Server Scheduling

Scheduling: The task of deciding which job to run at each time.

Non-Preemptive: job once scheduled, must be completed before scheduling another job.

   Example: FCFS (First Come, First Serve)

Preemptive: job service can be interrupted and another job can be scheduled.

   Example: Last Come, First Serve - with preemptive resume.

Performance metrics:

- Server utilization: whenever job is available, server should be serving.

- Fairness: we don’t want to starve a job for a long period of time. Further, it’s unfair to make a short job wait behind a long job for a long period of time.
Mean Response Time:

\[ s_i : \text{arrival time of } i \text{th job} \]
\[ t_i : \text{departure time of } i \text{th job} \]
\[ \bar{RT} = \frac{1}{N} \sum_{i=1}^{N} (t_i - s_i) \]

Analysis of FCFS

\[
\begin{align*}
\lambda & \sim \text{Poison arrivals} \\
\mu & \sim \text{Service requirement} \\
0 & < \lambda < 1 \\
& \sim \text{Exp}(1)
\end{align*}
\]

Global balance equations to find stationary distribution

\[ \pi_k = P(\theta = k), \quad k = 0, 1, 2, \ldots \]

\[ \pi_0 \lambda = \pi_1 \]

\[ \pi_i (\lambda + 1) = \pi_0 \lambda + \pi_{i-1} \]

\[ \pi_i (\lambda + 1) = \pi_{i-1} \lambda + \pi_{i+1} \]
Then solution is $\Pi_k = \lambda^k (1-\lambda)$, $k=0,1,\ldots$

average queue size $E[Q] = \sum_{i=0}^{\infty} i \Pi_i = \frac{\lambda}{1-\lambda}$

average delay ($\bar{W}$) $= E[D] = \frac{1}{1-\lambda}$ (by Little's law)

Little's law: $E[D] = \frac{1}{\lambda} E[Q]$. 

\[ Q(t) = \text{Arrivals}(0,t) - \text{Departures}(0,t) \]

So
\[ \bar{Q} = \frac{1}{t} \int_0^t Q(s) \, ds = \frac{1}{t} \sum_{i=1}^{\infty} (t_i - s_i) \]

\[ = \frac{\text{Arr}(t)}{t} \frac{1}{\text{Arr}(t)} \sum_{i=1}^{\infty} (t_i - s_i) \]

$\rightarrow \lambda \cdot \bar{Q}$ as $t \to \infty$

so FCFS is at most $\frac{1}{1-\lambda}$ factor worse than the minimum response time.

Delay optimal policy:

Offline policy: N jobs with service requirement $s_i$. So
available at time 0.

- Shortest Job First (SJF): Schedule jobs in the increasing order of their service requirements.

Proof:

$$\overline{RT}_N = \frac{1}{N} \sum_{i=1}^{N} t_i = \frac{1}{N} \int_0^\infty q(t) \, dt$$

Little's law graph

Now SJF minimizes $Q(t)$ at all $t \geq 0$. Thus SJF minimizes $\overline{RT}_N$.

Online Policy:

- Shortest remaining Processing Time First, at each (SRPT)

time schedule job that has the minimum remaining processing time among the existing jobs in the system.

Proof similar to the offline policy.

SRPT is $\overline{RT}$ optimal it can lead to starvation of long jobs.

(Thus not fair). When service requirements are not exponential.

To resolve this issue, one can divide the server to multi
Subservers & assign the jobs with similar service requirement to the same subserv. The choice of subserv & assignment must be done in a way to ensure that subservers are equally loaded.

Scheduling in Multi-Server System:

1. Job arrivals
2. Scheduling
3. Server 1
4. Server n

Join the least loaded (JLL) policy.

Instead of analyzing the above system, consider the system below with one common queue.

Whenever a server becomes free, it picks a job from the head of the common queue.
The departure times of each job from both systems is the same and they depart from the same server in both systems. (The same tie breaking rule must be used for both systems).

The proof is based on induction. Consider a sequence of \( k \) jobs. The result is obvious for \( k=1 \). Now suppose it's true for \( k \geq 1 \) and show that \( (k+1) \) th job is also depart from the same server in both systems.

Hence to analyze \( J_k \), we can analyze the second system which is simpler.

- Arrival \( \sim \) Poisson process at rate \( \lambda \).
- Service requirements \( \sim \) iid \( \exp \) with mean \( 1 \).
- \( 1 < n \) (n servers).

![Diagram of Markov Chain](image)

Markov chain. State of the system is the \# of jobs in the system.
Solving the global balance equations to find $\Pi_k$:

$$
\Pi_k = \begin{cases} 
\frac{1}{k!} \Pi_0 & k \leq n \\
\left(1 - \frac{\lambda}{\mu} \right) \frac{1}{n!} \Pi_0 & k > n
\end{cases}
$$

where

$$
\Pi_0 = \frac{1}{\sum_{k=0}^{n-1} \frac{\lambda}{k!} + \frac{\lambda^n}{n!} \left( \frac{1}{1 - \frac{\lambda}{\mu}} \right)}
$$

$$
\bar{Q} = \mathbb{E}[Q] = \sum_{k=0}^{\infty} k \Pi_k
$$

$$
\bar{D} = \frac{\bar{Q}}{\lambda} \quad \text{[Little's law]}
$$

Implementing JDL requires knowing the job service times which is not known a priori in practice. A good alternative policy is JSQ (Join the shortest queue).

JSQ is optimal (in terms of response time) under Poisson arrivals & exponential service requirements.

To show the optimality, it is enough to show that JSQ minimizes the average queue size.
suppose \( Q^1 \) is the queue under user JSQ, and \( Q^2 \) in 
the one under my other policy. We show that \( Q^1 \approx Q^2 \), 

i.e., \( Q^1 \) is stochastically dominated by \( Q^2 \).

Stochastic dominance: \( X = (x_1, \ldots, x_n) \preceq Y = (y_1, \ldots, y_n) \) if 

it's possible to construct \( X \) and \( Y \) on a joint probability 
space s.t. the marginal dist. of \( X, Y \) remain unchanged and 

\[
\sum_{j=1}^{k} x_j \leq \sum_{j=1}^{k} y_j \quad \forall k = 1, \ldots, n
\]

where \( \pi^x, \pi^y \) are permutation of indices s.t.

\[
\begin{align*}
\pi^x(1) & \to \ldots \to \pi^x(n) \\
\pi^y(1) & \to \ldots \to \pi^y(n)
\end{align*}
\]

Equivalent description of systems: A common Poisson process of

rate \( \lambda + n \). Each event in a job arrival w.p. \( \frac{1}{\lambda + n} \) or

a potential departure w.p. \( \frac{n}{\lambda + n} \). The potential departure

is uniformly at random from any of \( n \) servers. If the server

is idle, the potential departure is wasted, otherwise, a job

departs from its server.
Let \( \omega_1, \ldots, \omega_n \) be a sequence of arrivals/potential departures.

For \( k = 1 \), the stochastic dominance holds.

Suppose for \( k > 1 \) holds. We show that for a sequence of
length \( k + 1 \) also holds.

\( \omega_{k+1} \) is either an arrival or potential departure. If it's an arrival, it goes to \( \Pi^{k, \infty}_1 \) under JSD and \( \Pi^{k, \infty}_2 \) under any other policy. It's easy to see that

\[
Q^1 + e^{\Pi^{k, \infty}_1} \leq Q^2 + e^{\Pi^{k, \infty}_2} \quad \forall \epsilon > 0.
\]

If \( \omega_{k+1} \) a potential departure, suppose it's for game
\( \Pi^{k, (ij)} \) under JSD. We can assign this potential departure
to the \( k \)th game under the optimal policy. So that

\[
Q^1 - e^{\Pi^{k, (ij)}_1} + Q^2 - e^{\Pi^{k, (ij)}_2} \quad \forall \epsilon > 0
\]

and obviously this coupling preserves the uniform dist.
of potential departures (marginals).