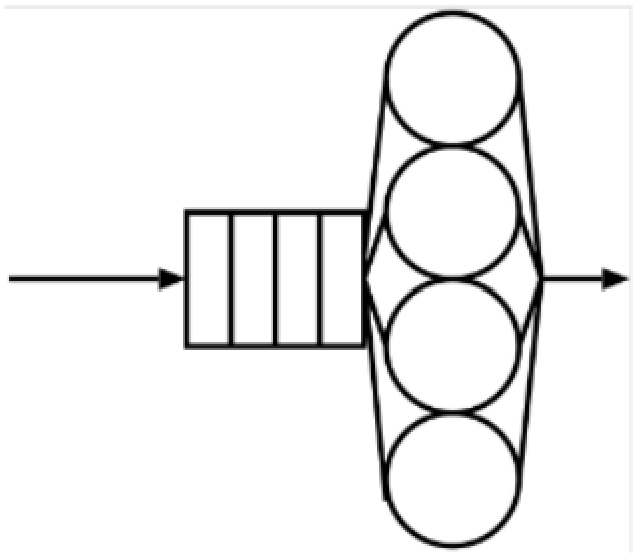
16.Finite Capacity and M-M-c-K queues

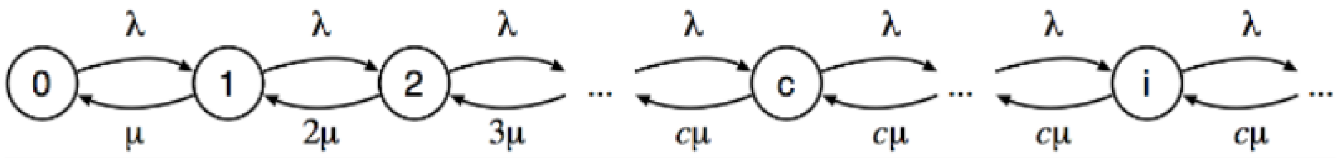
The idea of finite capacity is that a device can handle a maximum amount of jobs and can have a maximum queue that represent the maximum capacity it can have until there are losses.

M/M/c

Similar to the M/M/2 but it has c > 2 servers.

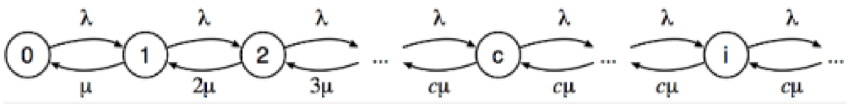


In this case, the death rate of the corresponding birth-death process linearly increases up to the state where all the c servers are busy. Then the death rate remains constant at c.μ for the remaining states in which queue is actually forming. As for the M/M/2 case, the arrival rate is constant for all the states and it is equal to λ:



The service rate at which the jobs finishes increases up to state c and then remain stable at μ. The state probability can again be computed exploiting the fact that the M/M/c queue is a birth-death process.

$$p_n = p_0 \cdot \prod_{i=1}^{n} \frac{\lambda_{i-1}}{\mu_i}$$

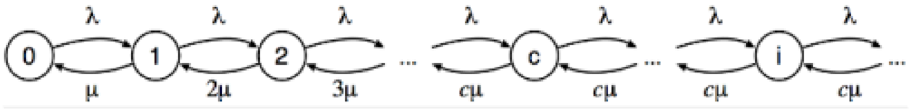


$$\pi_n = \begin{cases} \pi_0 \cdot \prod_{i=1}^{n} \frac{\lambda}{i \cdot \mu} & n \le c \\ \pi_0 \cdot \prod_{i=1}^{c} \frac{\lambda}{i \cdot \mu} \cdot \prod_{i=c+1}^{n} \frac{\lambda}{c \cdot \mu} & n > c \end{cases} \qquad \pi_n = \begin{cases} \frac{\pi_0}{n!} \cdot \left(\frac{\lambda}{\mu}\right)^n & n < c \\ \frac{\pi_0}{c! \cdot c^{n-c}} \cdot \left(\frac{\lambda}{\mu}\right)^n & n \ge c \end{cases}$$

$$\rho = \frac{\lambda}{c\mu} = \bar{U} = \frac{\lambda \cdot D}{c} \qquad \pi_n = \begin{cases} \frac{\pi_0}{n!} \cdot (c\rho)^n & n < c \\ \frac{\pi_0 c^c \rho^n}{c!} & n \ge c \end{cases} \qquad \pi_0 = \left[\underbrace{\frac{(c\rho)^c}{c!} \frac{1}{1-\rho}}_{n \ge c} + \underbrace{\sum_{k=0}^{c-1} \frac{(c\rho)^k}{k!}}_{n < c} \right]^{-1}$$

With extensions to the techniques we have seen for the M/M/1 and M/M/2, we can compute the average number of jobs in the queue:

$$\pi_n = \pi_0 \cdot \prod_{i=1}^{n} \frac{\lambda_{i-1}}{\mu_i}$$



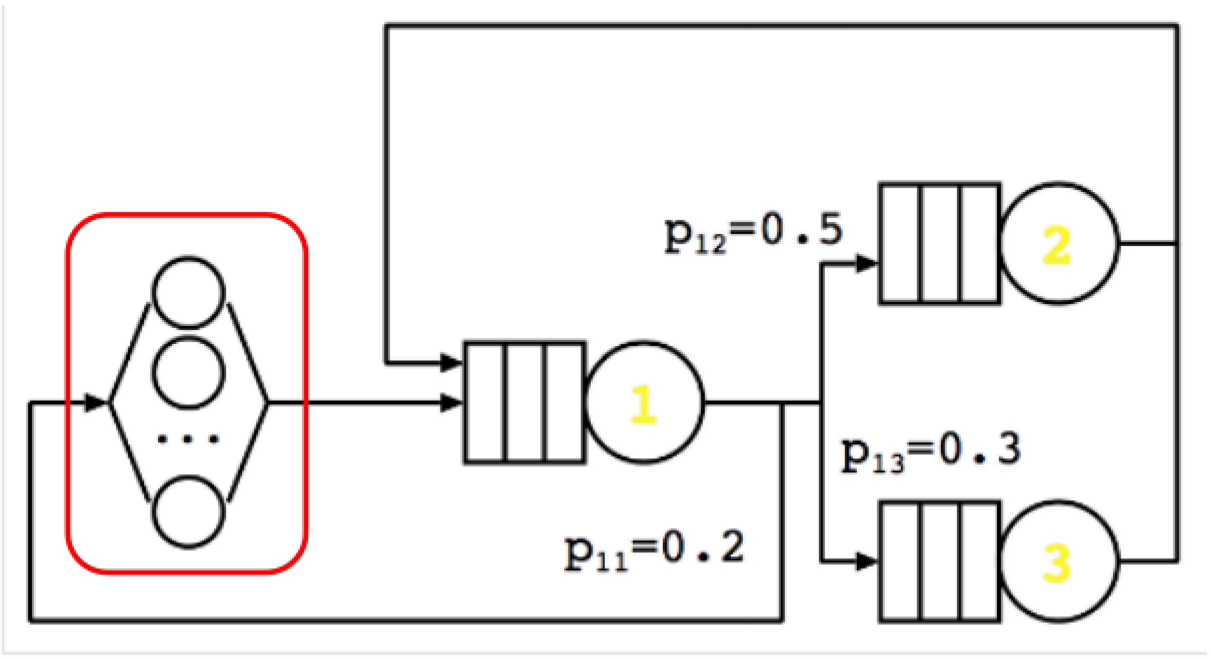
$$N = c\rho + \frac{\frac{\rho}{1-\rho}}{1 + (1-\rho)\left(\frac{c!}{(c\rho)^c}\right) \sum_{k=0}^{c-1} \frac{(c\rho)^k}{k!}}$$

We will not detail the proofs of these formula, but we will trust them, and use them when needed.

$$R = D + \Theta = D + \boxed{\frac{\frac{D}{c(1-\rho)}}{1 + (1-\rho)\left(\frac{c!}{(c\rho)^c}\right) \sum_{k=0}^{c-1} \frac{(c\rho)^k}{k!}}} = \Theta$$

The part in the dotted box, represents the time Θ spent in the queue before being served.

The number of servers can also be infinite so no jobs in queue. This is something done when we have parallel jobs. Infinite servers can be used to model activities that are performed by each job individually, and that result in a delay for a given amount of time (waiting time).
In queuing models, it is generally depicted as a series of servers, without a queue.



For infinite servers, the average utilization is zero, since the number of servers tends to infinity. The total utilization can instead be defined, since the busy time can be accounted for each job entering the system. In this case, the total utilization becomes identical to the average number of jobs in the system, and the response time corresponds to the average service time D.
R=D, U=N=λ.D
These models can be called M/M/oo and the markov chain is similar to M/M/2 but it goes to infinity.
MMoo the number of jobs in the system is distributed as a Poisson distribution.
In reality everything has finite capacity so sooner or later that capacity will be reached. What happens if this happens?
2 behaviour:

- Loss: if a system is full the job will not enter the system but dropped. This is used when the losses aren't important/can be restored. We can define a new performance index called the Drop rate: $D_r = \lim_{T \to \infty} \frac{L(T)}{T}$, $\lambda = D_r + X$ as the arrival rate is equal to the drop rate plus the throughput.
- Blocking: jobs are never lost. However the systems that are sending jobs will be stopped until the queue is more empty(either by an external source or by another station in the system). This idea may be complex if we have several queue. Several types of jobs are possible. The one that we will consider are: BBS(Blocking Before Service: when the station become full whatever we have before immediately stop working until we have the opportunuity of having an empty spot in the queue) and BAS(Blocking After Service: whenever the queue is full the station serves the next jobs and if the job is completed and cannot go to the other station the current station is blocked). BAS is more efficient as it only blocks when the next station is full but is harder to implement. These blocks can be applied only if the jobs can be stopped without cathastropic effects. We can define both for BBS and BAS the blocking probability, given $\beta(T)$ the time the system was blocked and

the blocking probability is $p_B = \lim_{T\to\infty} \frac{\beta(T)}{T}$. The relation between arrival rate and throughput can be defined as: $X = \lambda(1 - p_B)$

-M/M/1/K queues are used to model finite capacity systems. They correspond to a birth-death process with a constant death rate equal to μ, and constant arrival λi = λ until the system reaches its final capacity K, that is for 0 ≤ i < K.It is then λi=0 for i ≥ K, cutting out all the states with a population i > K.

In this case the service rate is constant and equal to μ. The arrival rate changes as is equal to zero from K on.

The state probability has the same expression as the one of the M/M/1 queue, but it is limited to a maximum population K. This leads to a different expression for the empty system probability p_0.

$$p_n = \begin{cases} p_0 \cdot \left(\frac{\lambda}{\mu}\right)^n & n \le K \\ 0 & n > K \end{cases} \qquad p_n = \begin{cases} p_0 \cdot \rho^n & n \le K \\ 0 & n > K \end{cases} \qquad \rho = \frac{\lambda}{\mu}$$

$$p_0 = \left[\sum_{n=0}^{K} \rho^n\right]^{-1} = \left[\frac{1-\rho^{K+1}}{1-\rho}\right]^{-1} = \frac{1-\rho}{1-\rho^{K+1}}$$

$$p_n = \begin{cases} \frac{1-\rho}{1-\rho^{K+1}} \cdot \rho^n & n \le K \\ 0 & n > K \end{cases}$$

We cannot have anything greater than K.

The utilization is:

$$U = 1 - p_0 = \frac{\rho - \rho^{K+1}}{1 - \rho^{K+1}}$$

It is less than the normal as we stop the system

The average number of jobs in the system:

$$N = \frac{\rho}{(1-\rho)} - \frac{(K+1)\rho^{K+1}}{1-\rho^{K+1}}$$

A system can be stable even if $\rho >> 1$ as we are dropping a lot of jobs

As introduced earlier, finite capacity systems are characterized by either loss or blocking.

The birth-death process just presented can model both loss and blocking before service.

The probability of having the system full, p_K , corresponds to either the loss probability p_L (loss systems) or the blocking probability p_B (BBS blocking):

$$p_L = p_B = p_K = \frac{\rho^K - \rho^{K+1}}{1 - \rho^{K+1}}$$

$$D_r = \lambda p_K = \lambda \frac{\rho^K - \rho^{K+1}}{1 - \rho^{K+1}}$$