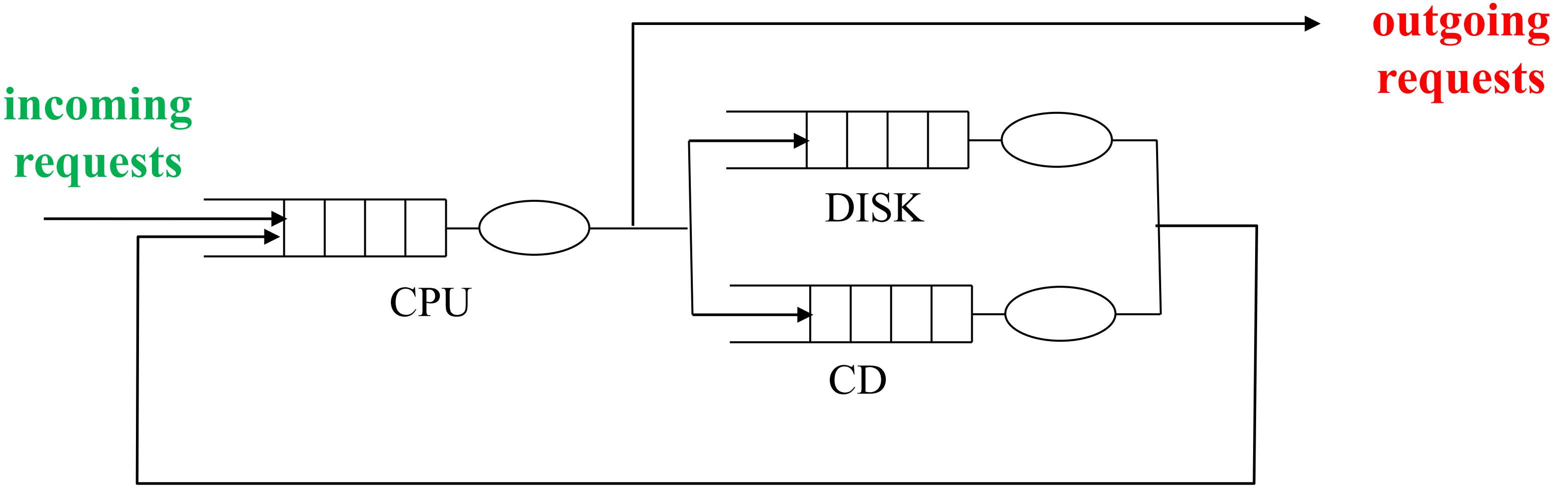
Queuing Networks

- **Outline of queuing networks**

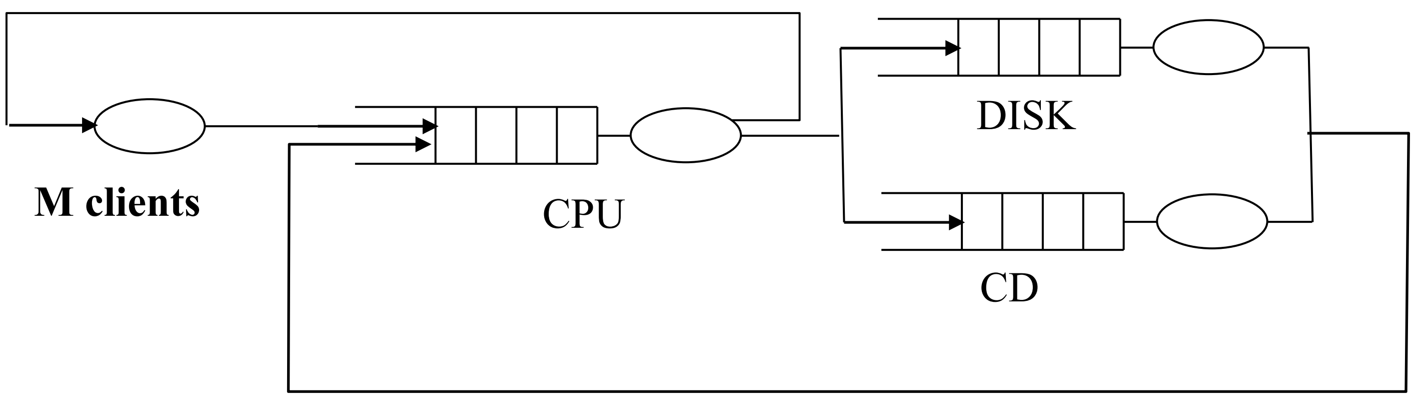
- **Mean Value Analisys (MVA) for open and closed queuing networks**

Open queuing networks

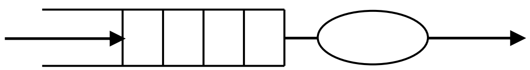


Closed queuing networks

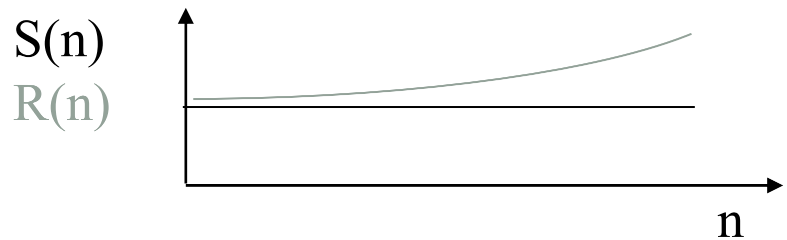
(finite number of users)

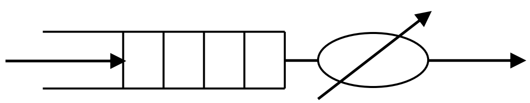


Kind of resources in a queuing network

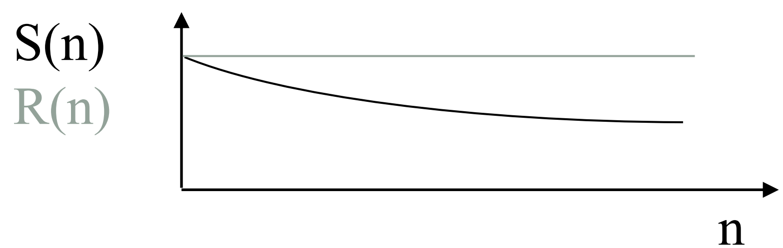


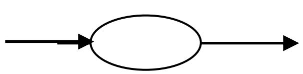
Load independent



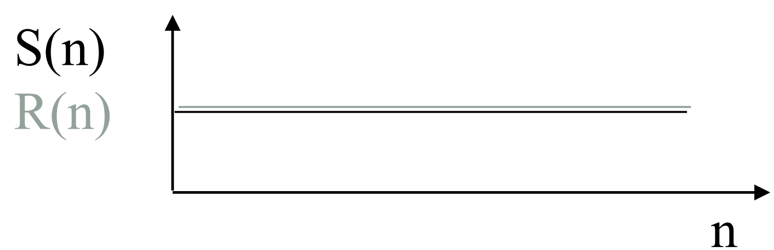


Load dependent





Delay



Definitions

K: number of queues

X_0: network average throughput. If open network in a stationary condition $X_0 = \lambda$

V_i: average number of visits a generic request makes to *i* server from its generation to its service time (request goes out from the system if open network)

S_i: average request <u>service</u> time at the server *i*

W_i: average request <u>waiting</u> time in the queue *i*

R_i: average request <u>response</u> time in the queue *i*

$$\mathbf{R_i = S_i + W_i}$$

Definitions

X_i: throughput for the *i*-th queue

$$X_i = X_0 V_i$$

R'_i: average request residence time in the queue *i* from its creation to its service completion time (request goes out from the system if open network)

$$R'_i = V_i R_i$$

D_i: request service demand to a server in a queue *i* from its creation to its service completion time (request goes out from the system if open network)

$$D_i = V_i S_i$$

Q_i: total time a request spends waiting in the queue *i* from its creation to its service time (request goes out from the system if open network)

$$Q_i = V_i W_i$$

$$R'_i = V_i R_i = V_i (W_i + S_i) = W_i V_i + S_i V_i = Q_i + D_i$$

R_0: total average request response time ((from the whole system)

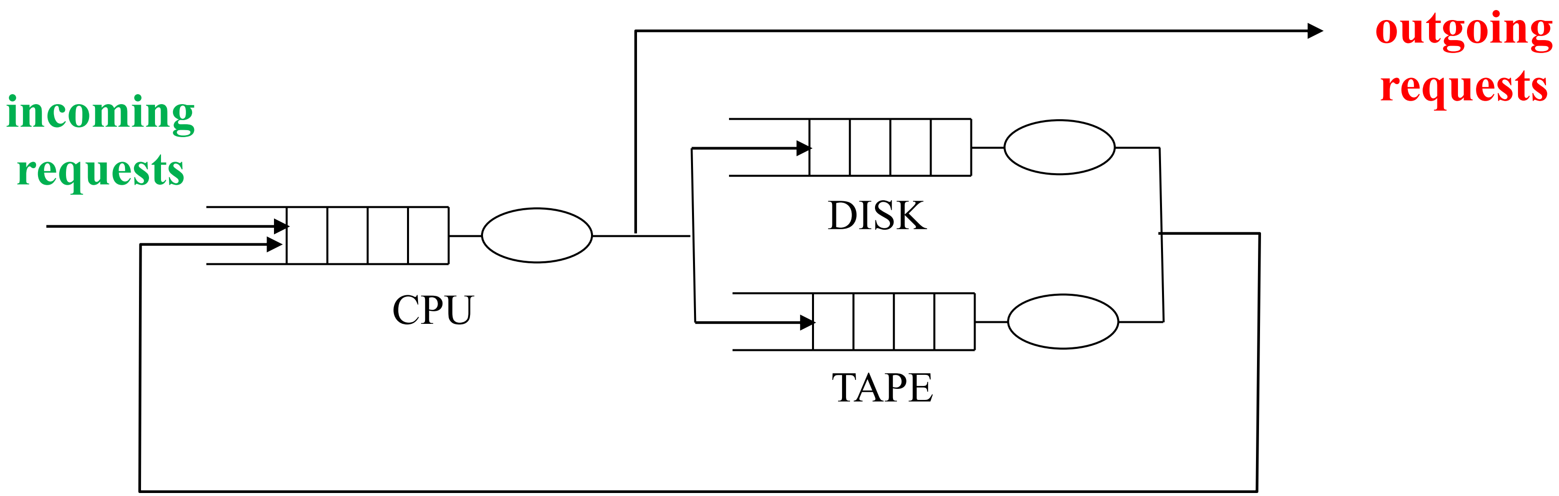
$$R_0 = \Sigma^k_{i=1} R'_i$$

n_i: average number of requests waiting or in service at the queue *i*

N: average number of requests in the system

$$N = \Sigma^k_{i=1} n_i$$

Open queuing networks



Open networks
(Single Class)

Equations:

Arrival theorem (for open networks): the average number of requests in a queue *i* that an incoming request find in the same queue (n^a_i), is equal to the average number of requests in the queue *i* (n_i).

$$R_i(n) = S_i + W_i(n) = S_i + n_i\ S_i$$

Using Little's Law ($n_i = X_i R_i$) and $U_i = X_i S_i$:

$$R_i = \frac{S_i}{(1-U_i)}$$

given that

$$R_i = S_i\,(1 + n_i) = S_i + S_i X_i R_i = S_i + U_i R_i$$

$$R_i\,(1- U_i) = S_i$$

Open networks
(Single Class)

Equations:

Then:

$$R'_i = V_i\, R_i = \frac{D_i}{(1-U_i)}$$

besides:

$$n_i = \frac{U_i}{(1-U_i)}$$

because

$n_i = X_i\, R_i$

$R_i = S_i \;/\; (1- U_i)$
$U_i = X_i\, S_i$

Open networks (Single Class)

Calculation of the greatest λ :

In an open network the average frequency of users incoming into the network is fixed. For λ too much big the network will become unstable, we are then interested in the greatest value of λ that we can apply to the network.

Given: $U_i = X_i S_i = \lambda V_i S_i$

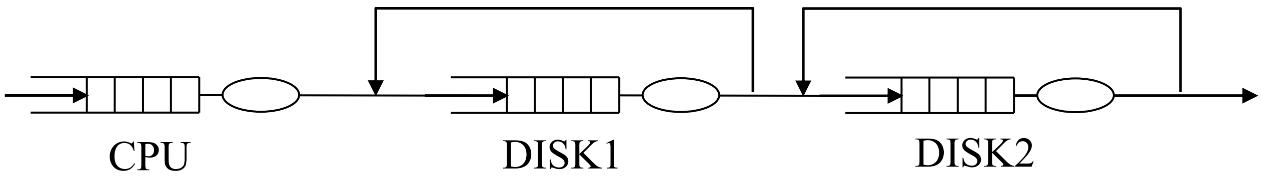
then: $\boldsymbol{\lambda = U_i / D_i}$ because $D_i = V_i S_i$

$\boldsymbol{U_i = 1}$ is the greatest utilization factor of a queue (i.e.= i), then we can calculate the greatest λ that doesn't make unstable the system as:

$$\lambda \leq \frac{1}{\max_{i=1}^{k} D_i}$$

DB Server

(example 9.1)



λ =10.800 requests per hour = 3 requests per sec = $\mathbf{X_0}$

D_{CPU} = 0,2 sec Service demand at CPU

V_{DISK1} = 5

V_{DISK2} = 3

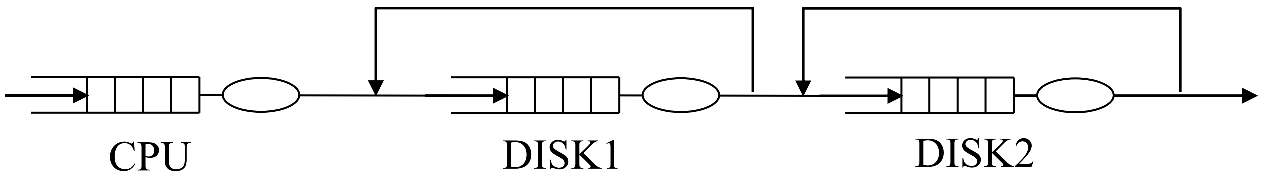
S_{DISK1} = S_{DISK2} = 15 msec

D_{DISK1} = V_{DISK1} * S_{DISK1} = 5 * 15 msec = 75 msec Service demand at disk 1

D_{DISK2} = V_{DISK2} * S_{DISK2} = 3 * 15 msec = 45 msec Service demand at disk 2

DB Server

(example 1)



.

Service Demand Law

$U_{CPU} = D_{CPU} * X_0 = 0{,}2$ sec/req * 3 req/sec = 0,6 CPU utilization

$U_{D1} = D_{DISK1} * X_0 =$ = 0,225 Disk1 utilization

$U_{D2} =$ = 0,135 Disk2 utilization

Residence time

$R'_{CPU} = D_{CPU} / (1 - U_{CPU}) = 0{,}5$ sec

$R'_{D1} = D_{DISK1} / (1 - U_{DISK1}) = 0{,}097$ sec

$R'_{D2} = D_{DISK2} / (1 - U_{DISK2}) = 0{,}052$ sec

Total response time

$R_0 = R'_{CPU} + R'_{D1} + R'_{D2} = 0{,}649$ sec

Average number of requests at each queue

$n_{CPU} = U_{CPU} / (1- U_{CPU}) = 0{,}6 / (1-0{,}6) = 1{,}5$

$n_{DISK1} = \quad = 0{,}29$

$n_{DISK2} = \quad = 0{,}16$

Total number of requests at the server

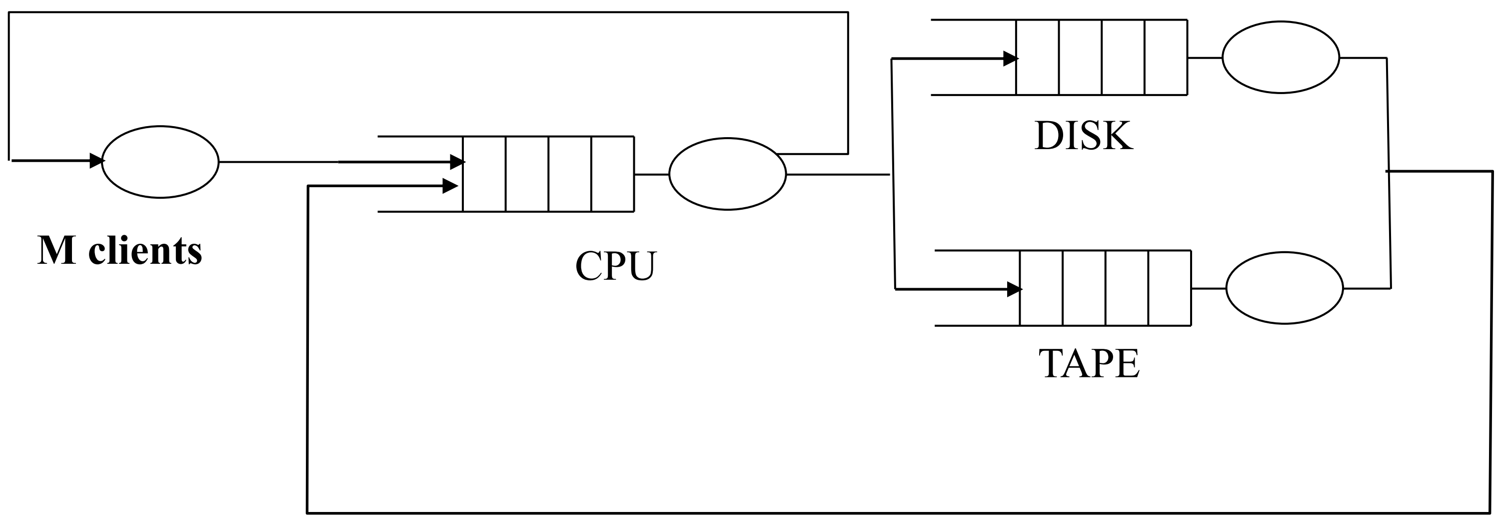
$N = n_{CPU} + n_{DISK2} + n_{DISK2} = 1{,}95$ requests

RMaximum arrival rate

$$\lambda = \frac{1}{max^{k}_{i=1} D_i} = \frac{1}{max\,(0{,}2;\, 0{,}075;\, 0{,}045)} = 5 \text{ req /sec}$$

Closed queue network

(finite number of users)



Closed networks
(Mean Value Analysis)

- Allows calculating the performance indeces (average response time, throughput, average queue lenght, etc…) for a closed network
- Iterative method based on the consideration that a queuing network results can be calculated from the same network results with a population reduced by one unit.
- Useful also for hybrid queuing networks

Definitions

. $\mathbf{X_0}$: average queuing network throughput.

. $\mathbf{V_i}$: average number of visits for a request at a queue *i*.

. $\mathbf{S_i}$: average service time for a request on the server *i*.

. $\mathbf{R_i}$: average response time for a request at the queue *i* (service+waiting time)

Closed networks
(Mean Value Analysis)

Definitions

- $\mathbf{R'_i}$: total average stay time for a request at the queue *i* considering all its visits at the queue. Equal to $\mathbf{V_i\,R_i}$
- $\mathbf{D_i}$: total average service time for a request at the queue *i* considering all its visits at the queue. Equal to $\mathbf{V_i\,S_i}$
- $\mathbf{R_0}$: average response time of the queuing network. Equal to the sum of the $\mathbf{R'_i}$
- $\mathbf{n_i^a}$: average number of the requests found by a request incoming in the queue.

Forced Flow Law

Then we have:

$$\mathbf{X_i = X_0\,V_i}$$

Mean Value Analysis
(Single class)

Equations:

$R_i(n) = S_i + W_i(n) = S_i + n_i^a(n)\, S_i = S_i\,(1+ n_i^a(n))$

Arrival Theorem: the average number of requests (n_i^a) in a queue *i* that an incoming request finds in the same queue is equal to the average number of requests in the queue *i* when *n-1* requests are in the queuing network ($n_i(n-1)$ that is *n* minus the incoming request that wants the service on the *i*-th queue)

in other words: $n_i^a(n) = n_i(n-1)$ (i.e n_i *is function of n-1)*

then: $R_i = S_i(1+n_i(n-1))$

and multiplying both members for V_i

→ $R'_i = D_i(1+n_i(n-1))$

Mean Value Analysis
(Single class)

Equations:

Applying Little's Law to the whole "queuing network" system ($n=X_0R_0$), we have:

$$\rightarrow \quad X_0 = n / R_0(n) = n / \Sigma^K_{r=1} R'_i(n)$$

Applying Little's Law and Forced Flow Law:

$$\rightarrow \quad n_i(n) = X_i(n)\, R_i(n) = X_0(n)\, V_i\, R_i(n) = X_0(n)\, R'_i(n)$$

Mean Value Analysis (Single class)

Three equations:

→ **Residence Time equation**

$$R'_i\,(n) = D_i[1+n_i(n-1)]$$

→ **Throughput equation**

$$X_0\,(n) = n \,/\, \sum^{K}_{r=1} R'_i(n)$$

→ **Queue lenght equation**

$$n_i(n) = X_0(n)\, R'_i(n)$$

Mean Value Analysis
(Single class)

Iterative procedure:

1. We know that $n_i(n) = 0$ for n=0: if no users is in the queuing network, then no users (requests) will be in every single queue.
2. Given $n_i(0)$ it's possible to evaluate all $R'_i(1)$
3. Given all $R'_i(1)$ it's possible to evaluate all $n_i(1)$ and $X_0(1)$
4. Given all $n_i(1)$ it's possible to evaluate all $R'_i(2)$
5. The procedure continues until all $n_i(n)$, $R'_i(n)$ and $X_0(n)$ are found, where n is the total number of users (requests) inside the network.

DB Server
(example 9.3)

- Requests from 50 clients
- Every request needs 5 record read from (visit to) a disk
- Average read time for a record (visit) = 9 msec
- Every request to DB needs 15 msec CPU

$D_{CPU} = S_{CPU} = 15$ msec **CPU service demand**

$D_{DISK} = S_{DISK} * V_{DISK} = 9 * 5 = 45$ msec **Disk service demand**

DB Server

(example 2)

Using MVA Equations

n = 0; Number of concurrent requests

$R'_{CPU} = 0;$ Residence time for CPU

$R'_{DISK} = 0;$ Residence time for disk

$R_0 = 0;$ Average response time

$X_0 = 0;$ Throughput

$n_{CPU} = 0;$ Queue lenght at CPU

$n_{DISK} = 0$ Queue lenght at disk

n = 1;

$R'_{CPU} = D_{CPU}\,(1+ n_{CPU}(0)) = D_{CPU} = 15$ msec;

$R'_{DISK} = D_{DISK}\,(1+ n_{DISK}(0)) = D_{DISK} = 45$ msec;

$R_0 = R'_{CPU} + R'_{DISK} = 60$ msec;

$X_0 = n/\,R_0 = 0{,}0167$ tx/msec

$n_{CPU} = X_0 * R'_{CPU} = 0{,}250$

$n_{DISK} = X_0 * R'_{DISK} = 0{,}750$

DB Server

(example 2)

n = 1;

$R'_{CPU} = D_{CPU}\ (1 + n_{CPU}(0)) = D_{CPU} = 15$ msec;

$R'_{DISK} = D_{DISK}\ (1 + n_{DISK}(0)) = D_{DISK} = 45$ msec;

$R_0 = R'_{CPU} + R'_{DISK} = 60$ msec;

$\mathbf{X_0 = 1 / R_0} = 0{,}0167$ tx/msec

$n_{CPU} = X_0 * R'_{CPU} = 0{,}250$

$n_{DISK} = 0{,}750$

n = 2;

$R'_{CPU} = D_{CPU}\ (1 + n_{CPU}(1)) = 15 * 1{,}25 = 18{,}75$ msec;

$R'_{DISK} = D_{DISK}\ (1 + n_{DISK}(1)) = 45 * 1{,}750 = 78{,}75$ msec;

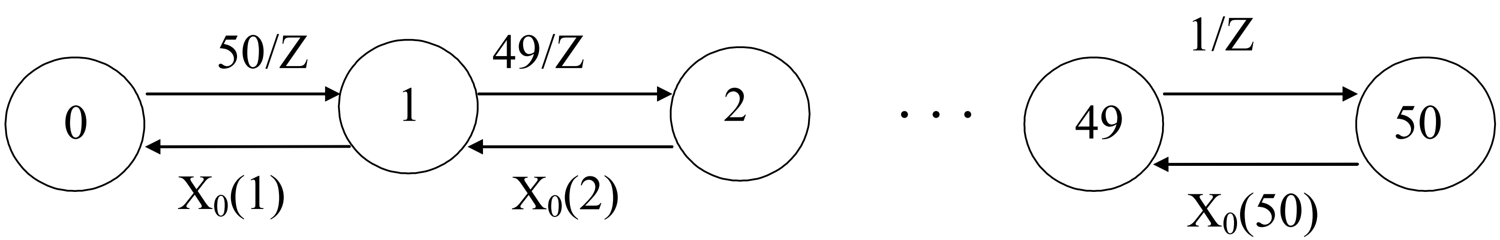
$R_0 = R'_{CPU} + R'_{DISK} = 97{,}5$ msec;

$\mathbf{X_0 = 2 / R_0} = 0{,}0205$ tx/msec

$n_{CPU} = X_0 * R'_{CPU} = 0{,}38$

$n_{DISK} = X_0 * R'_{DISK} = 1{,}62$

The related Markov process



Closed networks (Single Class) - Bounds

Bottleneck identification **(1/3)**

Usually the queuing network throughput will reach saturation if requests increase inside the system; we are then interested in finding the component in the system that causes saturation.

→ in open networks:

$$\lambda \leq \frac{1}{\max_{i=1}^{k} D_i}$$

and replacing λ with $X_0(n)$:

$$X_0(n) \leq \frac{1}{\max_{i=1}^{k} D_i}$$

Closed networks (Single Class) - Bounds

Bottleneck identification **(2/3)**

- from throughput equation of MVA, remembering that

$$R'_i(n) = D_i\,[1 + n_i(n-1)]$$

$\rightarrow$ $\mathbf{R'_i \geq D_i}$ for every queue *i*,

then we have (from Little's formula):

$$X_0(n) = \frac{n}{\sum_{r=1}^{K} R'_i} \leq \frac{n}{\sum_{r=1}^{K} D_i}$$

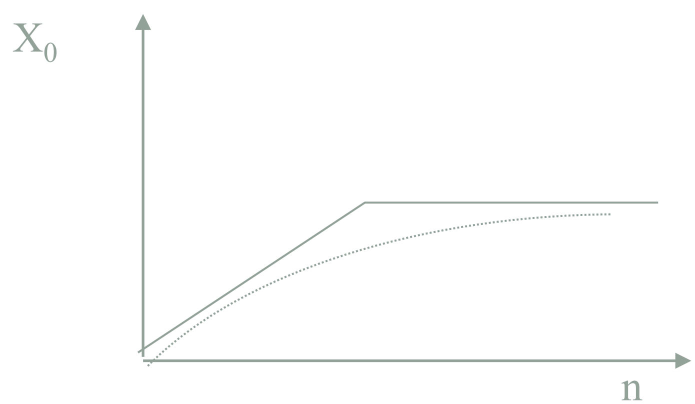
Closed networks
(Single Class) - Bounds

Bottleneck identification **(3/3)**

- Combining the preceding two equations we obtain:

$$\rightarrow X_0(n) \leq min \left(\frac{n}{\Sigma^K_{r=1} D_i}, \frac{1}{max^k_{i=1} D_i} \right)$$

For little n the throughput will increase at the most in a linear way with n, then becomes flat around the value $1/ max^k_{i=1} D_i$



Closed networks (Single Class) - Bounds

Average response time (1/2)

When throughput reaches its greatest value (that is for *n* big) the average response time is equivalent to:

$$R_0(n) \approx \frac{n}{\text{max throughput}}$$

Then for *n* big the response time increases in a linear way with *n*:

$$\rightarrow \; R_0(n) \approx n \; \max_{i=1}^{k} D_i$$

On the contrary, for small values of *n* (*n* near to 1) the average response time will be:

$$\rightarrow \; R_0(n) = \sum_{r=1}^{K} D_i$$

considering that all waiting times are **null**.

Closed networks (Single Class) - Bounds

<u>Average response time</u> (2/2)

We can establish a lower bound on average response time equal to:

$$\rightarrow \; R_0(n) \;\geq\; max \left(\Sigma^{K}_{i=1} D_i \;,\; n \cdot max^{k}_{i=1} D_i \right)$$

DB Server

(Example 9.4)

New scenarios with regard to previous example:

a. index variation in DB (# of disk access equal to 2,5 (before was 5))
b. 60% faster Disk (average service time = 5,63 msec)
c. faster CPU (service demand = 7,5 msec)

Scenario	Service demand D_{CPU}	Service demand D_{DISK}	ΣD_i	$1/ {}_{max}D_i$	Bottleneck
a	15	2,5 * 9 = 22,5	37,5	0,044	disk
b	15	5*5,63 = 28,15	43,15	0,036	disk
c	15/2 = 7,5	45	52,5	0,022	disk
a+b	15	2,5*5,63 = 14,08	29,08	0,067	CPU
a+c	15/2 = 7,5	2,5 * 9 = 22,5	30,0	0,044	disk