Lecture 14: M/G/1 Queueing System with Priority

Dr. Mohammed Hawa
Electrical Engineering Department
University of Jordan

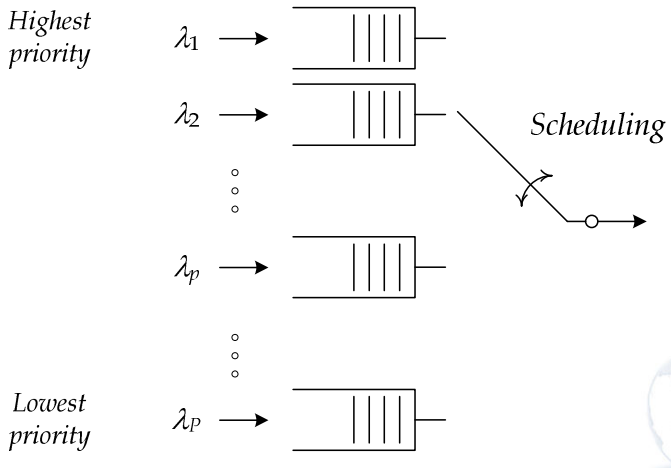
Priority Queueing Systems

- Until the moment, we assumed identical customers arriving at the queuing system (identical arrival statistics, identical service time statistics, and identical preference in the system).
- This is called a homogenous customer population.
- A heterogeneous customer population, on the other hand, consists of different classes of customers.
- For example, a group of customers can be given priority over other customers because they have different performance requirements (e.g., urgent management packets versus regular data packet).

Definitions…

- In a *priority* queuing system each arriving customer is treated differently based on its assigned priority.
- We will assume that the queuing system provides a different queue for each group of customers with a specific priority $p \in [1, P]$.
- All the queues are served by one server based on a certain *scheduling mechanism*.
- This model is called *M/G/1 with Priority*.

System Description

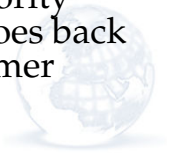


Scheduling Mechanisms (*disciplines* or *strategies*)

- Conventional (*simple*):
  - First-In First-Out (FIFO)
  - Last-In First-Out (LIFO)
- **Strict Priority scheduling**:
  - Preemptive Resume Strategy
  - Preemptive Non-Resume Strategy
  - Non-Preemptive Strategy
- Fair Queueing:
  - Weighted Fair Queueing (WFQ)
  - Self-Clocked Fair Queueing (SCFQ)
  - Start-Time Fair Queueing (STFQ)
  - Weighted Round Robin (WRR)
  - Deficit Round Robin (DRR)

Preemptive Resume Strategy

- In this strategy, customers with higher priority will always be served independently of whatever else is happening with lower priority customers.
- If a low priority customer was being served when another higher priority customer arrived, the server will *stop immediately* serving the low priority customer, and move to serve the higher priority customer.
- *After* finishing the service of the higher priority customer (and assuming no more high priority customers arrive meanwhile), the server goes back and *resumes* serving the low priority customer from the point where it stopped.



Preemptive Resume Strategy [2]

- In this strategy, higher priority customers do not feel the existence of lower priority customers, because no class p customers can be served if class $p-1$ customers exist, and no class $p-1$ customers can be served if class $p-2$ customers exist, etc.
- Cleary this strategy cannot be used in practical packet switched networks, since packets cannot be divide into arbitrary parts (depending on the time the server spends serving the packet). However, it can be used to describe a processor running a low-priority thread that is stopped by a high priority interrupt.

Preemptive Non-Resume Strategy

- In this strategy, if a low priority customer was being served when another higher priority customer arrived, the server will *stop immediately* serving the low priority customer, and move to serve the higher priority customer.
- After finishing the service of the higher priority customer (and assuming no more high priority customers arrive meanwhile), the server resumes serving the low priority customer from the beginning. In other words, it treats that customer as if it was a new customer, i.e., it makes no distinction between a new customer and the one that has been preempted (in terms of service time).

Preemptive Non-Resume Strategy [2]

- In this strategy, low priority customers are transparent to high priority customers since the high priority customers performance is unaffected by the low priority ones.
- Even though this strategy can be used in packet switched networks, it results in plenty of wasted server time and partially transmitted packets, hence, it is not used in practical systems.

Non-Preemptive Strategy

- In this strategy, if a high priority customer arrives while a low priority customer is being served, the low priority customer is allowed to finish the service before the server moves to serve the high priority customer (i.e., the low priority customer is not preempted, but no other low priority customer enters the server).
- The server does not go back to low priority customers until the high priority queue is empty.

Non-Preemptive Strategy [2]

- In this strategy, low priority customers affect (slightly) the performance of high priority customers, since a high priority customer arriving to an empty queue has to wait for the low priority customer to finish service.
- This is the strategy that is used in practical packet switched networks, and the one that we analyze in this course.

M/G/1 system with *non-preemptive strict priority*

- P: Number of traffic classes (priorities), where class 1 traffic is the highest priority and class P is the lowest priority.
- λ_i: Mean arrival rate of customers of class i (arrivals are Poisson).
- $E[B_i] = 1/\mu_i$: Mean service time of customers of class i (arbitrary distribution for service time B_i).
- $E[B_i^2]$: Second moment for the service time of customers of class i.
- $\rho_i = \lambda_i/\mu_i$: Traffic intensity of customers of class i.

Total Traffic

- Hence, the total traffic intensity to the system is:

$$\rho = \sum_{i=1}^{P} \rho_i$$

- We require that $\rho < 1$ to prevent buffers from accumulating to infinity (as this is infinite buffer case).

We define…

- W_k: Mean waiting time for a customer of priority k in the queue (not in the whole system).

$$W_k = W_0 + W_{Ak} + W_{Bk}$$

- W_0: Mean time until next customer in line for service starts service after an arbitrary arrival (i.e. if any customer in service, it has to finish its service).

Definitions

- W_{AK}: Mean waiting time for a customer of priority k due to customers of a **higher** or the **same** priority as k (i.e., priority $\leq k$) which already exist in the queue at the time of the customer arrival.
- W_{BK}: Mean waiting time for a customer of priority k due to customers of a **higher** priority as k (i.e., priority $< k$) that arrive after the k-priority customer arrival but before it starts its service (i.e., steps into the server).

Solution

$$W_k = \frac{W_0}{\left(1 - \sum_{i=1}^{k} \rho_i\right)\left(1 - \sum_{i=1}^{k-1} \rho_i\right)}$$

$$W_0 < W_1 < W_2 < \cdots < W_P$$

$$\overline{\tau_k} = \frac{1}{\mu_k} + W_k$$

