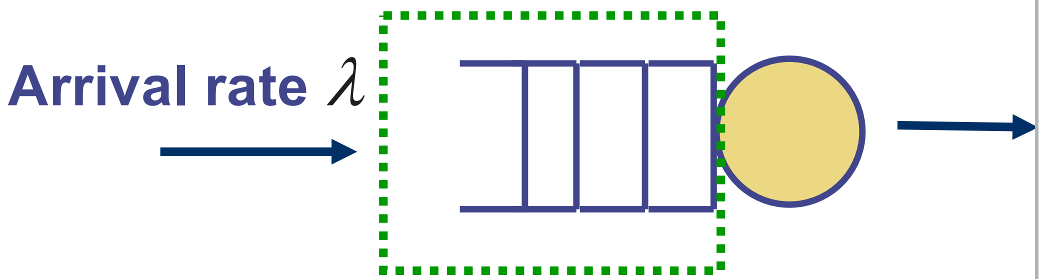
Priority queue



Let us derive the waiting time for P = 2

Lecture 4A: Deriving the P-K formula

- 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

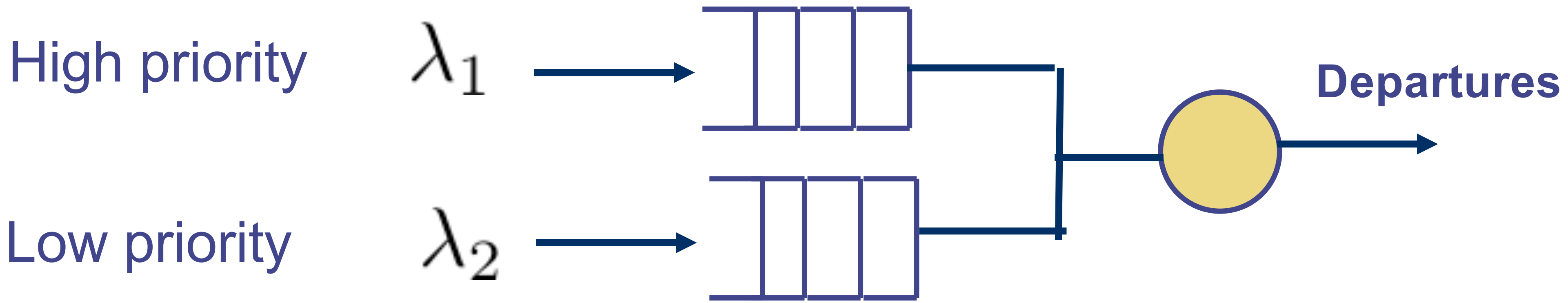
Substitution

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

Mean residual time R

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

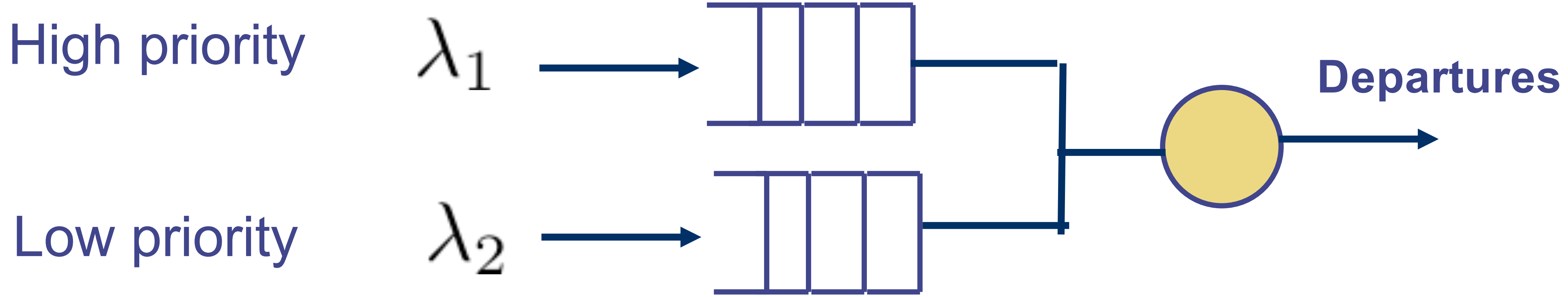
Deriving the non-preemptive queue result (1)



- 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)



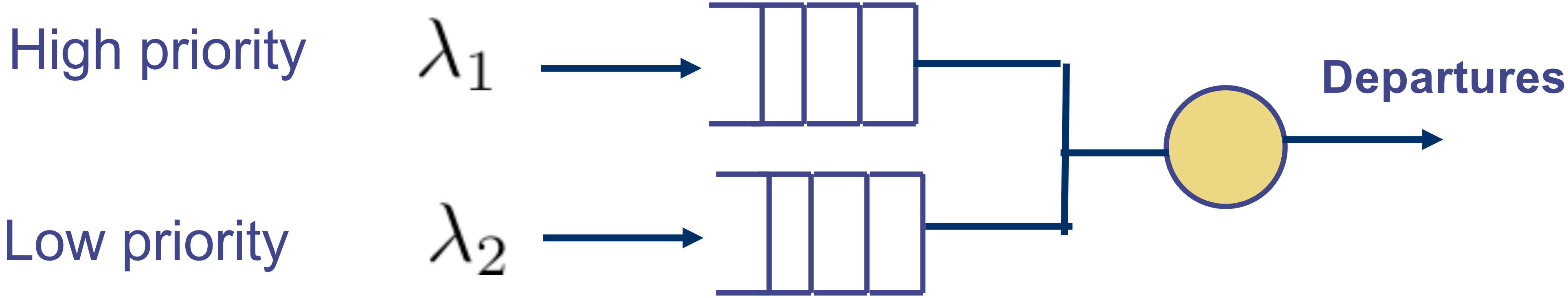
- To find the residual service time R, note that the customer in the server can be a high or low priority customer, we have

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

- The waiting time is therefore

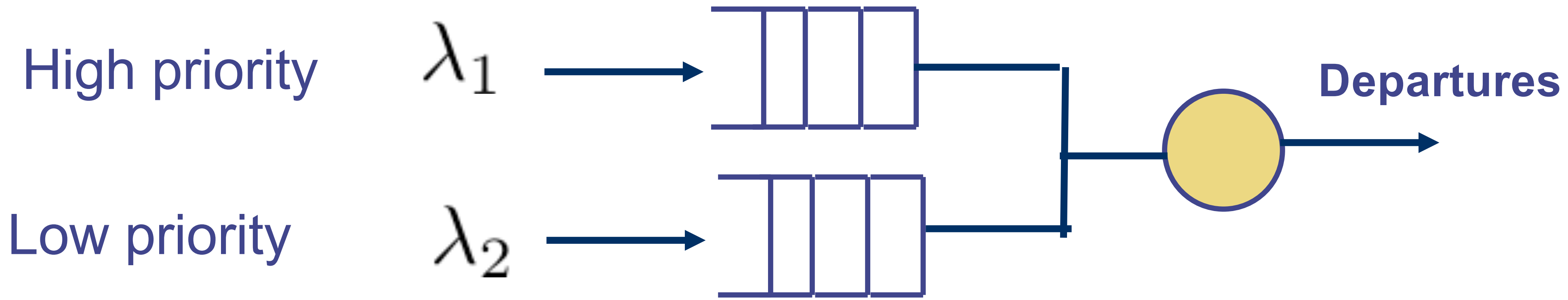
$$W_1 = \frac{R}{1-\rho_1} \quad \text{where} \quad \rho_1 = \lambda_1 E[S_1]$$

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:

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

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:

$$N_2 = \lambda_2 W_2$$

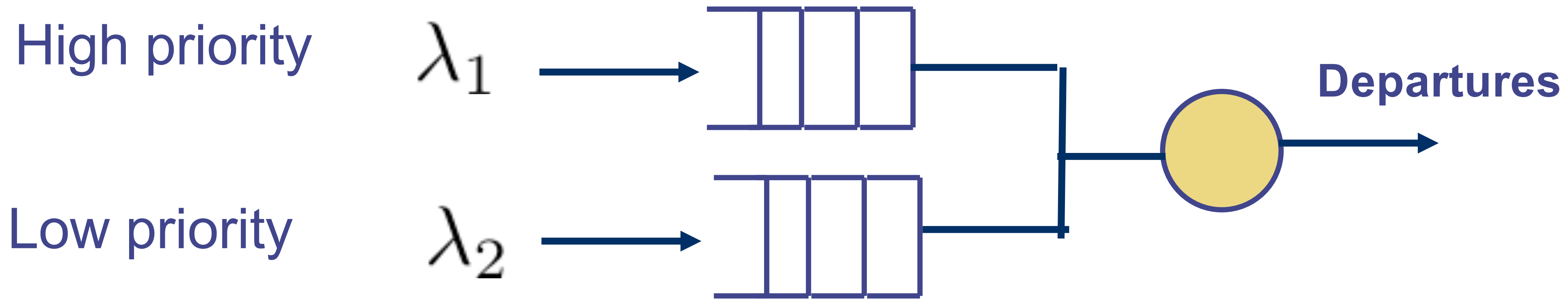
- Combining all of the above

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

Where

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

Deriving the non-preemptive queue result (6)



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

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

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

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

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

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

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:

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

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.

References

- Recommended reading
  - Bertsekas and Gallager, “Data Networks”
    - Section 3.5.3 for priority queuing
- Optional reading
  - Harchol-Balter, Chapter 22