Performance Evaluation : Contention and Queues

Stochastic Modeling of Computer Systems
MOSIG Master 2

Jean-Marc Vincent

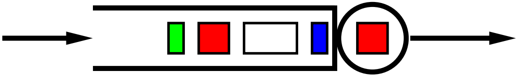
Laboratoire LIG, projet Inria-Mescal
UniversitéJoseph Fourier
Jean-Marc.Vincent@imag.fr

2012 October 19

Outline

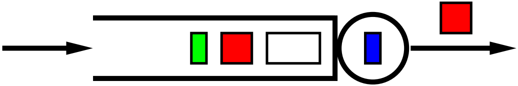
Queues

Queues are among simplest dynamic systems, but are still the source of many open problems.
Tasks do not have any constraints, sizes and arrival times are often independent.



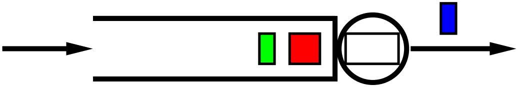
Queues

Queues are among simplest dynamic systems, but are still the source of many open problems.
Tasks do not have any constraints, sizes and arrival times are often independent.



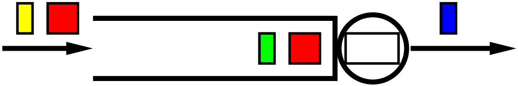
Queues

Queues are among simplest dynamic systems, but are still the source of many open problems.
Tasks do not have any constraints, sizes and arrival times are often independent.



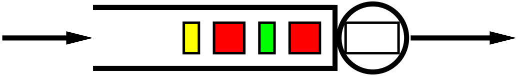
Queues

Queues are among simplest dynamic systems, but are still the source of many open problems.
Tasks do not have any constraints, sizes and arrival times are often independent.



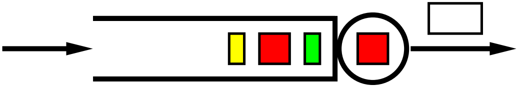
Queues

Queues are among simplest dynamic systems, but are still the source of many open problems.
Tasks do not have any constraints, sizes and arrival times are often independent.

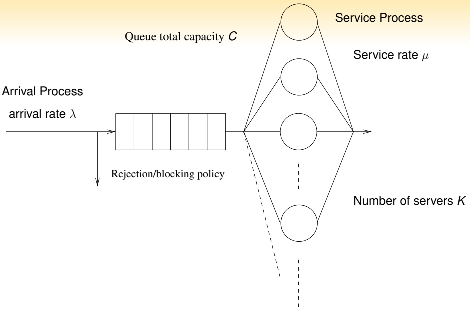


Queues

Queues are among simplest dynamic systems, but are still the source of many open problems.
Tasks do not have any constraints, sizes and arrival times are often independent.



Kendall's notation



Notation : $A/S/K/C/Disc$

- A : arrival process
- B : service process
- K : number of servers
- C : total queue capacity (including currently served customers)
- $Disc$: Service discipline (FIFO, LIFO, PS, Quantum, Priorities,...)

State variables

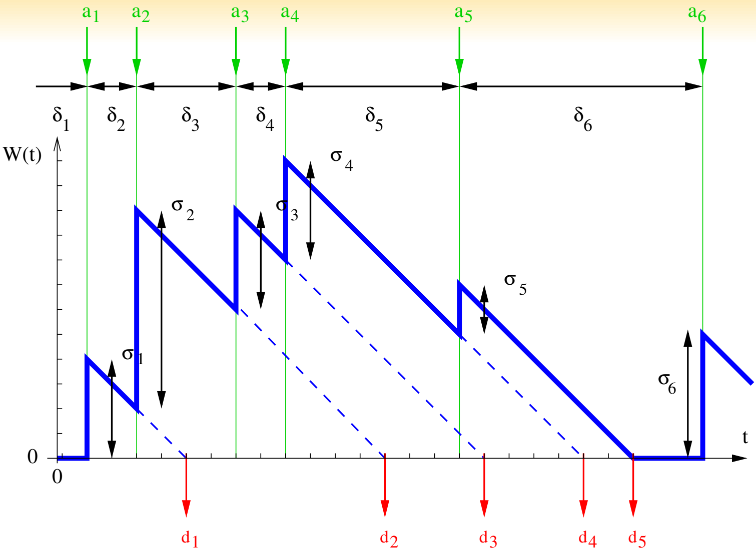
User variables

- Input rate λ or inter-arrival δ
- Service time σ or S (service rate μ)
- Waiting time W
- Response time R (in some books W)
- Rejection probability

Resource variables

- Resource utilisation (offered load) ρ
- Queue occupation N
- System availability

One Server Queue load



Lindley's formula (2)

W_n is the waiting time of the n-th task. It is a dynamical system of the form $W_n = \varphi(W_{n-1}, X_n)$ with $X_n = \sigma_{n-1} - \delta_n$ and φ defined by the

Lindley's equation:

$$W_n = \max\left(W_{n-1} + X_n, 0\right) .$$

- FIFO scheduling
- Non-linear evolution equation

Outline

Stability of the $G/G/1$ queue

$$
\begin{aligned}
W_n &= \max\left(W_{n-1} + X_n, 0\right), \\
&= \max\left(\max(W_{n-2} + X_{n-1}, 0) + X_n, 0\right), \\
&= \max\left(W_{n-2} + X_{n-1} + X_n, X_n, 0\right), \\
&= \max\left(W_{n-3} + X_{n-2} + X_{n-1} + X_n, X_{n-1} + X_n, X_n, 0\right), \\
&= \max\left(W_0 + X_1 + \cdots + X_{n-1} + X_n, \cdots, X_{n-1} + X_n, X_n, 0\right), \\
&= \max\left(X_1 + \cdots + X_{n-1} + X_n, \cdots, X_{n-1} + X_n, X_n, 0\right), \\
&\sim \max\left(X_n + \cdots + X_2 + X_1, \cdots, X_2 + X_1, X_1, 0\right), \\
&\stackrel{def}{=} M_n.
\end{aligned}
$$

$$
W_n =_{st} M_n = \max\left(M_{n-1}, X_1 + \cdots + X_n\right) .
$$

Stability of the $G/G/1$ queue (2)

$$W_n =_{st} M_n = \max\left(M_{n-1}, X_1 + \cdots + X_n\right).$$

M_n is a non-decreasing sequence
Either $M_n \longrightarrow M_\infty$ or $M_n \longrightarrow +\infty$

Stability

- $\mathbb{E}X = \mathbb{E}(\sigma - \delta) < 0$ The system is **Stable**

  $$M_\infty =_{st} \max(M_\infty + X, 0).$$

  Functional equation on the distribution

  $$\mathbb{P}(M_\infty < x) \stackrel{def}{=} F(x) = \int F(x-u)dF_X(u).$$

  Condition : $\mathbb{E}\sigma < \mathbb{E}\delta$ or $\lambda < \mu$
- $\mathbb{E}X = \mathbb{E}(\sigma - \delta) > 0$ The system is **Unstable**

Depends only on service and inter-arrival expectation

Loynes' scheme

Theorem

$W_n \leqslant_{st} W_{n+1}$ in a G/G/1 queue, initialy empty.

Proof. done by a backward coupling known as the Loynes' scheme.
Construct on a common probability space two trajectories by going backward in time: $S^1_{i-n}(\omega) = S^2_{i-n-1}(\omega)$ with distribution S_i and $T^1_{i-n}(\omega) = T^2_{i-n-1}(\omega)$, with distribution $T_i - T_{n+1}$ for all $0 \leqslant i \leqslant n+1$ and $S^1_{-n-1}(\omega) = 0$.
By construction, $W^1_0 =_{st} W_n$ and $W^2_0 =_{st} W_{n+1}$. Also, it should be clear that $0 = W^1_{-n+1}(\omega) \leqslant W^2_{-n+1}(\omega)$ for all ω.
This implies $W^1_{-i}(\omega) \leqslant W^2_{-i}(\omega)$ so that $W_n \leqslant_{st} W_{n+1}$.

Loynes' scheme

Theorem

$W_n \leqslant_{st} W_{n+1}$ in a G/G/1 queue, initialy empty.

Proof. done by a backward coupling known as the Loynes' scheme. Construct on a common probability space two trajectories by going backward in time: $S^1_{i-n}(\omega) = S^2_{i-n-1}(\omega)$ with distribution S_i and $T^1_{i-n}(\omega) = T^2_{i-n-1}(\omega)$, with distribution $T_i - T_{n+1}$ for all $0 \leqslant i \leqslant n+1$ and $S^1_{-n-1}(\omega) = 0$.

By construction, $W^1_0 =_{st} W_n$ and $W^2_0 =_{st} W_{n+1}$. Also, it should be clear that $0 = W^1_{-n+1}(\omega) \leqslant W^2_{-n+1}(\omega)$ for all ω.

This implies $W^1_{-i}(\omega) \leqslant W^2_{-i}(\omega)$ so that $W_n \leqslant_{st} W_{n+1}$.

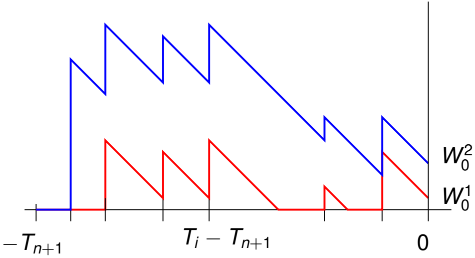
Loynes' scheme

Theorem

$W_n \leqslant_{st} W_{n+1}$ in a G/G/1 queue, initialy empty.

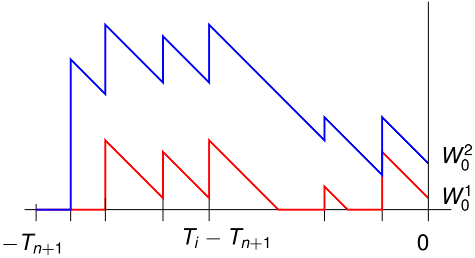
Proof. done by a backward coupling known as the Loynes' scheme.
Construct on a common probability space two trajectories by going backward in time: $S^1_{i-n}(\omega) = S^2_{i-n-1}(\omega)$ with distribution S_i and $T^1_{i-n}(\omega) = T^2_{i-n-1}(\omega)$, with distribution $T_i - T_{n+1}$ for all $0 \leqslant i \leqslant n+1$ and $S^1_{-n-1}(\omega) = 0$.
By construction, $W^1_0 =_{st} W_n$ and $W^2_0 =_{st} W_{n+1}$. Also, it should be clear that $0 = W^1_{-n+1}(\omega) \leqslant W^2_{-n+1}(\omega)$ for all ω.
This implies $W^1_{-i}(\omega) \leqslant W^2_{-i}(\omega)$ so that $W_n \leqslant_{st} W_{n+1}$.

Loynes' scheme



This has many consequences in terms of existence and uniqueness of a stationary (or limit) regime for the G/G/1 queue Baccelli Bremaud, 2002).

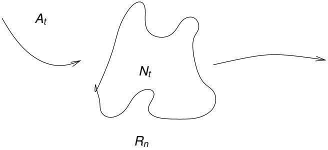
Loynes' scheme



This has many consequences in terms of existence and uniqueness of a stationary (or limit) regime for the G/G/1 queue Baccelli Bremaud, 2002).

Outline

Little's Formula



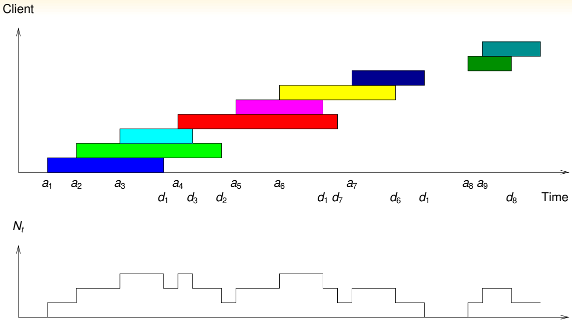
Assumptions

$$\lim_{t\to+\infty} \frac{A_t}{t} = \lambda, \quad \lim_{t\to+\infty} \frac{1}{t}\int_0^t N_s ds = \mathbb{E}N \text{ and } \lim_{n\to+\infty} \frac{1}{n}\sum_{i=1}^n R_i = \mathbb{E}R,$$

Little's Formula

$$\mathbb{E}N = \lambda \mathbb{E}R.$$

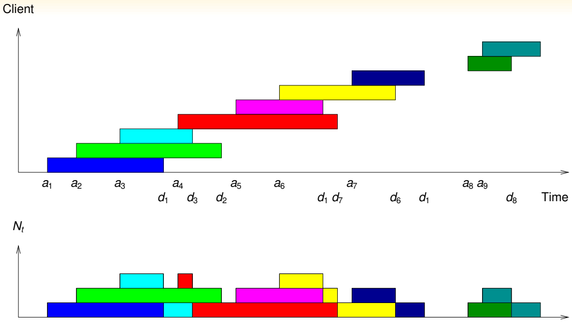
Little's Formula (proof)



$$\frac{1}{T}\int_0^T N_s ds = \frac{A_T}{T}\frac{1}{A_T}\sum_{i=1}^{A_T} R_i.$$

$T \to \infty$ implies $\mathbb{E}N = \lambda \mathbb{E}R$.

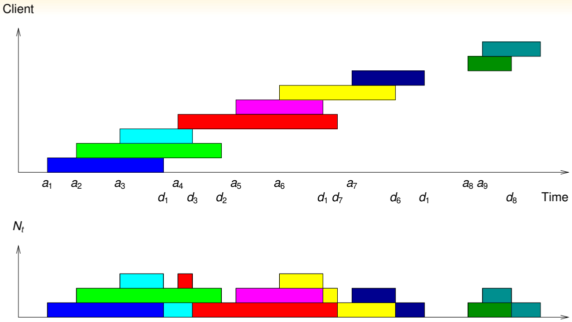
Little's Formula (proof)



$$\frac{1}{T}\int_0^T N_s ds = \frac{A_T}{T}\frac{1}{A_T}\sum_{i=1}^{A_T} R_i.$$

$T \to \infty$ implies $\mathbb{E}N = \lambda \mathbb{E}R$.

Little's Formula (proof)



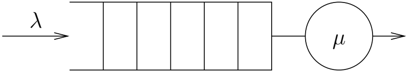
$$\frac{1}{T}\int_0^T N_s ds = \frac{A_T}{T}\frac{1}{A_T}\sum_{i=1}^{A_T} R_i.$$

$T \to \infty$ implies $\mathbb{E}N = \lambda \mathbb{E}R$.

Outline

1. Queues
2. Stability
3. Average
4. **Computable queues**
   - Single server queue
   - Limited capacity
5. Networks
6. Multiclass networks

M/M/1



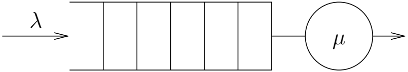
M/M/1 queue

- Infinite capacity
- Poisson(λ) arrivals
- Exp(μ) service times
- FIFO discipline

Definition

$\rho = \frac{\lambda}{\mu}$ is the traffic intensity of the queueing system.

M/M/1



M/M/1 queue

- Infinite capacity
- Poisson(λ) arrivals
- Exp(μ) service times
- FIFO discipline

Definition

$\rho = \frac{\lambda}{\mu}$ is the traffic intensity of the queueing system.

M/M/1

Let $X(t)$ the number of clients in the system at time t. $X(t)$ is a birth and death process.

Results for M/M/1 queue

1. Stable if and only if $\rho < 1$
2. Clients follow a geometric distribution $\forall i \in \mathbb{N}, \pi_i = (1-\rho)\rho^i$
3. Mean number of clients $\mathbb{E}X = \frac{\rho}{(1-\rho)}$
4. Average response time $\mathbb{E}T = \frac{1}{\mu - \lambda}$

M/M/1

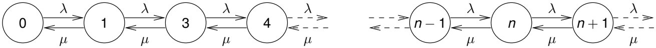
Let $X(t)$ the number of clients in the system at time t. $X(t)$ is a birth and death process.

Results for M/M/1 queue

1. Stable if and only if $\rho < 1$
2. Clients follow a geometric distribution $\forall i \in \mathbb{N}, \pi_i = (1-\rho)\rho^i$
3. Mean number of clients $\mathbb{E}X = \frac{\rho}{(1-\rho)}$
4. Average response time $\mathbb{E}T = \frac{1}{\mu-\lambda}$

M/M/1

Let $X(t)$ the number of clients in the system at time t. $X(t)$ is a birth and death process.

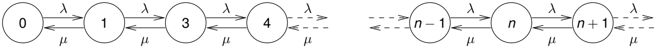


Results for M/M/1 queue

1. Stable if and only if $\rho < 1$
2. Clients follow a geometric distribution $\forall i \in \mathbb{N}, \pi_i = (1-\rho)\rho^i$
3. Mean number of clients $\mathbb{E}X = \frac{\rho}{(1-\rho)}$
4. Average response time $\mathbb{E}T = \frac{1}{\mu-\lambda}$

M/M/1

Let $X(t)$ the number of clients in the system at time t. $X(t)$ is a birth and death process.

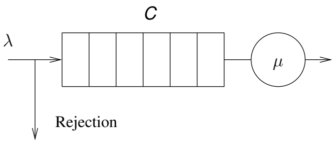


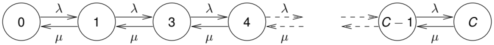
Results for M/M/1 queue

1. Stable if and only if $\rho < 1$
2. Clients follow a geometric distribution $\forall i \in \mathbb{N}, \pi_i = (1-\rho)\rho^i$
3. Mean number of clients $\mathbb{E}X = \frac{\rho}{(1-\rho)}$
4. Average response time $\mathbb{E}T = \frac{1}{\mu-\lambda}$

M/M/1/C

In reality, buffers are finite: M/M/1/C is a queueing system with rejection.





Results for M/M/1/C queue

Geometric distribution with finite state space

$$\pi(i) = \frac{(1-\rho)\rho^i}{1-\rho^{C+1}}$$

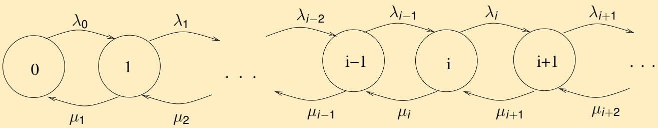
Outline

Reversibility

Proposition

An ergodic birth and death process is time-reversible.

Proof



By induction:

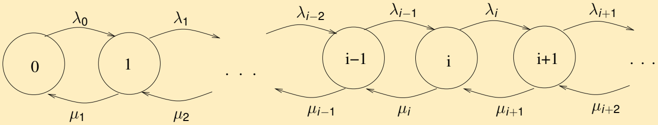
1. $\pi_0 \lambda_0 = \pi_1 \mu_1$
2. Suppose $\pi_{i-1}\lambda_{i-1} = \pi_i \mu_i$. Then
   $\pi_i(\lambda_i + \mu_i) = \pi_{i+1}\mu_{i+1} + \pi_{i-1}\lambda_{i-1}$
   Which gives $\pi_i \lambda_i = \pi_{i+1}\mu_{i+1}$.

Reversibility

Proposition

An ergodic birth and death process is time-reversible.

Proof



By induction:

1. $\pi_0\lambda_0 = \pi_1\mu_1$
2. Suppose $\pi_{i-1}\lambda_{i-1} = \pi_i\mu_i$. Then
   $\pi_i(\lambda_i + \mu_i) = \pi_{i+1}\mu_{i+1} + \pi_{i-1}\lambda_{i-1}$
   Which gives $\pi_i\lambda_i = \pi_{i+1}\mu_{i+1}$.

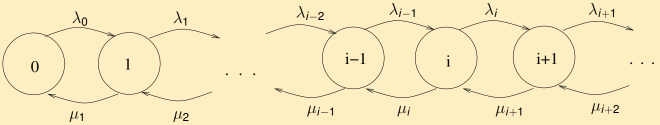
□

Reversibility

Proposition

An ergodic birth and death process is time-reversible.

Proof



By induction:

1. $\pi_0 \lambda_0 = \pi_1 \mu_1$
2. Suppose $\pi_{i-1}\lambda_{i-1} = \pi_i \mu_i$. Then
   $\pi_i(\lambda_i + \mu_i) = \pi_{i+1}\mu_{i+1} + \pi_{i-1}\lambda_{i-1}$
   Which gives $\pi_i \lambda_i = \pi_{i+1}\mu_{i+1}$.

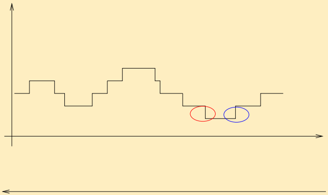
□

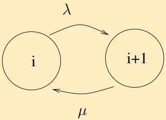
Burke's theorem

Theorem

The output process of an M/M/s queue is a Poisson process that is independent of the number of customers in the queue.

Sketch of Proof.





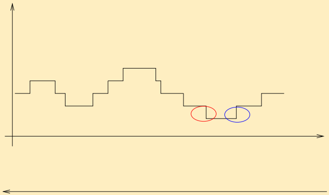
$X(t)$ increases by 1 at rate $\lambda\pi_i$ (Poisson process λ). Reverse process increases by 1 at rate $\mu\pi_{i+1} = \lambda\pi_i$ by reversibility.

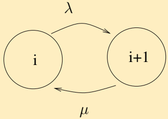
Burke's theorem

Theorem

The output process of an M/M/s queue is a Poisson process that is independent of the number of customers in the queue.

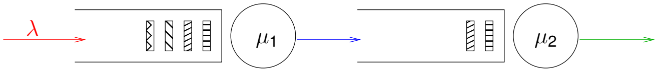
Sketch of Proof.





$X(t)$ increases by 1 at rate $\lambda\pi_i$ (Poisson process λ). Reverse process increases by 1 at rate $\mu\pi_{i+1} = \lambda\pi_i$ by reversibility. □

Open Queueing Networks



Let X_1 and X_2 denote the number of clients in queues 1 and 2 respectively.

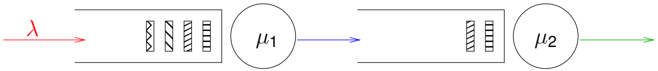
Lemma

X_1 and X_2 are independent rv's.

Proof

Arrival process at queue 1 is Poisson(λ) so future arrivals are independent of $X_1(t)$.
By time reversibility $X_1(t)$ is independent of past departures.
Since these departures are the arrival process of queue 2, $X_1(t)$ and $X_2(t)$ are independent. □

Open Queueing Networks



Let X_1 and X_2 denote the number of clients in queues 1 and 2 respectively.

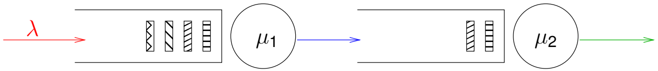
Lemma

X_1 and X_2 are independent rv's.

Proof

Arrival process at queue 1 is Poisson(λ) so future arrivals are independent of $X_1(t)$.
By time reversibility $X_1(t)$ is independent of past departures.
Since these departures are the arrival process of queue 2, $X_1(t)$ and $X_2(t)$ are independent. □

Open Queueing Networks



Let X_1 and X_2 denote the number of clients in queues 1 and 2 respectively.

Lemma

X_1 and X_2 are independent rv's.

Proof

Arrival process at queue 1 is Poisson(λ) so future arrivals are independent of $X_1(t)$.
By time reversibility $X_1(t)$ is independent of past departures.
Since these departures are the arrival process of queue 2, $X_1(t)$ and $X_2(t)$ are independent. □

Open Queueing Networks

Theorem

The number of clients at server 1 and 2 are independent and

$$P(n_1, n_2) = \left(\frac{\lambda}{\mu_1}\right)^{n_1} \left(1 - \frac{\lambda}{\mu_1}\right) \left(\frac{\lambda}{\mu_2}\right)^{n_2} \left(1 - \frac{\lambda}{\mu_1}\right)$$

Proof

By independence of X_1 and X_2 the joint probability is the product of M/M/1 distributions. □

This result is called a product-form result for the tandem queue.
This product form also appears in more general networks of queues.

Open Queueing Networks

Theorem

The number of clients at server 1 and 2 are independent and

$$P(n_1, n_2) = \left(\frac{\lambda}{\mu_1}\right)^{n_1}\left(1 - \frac{\lambda}{\mu_1}\right)\left(\frac{\lambda}{\mu_2}\right)^{n_2}\left(1 - \frac{\lambda}{\mu_1}\right)$$

Proof

By independence of X_1 and X_2 the joint probability is the product of M/M/1 distributions. □

This result is called a product-form result for the tandem queue.
This product form also appears in more general networks of queues.

Open Queueing Networks

Theorem

The number of clients at server 1 and 2 are independent and

$$P(n_1, n_2) = \left(\frac{\lambda}{\mu_1}\right)^{n_1} \left(1 - \frac{\lambda}{\mu_1}\right) \left(\frac{\lambda}{\mu_2}\right)^{n_2} \left(1 - \frac{\lambda}{\mu_1}\right)$$

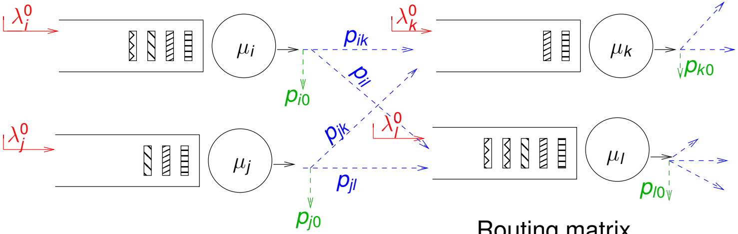
Proof

By independence of X_1 and X_2 the joint probability is the product of M/M/1 distributions. □

This result is called a product-form result for the tandem queue.
This product form also appears in more general networks of queues.

Open Queueing Networks

Example of a feed-forward network:



- Exponential service times
- output of i is routed to j with probability p_{ij}
- external traffic arrives at i with rate λ_i^0
- packets exiting queue i leave the system with probability p_{i0}.

Routing matrix

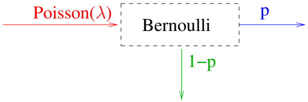
$$R = \begin{pmatrix} 0 & p_{ij} & p_{ik} & p_{il} \\ p_{ji} & 0 & p_{jk} & p_{jl} \\ p_{ki} & p_{kj} & 0 & p_{jl} \\ p_{li} & p_{lj} & p_{lk} & 0 \end{pmatrix}$$

Open Queueing Networks

Reminder

- $N(t)$ Poisson process with rate λ
- $Z(n)$ sequence of iid rv's $\sim$ Bernoulli(p) independent of N.

Suppose the nth trial is performed at the nth arrival of the Poisson process.

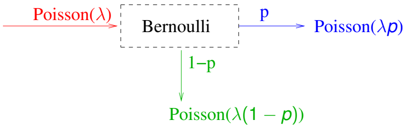


Open Queueing Networks

Reminder

- $N(t)$ Poisson process with rate λ
- $Z(n)$ sequence of iid rv's $\sim$ Bernoulli(p) independent of N.

Suppose the nth trial is performed at the nth arrival of the Poisson process. The resulting process $M(t)$ of successes is a Poisson process with rate λp.

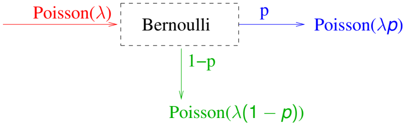


Open Queueing Networks

Reminder

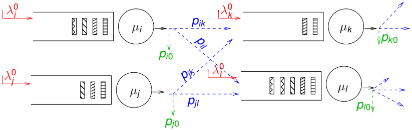
- $N(t)$ Poisson process with rate λ
- $Z(n)$ sequence of iid rv's $\sim$ Bernoulli(p) independent of N.

Suppose the nth trial is performed at the nth arrival of the Poisson process. The resulting process $M(t)$ of successes is a Poisson process with rate λp. The process of failures $L(t)$ is a Poisson process with rate $\lambda(1-p)$ and is independent of $M(t)$.



Open Queueing Networks

Define λ_i the total arrival rate at queue i, $1 \leqslant i \leqslant K$.



No feedback : from Burke we can consider K independent M/M/1 queues with Poisson arrivals with rate λ_i, where

$$\lambda_i = \lambda_i^0 + \sum_{j=0}^{K} \lambda_j p_{ji}$$

$$\vec{\Lambda} = \vec{\Lambda}^0 + \vec{\Lambda}\mathbf{R}$$

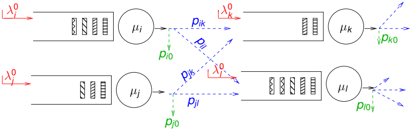
in matrix notation.

Stability condition

$\lambda_i < \mu_i, \forall i = 1, 2, \ldots, K.$

Open Queueing Networks

Define λ_i the total arrival rate at queue i, $1 \leqslant i \leqslant K$.



No feedback : from Burke we can consider K independent M/M/1 queues with Poisson arrivals with rate λ_i, where

$$\lambda_i = \lambda_i^0 + \sum_{j=0}^{K} \lambda_j p_{ji}$$

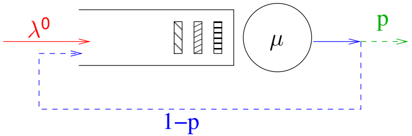
$$\vec{\Lambda} = \vec{\Lambda}^0 + \vec{\Lambda}\mathbf{R}$$

in matrix notation.

Stability condition

$\lambda_i < \mu_i, \forall i = 1, 2, \ldots, K.$

Open Queueing Networks



Remark

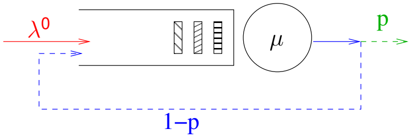
Arrivals are not Poisson anymore!

Result

The departure process is still Poisson with rate λp.

Proof in [Walrand, An Introduction to Queueing Networks, 1988].

Open Queueing Networks



Remark

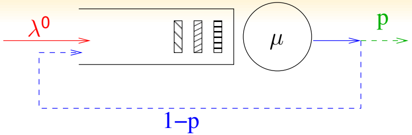
Arrivals are not Poisson anymore!

Result

The departure process is still Poisson with rate λp.

Proof in [Walrand, An Introduction to Queueing Networks, 1988].

Open Queueing Networks



Balance equations:

$$\begin{aligned}\pi(0)\lambda^0 &= \mu p \pi(1)\\ \pi(n)(\lambda^0 + p\mu) &= \lambda^0 \pi(n-1) + \mu p \pi(n+1), \quad n > 0\end{aligned}$$

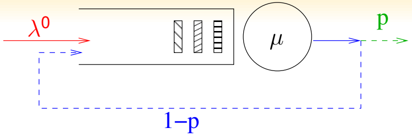
Actual arrival rate $\lambda = \lambda^0 + (1-p)\lambda$, so $\lambda^0 = \lambda p$ which gives

$$\begin{aligned}\pi(0)\lambda &= \mu\pi(1)\\ \pi(n)(\lambda + \mu) &= \lambda\pi(n-1) + \mu\pi(n+1), \quad n > 0 \qquad \text{M/M/1!}\end{aligned}$$

The unique solution is:

$$\pi(n) = \left(1 - \frac{\lambda}{\mu}\right)\left(\frac{\lambda}{\mu}\right)^n = \left(1 - \frac{\lambda^0}{p\mu}\right)\left(\frac{\lambda^0}{p\mu}\right)^n$$

Open Queueing Networks



Balance equations:

$$\pi(0)\lambda^0 = \mu p \pi(1)$$
$$\pi(n)(\lambda^0 + p\mu) = \lambda^0 \pi(n-1) + \mu p \pi(n+1), \quad n > 0$$

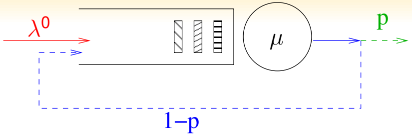
Actual arrival rate $\lambda = \lambda^0 + (1-p)\lambda,$ so $\lambda^0 = \lambda p$ which gives

$$\pi(0)\lambda = \mu\pi(1)$$
$$\pi(n)(\lambda + \mu) = \lambda\pi(n-1) + \mu\pi(n+1), \quad n > 0 \qquad \text{M/M/1!}$$

The unique solution is:

$$\pi(n) = \left(1 - \frac{\lambda}{\mu}\right)\left(\frac{\lambda}{\mu}\right)^n = \left(1 - \frac{\lambda^0}{p\mu}\right)\left(\frac{\lambda^0}{p\mu}\right)^n$$

Open Queueing Networks



Balance equations:

$$\begin{aligned} \pi(0)\lambda^0 &= \mu p \pi(1) \\ \pi(n)(\lambda^0 + p\mu) &= \lambda^0 \pi(n-1) + \mu p \pi(n+1), \quad n > 0 \end{aligned}$$

Actual arrival rate $\lambda = \lambda^0 + (1-p)\lambda$, so $\lambda^0 = \lambda p$ which gives

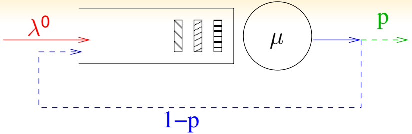
$$\begin{aligned} \pi(0)\lambda &= \mu \pi(1) \\ \pi(n)(\lambda + \mu) &= \lambda \pi(n-1) + \mu \pi(n+1), \quad n > 0 \end{aligned}$$

M/M/1!

The unique solution is:

$$\pi(n) = \left(1 - \frac{\lambda}{\mu}\right)\left(\frac{\lambda}{\mu}\right)^n = \left(1 - \frac{\lambda^0}{p\mu}\right)\left(\frac{\lambda^0}{p\mu}\right)^n$$

Open Queueing Networks



Balance equations:

$$\begin{aligned} \pi(0)\lambda^0 &= \mu p \pi(1) \\ \pi(n)(\lambda^0 + p\mu) &= \lambda^0 \pi(n-1) + \mu p \pi(n+1), \quad n > 0 \end{aligned}$$

Actual arrival rate $\lambda = \lambda^0 + (1-p)\lambda$, so $\lambda^0 = \lambda p$ which gives

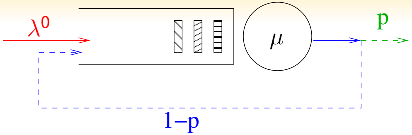
$$\begin{aligned} \pi(0)\lambda &= \mu \pi(1) \\ \pi(n)(\lambda + \mu) &= \lambda \pi(n-1) + \mu \pi(n+1), \quad n > 0 \end{aligned}$$

M/M/1!

The unique solution is:

$$\pi(n) = \left(1 - \frac{\lambda}{\mu}\right)\left(\frac{\lambda}{\mu}\right)^n = \left(1 - \frac{\lambda^0}{p\mu}\right)\left(\frac{\lambda^0}{p\mu}\right)^n$$

Open Queueing Networks



Balance equations:

$$\begin{aligned}\pi(0)\lambda^0 &= \mu p \pi(1) \\ \pi(n)(\lambda^0 + p\mu) &= \lambda^0 \pi(n-1) + \mu p \pi(n+1), \quad n > 0\end{aligned}$$

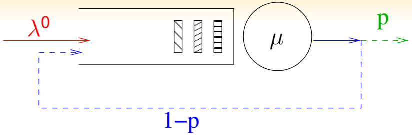
Actual arrival rate $\lambda = \lambda^0 + (1-p)\lambda$, so $\lambda^0 = \lambda p$ which gives

$$\begin{aligned}\pi(0)\lambda &= \mu \pi(1) \\ \pi(n)(\lambda + \mu) &= \lambda \pi(n-1) + \mu \pi(n+1), \quad n > 0\end{aligned} \qquad \text{M/M/1!}$$

The unique solution is:

$$\pi(n) = \left(1 - \frac{\lambda}{\mu}\right)\left(\frac{\lambda}{\mu}\right)^n = \left(1 - \frac{\lambda^0}{p\mu}\right)\left(\frac{\lambda^0}{p\mu}\right)^n$$

Open Queueing Networks



Balance equations:

$$\begin{aligned}\pi(0)\lambda^0 &= \mu p \pi(1)\\ \pi(n)(\lambda^0 + p\mu) &= \lambda^0 \pi(n-1) + \mu p \pi(n+1), \quad n > 0\end{aligned}$$

Actual arrival rate $\lambda = \lambda^0 + (1-p)\lambda$, so $\lambda^0 = \lambda p$ which gives

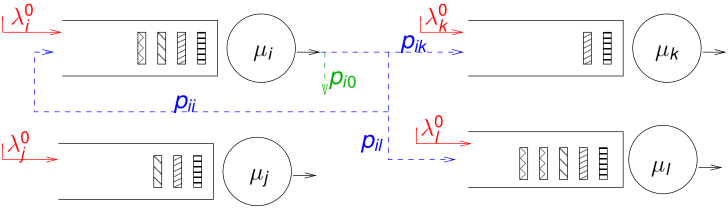
$$\begin{aligned}\pi(0)\lambda &= \mu \pi(1)\\ \pi(n)(\lambda + \mu) &= \lambda \pi(n-1) + \mu \pi(n+1), \quad n > 0\end{aligned}$$

M/M/1!

The unique solution is:

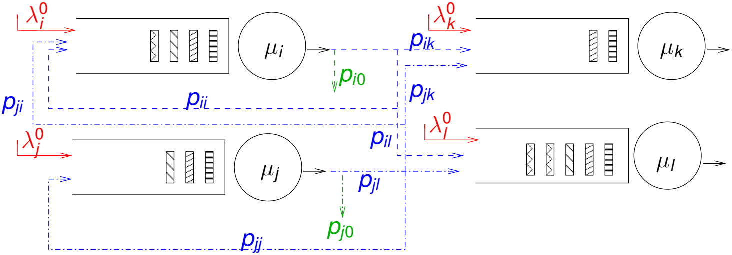
$$\pi(n) = \left(1 - \frac{\lambda}{\mu}\right)\left(\frac{\lambda}{\mu}\right)^n = \left(1 - \frac{\lambda^0}{p\mu}\right)\left(\frac{\lambda^0}{p\mu}\right)^n$$

Open Queueing Networks



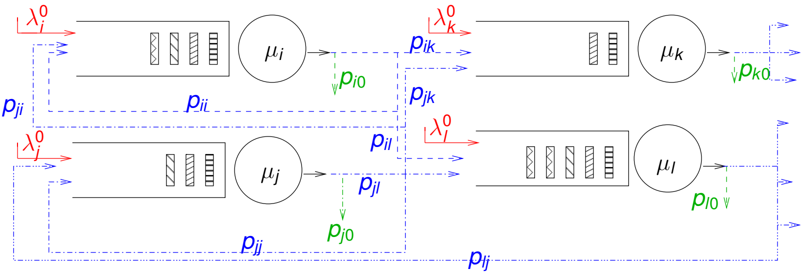
Backfeeding allowed.

Open Queueing Networks



Backfeeding allowed.

Open Queueing Networks



Backfeeding allowed.

Open Queueing Networks

Theorem (Jackson, 1957)

If $\lambda_i < \mu_i$ (stability condition), $\forall i = 1, 2, \ldots K$ then

$$\pi(\vec{n}) = \prod_{i=1}^{K} \left(1 - \frac{\lambda_i}{\mu_i}\right) \left(\frac{\lambda_i}{\mu_i}\right)^{n_i} \quad \forall \vec{n} = (n_1, \ldots, n_K) \in \mathbb{N}^K.$$

where $\lambda_1, \ldots, \lambda_K$ are the unique solution of the system

$$\lambda_i = \lambda_i^0 + \sum_{j=0}^{K} \lambda_j p_{ji}$$

Product form even with backfeeding!

Open Queueing Networks

Theorem (Jackson, 1957)

If $\lambda_i < \mu_i$ (stability condition), $\forall i = 1, 2, \ldots K$ then

$$\pi(\vec{n}) = \prod_{i=1}^{K} \left(1 - \frac{\lambda_i}{\mu_i}\right) \left(\frac{\lambda_i}{\mu_i}\right)^{n_i} \quad \forall \vec{n} = (n_1, \ldots, n_K) \in \mathbb{N}^K.$$

where $\lambda_1, \ldots, \lambda_K$ are the unique solution of the system

$$\lambda_i = \lambda_i^0 + \sum_{j=0}^{K} \lambda_j p_{ji}$$

Product form even with backfeeding!

Open Queueing Networks

Derive balance equations:

$$\pi(\vec{n})\left(\sum_{i=1}^{K}\lambda_i^0 + \sum_{i=1}^{K}\mathbb{1}\, n_i > 0\mu_i\right) = \sum_{i=1}^{K}\mathbb{1}\, n_i > 0\lambda_i^0\pi(\vec{n}-\vec{e}_i)$$
$$+\sum_{i=1}^{K}p_{i0}\mu_i\pi(\vec{n}+\vec{e}_i)$$
$$+\sum_{i=1}^{K}\sum_{j=1}^{K}\mathbb{1}\, n_j > 0p_{ij}\mu_i\pi(\vec{n}+\vec{e}_i-\vec{e}_j)$$

Then check that $\pi(\vec{n}) = \prod_{i=1}^{K}\left(1-\frac{\lambda_i}{\mu_i}\right)\left(\frac{\lambda_i}{\mu_i}\right)^{n_i}$ satisfies the balance equations with $\lambda_i = \lambda_i^0 + \sum_{j=0}^{K}\lambda_j p_{ji}$.

Open Queueing Networks

Derive balance equations:

$$\pi(\vec{n})\left(\sum_{i=1}^{K}\lambda_i^0+\sum_{i=1}^{K}\mathbb{1}\, n_i>0\mu_i\right) = \sum_{i=1}^{K}\mathbb{1}\, n_i>0\lambda_i^0\pi(\vec{n}-\vec{e}_i)$$

$$+\sum_{i=1}^{K}p_{i0}\mu_i\pi(\vec{n}+\vec{e}_i)$$

$$+\sum_{i=1}^{K}\sum_{j=1}^{K}\mathbb{1}\, n_j>0p_{ij}\mu_i\pi(\vec{n}+\vec{e}_i-\vec{e}_j)$$

Then check that $\pi(\vec{n})=\prod_{i=1}^{K}\left(1-\frac{\lambda_i}{\mu_i}\right)\left(\frac{\lambda_i}{\mu_i}\right)^{n_i}$ satisfies the balance equations with $\lambda_i=\lambda_i^0+\sum_{j=0}^{K}\lambda_j p_{ji}$. □

Open Queueing Networks

Derive balance equations:

$$\begin{aligned}
\pi(\vec{n})\left(\sum_{i=1}^{K}\lambda_i^0+\sum_{i=1}^{K}\mathbb{1}\, n_i>0\mu_i\right) \quad &= \quad \sum_{i=1}^{K}\mathbb{1}\, n_i>0\lambda_i^0\pi(\vec{n}-\vec{e}_i)\\
&\quad+\sum_{i=1}^{K}p_{i0}\mu_i\pi(\vec{n}+\vec{e}_i)\\
&\quad+\sum_{i=1}^{K}\sum_{j=1}^{K}\mathbb{1}\, n_j>0p_{ij}\mu_i\pi(\vec{n}+\vec{e}_i-\vec{e}_j)
\end{aligned}$$

Then check that $\pi(\vec{n})=\prod_{i=1}^{K}\left(1-\frac{\lambda_i}{\mu_i}\right)\left(\frac{\lambda_i}{\mu_i}\right)^{n_i}$ satisfies the balance equations

with $\lambda_i=\lambda_i^0+\sum_{j=0}^{K}\lambda_j p_{ji}$. □

Open Queueing Networks

Derive balance equations:

$$\pi(\vec{n}) \left(\sum_{i=1}^{K} \lambda_i^0 + \sum_{i=1}^{K} \mathbb{1} n_i > 0 \mu_i \right) = \sum_{i=1}^{K} \mathbb{1} n_i > 0 \lambda_i^0 \pi(\vec{n} - \vec{e}_i)$$
$$+ \sum_{i=1}^{K} p_{i0} \mu_i \pi(\vec{n} + \vec{e}_i)$$
$$+ \sum_{i=1}^{K} \sum_{j=1}^{K} \mathbb{1} n_j > 0 p_{ij} \mu_i \pi(\vec{n} + \vec{e}_i - \vec{e}_j)$$

Then check that $\pi(\vec{n}) = \prod_{i=1}^{K} \left(1 - \frac{\lambda_i}{\mu_i}\right) \left(\frac{\lambda_i}{\mu_i}\right)^{n_i}$ satisfies the balance equations with $\lambda_i = \lambda_i^0 + \sum_{j=0}^{K} \lambda_j p_{ji}$. □

Open Queueing Networks

Derive balance equations:

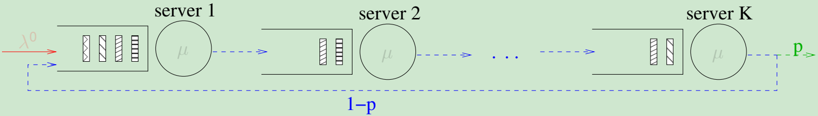
$$\pi(\vec{n})\left(\sum_{i=1}^{K}\lambda_i^0+\sum_{i=1}^{K}1\!\!1\, n_i>0\mu_i\right) = \sum_{i=1}^{K}1\!\!1\, n_i>0\lambda_i^0\pi(\vec{n}-\vec{e}_i)$$
$$+\sum_{i=1}^{K}p_{i0}\mu_i\pi(\vec{n}+\vec{e}_i)$$
$$+\sum_{i=1}^{K}\sum_{j=1}^{K}1\!\!1\, n_j>0p_{ij}\mu_i\pi(\vec{n}+\vec{e}_i-\vec{e}_j)$$

Then check that $\pi(\vec{n})=\prod_{i=1}^{K}\left(1-\frac{\lambda_i}{\mu_i}\right)\left(\frac{\lambda_i}{\mu_i}\right)^{n_i}$ satisfies the balance equations with $\lambda_i=\lambda_i^0+\sum_{j=0}^{K}\lambda_j p_{ji}$. □

Open Queueing Networks

Example

Switch**es** transmitting frames with random errors.



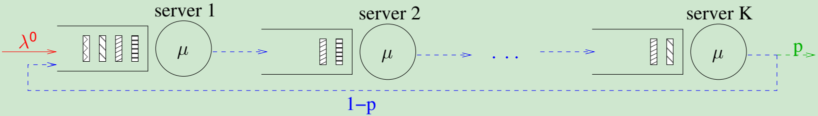
Traffic equations give $\lambda_i = \lambda_{i-1}$ for $i \geqslant 2$ and $\lambda_1 = \lambda^0 + (1-p)\lambda_K$.The unique solution is clearly $\lambda_i = \frac{\lambda^0}{p}$ for $1 \leqslant i \leqslant K$. Apply Jackson's theorem:

$$\pi(\vec{n}) = \left(1 - \frac{\lambda^0}{p\mu}\right)^K \left(\frac{\lambda^0}{p\mu}\right)^{n_1+\ldots+n_K} \qquad \forall \vec{n} = (n_1, \ldots, n_K) \in \mathbb{N}^K.$$

Open Queueing Networks

Example

Switches transmitting frames with random errors.



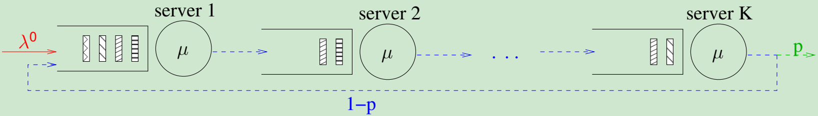
Traffic equations give $\lambda_i = \lambda_{i-1}$ for $i \geqslant 2$ and $\lambda_1 = \lambda^0 + (1-p)\lambda_K$.The unique solution is clearly $\lambda_i = \frac{\lambda^0}{p}$ for $1 \leqslant i \leqslant K$. Apply Jackson's theorem:

$$\pi(\vec{n}) = \left(1 - \frac{\lambda^0}{p\mu}\right)^K \left(\frac{\lambda^0}{p\mu}\right)^{n_1+\ldots+n_K} \qquad \forall \vec{n} = (n_1, \ldots, n_K) \in \mathbb{N}^K.$$

Open Queueing Networks

Example

Switches transmitting frames with random errors.



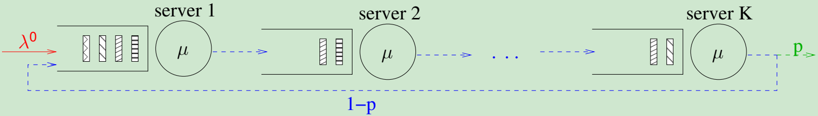
Traffic equations give $\lambda_i = \lambda_{i-1}$ for $i \geqslant 2$ and $\lambda_1 = \lambda^0 + (1-p)\lambda_K$. The unique solution is clearly $\lambda_i = \frac{\lambda^0}{p}$ for $1 \leqslant i \leqslant K$. Apply Jackson's theorem:

$$\pi(\vec{n}) = \left(1 - \frac{\lambda^0}{p\mu}\right)^K \left(\frac{\lambda^0}{p\mu}\right)^{n_1+\ldots+n_K} \qquad \forall \vec{n} = (n_1, \ldots, n_K) \in \mathbb{N}^K.$$

Open Queueing Networks

Example

Switches transmitting frames with random errors.



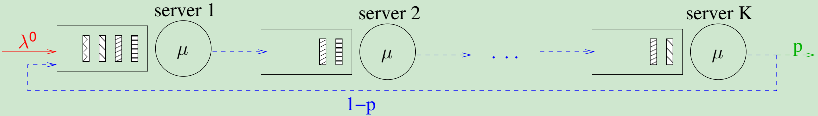
Traffic equations give $\lambda_i = \lambda_{i-1}$ for $i \geqslant 2$ and $\lambda_1 = \lambda^0 + (1-p)\lambda_K$.The unique solution is clearly $\lambda_i = \frac{\lambda^0}{p}$ for $1 \leqslant i \leqslant K$. Apply Jackson's theorem:

$$\pi(\vec{n}) = \left(1 - \frac{\lambda^0}{p\mu}\right)^K \left(\frac{\lambda^0}{p\mu}\right)^{n_1+\ldots+n_K} \qquad \forall \vec{n} = (n_1, \ldots, n_K) \in \mathbb{N}^K.$$

Open Queueing Networks

Example

Switches transmitting frames with random errors.



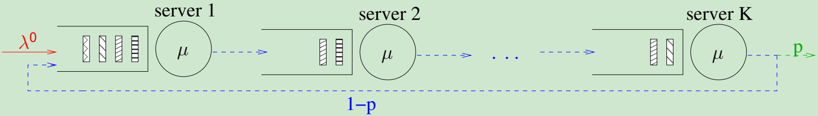
Traffic equations give $\lambda_i = \lambda_{i-1}$ for $i \geqslant 2$ and $\lambda_1 = \lambda^0 + (1-p)\lambda_K$.The unique solution is clearly $\lambda_i = \frac{\lambda^0}{p}$ for $1 \leqslant i \leqslant K$. Apply Jackson's theorem:

$$\pi(\vec{n}) = \left(1 - \frac{\lambda^0}{p\mu}\right)^K \left(\frac{\lambda^0}{p\mu}\right)^{n_1+\ldots+n_K} \qquad \forall \vec{n} = (n_1, \ldots, n_K) \in \mathbb{N}^K.$$

Open Queueing Networks

Example

Switches transmitting frames with random errors.



Using M/M/1 results for each queue we get the mean number of frames at each queue $\mathbb{E}X_i = \frac{\lambda^0}{p\mu - \lambda^0}$

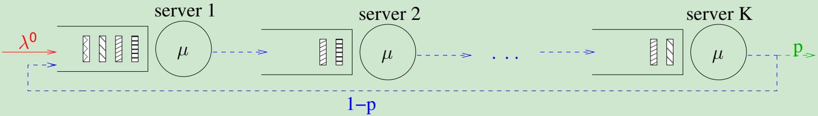
The expected transmission time of a frame is therefore (Little)

$$\mathbb{E}T = \frac{1}{\lambda^0}\mathbb{E}X = \frac{1}{\lambda^0}\sum_{i=1}^{K}\mathbb{E}X_i = \frac{K}{p\mu - \lambda^0}$$

Open Queueing Networks

Example

Switches transmitting frames with random errors.



Using M/M/1 results for each queue we get the mean number of frames at each queue $\mathbb{E}X_i = \frac{\lambda^0}{p\mu - \lambda^0}$

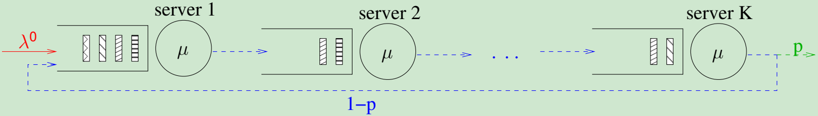
The expected transmission time of a frame is therefore (Little)

$$\mathbb{E}T = \frac{1}{\lambda^0}\mathbb{E}X = \frac{1}{\lambda^0}\sum_{i=1}^{K}\mathbb{E}X_i = \frac{K}{p\mu - \lambda^0}$$

Open Queueing Networks

Example

Switches transmitting frames with random errors.



Using M/M/1 results for each queue we get the mean number of frames at each queue $\mathbb{E}X_i = \frac{\lambda^0}{p\mu - \lambda^0}$

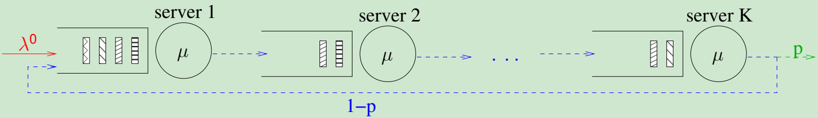
The expected transmission time of a frame is therefore (Little)

$$\mathbb{E}T = \frac{1}{\lambda^0}\mathbb{E}X = \frac{1}{\lambda^0}\sum_{i=1}^{K}\mathbb{E}X_i = \frac{K}{p\mu - \lambda^0}$$

Open Queueing Networks

Example

Switches transmitting frames with random errors.



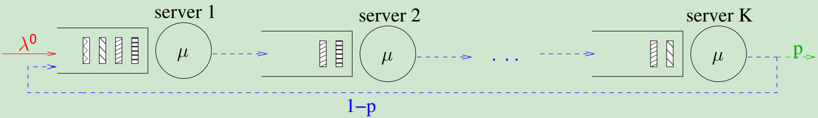
Using M/M/1 results for each queue we get the mean number of frames at each queue $\mathbb{E}X_i = \frac{\lambda^0}{p\mu - \lambda^0}$
The expected transmission time of a frame is therefore (Little)

$$\mathbb{E}T = \frac{1}{\lambda^0}\mathbb{E}X = \frac{1}{\lambda^0}\sum_{i=1}^{K}\mathbb{E}X_i = \frac{K}{p\mu - \lambda^0}$$

Open Queueing Networks

Example

Switches transmitting frames with random errors.



Using M/M/1 results for each queue we get the mean number of frames at each queue $\mathbb{E}X_i = \frac{\lambda^0}{p\mu - \lambda^0}$

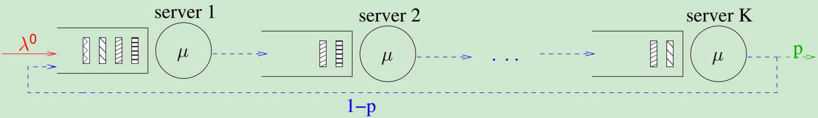
The expected transmission time of a frame is therefore (Little)

$$\mathbb{E}T = \frac{1}{\lambda^0}\mathbb{E}X = \frac{1}{\lambda^0}\sum_{i=1}^{K}\mathbb{E}X_i = \frac{K}{p\mu - \lambda^0}$$

Open Queueing Networks

Example

Switches transmitting frames with random errors.



Using M/M/1 results for each queue we get the mean number of frames at each queue $\mathbb{E}X_i = \frac{\lambda^0}{p\mu - \lambda^0}$
The expected transmission time of a frame is therefore (Little)

$$\mathbb{E}T = \frac{1}{\lambda^0}\mathbb{E}X = \frac{1}{\lambda^0}\sum_{i=1}^{K}\mathbb{E}X_i = \frac{K}{p\mu - \lambda^0}$$

Open Queueing Networks

Theorem

Consider an open network of K M/M/c_i queues. Let $\mu_i(n) = \mu_i \min(n, c_i)$ *and* $\rho_i = \frac{\lambda_i}{\mu_i}$.
Then if $\rho_i < c_i$ *for all* $1 \leqslant i \leqslant K$ *then*

$$\pi(\vec{n}) = \prod_{i=1}^{K} C_i \left(\frac{\lambda_i^{n_i}}{\prod_{m=1}^{n_i} \mu_i(m)} \right) \quad \forall \vec{n} = (n_1, \ldots, n_K) \in \mathbb{N}^K$$

where $(\lambda_1, \ldots, \lambda_K)$ *is the unique positive solution of the traffic equations*

$$\lambda_i = \lambda_i^0 + \sum_{j=0}^{K} \lambda_j p_{ji}, \quad \text{and where } C_i = \left(\sum_{m=1}^{c_i - 1} \frac{\rho_i}{i!} + \frac{\rho_i^{c_i}}{c_i!(1 - \rho_i / c_i)} \right)^{-1}$$

Closed Queueing Networks

Computing the normalization factor $C(N, K)$ is a heavy task!

$$C(n,k) = \sum_{\vec{n}\in S(n,k)} \prod_{i=1}^{k} \left(\frac{\lambda_i}{\mu_i}\right)^{n_i} = \sum_{m=0}^{n} \sum_{\substack{\vec{n} \in S(n,k) \\ n_k = m}} \prod_{i=1}^{k} \left(\frac{\lambda_i}{\mu_i}\right)^{n_i}$$

$$= \sum_{m=0}^{n} \left(\frac{\lambda_k}{\mu_k}\right)^{m} \sum_{\vec{n}\in S(n-m,k-1)} \prod_{i=1}^{k-1} \left(\frac{\lambda_i}{\mu_i}\right)^{n_i}$$

Convolution algorithm (Buzen,1973)

$$C(n,k) = \sum_{m=0}^{n} \left(\frac{\lambda_k}{\mu_k}\right)^{m} C(k-m, k-1) \text{ and } \begin{cases} C(n,1) & = \left(\frac{\lambda_1}{\mu_1}\right)^{n} \\ C(0,k) & = 1, \forall 1 \leqslant i \leqslant K \end{cases}$$

Closed Queueing Networks

Computing the normalization factor $C(N, K)$ is a heavy task!

$$C(n,k) = \sum_{\vec{n} \in S(n,k)} \prod_{i=1}^{k} \left(\frac{\lambda_i}{\mu_i}\right)^{n_i} = \sum_{m=0}^{n} \sum_{\substack{\vec{n} \in S(n,k) \\ n_k = m}} \prod_{i=1}^{k} \left(\frac{\lambda_i}{\mu_i}\right)^{n_i}$$

$$= \sum_{m=0}^{n} \left(\frac{\lambda_k}{\mu_k}\right)^{m} \sum_{\vec{n} \in S(n-m,k-1)} \prod_{i=1}^{k-1} \left(\frac{\lambda_i}{\mu_i}\right)^{n_i}$$

Convolution algorithm (Buzen,1973)

$$C(n,k) = \sum_{m=0}^{n} \left(\frac{\lambda_k}{\mu_k}\right)^{m} C(k-m, k-1) \text{ and } \begin{cases} C(n,1) & = \left(\frac{\lambda_1}{\mu_1}\right)^{n} \\ C(0,k) & = 1, \forall 1 \leqslant i \leqslant K \end{cases}$$

Closed Queueing Networks

Computing the normalization factor $C(N, K)$ is a heavy task!

$$C(n,k) = \sum_{\vec{n}\in S(n,k)} \prod_{i=1}^{k} \left(\frac{\lambda_i}{\mu_i}\right)^{n_i} = \sum_{m=0}^{n} \sum_{\substack{\vec{n} \in S(n,k) \\ n_k = m}} \prod_{i=1}^{k} \left(\frac{\lambda_i}{\mu_i}\right)^{n_i}$$

$$= \sum_{m=0}^{n} \left(\frac{\lambda_k}{\mu_k}\right)^{m} \sum_{\vec{n}\in S(n-m,k-1)} \prod_{i=1}^{k-1} \left(\frac{\lambda_i}{\mu_i}\right)^{n_i}$$

Convolution algorithm (Buzen,1973)

$$C(n,k) = \sum_{m=0}^{n} \left(\frac{\lambda_k}{\mu_k}\right)^{m} C(k-m, k-1) \text{ and } \begin{cases} C(n,1) & = \left(\frac{\lambda_1}{\mu_1}\right)^{n} \\ C(0,k) & = 1, \forall 1 \leqslant i \leqslant K \end{cases}$$

Closed Queueing Networks

Computing the normalization factor $C(N, K)$ is a heavy task!

$$C(n,k) = \sum_{\vec{n} \in S(n,k)} \prod_{i=1}^{k} \left(\frac{\lambda_i}{\mu_i}\right)^{n_i} = \sum_{m=0}^{n} \sum_{\substack{\vec{n} \in S(n,k) \\ n_k = m}} \prod_{i=1}^{k} \left(\frac{\lambda_i}{\mu_i}\right)^{n_i}$$

$$= \sum_{m=0}^{n} \left(\frac{\lambda_k}{\mu_k}\right)^{m} \sum_{\vec{n} \in S(n-m,k-1)} \prod_{i=1}^{k-1} \left(\frac{\lambda_i}{\mu_i}\right)^{n_i}$$

Convolution algorithm (Buzen,1973)

$$C(n,k) = \sum_{m=0}^{n} \left(\frac{\lambda_k}{\mu_k}\right)^{m} C(k-m, k-1) \text{ and } \begin{cases} C(n,1) & = & \left(\frac{\lambda_1}{\mu_1}\right)^{n} \\ C(0,k) & = & 1, \forall 1 \leqslant i \leqslant K \end{cases}$$

Closed Queueing Networks

Computing the normalization factor $C(N, K)$ is a heavy task!

$$C(n,k) = \sum_{\vec{n}\in S(n,k)} \prod_{i=1}^{k} \left(\frac{\lambda_i}{\mu_i}\right)^{n_i} = \sum_{m=0}^{n} \sum_{\substack{\vec{n} \in S(n,k) \\ n_k = m}} \prod_{i=1}^{k} \left(\frac{\lambda_i}{\mu_i}\right)^{n_i}$$

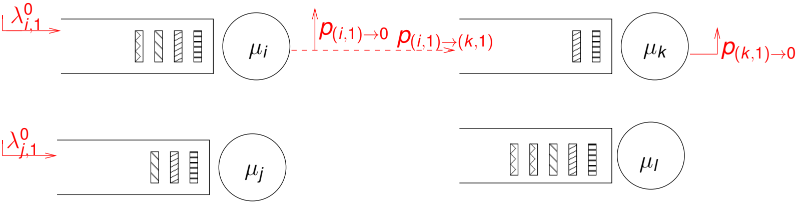
$$= \sum_{m=0}^{n} \left(\frac{\lambda_k}{\mu_k}\right)^{m} \sum_{\vec{n}\in S(n-m,k-1)} \prod_{i=1}^{k-1} \left(\frac{\lambda_i}{\mu_i}\right)^{n_i}$$

Convolution algorithm (Buzen,1973)

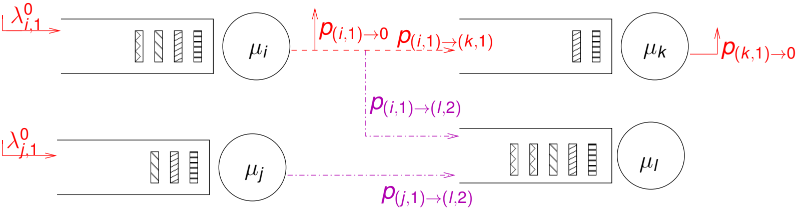
$$C(n,k) = \sum_{m=0}^{n} \left(\frac{\lambda_k}{\mu_k}\right)^{m} C(k-m, k-1) \text{ and } \begin{cases} C(n,1) &= \left(\frac{\lambda_1}{\mu_1}\right)^{n} \\ C(0,k) &= 1, \forall 1 \leqslant i \leqslant K \end{cases}$$

Outline

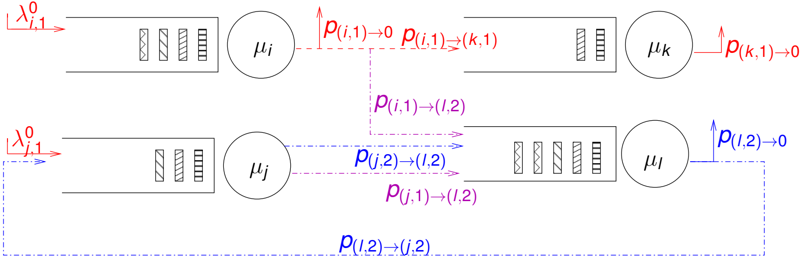
Multiclass Networks



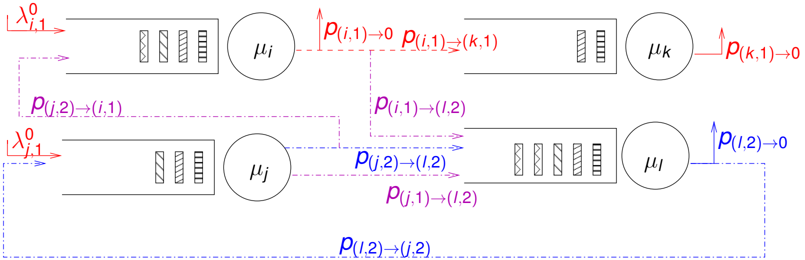
Multiclass Networks



Multiclass Networks



Multiclass Networks



Multiclass Networks

- $K < \infty$ nodes and $R < \infty$ classes
- Customer at node i in class r will go to node j with class s with probability $p_{(i,r);(j,s)}$
- (i, r) and (j, s) belong to the same subchain if $p_{(i,r);(j,s)} > 0$
- FIFO discipline and exponential service times

Definition

A subchain is open iff there exist one pair (i, r) for which $\lambda^0_{(i,r)} > 0$.

Multiclass Networks

- $K < \infty$ nodes and $R < \infty$ classes
- Customer at node i in class r will go to node j with class s with probability $p_{(i,r);(j,s)}$
- (i, r) and (j, s) belong to the same subchain if $p_{(i,r);(j,s)} > 0$
- FIFO discipline and exponential service times

Definition

A subchain is open iff there exist one pair (i, r) for which $\lambda^0_{(i,r)} > 0$.

Multiclass Networks

- $K < \infty$ nodes and $R < \infty$ classes
- Customer at node i in class r will go to node j with class s with probability $p_{(i,r);(j,s)}$
- (i, r) and (j, s) belong to the same subchain if $p_{(i,r);(j,s)} > 0$
- FIFO discipline and exponential service times

Definition

A subchain is open iff there exist one pair (i, r) for which $\lambda^0_{(i,r)} > 0$.

Multiclass Networks

- $K < \infty$ nodes and $R < \infty$ classes
- Customer at node i in class r will go to node j with class s with probability $p_{(i,r);(j,s)}$
- (i, r) and (j, s) belong to the same subchain if $p_{(i,r);(j,s)} > 0$
- FIFO discipline and exponential service times

Definition

A subchain is open iff there exist one pair (i, r) for which $\lambda^0_{(i,r)} > 0$.

Definition

A mixed system contains at least one open subchain and one closed subchain.

Multiclass Networks

- $K < \infty$ nodes and $R < \infty$ classes
- Customer at node i in class r will go to node j with class s with probability $p_{(i,r);(j,s)}$
- (i, r) and (j, s) belong to the same subchain if $p_{(i,r);(j,s)} > 0$
- FIFO discipline and exponential service times

Definition

A subchain is open iff there exist one pair (i, r) for which $\lambda^0_{(i,r)} > 0$.

Definition

A mixed system contains at least one open subchain and one closed subchain.

Multiclass Networks

- $K < \infty$ nodes and $R < \infty$ classes
- Customer at node i in class r will go to node j with class s with probability $p_{(i,r);(j,s)}$
- (i, r) and (j, s) belong to the same subchain if $p_{(i,r);(j,s)} > 0$
- FIFO discipline and exponential service times

Definition

A subchain is open iff there exist one pair (i, r) for which $\lambda^0_{(i,r)} > 0$.

Definition

A mixed system contains at least one open subchain and one closed subchain.

Multiclass Networks

- $K < \infty$ nodes and $R < \infty$ classes
- Customer at node i in class r will go to node j with class s with probability $p_{(i,r);(j,s)}$
- (i, r) and (j, s) belong to the same subchain if $p_{(i,r);(j,s)} > 0$
- FIFO discipline and exponential service times

Definition

A subchain is open iff there exist one pair (i, r) for which $\lambda^0_{(i,r)} > 0$.

Definition

A mixed system contains at least one open subchain and one closed subchain.

Multiclass Networks

The state of a multiclass network may be characterized by the number of customers of each class at each node

$$\vec{Q}(t) = (\vec{Q}_1(t), \vec{Q}_2(t), \ldots, \vec{Q}_K(t)) \text{ with } \vec{Q}_i(t) = (Q_{i1}(t)), \ldots, Q_{iR(t)})$$

Problem

$\vec{Q}(t)$ is not a CMTC!

To see why, consider the FIFO discipline: how do you know the class of the next customer?

Multiclass Networks

The state of a multiclass network may be characterized by the number of customers of each class at each node

$$\vec{Q}(t) = (\vec{Q}_1(t), \vec{Q}_2(t), \ldots, \vec{Q}_K(t)) \text{ with } \vec{Q}_i(t) = (Q_{i1}(t)), \ldots, Q_{iR(t)})$$

Problem

$\vec{Q}(t)$ is not a CMTC!

To see why, consider the FIFO discipline: how do you know the class of the next customer?

Multiclass Networks

The state of a multiclass network may be characterized by the number of customers of each class at each node

$$\vec{Q}(t) = (\vec{Q}_1(t), \vec{Q}_2(t), \ldots, \vec{Q}_K(t)) \text{ with } \vec{Q}_i(t) = (Q_{i1}(t)), \ldots, Q_{iR(t)})$$

Problem

$\vec{Q}(t)$ is not a CMTC!

To see why, consider the FIFO discipline: how do you know the class of the next customer?

Multiclass Networks

Define $\vec{X}_i(t) = (I_{i1}(t), \ldots, I_{iQ_i(t)}(t))$ with $I_{ij}(t)$ the class of the jth customer at node i.

Proposition

$\vec{X}(t)$ is a CMTC!

Solving the balance equations for X gives a product-form solution. The steady-state distribution of $\vec{X}(t)$ also gives the distribution of $\vec{Q}(t)$ by aggregation of states.

Multiclass Networks

Define $\vec{X}_i(t) = (I_{i1}(t), \ldots, I_{iQ_i(t)}(t))$ with $I_{ij}(t)$ the class of the jth customer at node i.

Proposition

$\vec{X}(t)$ is a CMTC!

Solving the balance equations for X gives a product-form solution. The steady-state distribution of $\vec{X}(t)$ also gives the distribution of $\vec{Q}(t)$ by aggregation of states.

Multiclass Networks

Define $\vec{X}_i(t) = (I_{i1}(t), \ldots, I_{iQ_i(t)}(t))$ with $I_{ij}(t)$ the class of the jth customer at node i.

Proposition

$\vec{X}(t)$ is a CMTC!

Solving the balance equations for X gives a product-form solution. The steady-state distribution of $\vec{X}(t)$ also gives the distribution of $\vec{Q}(t)$ by aggregation of states.

Other queueing networks

Jackson networks imply

- FIFO discipline
- probabilistic routing

These assumptions can be relaxed using BCMP and Kelly networks.

BCMP networks

Definition

BCMP networks are multiclass networks with exponential service times and c_i servers at node i.

Service disciplines may be:

- FCFS
- Processor Sharing
- Infinite Server
- LCFS

BCMP networks also have product-form solution!

BCMP networks

Definition

BCMP networks are multiclass networks with exponential service times and c_i servers at node i.

Service disciplines may be:

- FCFS
- Processor Sharing
- Infinite Server
- LCFS

BCMP networks also have product-form solution!

BCMP networks

Consider an open/closed/mixed BCMP network with K nodes and R classes in which each node is either FIFO,PS,LIFO or IS. Define

- $\rho_{ir} = \frac{\lambda_{ir}}{\mu_{ir}}$ for LIFO, IS and PS nodes
- $\rho_{ir} = \frac{\lambda_{ir}}{\mu_{i}}$ for FIFO nodes
- $\lambda_{ir} = \lambda_{ir}^0 + \sum_{(j,s)\in E_k} \lambda_{js} p_{(i,r);(j,s)}$ for any (i,r) of each open subchain E_k
- $\lambda_{ir} = \sum_{(j,s)\in E_m} \lambda_{js} p_{(i,r);(j,s)}$ for any (i,r) of each closed subchain E_m

BCMP networks

Consider an open/closed/mixed BCMP network with K nodes and R classes in which each node is either FIFO,PS,LIFO or IS. Define

- $\rho_{ir} = \frac{\lambda_{ir}}{\mu_{ir}}$ for LIFO, IS and PS nodes
- $\rho_{ir} = \frac{\lambda_{ir}}{\mu_i}$ for FIFO nodes
- $\lambda_{ir} = \lambda_{ir}^0 + \sum_{(j,s)\in E_k} \lambda_{js} p_{(i,r);(j,s)}$ for any (i, r) of each open subchain E_k
- $\lambda_{ir} = \sum_{(j,s)\in E_m} \lambda_{js} p_{(i,r);(j,s)}$ for any (i, r) of each closed subchain E_m

BCMP networks

Theorem

The steady-state distribution is given by: for all $\vec{n}$ in state space $\mathcal{S}$,

$$\pi(\vec{n}) = \frac{1}{G}\prod_{i=1}^{K} f_i(\vec{n}_i) \quad \textit{with } G = \sum_{\vec{n}\in\mathcal{S}}\prod_{i=1}^{K} f_i(\vec{n}_i)$$

with $\vec{n} = (\vec{n}_1, \ldots, \vec{n}_K) \in \mathcal{S}$ and $\vec{n}_i = (n_{i1}, \ldots, n_{iR})$, if and only if (stability condition for open subchains) $\sum_{r:(i,r)\in \text{ any open } E_k} \rho_{ir} < 1, \quad \forall 1 \leqslant i \leqslant K.$

Moreover, $f_i(\vec{n}_i)$ has an explicit expression for each service discipline.

BCMP networks

FIFO $f_i(\vec{n}_i) = |n_i|! \prod_{j=1}^{|n_i|} \frac{1}{\alpha_i(j)} \prod_{r=1}^{R} \frac{\rho_{ir}^{n_{ir}}}{n_{ir}!}$ with $\alpha_j(j) = min(c_i, j)$.

PS or LIFO $f_i(\vec{n}_i) = |n_i|! \prod_{r=1}^{R} \frac{\rho_{ir}^{n_{ir}}}{n_{ir}!}$

IS $f_i(\vec{n}_i) = \prod_{r=1}^{R} \frac{\rho_{ir}^{n_{ir}}}{n_{ir}!}$

Extensions

the BCMP product form result may be extended to the following cases:

- state-dependent routing probabilities
- arrivals depending on the number of customers in the corresponding subchain

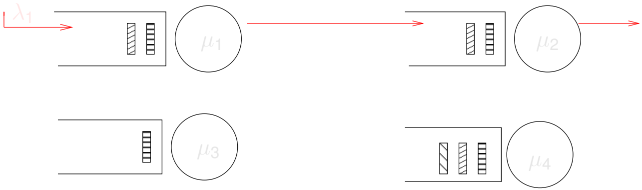
Kelly networks

In Kelly networks the routing is deterministic. The network is then characterized by its set of nodes and its set of routes.

Definition

In a Kelly network, each class of customers corresponds to a route.

Let λ_k be the arrival rate of class k clients (Poisson process). Note that class k customers can only arrive at one single node.



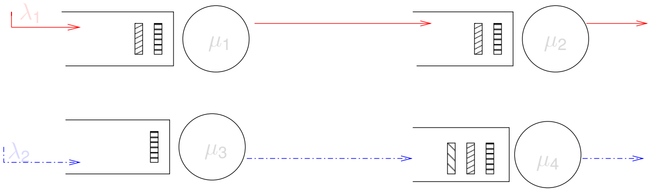
Kelly networks

In Kelly networks the routing is deterministic. The network is then characterized by its set of nodes and its set of routes.

Definition

In a Kelly network, each class of customers corresponds to a route.

Let λ_k be the arrival rate of class k clients (Poisson process). Note that class k customers can only arrive at one single node.



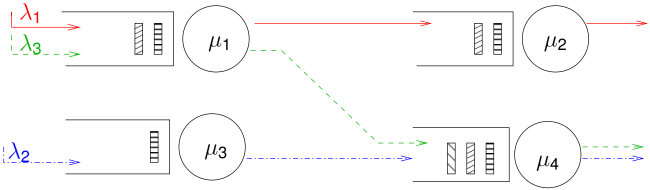
Kelly networks

In Kelly networks the routing is deterministic. The network is then characterized by its set of nodes and its set of routes.

Definition

In a Kelly network, each class of customers corresponds to a route.

Let λ_k be the arrival rate of class k clients (Poisson process). Note that class k customers can only arrive at one single node.



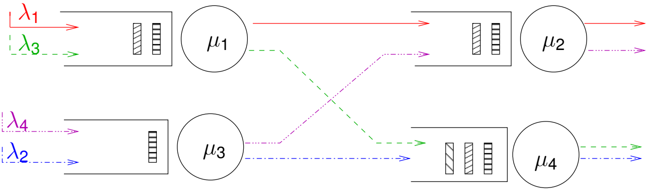
Kelly networks

In Kelly networks the routing is deterministic. The network is then characterized by its set of nodes and its set of routes.

Definition

In a Kelly network, each class of customers corresponds to a route.

Let λ_k be the arrival rate of class k clients (Poisson process). Note that class k customers can only arrive at one single node.



Kelly networks

the state space of a Kelly network is the set of $N \times K$ matrices
$M = ((m_{i,k}))$ with $m_{i,k}$ is the number of class k clients in queue i

Theorem (Kelly)

$$\pi_M = \prod_{i=1}^{N} \left(1 - \frac{\hat{\lambda}_i}{\mu_i}\right) \frac{\hat{m}_i!}{m_{i,1}! \cdots m_{i,K}!} \left(\frac{\hat{\lambda}_{i,1}}{\mu_i}\right)^{m_{i,1}} \cdots \left(\frac{\hat{\lambda}_{i,K}}{\mu_i}\right)^{m_{i,K}}$$

with $\hat{\lambda}_{i,k}$ global input rate of class k clients in queue i
with $\hat{\lambda}_i = \sum_k \hat{\lambda}_{i,k}$ global input rate queue i
and $\hat{m}_i = \sum_k m_{i,k}$