Queueing Systems

Shankar

March 31, 2022
Queueing Overview

- Queueing system
  - servers + waiting rooms
  - customers arrive, wait, get served, depart or go to next server
  - queueing disciplines
    - non-preemptive: fifo, priority, ...
    - preemptive: round-robin, multi-level feedback, ...

- Operating systems are examples of queueing systems
  - servers: hw/sw resources (cpu, disk, req handler, ...)
  - customers: PCBs, TCBs, ...

- Given: arrival rates, service times, queueing disciplines, ...
- Obtain: queue sizes, response times, fairness, bottlenecks, ...
Why do queues arise: bursty traffic

- Consider cars traveling on a road with a turn
  - each car takes 3 seconds to go through the turn
  - at most one car can be in the turn at any time
- \( N(t) \): \# cars in the turn and waiting to enter the turn

![Diagram](image)

- Arrival rate 1/4, load 3/4 (uniform)
- Arrival rate 1/4, load 3/4 (bursty)

Load \(< 1\): stable with waits depending on burstiness
Load \(> 1\): unstable, ever-increasing waits // not relevant
Single Queue

- **Customer $i$:**
  - arrival time // when it arrives
  - service time // duration of service needed
  - departure time // when it departs
  - response time // departure time – arrival time
  - wait time // response time – service time

- **Queue**
  - number of customers in queue at time $t$
  - unfinished work in queue at time $t$
Steady-state metrics

- Assume unending stream of customers
  - arrival rate  \( \text{// # arrivals per second averaged over all time} \)
  - average service time  \( \text{// averaged over all customers} \)
  - average response time  \( \text{// averaged over all customers} \)
  - load  \( \text{// work arriving per second averaged over all time} \)
  - throughput (aka departure rate):  \( \text{// # departures per second averaged over all time} \)
  - average queue size  \( \text{// averaged over all time} \)
  - utilization  \( \text{// fraction of time server is busy} \)

- Typical goal
  - Given: arrival rate, average service time, queueing discipline
  - Obtain: average response time, average queue size
Some Steady-state Relationships

- Load  = arrival_rate × average_service_time
- System is unstable if load > 1
  - avg queue size and avg response time are not defined
  - throughput = 1/service_time
  - utilization = 1
- System is stable if load ≤ 1
  - throughput = arrival_rate
  - utilization = load

- Little’s Law
  - avg_queue_size = avg_response_time × arrival_rate
  - holds for any queueing (sub)system: eg, a class of customers
Steady-state: Queue Size vs Load

- Avg queue size $N$ increases “exponentially” as load $\rho$ increases, becoming $\infty$ as $\rho \to 1$
- $N$ increases as burstiness increases
Steady-state: Wait time vs Service time

- Queuing disciplines can discriminate based on service times
- \( W(S) \): avg wait time for customers with service time \( S \)
- Favor customers with small \( S \)
  - SJF-preemptive > SJF > RR > FIFO, LIFO
  - RR w quantum \( \rightarrow 0 \): linear discrimination // ignoring overhead

\[ W(S) \]

\[ 0 \quad S \]

\( \text{SJF} \)

\( \text{RR (qs} \rightarrow 0) \)

\( \text{FIFO} \)
Server cycles between idle periods and busy periods

Work-conserving discipline: server not idle when customer present

For work-conserving disciplines:
the sequence of idle and busy periods, hence utilization, is independent of queueing discipline.

Proof: Consider the evolution of unfinished work $Y(t)$
- arrival increases $Y(t)$ by arrival’s service time
- while $Y(t) > 0$ holds, it decreases with slope $-1$
Evolution of unfinished work $Y(t)$