Priority Based Scheduling in Queuing Systems

PMCS Project

Team 4

Problem Statement in Brief

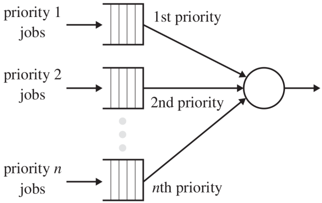


Figure 31.1. When a server frees up, it takes the job at the head of the highest priority, non-empty queue.

- Queues covered in class followed the FCFS scheduling policy.
- We shall explore priority-based scheduling where priorities are assigned according to job size (shortest job has the highest priority).
- Each class is an imaginary M/G/1 queue. When the server becomes free, it always chooses the job at the head of the highest priority non-empty queue to work on.

Problem Statement in Brief

- How will such a queue be modelled as a CTMC ?
- What are conditions that ensure queue is stable?
- What are the average performance metric formulas such as the mean sojourn time, mean number of jobs in the system, or blocking probabilities ?
- Simulations of the queuing system to verify the results derived

Notations

- S_k = size of priority k job
- $E[N_Q(k)]$ = average number of priority k jobs in the queue
- $E[T_Q(k)]$ = average time in queue for priority k jobs
- $E[T(k)]$ = average time in system for priority k jobs
- $\lambda_k = \lambda.p_k$ = average arrival rate of jobs of priority k
- $\rho_k = \lambda_k.E[S_k]$ = contribution to the load due to jobs of priority k
- $\sum_{i=1}^{n} \rho_i = \sum_{i=1}^{n} \lambda_i E[S_i] = \sum_{i=1}^{n} \lambda_i p_i E[S_i] = \lambda E[S] = \rho$
- ρ is the server utilization (< 1)

Notations

- Let S denote size of arbitrary job
- $E[S] = \sum_{k=1}^{n} p_k E[S_k]$
- $E[S^2] = \sum_{k=1}^{n} p_k E[S_k^2]$
- Using Hitchhiker's Paradox,

$$E[S_e] = \text{mean remaining time of a job}$$

$$E[S_e] = \frac{E[S^2]}{2E[S]}$$

Deriving $T_Q(k)$ -Time in Queue for Jobs of Priority k

- Consider a priority k arrival. That arrival has to wait for
  - The job currently in service, if there is one.
  - All jobs of priority 1, 2, . . . , k in queue when the job arrives.
  - All jobs of priority 1, 2, . . . , k 1 that arrive while the new job is waiting.
- It can be shown that
  $E[T_Q(k)]^{NP-Priority} = \frac{\rho E[S_e]}{(1-\sum_{i=1}^{k} \rho_i)(1-\sum_{i=1}^{k-1} \rho_i)}$

Deriving $T_Q(1)$ -Time in Queue for Jobs of Priority 1

- Consider a priority 1 arrival. That arrival has to wait for
  - The job currently in service, if there is one.
  - All jobs of priority 1 in queue when the job arrives.
- $E[T_Q(1)] = P(Serverbusy).E[S_e] + E[N_Q(1)].E[S_1]$
  $= \rho E[S_e] + E[T_Q(1)].\lambda_1.E[S_1]$
  $= \rho E[S_e] + E[T_Q(1)].\rho_1$
  $= \frac{\rho E[S_e]}{1-\rho_1}$

Deriving $T_Q(2)$ -Time in Queue for Jobs of Priority 2

- Consider a priority 2 arrival. That arrival has to wait for
  - The job currently in service, if there is one.
  - All jobs of priority 1 in queue when the job arrives.
  - All jobs of priority 1 that arrive while the new job is waiting (not in service).
- $E[T_Q(2)] = \rho E[S_e] + E[N_Q(1)].E[S_1] + E[N_Q(2)].E[S_2] + E[T_Q(2)].\lambda_1 E[S_1]$
  $= \rho E[S_e] + E[T_Q(1)].\rho_1 + E[T_Q(2)].\rho_2 + E[T_Q(2)].\rho_1$
  $\mathrm{E}[\mathrm{T}_Q(2)].(1-\rho_1-\rho_2) = \rho E[S_e] + \rho_1.E[T_Q(1)]$
  $\mathrm{E}[\mathrm{T}_Q(2)] = \frac{\rho E[S_e]}{(1-\rho_1)(1-\rho_1-\rho_2)}$
- Similarly using induction, we can derive for formula $E[T_Q(k)]^{NP-Priority} = \frac{\rho E[S_e]}{(1-\sum_{i=1}^{k}\rho_i)(1-\sum_{i=1}^{k-1}\rho_i)}$

Interpreting the formula

- $E[T_Q(k)]^{NP-Priority} = \frac{\rho E[S_e]}{(1-\sum_{i=1}^{k}\rho_i)(1-\sum_{i=1}^{k-1}\rho_i)}$
- $\rho E[S_e]$ is due to waiting for the job in service
- $(1-\sum_{i=1}^{k}\rho_i)$ due to waiting for jobs in the queue of higher or equal priority
- $(1-\sum_{i=1}^{k-1}\rho_i)$ is due to those jobs that arrive after our job, but have strictly higher priority than our tagged job
- This is the average waiting time for a job of class k. How do we obtain the average waiting time across all classes $E[T_Q]^{NP-Priority} = \sum_{k=1}^{n} E[T_Q(k)].p_k = \sum_{k=1}^{n} E[T_Q(k)].\frac{\lambda_k}{\lambda}$

Shortest-Job-First

- Consider again the situation of n priority classes. Let's assume that the job sizes range between $x_0 = 0$ and x_n. Job sizes are generated as instances of exponential random variable(μ).
- Define boundary points $x_1, x_2, ...x_n$ such that $x_0 < x_1 < x_2 < ... < x_{n-1} < x_n$.
- Assign all jobs of size (x_{k-1}, x_k) to class k.

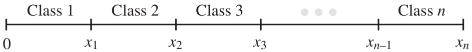


Figure 31.2. Defining classes based on job size.

Shortest-Job-First

- Now consider the situation where $n \to \infty$ and $(x_k - x_{k-1}) \to 0$. That is, the number of classes, n, is allowed to grow to ∞, in such a way that $(x_k - x_{k-1})$ becomes arbitrarily small.
- We are interested in the expected waiting time for a job of size x_k

Expected Waiting Time in SJF

- Load of jobs in class 1 to k = Load of jobs of size $< x_k = \lambda \int_{t=0}^{x_k} tf(t)dt$
- Load of jobs in class 1 to k-1 = Load of jobs of size $< x_{k-1} = \lambda \int_{t=0}^{x_{k-1}} tf(t)dt = \lambda \int_{t=0}^{x_k} tf(t)dt$
- $E[T_Q(x)]^{SJF} = \frac{\rho E[S_e]}{(1-\lambda \int_{t=0}^{x} tf(t)dt)^2}$
- $E[T_Q]^{SJF} = \int_{x=0}^{x_n} E[T_Q(x)]f(x)dx = \rho E[S_e] \int_{x=0}^{x_n} \frac{f(x)dx}{(1-\lambda \int_{t=0}^{x} tf(t)dt)^2}$

Motivation

- Scheduling policies can be categorized into non-preemptive and preemptive types.
- Non-preemptive policies exhibit drawbacks, particularly in scenarios with highly variable job sizes, leading to longer mean response times.
- Preemptive policies offer better performance in such scenarios by interrupting larger jobs to serve smaller ones, reducing response times.
- Policies utilizing job size or age can further improve performance by favoring smaller jobs, as seen in the Feedback (FB) scheduling policy.

Preemptive Priority Queueing

- We continue with the assumptions from NP-Priority:
  - There are n classes, with Class 1 having the highest priority.
  - Class k jobs arrive according to a Poisson process with rate $\lambda_k = \lambda \cdot p_k$.
  - Class k jobs have service requirements with moments $E[S_k]$ and $E[S_k^2]$.
  - The load of class k is $\rho_k = \lambda_k \cdot E[S_k]$.
- Preemptive priority queueing differs from non-preemptive queueing in that whenever a job arrives with a higher priority than the job currently in service, the job in service is preempted, and the higher priority job begins service.
- No work is lost under preemptions.
- We will compute $E[T(k)]^{P-Priority}$, the mean time in system for a job of priority k in a system with preemptive priority.

Calculating Mean Time in System for Priority k

- To compute $E[T(k)]^{P-Priority}$, the mean time in system for a job of priority k in a preemptive priority system, we break it down into three components:
  1. $E[S_k]$: Mean service time for a job of priority class k.
  2. Expected time to complete service on all jobs of priority 1 to k already in the system when our arrival walks in.
  3. Expected total service time required for all jobs of priority 1 to $k-1$ that arrive before our arrival leaves.

Calculating Component (3)

- Component (3) is the expected total service time required for all jobs of priority 1 to $k-1$ that arrive before our arrival leaves.
- This is calculated as:

$$(3) = \sum_{i=1}^{k-1} E[T(k)] \cdot \lambda_i \cdot E[S_i] = E[T(k)] \sum_{i=1}^{k-1} \rho_i$$

Calculating Component (2)

- We cannot directly sum up the expected number of jobs in each class for classes 1 to k, weighted by the mean job size for that class, as in non-preemptive priority queueing.
- We make the following arguments to determine (2):
  1. Expected remaining work in the system due to only jobs of priority 1 through k.
  2. Total expected remaining work in preemptive priority system if the system only ever had arrivals of priority 1 through k.
  3. Total expected remaining work in the system if the system only ever had arrivals of class 1 through k and the scheduling order was any work-conserving order.

Calculating Component (2) - Continued

- Using these arguments, we arrive at:

$$(2) = \frac{\lambda \sum_{i=1}^{k} p_i E[S_i]}{1 - \sum_{i=1}^{k} \rho_i} \cdot \frac{\sum_{i=1}^{k} p_i E[S_i^2]}{2 \sum_{i=1}^{k} p_i E[S_i]} = \frac{\lambda \sum_{i=1}^{k} \rho_i E[S_i^2]}{2(1 - \sum_{i=1}^{k} \rho_i)}$$

$$(2) = \frac{\sum_{i=1}^{k} \rho_i \frac{E[S_i^2]}{2E[S_i]}}{1 - \sum_{i=1}^{k} \rho_i}$$

Combining Components

- Finally, adding (1), (2), and (3), we have:

$$E[T(k)]^{P-Priority} = E[S_k] + \frac{\sum_{i=1}^{k} \rho_i \frac{E[S_i^2]}{2E[S_i]}}{1 - \sum_{i=1}^{k} \rho_i} + E[T(k)] \sum_{i=1}^{k-1} \rho_i$$

$$E[T(k)](1 - \sum_{i=1}^{k} \rho_i) = E[S_k] + \frac{\lambda \sum_{i=1}^{k} \rho_i \frac{E[S_i^2]}{2E[S_i]}}{1 - \sum_{i=1}^{k} \rho_i}$$

$$E[T(k)]^{P-Priority} = \frac{E[S_k]}{(1 - \sum_{i=1}^{k} \rho_i)} + \frac{\sum_{i=1}^{k} \rho_i \frac{E[S_i^2]}{2E[S_i]}}{(1 - \sum_{i=1}^{k-1} \rho_i)(1 - \sum_{i=1}^{k} \rho_i)}$$

Interpretation of $E[T(k)]^{P-Priority}$

- For preemptive service disciplines, the time in the system of a job can be divided into two components:
  1. Time until the job first starts serving (waiting time), denoted by *Wait*.
  2. Time from when the job first receives some service until it leaves the system (residence time), denoted by *Res*.

Understanding Mean Time in System for Priority k

- **Question**: Is residence time the same as service time?
- **Answer**: No. The residence time is longer as it includes interruptions.
- **Question**: What does the first term $E[S_k]/(1 - \sum_{i=1}^{k-1} \rho_i)$ in expression $E[T(k)]^{P-Priority}$ represent?
- **Answer**: This represents the mean residence time of the job of class k, denoted as $E[Res(k)]$. It signifies the expected length of a busy period started by a job of size $E[S_k]$, where only jobs of class 1 through $k-1$ are allowed in the busy period.

Understanding Mean Time in System for Priority k (Continued)

- **Question**: What does the second term in the expression $E[T(k)]^{P-Priority}$ means ?.
- **Answer**: The remaining term in expression $E[T(k)]^{P-Priority}$ represents the mean time until the job of priority k first receives service. It is calculated as:

$$E[Wait(k)] = \frac{\sum_{i=1}^{k} \rho_i \left(\frac{E[S_{2i}]}{2E[S_i]} \right)}{\left(1 - \sum_{i=1}^{k-1} \rho_i\right) \left(1 - \sum_{i=1}^{k} \rho_i\right)}$$

This term is similar to $E[T_Q(k)]$ for the non-preemptive priority queue, but only considers interruptions by jobs of class 1 through k. This reflects the fact that a job of class greater than k will be preempted immediately.

Understanding Mean Time in System for Priority k (Continued)

- It is sometimes convenient to rewrite the expression $E[T(k)]^{P-Priority}$ as:

$$E[T(k)]^{P-Priority} = \frac{E[S_k]}{1 - \sum_{i=1}^{k-1} \rho_i} + \frac{\frac{\lambda}{2} \sum_{i=1}^{k} p_i E[S_{2i}]}{(1 - \sum_{i=1}^{k-1} \rho_i)(1 - \sum_{i=1}^{k} \rho_i)}$$

Comparison with Non-Preemptive Priority Queueing

- **Question**: Does a high priority job obtain better performance in preemptive priority queueing?
- **Answer**: Yes. Both terms in $E[T(k)]^{P-Priority}$ depend only on the first k priority classes. Thus, a high priority job in preemptive priority queueing indeed performs better, even in a high-variability job size distribution, compared to non-preemptive priority queueing.
- Note: In the non-preemptive case, the expression for $E[T_Q(k)]^{NP-Priority}$ has a similar denominator, but the numerator involves all n classes contributing to the excess, unlike in the preemptive case.

Preemptive Shortest Job First(PSJF)

- A preemption only occurs when a new job arrives whose size is smaller than the original size of the job in service.
- Recall that
  $E[T(k)]^{P-Priority} = \frac{E[S_k]}{1-\sum_{i=1}^{k-1}\rho_i} + \frac{\frac{\lambda}{2}\sum_{i=1}^{k} p_i E[S_i^2]}{(1-\sum_{i=1}^{k-1}\rho_i)(1=\sum_{i=1}^{k}\rho_i)}$
- Performing the same limiting operations as we did in analyzing SJF (where we imagine that jobs of size x form one "class" and that there are an infinite number of classes),we get
  $E[T(x)]^{PSJF} = \frac{x}{1-\rho_x} + \frac{\frac{\lambda}{2}\int_0^x f(t)t^2dt}{(1-\rho_x)^2}$
- Here $f(t)$ is the p.d.f. of job size, S and $\rho_x = \lambda\int_0^x tf(t)dt$ is defined to be the load made up by jobs of size less than x

Preemptive Shortest Job First(PSJF)- Busy Period Analysis

- We start by breaking up response time into waiting time and residence time
- $E[T(x)]^{PSJF} = E[Wait(x)]^{PSJF} + E[Res(x)]^{PSJF}$
- $Res(x)$ is just the duration of a busy period started by a job of size x, where the only jobs that make up this busy period are jobs of size $\leq$ x.
  $E[Res(x)]^{PSJF} = \frac{x}{1-\rho_x}$

Preemptive Shortest Job First(PSJF)- Busy Period Analysis

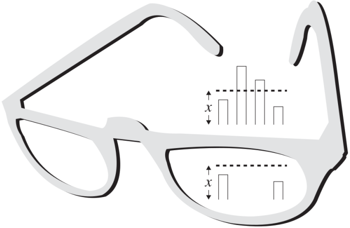


Figure 32.1. Transformer glasses for PSJF. Whereas in FB, the transformer glasses truncate all jobs of size $> x$ to size x, in PSJF, the transformer glasses make jobs of size $> x$ invisible.

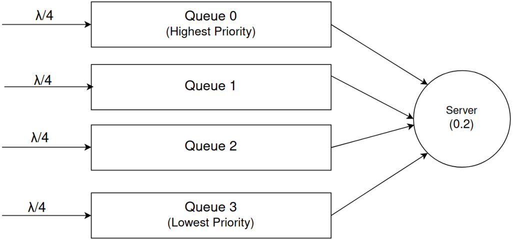
- Let us now calculate $E[Wait(x)]^{PSJF}$
- For a job of size x, busy period is the job of sizes $\leq$ x.Let us call this W_x
- We will use S_x to denote the size of a job of size $\leq$ x. The density of S_x is $\frac{f(t)}{F(x)}$ where $f(t)$ is density of S.

Preemptive Shortest Job First(PSJF)- Busy Period Analysis

- $E[Wait(x)]^{PSJF} = \frac{E[W_x]}{1-\rho_x}$
  $= \frac{E[T_Q|S_x]^{FCFS}}{1-\rho_x}$
  $= \frac{\frac{\lambda F(x) E[S_x^2]}{2(1-\rho_x)}}{1-\rho_x}$
  $= \frac{\lambda F(x) \int_0^x t^2 \frac{f(t)}{F(x)} dt}{2(1-\rho_x)^2}$
  $= \frac{\lambda \int_0^x t^2 f(t) dt}{2(1-\rho_x)^2}$
- Thus, $E[T(x)]^{PSJF} = \frac{x}{1-\rho_x} + \frac{\lambda \int_0^x t^2 f(t) dt}{2(1-\rho_x)^2}$

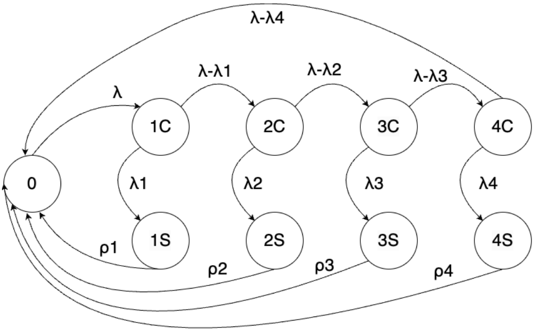
Simulations

Simulations- Details of the System



- We have implemented a M/M/1 queue where arrival rate $= Exp(\lambda) = 0.1$ and service requirements, $S_x = Exp(\mu) = 0.2$
- The system has 4 queues and 1 server. Each queue is for a single priority. Priority is assigned according to job sizes.

Modelling System as CTMC (Non - Preemptive)



0 - server is idle

i_C - checking if any job is present in ith queue

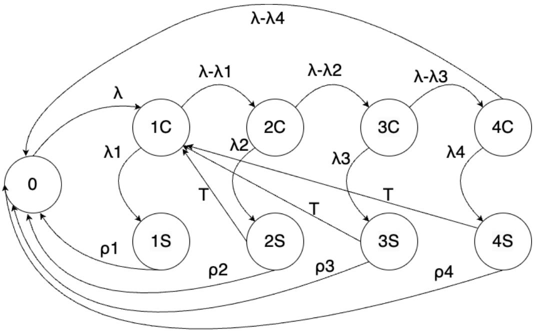
i_S - serving serving priority i job

λ - arrival rate

λ_i - arrival rate of priority i job

ρ_i - load that priority i job brings to the server

Modelling System as CTMC (Preemptive)



0 - server is idle

i_C - checking if any job is present in ith queue

i_S - serving serving priority i job

λ - arrival rate

λ_i - arrival rate of priority i job

ρ_i - load that priority i job brings to the server

T - rate of preemption

Non-Preemptive: Theoretical Calculations

- Given parameters:

$$
\begin{aligned}
\lambda &= 0.1 \\
E[S] &= \frac{1}{0.2} = 5 \\
\rho &= \lambda \cdot E[S] = 0.5 \\
E[S^2] &= \frac{1}{(0.2)^2} + \frac{1}{0.04} = 50 \\
\frac{\rho \cdot E[S^2]}{2 \cdot E[S]} &= 2.5
\end{aligned}
$$

Theoretical Calculations (Continued)

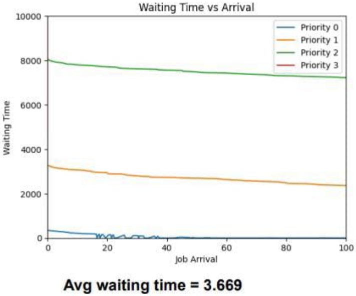
- We calculate the integral:

$$\int_0^\infty \frac{e^{-0.2x}}{\left(1-\left(0.5-\left(0.5x+2.5\right)e^{-0.2x}\right)\right)^2}dx = 1.45231388191$$

- Then, we multiply it by $\frac{p \cdot E[S^2]}{2 \cdot E[S]}$:

$$\frac{\rho \cdot E[S^2]}{2 \cdot E[S]} \cdot \int_0^\infty \frac{e^{-0.2x}}{\left(1-\left(0.5-\left(0.5x+2.5\right)e^{-0.2x}\right)\right)^2}dx = 3.63078470$$

Results obtained through Simulations



- Average Waiting Time: 3.669

Conclusions

- Both theoretical and simulation results match, indicating the accuracy of the theoretical model in predicting the average waiting time.

Preemptive: Theoretical Calculations

- Given parameters:

$$\lambda = 0.1$$
$$\mu = 0.2$$

Theoretical Calculations (Continued)

- First let's calculate loads made up by jobs each queue less than x:

$$\rho_x = \lambda \cdot \int_0^x tf(t)dt$$

- We have four bins $[0, 2.5], [2.5, 5], [5, 7.5], [7.5, \infty)$:

$$\rho_1 = \lambda \cdot \int_0^{2.5} tf(t)dt$$

$$\rho_1 = \lambda \cdot \int_0^{2.5} x \cdot \mu e^{-\mu \cdot x} dx$$

$$\rho_1 = 0.04$$

$$\rho_2 = \lambda \cdot \int_0^{5} tf(t)dt$$

$$\rho_2 = \lambda \cdot \int_0^{5} x \cdot \mu e^{-\mu \cdot x} dx$$

$$\rho_2 = 0.13$$

Theoretical Calculations (Continued)

$$\rho_3 = \lambda \cdot \int_0^{7.5} tf(t)\, dt$$

$$\rho_3 = \lambda \cdot \int_0^{7.5} x \cdot \mu e^{-\mu \cdot x}\, dx$$

$$\rho_3 = 0.22$$

$$\rho_4 = \lambda \cdot \int_0^{\infty} tf(t)\, dt$$

$$\rho_4 = \lambda \cdot \int_0^{\infty} x \cdot \mu e^{-\mu \cdot x}\, dx$$

$$\rho_4 = 0.5$$

Theoretical Calculations (Continued)

- $$E[T(x)]^{\text{PSJF}} = \frac{x}{1-\rho_x} + \frac{\lambda \int_0^x t^2 f(t)\, dt}{2(1-\rho_x)^2}$$

  Using the above formula we get the following values:

$$E[T(2.5)] = 1.08$$

$$E[T(5)] = 1.414$$

$$E[T(7.5)] = 2.065$$

$$E[T(\infty)] = 12$$

Preemptive SJF-Theoretical Results

- Average waiting time =

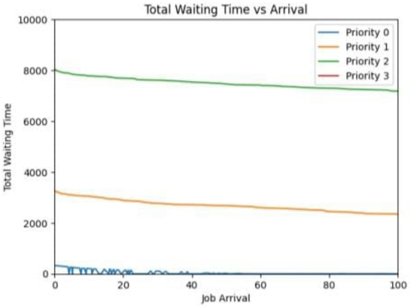
$$E[T(2.5)] \cdot F_x(2.5) + E[T(5)] \cdot [F_x(5) - F_x(2.5)]$$

$$+E[T(7.5)] \cdot [F_x(7.5) - F_x(5)] + E[T(\infty)] \cdot (1 - F_x(7.5))$$

$$= 0.432 + 0.4242 + 0.1652 + 2.64$$

$$= 3.6614$$

Results obtained through Simulations



Average waiting time = 3.7

- Average Waiting Time: 3.7

Conclusions

- Both theoretical and simulation results match, indicating the accuracy of the theoretical model in predicting the average waiting time.