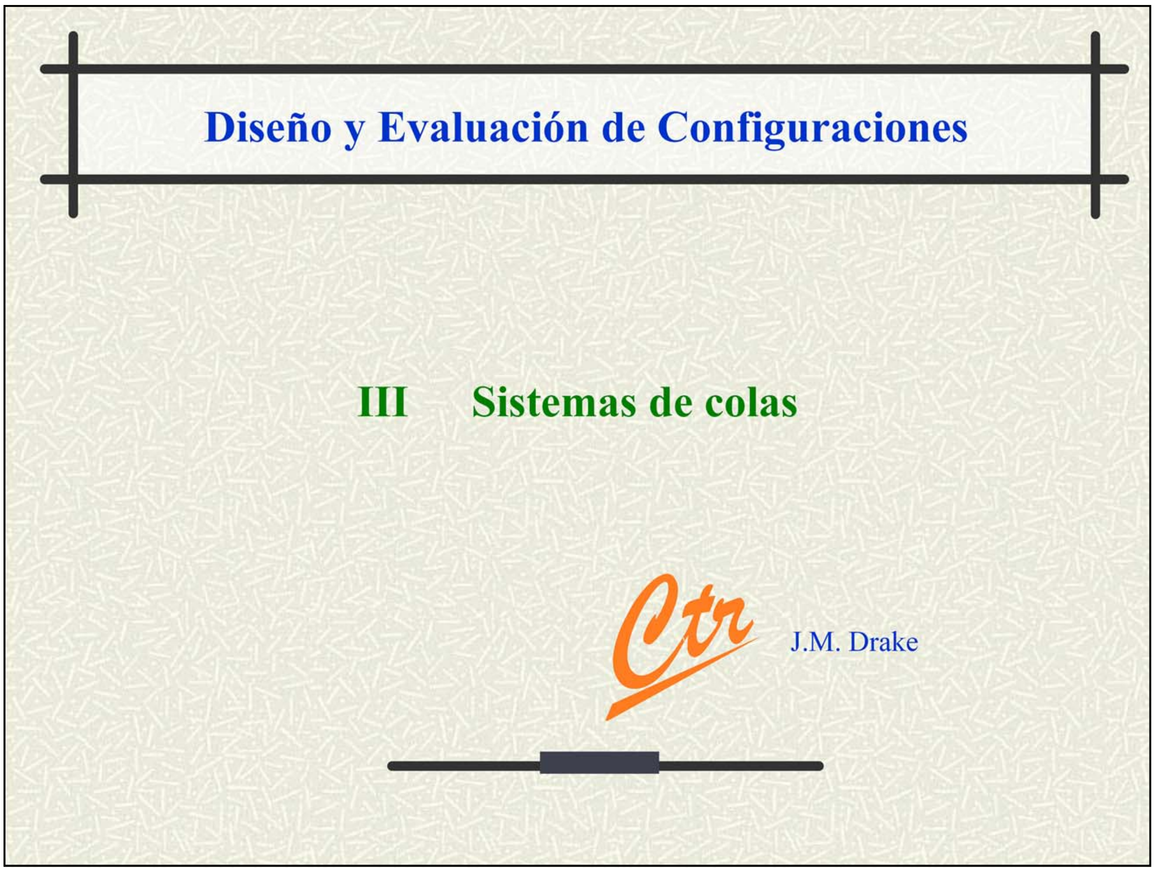
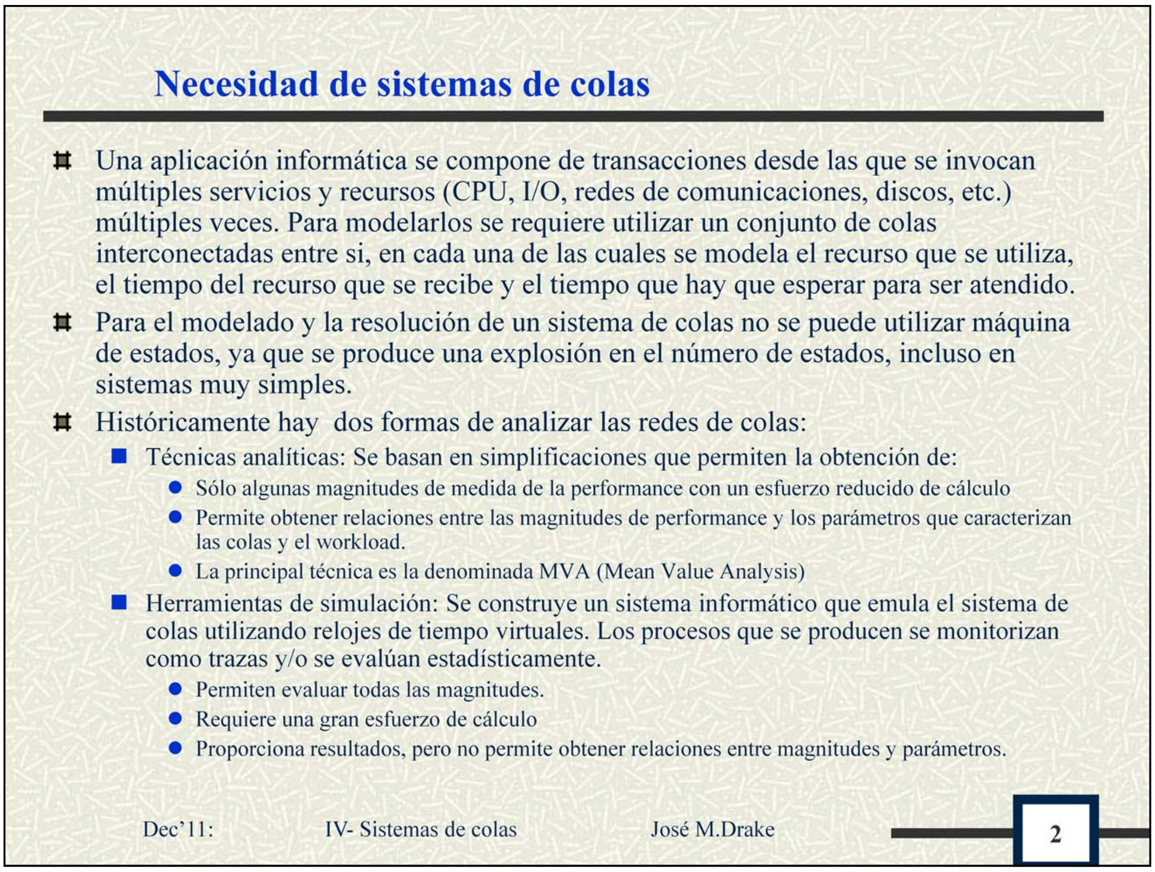


Diseño y Evaluación de Configuraciones

III Sistemas de colas

J.M. Drake

Notas:



Necesidad de sistemas de colas

- Una aplicación informática se compone de transacciones desde las que se invocan múltiples servicios y recursos (CPU, I/O, redes de comunicaciones, discos, etc.) múltiples veces. Para modelarlos se requiere utilizar un conjunto de colas interconectadas entre si, en cada una de las cuales se modela el recurso que se utiliza, el tiempo del recurso que se recibe y el tiempo que hay que esperar para ser atendido.
- Para el modelado y la resolución de un sistema de colas no se puede utilizar máquina de estados, ya que se produce una explosión en el número de estados, incluso en sistemas muy simples.
- Históricamente hay dos formas de analizar las redes de colas:
  - Técnicas analíticas: Se basan en simplificaciones que permiten la obtención de:
    - Sólo algunas magnitudes de medida de la performance con un esfuerzo reducido de cálculo
    - Permite obtener relaciones entre las magnitudes de performance y los parámetros que caracterizan las colas y el workload.
    - La principal técnica es la denominada MVA (Mean Value Analysis)
  - Herramientas de simulación: Se construye un sistema informático que emula el sistema de colas utilizando relojes de tiempo virtuales. Los procesos que se producen se monitorizan como trazas y/o se evalúan estadísticamente.
    - Permiten evaluar todas las magnitudes.
    - Requiere una gran esfuerzo de cálculo
    - Proporciona resultados, pero no permite obtener relaciones entre magnitudes y parámetros.

Notas:

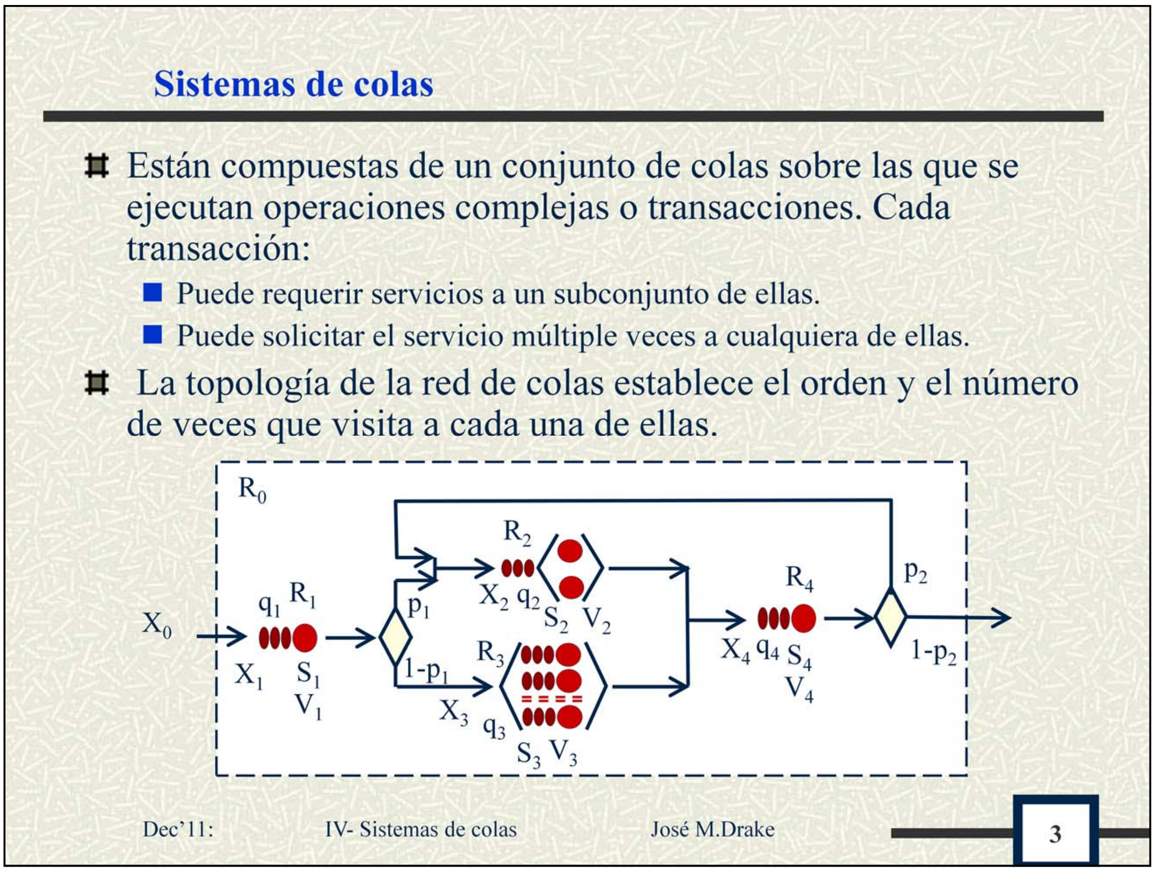
Hay un gran número de sistema que requieren modelar diferentes colas. Una transaccion puede requerir servicios de uno o mas recursos o servicios

En el caso general se necesitan modos de analizar sistemas de colas a fin de evaluar el comportamiento de sistemas informáticos compuestos por múltiples recursos (CPU, memoria, unidades I/O , etc.). En la nomenclatura de los modelos de colas, esto corresponde a decir que dentro de una transacción se ha de esperar a múltiples colas, o incluso realizar dentro de la misma transacción diferentes requerimientos a un mismo recurso. Por ejemplo, la transacción puede requerir ejecutar deferentes secciones de código en la CPU, realizar diferentes lecturas y escrituras sobre el disco, y transmitir y recibir diferentes datos por la unidad de I/O.

La resolución de un circuito de colas no se puede basar en maquinas de estados, ya que el estado de un sistema de colas es el producto cartesiano de los estados de cada una de las colas, y ello introduce una complejidad inabordable. Hay dos soluciones:

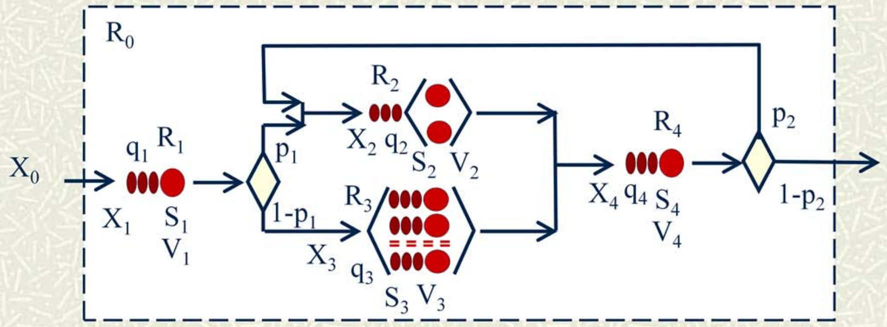
-Utilizar una herramienta de simulación: Con ella se pueden realizar resultados muy detallados y precisos, pero presenta el problema de requerir una gran capacidad de computo, y además se obtienen resultados sin saber a que características del modelo se deben.

-Utilizar métodos analíticos: Para obtener soluciones hay que realizar suposiciones simplificadoras que conduce a cálculos aproximados, pero a cambio de ello, los requerimientos de computación son bajos, y se obtienen relaciones entre los resultados y los parámetros de los modelos.



Sistemas de colas

- Están compuestas de un conjunto de colas sobre las que se ejecutan operaciones complejas o transacciones. Cada transacción:
  - Puede requerir servicios a un subconjunto de ellas.
  - Puede solicitar el servicio múltiple veces a cualquiera de ellas.
- La topología de la red de colas establece el orden y el número de veces que visita a cada una de ellas.



Dec'11: IV- Sistemas de colas José M.Drake 3

Notas:

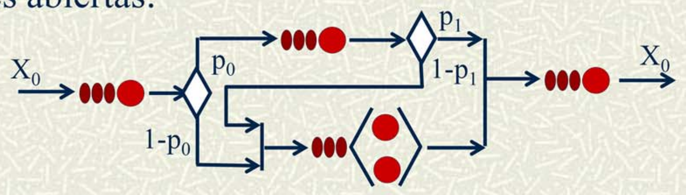
Un sistema de colas puede clasificarse como un conjunto de colas que modelan los recursos del sistema y un conjunto de arcos que describen las secuencias de requerimientos de una a otras. Las reglas que determinan el flujo de los eventos son probabilísticas y definen la probabilidad de que en una transacción, después de acceder a un recurso, aceda a uno u otro recurso.

Las magnitudes que se utilizan son:

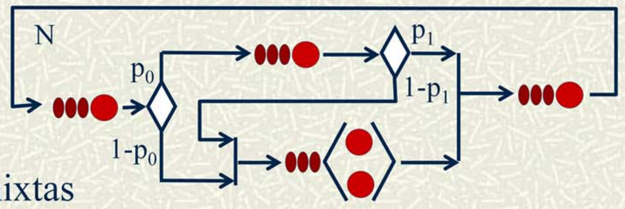
- X_i : flujo de requerimientos (throughput medidos como requerimientos/s) que se realizan en la cola i.
- S_i: Tiempo de servicio en el recurso de la cola i
- R_i: Tiempo de estancia en la cola i
- V_i: Tasa de visitas. Número de veces que dentro de una misma activación de la transacción se accede al recurso i
- qi: Numero medio de clientes en espera en la cola i.
- p_j: probabilidad de que un evento que llega un punto de bifurcación siga una rama.
- $D_i=V_iS_i$: Demanda del servicio i por la transacción.

Redes abiertas y cerradas

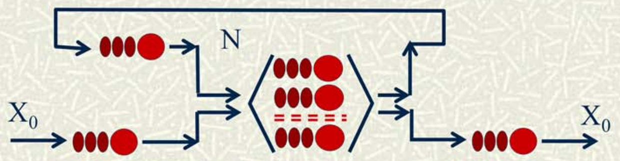
- Redes abiertas:



- Redes cerradas:



- Redes mixtas



Notas:

The simplest way to classify a queueing network is either open or closed. An **open queueing network** has external arrivals and departures, as shown in Figure (1). The jobs enter the system at "In" and exit at "Out." The number of jobs in the system varies with time. In analyzing an open system, we assume that the throughput is known (to be equal to the arrival rate), and the goal is to characterize the distribution of number of jobs in the system.

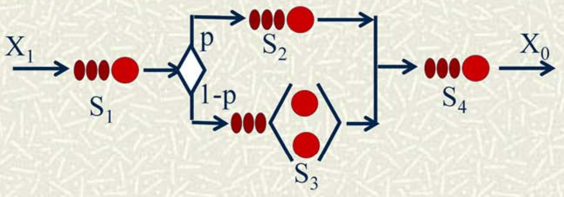
A **closed queueing network** has no external arrivals or departures. As shown in Figure 2, the jobs in the system keep circulating from one queue to the next. The total number of jobs in the system is constant. It is possible to view a closed system as a system where the Out is connected back to the In. The jobs exiting the system immediately reenter the system. The flow of jobs in the Out-to-In link defines the throughput of the closed system. In analyzing a closed system, we assume that the number of jobs is given, and we attempt to determine the throughput (or the job completion rate).

It is also possible to have **mixed queueing networks** that are open for some workloads and closed for others. Figure 3 shows an example of a mixed system with two *classes* of jobs. The system is closed for interactive jobs and is open for batch jobs. The term *class* refers to types of jobs. All jobs of a single class have the same service demands and transition probabilities. Within each class, the jobs are indistinguishable.

Redes con un workload o con múltiples workload

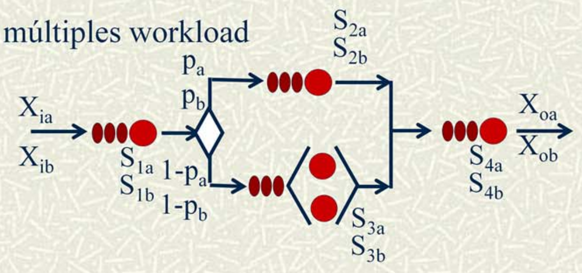
- Cada clase de workload define su propios flujos y los tiempos de servicios en cada cola:

Red con un workload



Clase C

Red con múltiples workload



Clase C_a

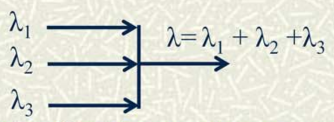
Clase C_b

Notas:

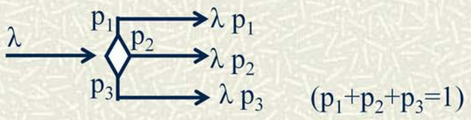
Sistemas con múltiples workload que se componen de varios tipos de clientes que fluyen con diferentes throughput, requieren de los mismos recursos servicios con tiempos diferentes, y que visitan las colas con diferentes tasas.

Propiedades de las redes de Poisson

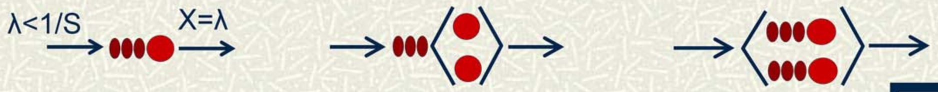
- Los intervalos entre requerimientos tienen una distribución de tipo exponencial.
- Propiedades:
  - Cuando múltiples flujos de Poisson convergen el flujo resultante es también de Poisson.



  - Cuando un flujo se diversifica de acuerdo con una ley probabilística los flujos que resultan son también de Poisson



  - Cuando un flujo atraviesa una cola M/M/1, M/M/m , m<M/M/1> el flujo que resulta es también de Poisson (Teorema de Burke)



Notas:

Poisson Processes: If the interarrival times are IID and exponentially distributed, the number of arrivals *n* over a given interval (*t*, *t* + *x*) has a Poisson distribution, and therefore, the arrival process is referred to as a **Poisson process** or a **Poisson stream.** The Poisson streams are popular in queueing

theory because the arrivals are then memoryless as the interarrival time is exponentially distributed. In addition, Poisson streams have the following properties:

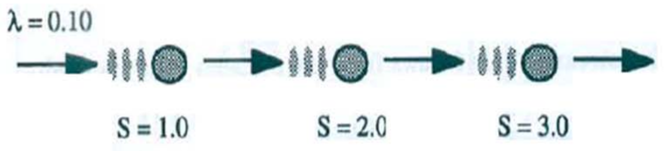
(a) Merging of *k* Poisson streams with mean rate λ_i results in a Poisson stream with mean rate $\lambda = \sum \lambda_i$

(b) If a Poisson Stream is split into *k* substreams such that the probability of a job going to the *i*th substream is p_i, each substream is also Poisson with a mean rate of $p_i\lambda$.

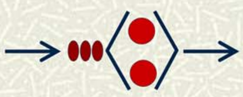
(c) If the arrivals to a single server with exponential service time are Poisson with mean rate λ, the departures are also Poisson with the same rate λ,, provided the arrival rate λ is less than the service rate μ.

(d) If the arrivals to a service facility with *m* service centers are Poisson with a mean rate λ, the departures also constitute a Poisson stream with the same rate λ, provided the arrival rate λ is less than the total service rate Here, the servers are assumed to have exponentially distributed service times.

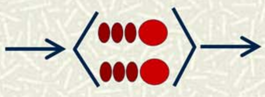
Ejemplo de red abierta y sin realimentación (feedforward)



	Metric	Center 1	Center 2	Center 3	Total
λ=X	Arrival rate	0.10	0.10	0.10	0.10
S	Service time	1.00	2.00	3.00	
U=XS	Utilization	0.10	0.20	0.30	
R=S/(1-U)	Residence time	1.11	2.50	4.29	7.90
Q=U/(1-U)	Queue length	0.11	0.25	0.43	0.79
Q=XR	Little's law				0.79



$$R \approx \frac{S}{1-\rho^m} \quad Q \approx \frac{m\rho}{1-\rho^m} \left(siendo\ \rho = \frac{U}{m} \right)$$



$$R \approx \frac{S}{1-\rho} \quad Q_t \approx \frac{XS}{1-\rho} \quad Q_c = \frac{XS}{m(1-\rho)}$$

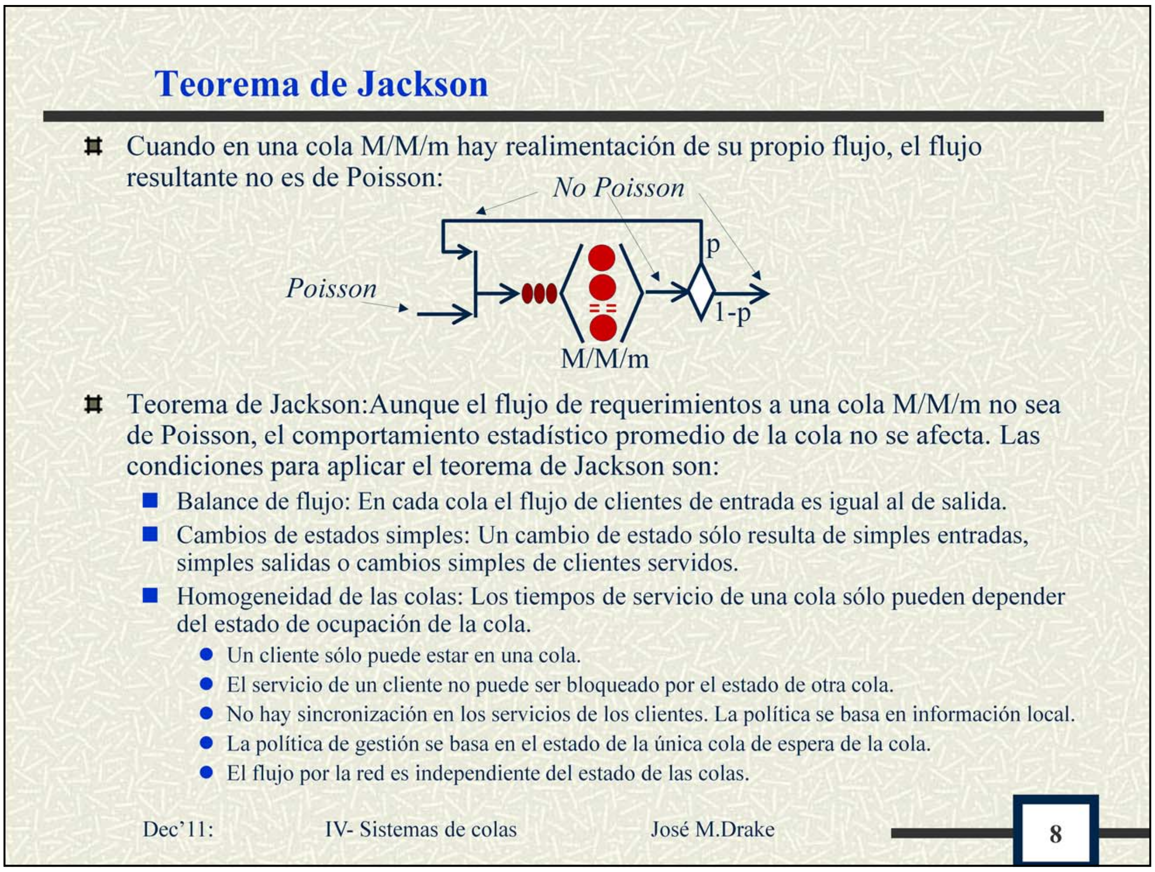
Notas:

Considerese un circuito simple compuesto por tres colas M/M/1 en seri con tiempos de servicio S1=1.0 s, S2=2.0 s y S3=3.0 s.y que recibe requerimientos de una transacción que se ejecuta con una frecuencia de λ=0.1 invocación/s. Debido a las propiedades de las distribuciones de Poisson, se puede tratar como tres colas independientes, todas ellas (en este caso) solicitadas con el mismo throughput λ.

En la tabla se calculan la utilización, el tiempo de estancia, y la longitud de la espera en cada cola.

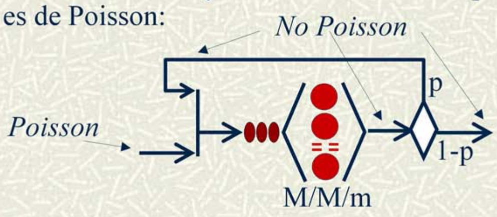
En la última línea se comprueba que la fórmula de Little se puede aplicar globalmente al sistema como si fuese una cola.

Si las colas fuesen de múltiples servidores, o fuesen múltiples colas, habría que utilizar las correspondientes ecuaciones para calcular el tiempo de estancia R y el número medio de elementos en las colas.



Teorema de Jackson

- Cuando en una cola M/M/m hay realimentación de su propio flujo, el flujo resultante no es de Poisson:



- Teorema de Jackson:Aunque el flujo de requerimientos a una cola M/M/m no sea de Poisson, el comportamiento estadístico promedio de la cola no se afecta. Las condiciones para aplicar el teorema de Jackson son:
  - Balance de flujo: En cada cola el flujo de clientes de entrada es igual al de salida.
  - Cambios de estados simples: Un cambio de estado sólo resulta de simples entradas, simples salidas o cambios simples de clientes servidos.
  - Homogeneidad de las colas: Los tiempos de servicio de una cola sólo pueden depender del estado de ocupación de la cola.
    - Un cliente sólo puede estar en una cola.
    - El servicio de un cliente no puede ser bloqueado por el estado de otra cola.
    - No hay sincronización en los servicios de los clientes. La política se basa en información local.
    - La política de gestión se basa en el estado de la única cola de espera de la cola.
    - El flujo por la red es independiente del estado de las colas.

Dec'11: IV- Sistemas de colas José M.Drake 8

Notas:

Jackson (1963) showed that the above method of computing joint probability is valid for any arbitrary open network of *m*-server queues with exponentially distributed service times (see the Figure). In particular, if all queues are single-server queues, the queue length distribution is exponencial. However, it is not correct to assume that each queue becomes an independent M/M/1 queue with a Poisson arrival process. In general, the internal flow in such networks is not Poisson. Particularly, if there is any feedback in the network, so that jobs can return to previously visited service centers, the internal flows are not Poisson. It is surprising that even though the flows are not Poisson, the queues are separable and can be analyzed as if they were Jackson's results were later extended to closed networks by Gordon and Newell (1967). They showed that any arbitrary closed networks of *m*-server queues with exponentially distributed service times also have a product form solution. showed that product form solutions exist for an even broader class of networks. This class consists of networks satisfying the following criteria:

1. *Job Flow Balance*: For each class, the number of arrivals to a device must equal the number of departures from the device.

2. *One-Step Behavior*: A state change can result only from single jobs entering the system, moving between pairs of devices in the system, or exiting from the system. This assumption asserts that

simultaneous job moves will not be observed.

3. *Device Homogeneity*: A device's service rate for a particular class does not depend on the state of the system in any way except for the total device queue length and the designated class's queue length. This assumption implies the following:

(a) *Single-Resource Possession*: A job may not be present (waiting for service or receiving service) at two or more devices at the same time.

(b) *No Blocking*: A device renders service whenever jobs are present; its ability to render service is not controlled by any other device.

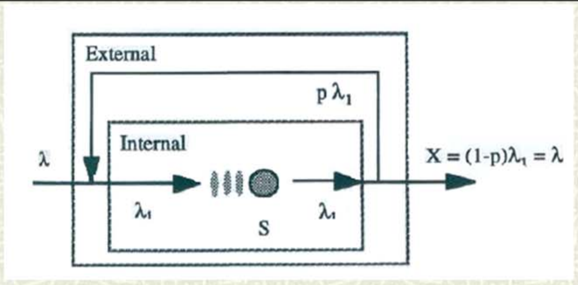
(c) *Independent Job Behavior*: Interaction among jobs is limited to queueing for physical devices; for example, there should not be any synchronization requirements.

(d) *Local Information*: A device's service rate depends only on local queue length and not on the state of the rest of the system.

(e) *Fair Service*: If service rates differ by class, the service rate for a class depends only on the queue length of that class at the device and not on the queue lengths of other classes. This means that the servers do not discriminate against jobs in a class depending on the queue lengths of other classes.

(f) *Routing Homogeneity*: The job routing should be state independent. In the last condition, the term *routing* is used to denote a job's path in the network. The routing homogeneity condition implies that the probability of a job going from one device to another device does not depend upon the number of jobs at various devices.

Consecuencias del teorema de Jackson



$$\lambda_1 = \lambda + p\lambda_1 \qquad \Rightarrow \lambda_1 = \frac{\lambda}{1-p} = V\lambda \qquad siendo \left| \begin{array}{l} V = 1/(1-p) \quad \textit{Tasa media de visita} \\ D = VS = \textit{Demada del servicio} \end{array} \right.$$

$$\left.\begin{array}{l} \textit{El tiempo de respuesta por transacción}: R = \dfrac{D}{1-\lambda D} = \dfrac{VS}{1-\lambda VS} \\ \textit{El tiempo de respuesta por visita}: R_v = \dfrac{S}{1-\lambda_1 S} \end{array}\right| \Rightarrow \begin{array}{l} R = VR_v \\ R_v = \dfrac{S}{1-\lambda VS} \end{array}$$

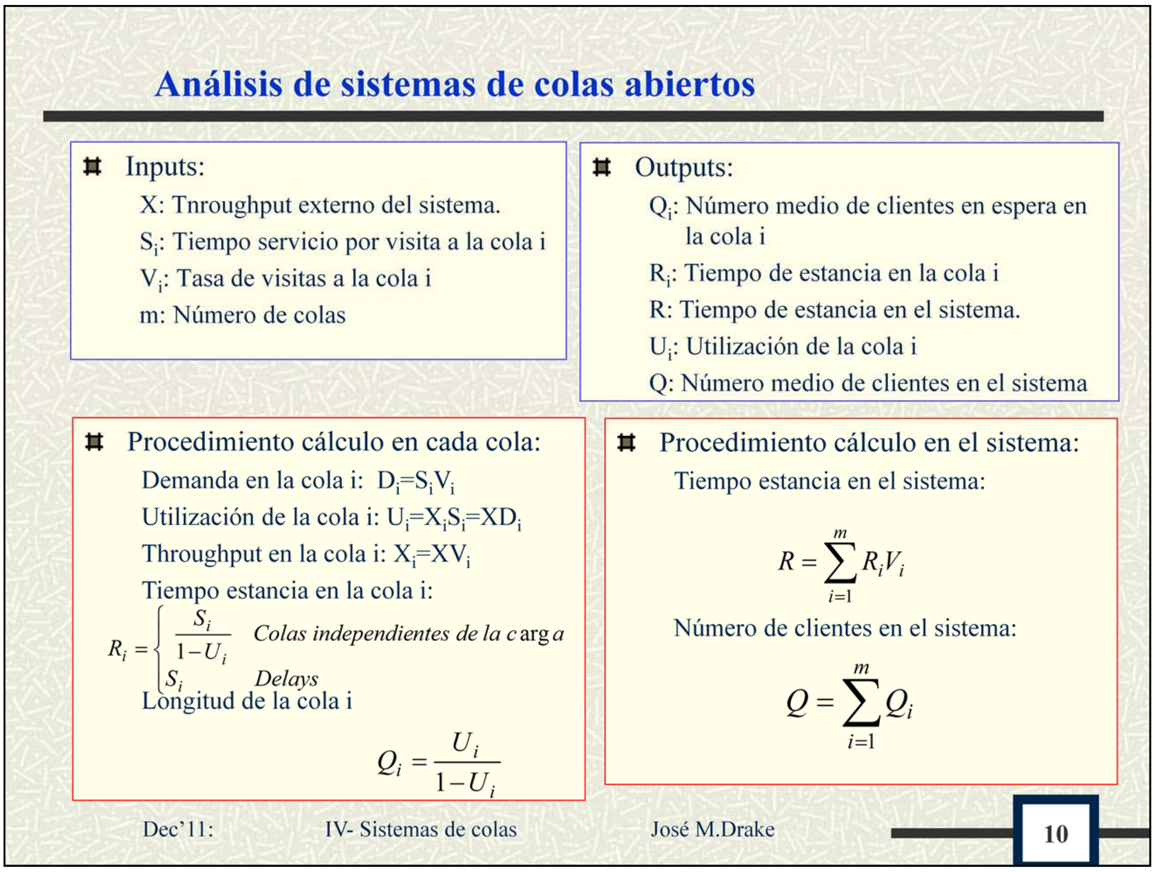
Notas:

En la red simple de la figura, los clientes llegan a una frecuencia λ de Poisson. Dado que a la salida hay una bifurcación que hace retornar a la entrada de la cola, se verifica $\lambda_1 = \lambda + p\,\lambda_1$. Despejando, resulta la tasa de visita V =1/(1-p) y la demanda de servicio D=VS.

La utilización del recurso es $U = \lambda_1 S$.

Aplicando la ecuación de una cola M/M/1, $R_v = S/(1-\lambda_1 S)$

El tiempo de respuesta global del sistema es $R = VR_v = VS/(1- V\lambda S) = D/(1-\lambda D)$



Análisis de sistemas de colas abiertos

- Inputs:
  - X: Tnroughput externo del sistema.
  - S_i: Tiempo servicio por visita a la cola i
  - V_i: Tasa de visitas a la cola i
  - m: Número de colas

- Outputs:
  - Q_i: Número medio de clientes en espera en la cola i
  - R_i: Tiempo de estancia en la cola i
  - R: Tiempo de estancia en el sistema.
  - U_i: Utilización de la cola i
  - Q: Número medio de clientes en el sistema

- Procedimiento cálculo en cada cola:
  - Demanda en la cola i: $D_i=S_iV_i$
  - Utilización de la cola i: $U_i=X_iS_i=XD_i$
  - Throughput en la cola i: $X_i=XV_i$
  - Tiempo estancia en la cola i:

$$R_i = \begin{cases} \dfrac{S_i}{1-U_i} & \textit{Colas independientes de la carga} \\ S_i & \textit{Delays} \end{cases}$$

  - Longitud de la cola i

$$Q_i = \frac{U_i}{1-U_i}$$

- Procedimiento cálculo en el sistema:
  - Tiempo estancia en el sistema:

$$R = \sum_{i=1}^{m} R_i V_i$$

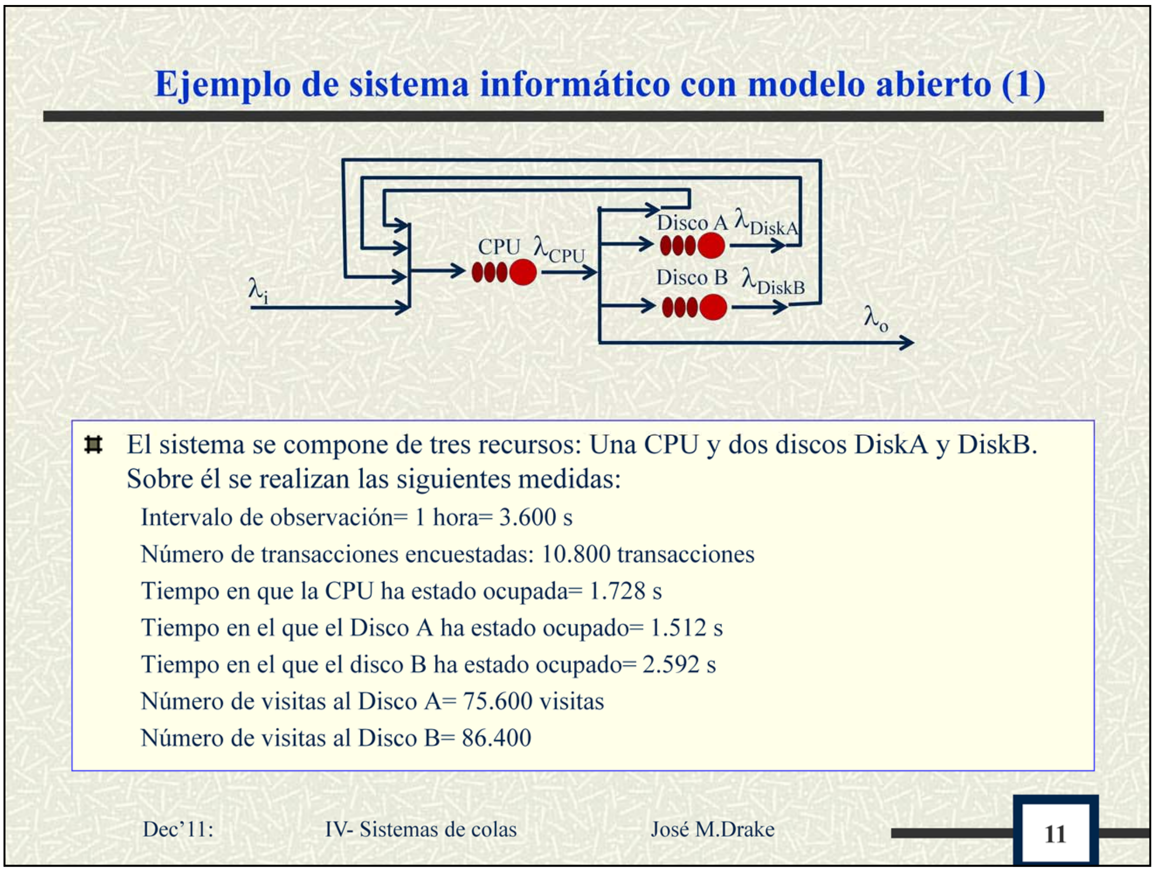
  - Número de clientes en el sistema:

$$Q = \sum_{i=1}^{m} Q_i$$

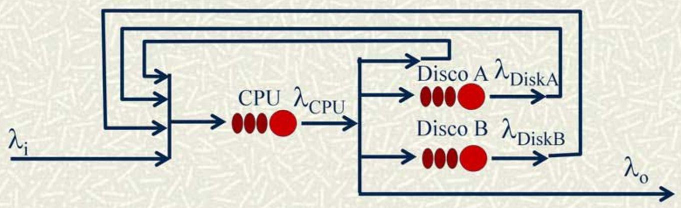
Dec'11: IV- Sistemas de colas José M.Drake 10

Notas:

Open queueing network models are used to represent transaction processing systems such as airline reservation systems or banking systems. In these systems, the transaction arrival rate is not dependent on the load on the computer system. The transaction arrivals are modeled as a Poisson process with a mean arrival rate ». In this section, we present equations for an exact analysis of such systems. We assume that all devices in the computer system can be modeled as either fixed-capacity service centers (single server with exponentially distributed service time) or delay centers (infinite servers with exponentially distributed service time).



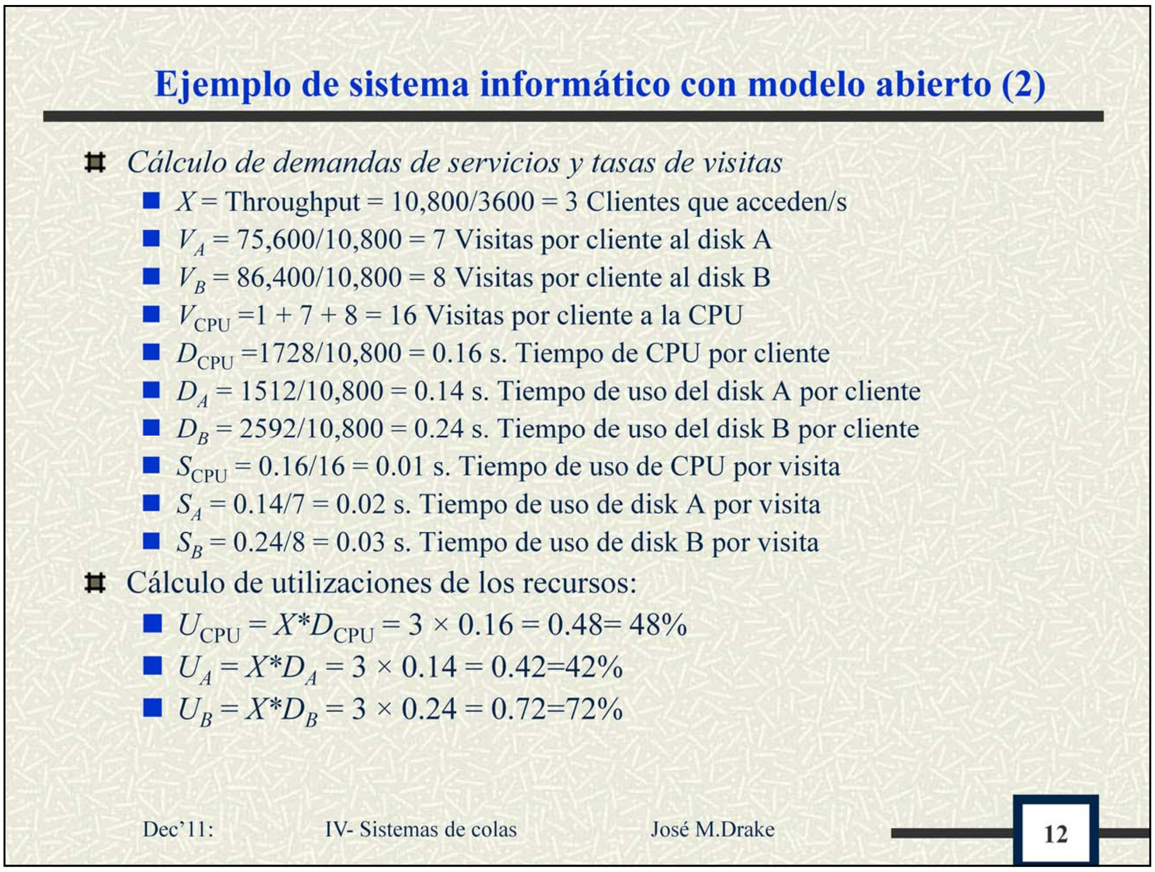
Ejemplo de sistema informático con modelo abierto (1)



- El sistema se compone de tres recursos: Una CPU y dos discos DiskA y DiskB. Sobre él se realizan las siguientes medidas:
  - Intervalo de observación= 1 hora= 3.600 s
  - Número de transacciones encuestadas: 10.800 transacciones
  - Tiempo en que la CPU ha estado ocupada= 1.728 s
  - Tiempo en el que el Disco A ha estado ocupado= 1.512 s
  - Tiempo en el que el disco B ha estado ocupado= 2.592 s
  - Número de visitas al Disco A= 75.600 visitas
  - Número de visitas al Disco B= 86.400

Notas:

The figure shown represents a queueing model of a file server consisting of a CPU and two disks, A and B. Measurements on a distributed system with six clients systems making requests to the file server produced.

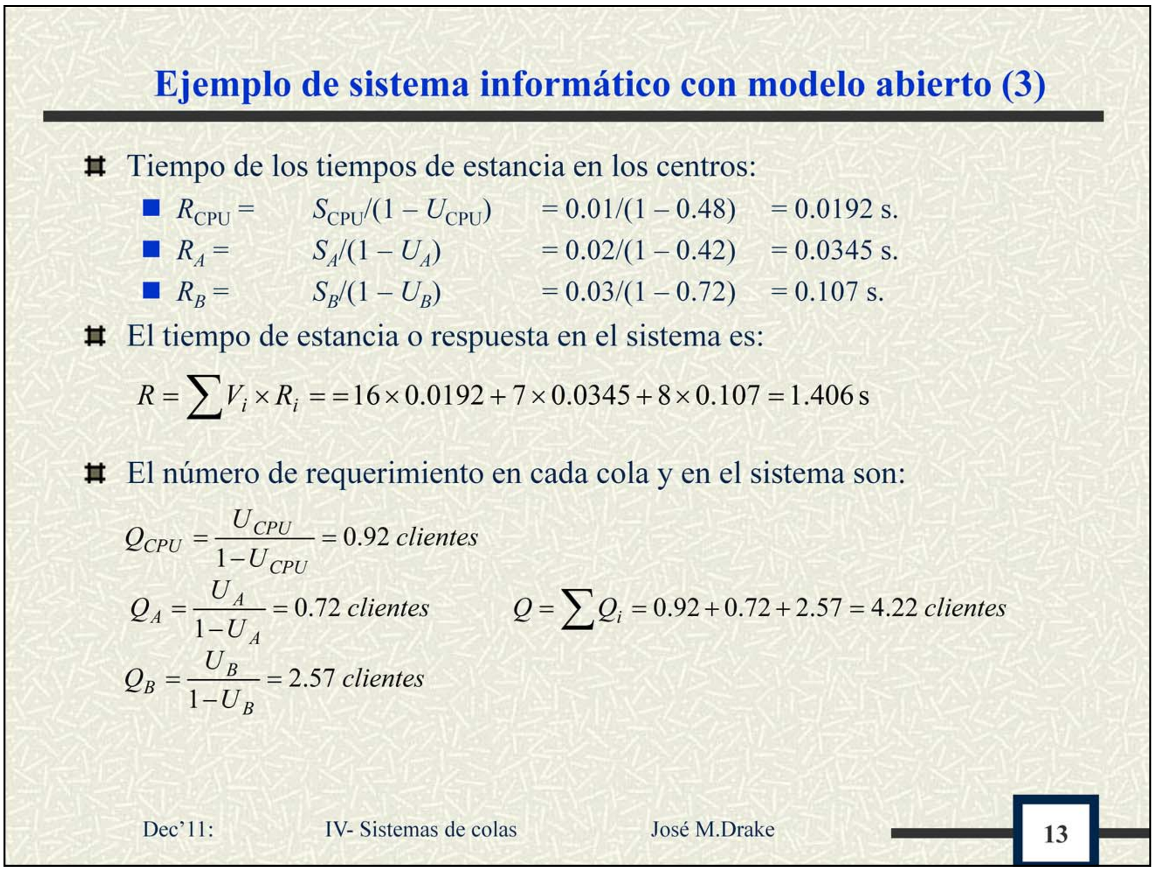


Ejemplo de sistema informático con modelo abierto (2)

- *Cálculo de demandas de servicios y tasas de visitas*
  - X = Throughput = 10,800/3600 = 3 Clientes que acceden/s
  - V_A = 75,600/10,800 = 7 Visitas por cliente al disk A
  - V_B = 86,400/10,800 = 8 Visitas por cliente al disk B
  - V_{CPU} =1 + 7 + 8 = 16 Visitas por cliente a la CPU
  - D_{CPU} =1728/10,800 = 0.16 s. Tiempo de CPU por cliente
  - D_A = 1512/10,800 = 0.14 s. Tiempo de uso del disk A por cliente
  - D_B = 2592/10,800 = 0.24 s. Tiempo de uso del disk B por cliente
  - S_{CPU} = 0.16/16 = 0.01 s. Tiempo de uso de CPU por visita
  - S_A = 0.14/7 = 0.02 s. Tiempo de uso de disk A por visita
  - S_B = 0.24/8 = 0.03 s. Tiempo de uso de disk B por visita
- Cálculo de utilizaciones de los recursos:
  - $U_{\mathrm{CPU}} = X{*}D_{\mathrm{CPU}} = 3 \times 0.16 = 0.48$= 48%
  - $U_A = X{*}D_A = 3 \times 0.14 = 0.42$=42%
  - $U_B = X{*}D_B = 3 \times 0.24 = 0.72$=72%

Dec'11: IV- Sistemas de colas José M.Drake 12

Notas:



Ejemplo de sistema informático con modelo abierto (3)

- Tiempo de los tiempos de estancia en los centros:
  - $R_{\mathrm{CPU}} = S_{\mathrm{CPU}}/(1 - U_{\mathrm{CPU}}) = 0.01/(1 - 0.48) = 0.0192$ s.
  - $R_A = S_A/(1 - U_A) = 0.02/(1 - 0.42) = 0.0345$ s.
  - $R_B = S_B/(1 - U_B) = 0.03/(1 - 0.72) = 0.107$ s.
- El tiempo de estancia o respuesta en el sistema es:

$$R = \sum V_i \times R_i = = 16 \times 0.0192 + 7 \times 0.0345 + 8 \times 0.107 = 1.406\,\mathrm{s}$$

- El número de requerimiento en cada cola y en el sistema son:

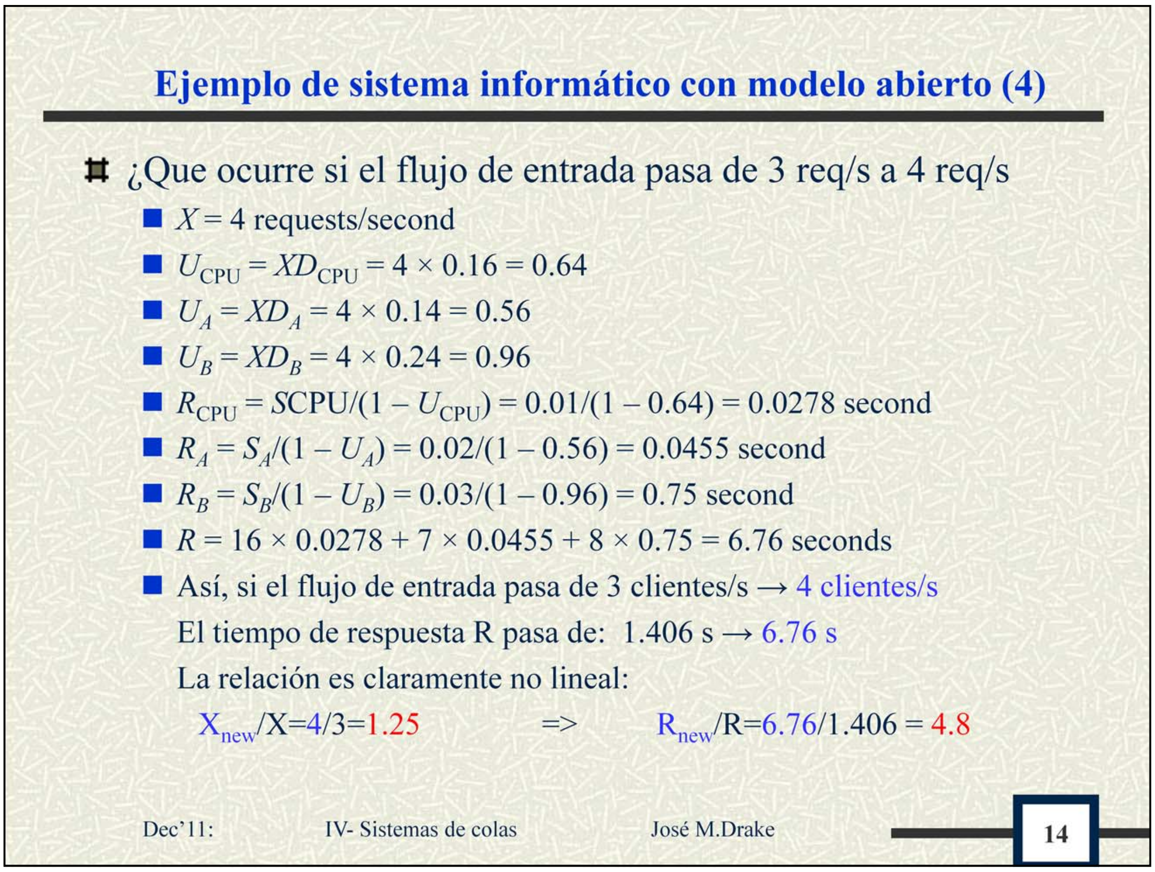
$$Q_{CPU} = \frac{U_{CPU}}{1 - U_{CPU}} = 0.92\ clientes$$

$$Q_A = \frac{U_A}{1 - U_A} = 0.72\ clientes \qquad Q = \sum Q_i = 0.92 + 0.72 + 2.57 = 4.22\ clientes$$

$$Q_B = \frac{U_B}{1 - U_B} = 2.57\ clientes$$

Dec'11: IV- Sistemas de colas José M.Drake 13

Notas:



Ejemplo de sistema informático con modelo abierto (4)

- ¿Que ocurre si el flujo de entrada pasa de 3 req/s a 4 req/s
  - $X = 4$ requests/second
  - $U_{\mathrm{CPU}} = XD_{\mathrm{CPU}} = 4 \times 0.16 = 0.64$
  - $U_A = XD_A = 4 \times 0.14 = 0.56$
  - $U_B = XD_B = 4 \times 0.24 = 0.96$
  - $R_{\mathrm{CPU}} = S\mathrm{CPU}/(1 - U_{\mathrm{CPU}}) = 0.01/(1 - 0.64) = 0.0278$ second
  - $R_A = S_A/(1 - U_A) = 0.02/(1 - 0.56) = 0.0455$ second
  - $R_B = S_B/(1 - U_B) = 0.03/(1 - 0.96) = 0.75$ second
  - $R = 16 \times 0.0278 + 7 \times 0.0455 + 8 \times 0.75 = 6.76$ seconds
  - Así, si el flujo de entrada pasa de 3 clientes/s → 4 clientes/s

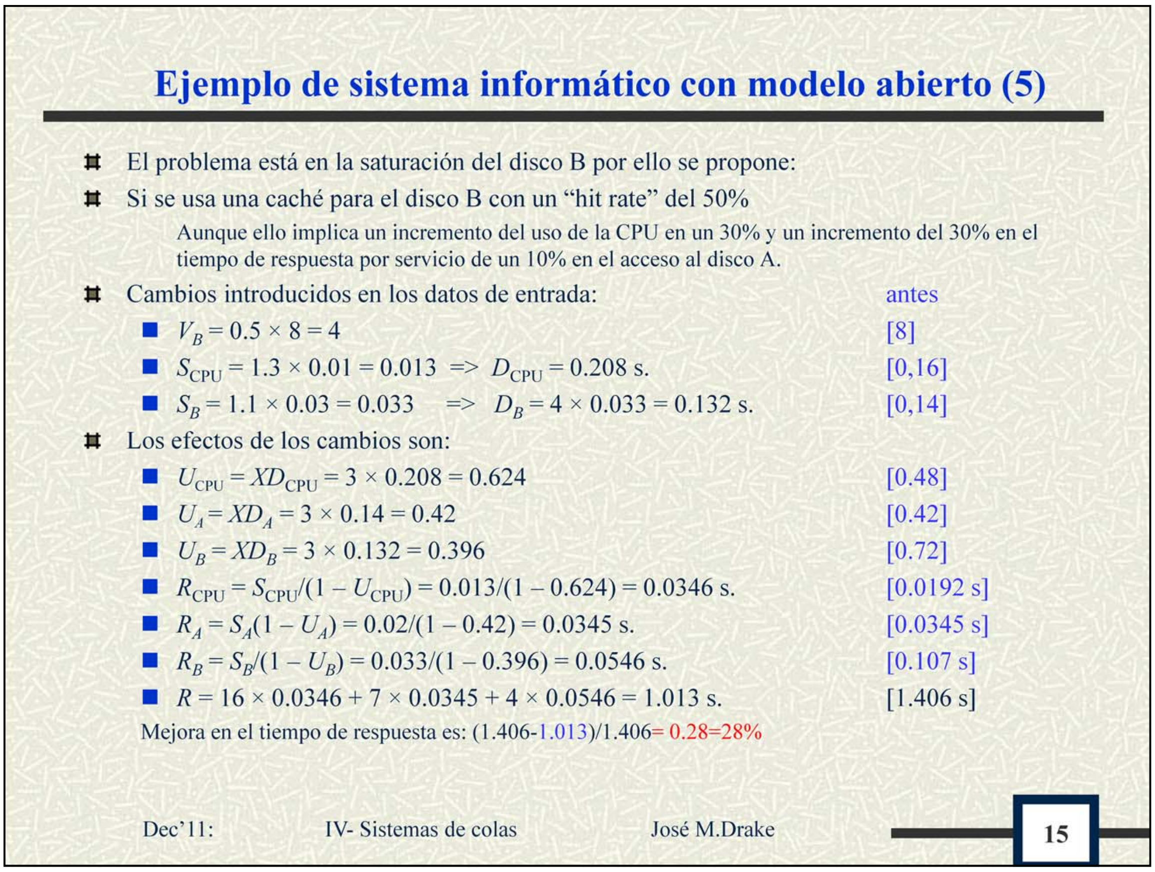
    El tiempo de respuesta R pasa de: 1.406 s → 6.76 s

    La relación es claramente no lineal:

    $X_{new}/X=4/3=1.25$ => $R_{new}/R=6.76/1.406 = 4.8$

Dec'11: IV- Sistemas de colas José M.Drake 14

Notas:



Ejemplo de sistema informático con modelo abierto (5)

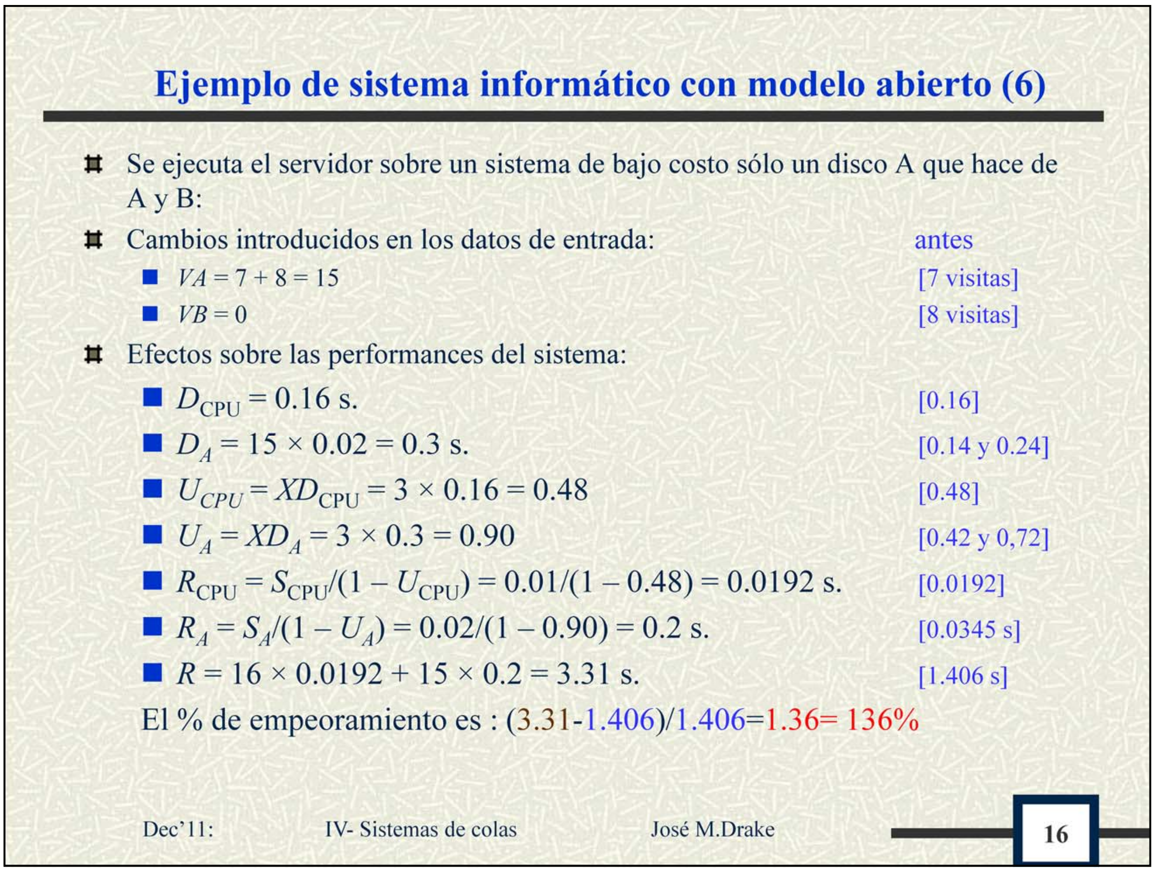
- El problema está en la saturación del disco B por ello se propone:
- Si se usa una caché para el disco B con un "hit rate" del 50%

  Aunque ello implica un incremento del uso de la CPU en un 30% y un incremento del 30% en el tiempo de respuesta por servicio de un 10% en el acceso al disco A.

	antes
Cambios introducidos en los datos de entrada:	
■ $V_B = 0.5 \times 8 = 4$	[8]
■ $S_{\text{CPU}} = 1.3 \times 0.01 = 0.013$ => $D_{\text{CPU}} = 0.208$ s.	[0,16]
■ $S_B = 1.1 \times 0.03 = 0.033$ => $D_B = 4 \times 0.033 = 0.132$ s.	[0,14]
Los efectos de los cambios son:	
■ $U_{\text{CPU}} = XD_{\text{CPU}} = 3 \times 0.208 = 0.624$	[0.48]
■ $U_A = XD_A = 3 \times 0.14 = 0.42$	[0.42]
■ $U_B = XD_B = 3 \times 0.132 = 0.396$	[0.72]
■ $R_{\text{CPU}} = S_{\text{CPU}}/(1 - U_{\text{CPU}}) = 0.013/(1 - 0.624) = 0.0346$ s.	[0.0192 s]
■ $R_A = S_A(1 - U_A) = 0.02/(1 - 0.42) = 0.0345$ s.	[0.0345 s]
■ $R_B = S_B/(1 - U_B) = 0.033/(1 - 0.396) = 0.0546$ s.	[0.107 s]
■ $R = 16 \times 0.0346 + 7 \times 0.0345 + 4 \times 0.0546 = 1.013$ s.	[1.406 s]

Mejora en el tiempo de respuesta es: (1.406-1.013)/1.406= 0.28=28%

Notas:



Ejemplo de sistema informático con modelo abierto (6)

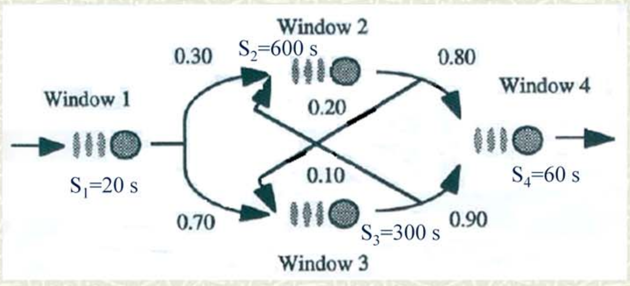
- Se ejecuta el servidor sobre un sistema de bajo costo sólo un disco A que hace de A y B:
- Cambios introducidos en los datos de entrada: antes
  - $VA = 7 + 8 = 15$ [7 visitas]
  - $VB = 0$ [8 visitas]
- Efectos sobre las performances del sistema:
  - $D_{\text{CPU}} = 0.16$ s. [0.16]
  - $D_A = 15 \times 0.02 = 0.3$ s. [0.14 y 0.24]
  - $U_{CPU} = XD_{\text{CPU}} = 3 \times 0.16 = 0.48$ [0.48]
  - $U_A = XD_A = 3 \times 0.3 = 0.90$ [0.42 y 0,72]
  - $R_{\text{CPU}} = S_{\text{CPU}}/(1 - U_{\text{CPU}}) = 0.01/(1 - 0.48) = 0.0192$ s. [0.0192]
  - $R_A = S_A/(1 - U_A) = 0.02/(1 - 0.90) = 0.2$ s. [0.0345 s]
  - $R = 16 \times 0.0192 + 15 \times 0.2 = 3.31$ s. [1.406 s]

El % de empeoramiento es : (3.31-1.406)/1.406=1.36= 136%

Notas:

Ejemplo de resolución del un circuito con realimentación

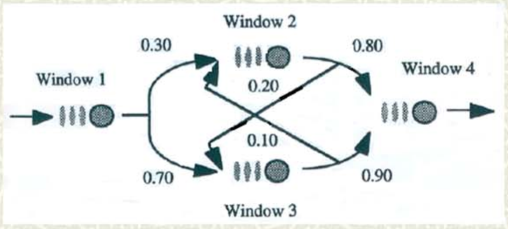
- Considérese el proceso de obtención del pasaporte en una comisaría de policía.
  - En la ventanilla 1 todos chequean sus documentos.
  - En la ventanillas 2 se presenta el certificado de nacimiento. Si no tienes la fotografía adecuada vas a la ventanilla 3
  - Desde la ventanilla 3, o bien se retorna a la ventanilla 2 para presentar la partida de nacimiento, o se va a la ventanilla 4 para finalizar el proceso.



$$\lambda_1 = \lambda$$
$$\lambda_2 = 0.3\lambda + 0.1\lambda_3 \Rightarrow \lambda_2 = 0.38\lambda$$
$$\lambda_3 = 0.7\lambda + 0.2\lambda_2 \Rightarrow \lambda_3 = 0.78\lambda$$
$$\lambda_4 = \lambda$$

Notas:

Ejemplo de resolución del un circuito con realimentación (2)



$$U_1 = 20\lambda_1 = 20\lambda$$
$$U_2 = 600\lambda_2 = 226.8\lambda$$
$$U_3 = 300\lambda_3 = 232.8\lambda$$
$$U_4 = 60\lambda_4 = 60\lambda$$

$\Rightarrow$

$$Q_1 = \frac{U_1}{1-U_1} = \frac{20\lambda}{1-20\lambda}$$
$$Q_2 = \frac{U_2}{1-U_2} = \frac{226.8\lambda}{1-226.8\lambda}$$
$$Q_3 = \frac{U_3}{1-U_3} = \frac{232.8\lambda}{1-232.8\lambda}$$
$$Q_4 = \frac{U_4}{1-U_4} = \frac{60\lambda}{1-60\lambda}$$

El tiempo para obtener un pasaporte, se puede obtener aplicando la formula de Little:

$$R = \frac{Q_1+Q_2+Q_3+Q_4}{\lambda}$$

Si el workload es $\lambda = 15.36\ clientes/h = 15.36/3600 = 0.0043\ clientes/s$

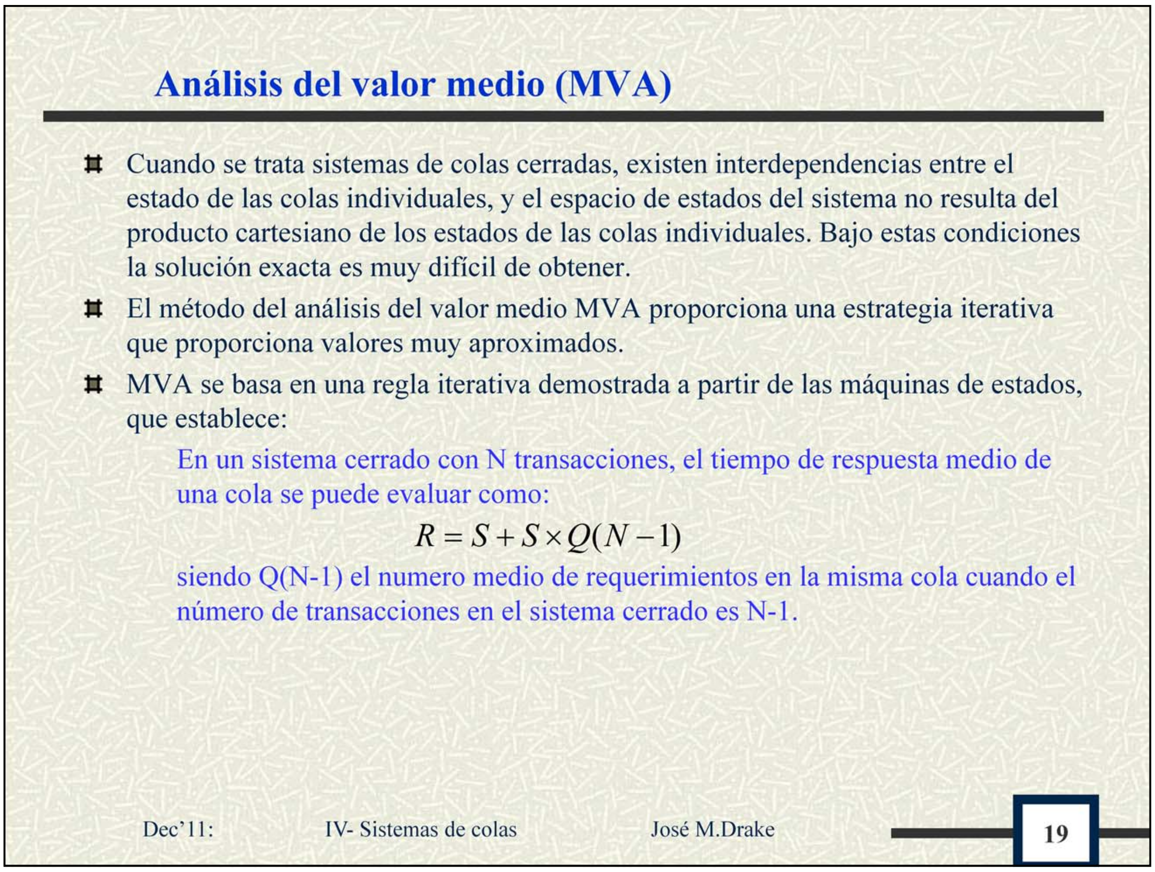
$U_1 = 8.53\%$ $\quad Q_1 = 0.0933$

$U_2 = 96.77\%$ $\quad Q_2 = 29.94$

$U_3 = 99.33\%$ $\quad Q_3 = 147.81$

$U_4 = 25.6\%$ $\quad Q_4 = 0.344 \quad \Rightarrow \quad R = 41762.72\ s = 11.60\ horas$

Notas:



Análisis del valor medio (MVA)

- Cuando se trata sistemas de colas cerradas, existen interdependencias entre el estado de las colas individuales, y el espacio de estados del sistema no resulta del producto cartesiano de los estados de las colas individuales. Bajo estas condiciones la solución exacta es muy difícil de obtener.
- El método del análisis del valor medio MVA proporciona una estrategia iterativa que proporciona valores muy aproximados.
- MVA se basa en una regla iterativa demostrada a partir de las máquinas de estados, que establece:

  En un sistema cerrado con N transacciones, el tiempo de respuesta medio de una cola se puede evaluar como:

  $$R = S + S \times Q(N-1)$$

  siendo Q(N-1) el numero medio de requerimientos en la misma cola cuando el número de transacciones en el sistema cerrado es N-1.

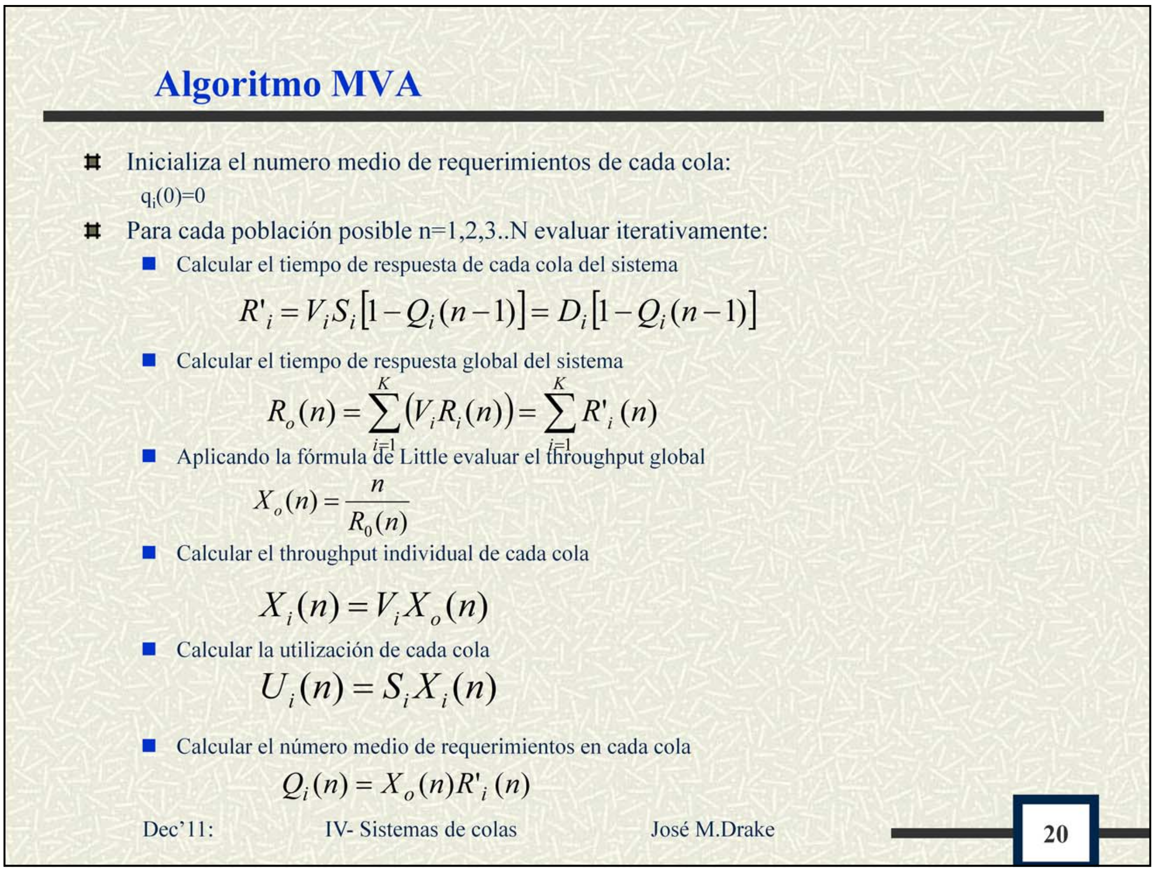
Notas:

Mean-Value Analysis (MVA) allows solving closed queueing networks in a manner similar to that used for open queueing networks. As the name implies, it gives the mean performance. The variance computation is not possible using this technique. Mean-value analysis applies to networks with a variety of service disciplines and service time distributions. However, we will initially limit our discussion here to fixed-capacity service centers. Delay centers are considered later. Load-dependent service centers are considered also later. Given a closed queueing network with *N* jobs, Reiser and Lavenberg (1980) showed that the response time of the ith device is given by

$$R_i(N) = S_i[1 + Q_i(N-1)]$$

Here, $Q_i(N-1)$ is the mean queue length at the ith device with $N-1$ jobs in the network. Notice the similarity between this equation and la equación habitual en colas simples $R_i(N) = S_i[1 + Q_i]$ for open queueing networks. As before, one way to explain this equation is to consider what happens when we add a tagged *N*th job to the network with $N-1$ jobs. On arrival at the *i*th service center, the tagged job sees $Q_i\ (N-1)$ jobs ahead (including the one in service) and expects to wait $Q_i(N-1)S_i$ seconds before receiving service. Including the service time for itself, the job should expect a total response time of $S_i[1 + Q_i\ (N-1)]$. This is not an operational law. It assumes that the service is memoryless, an assumption that is not operationally testable.

Given the performance for $N-1$ users, is sufficient to allow us to compute the performance for *N* users. Since the performance with no users ($N = 0$) can be easily computed, performance for any number of users can be computed iteratively, as we show next.



Algoritmo MVA

- Inicializa el numero medio de requerimientos de cada cola:

  $q_i(0)=0$
- Para cada población posible n=1,2,3..N evaluar iterativamente:
  - Calcular el tiempo de respuesta de cada cola del sistema

  $$R'_i = V_i S_i \left[1 - Q_i(n-1)\right] = D_i\left[1 - Q_i(n-1)\right]$$
  - Calcular el tiempo de respuesta global del sistema

  $$R_o(n) = \sum_{i=1}^{K}\left(V_i R_i(n)\right) = \sum_{i=1}^{K} R'_i(n)$$
  - Aplicando la fórmula de Little evaluar el throughput global

  $$X_o(n) = \frac{n}{R_0(n)}$$
  - Calcular el throughput individual de cada cola

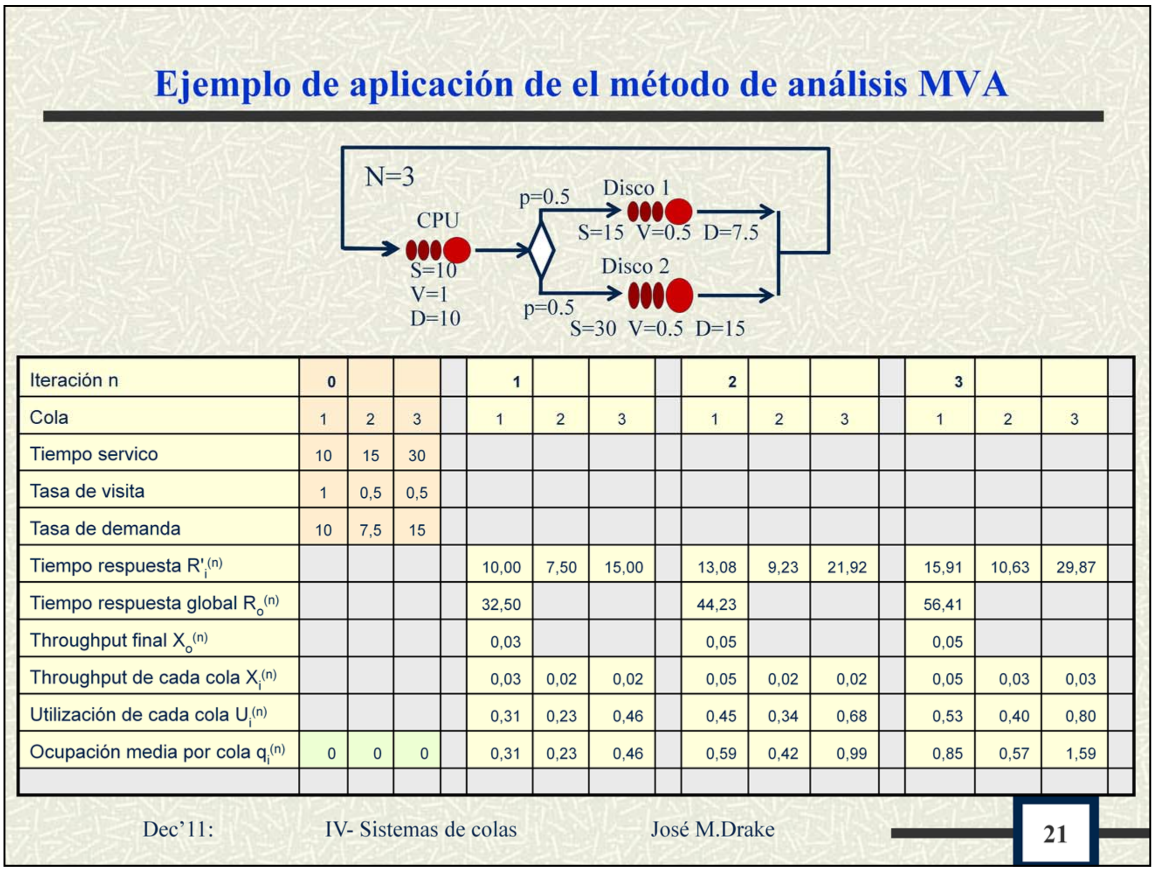
  $$X_i(n) = V_i X_o(n)$$
  - Calcular la utilización de cada cola

  $$U_i(n) = S_i X_i(n)$$
  - Calcular el número medio de requerimientos en cada cola

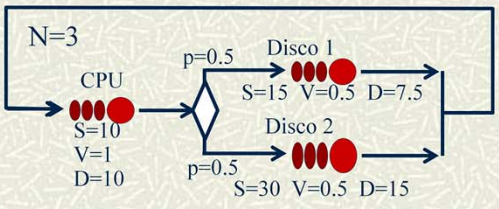
  $$Q_i(n) = X_o(n) R'_i(n)$$

Dec'11: IV- Sistemas de colas José M.Drake 20

Notas:



Ejemplo de aplicación de el método de análisis MVA



Iteración n	0				1				2				3			
Cola	1	2	3		1	2	3		1	2	3		1	2	3	
Tiempo servico	10	15	30													
Tasa de visita	1	0,5	0,5													
Tasa de demanda	10	7,5	15													
Tiempo respuesta $R'^{(n)}_i$					10,00	7,50	15,00		13,08	9,23	21,92		15,91	10,63	29,87	
Tiempo respuesta global $R^{(n)}_o$					32,50				44,23				56,41			
Throughput final $X^{(n)}_o$					0,03				0,05				0,05			
Throughput de cada cola $X^{(n)}_i$					0,03	0,02	0,02		0,05	0,02	0,02		0,05	0,03	0,03	
Utilización de cada cola $U^{(n)}_i$					0,31	0,23	0,46		0,45	0,34	0,68		0,53	0,40	0,80	
Ocupación media por cola $q^{(n)}_i$	0	0	0		0,31	0,23	0,46		0,59	0,42	0,99		0,85	0,57	1,59	

Notas:

Aproximación de Schweitzer

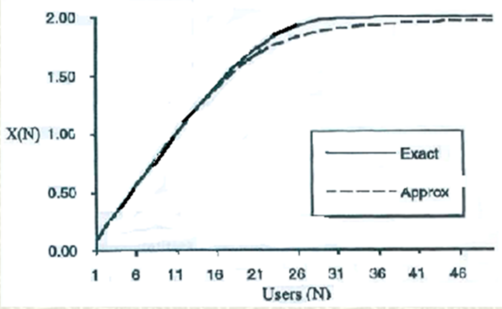
- Cuando el número de transacciones es alto, cabe considerar la aproximación:

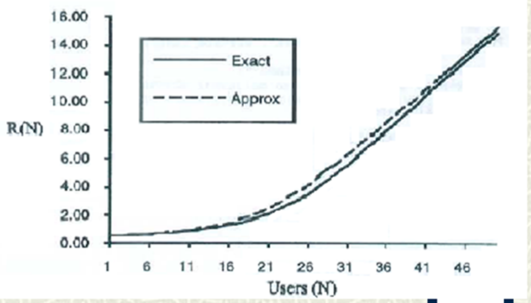
$$\frac{Q_i(N)}{N} \propto constante \quad \Leftrightarrow \quad \frac{Q_i(N)}{N} \approx \frac{Q_i(N-1)}{N-1} \quad \Rightarrow Q_i(N-1) = \frac{N-1}{N} Q_i(N)$$

- El tiempo de respuesta de la cola i resulta ser:

$$R_i(N) = D_i\left(1 + \frac{N-1}{N} Q_i(N)\right) \qquad X(N) = \frac{N}{Z + \sum V_i R_i(N)} \qquad Q_i(N) = X(N) V_i R_i(N)$$

- La aproximación es bastantes correcta salvo en la zona que corresponde a la saturación que proporciona una subestimación del throughput





Notas:

Mean-value analysis is a recursive algorithm. Computation of performance with *N* jobs in the network requires knowledge of performance with *N* – 1 jobs. Since performance with *N* = 0 is trivially known, we always start the analysis with *N* = 0 and compute the performance for *N* = 1,2,... successively. For small values of *N*, this procedure is not computationally too expensive. However, for large values of *N*, particularly if the performance for smaller values of N is not required, it would be preferable to avoid this recursion.

Several approximate analysis techniques have been developed with this goal. Here we describe one such approximation technique known as **Schweitzer's approximation**. It avoids the recursion in MVA by appropriately estimating the queue lengths with *N* jobs and computing the response times and throughputs. The values so computed can be used to recompute the queue lengths, and if the previous estimate was good, the new computed value would be close to the estimate.

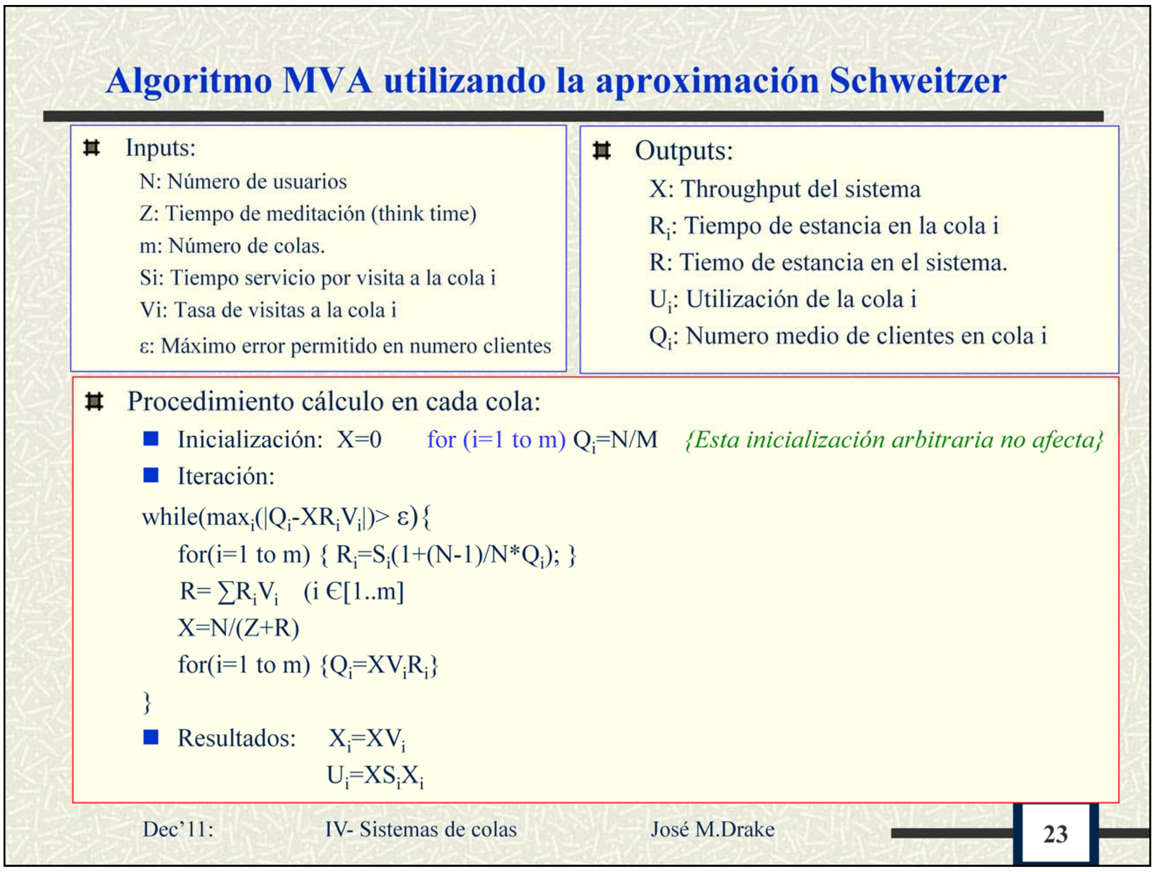
The approximation, due to Schweitzer (1979), is based on the assumption that as the number of jobs in a network increases, the queue length at each device increases proportionately. For example, doubling the

number of jobs in the network will result in doubling the number of jobs at each device. Analytically:

In particular, this implies

or

The MVA equations can therefore be written as follows:



Algoritmo MVA utilizando la aproximación Schweitzer

⌗ Inputs:
- N: Número de usuarios
- Z: Tiempo de meditación (think time)
- m: Número de colas.
- Si: Tiempo servicio por visita a la cola i
- Vi: Tasa de visitas a la cola i
- ε: Máximo error permitido en numero clientes

⌗ Outputs:
- X: Throughput del sistema
- R_i: Tiempo de estancia en la cola i
- R: Tiemo de estancia en el sistema.
- U_i: Utilización de la cola i
- Q_i: Numero medio de clientes en cola i

⌗ Procedimiento cálculo en cada cola:
- Inicialización: X=0 for (i=1 to m) Q_i=N/M *{Esta inicialización arbitraria no afecta}*
- Iteración:

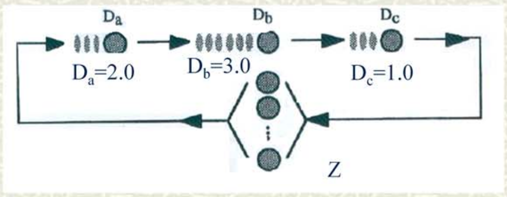
```
while(max_i(|Q_i-XR_iV_i|)> ε){
    for(i=1 to m) { $R_i=S_i(1+(N-1)/N*Q_i)$; }
    $R= \sum R_iV_i$   (i ∈[1..m])
    $X=N/(Z+R)$
    for(i=1 to m) {$Q_i=XV_iR_i$}
}
```

- Resultados: $X_i=XV_i$
  $U_i=XS_iX_i$

Notas:

Cotas de un sistema de colas

- Es muy instructivo determinar analizar un sistema a través de las cotas del throughput y de los tiempos de respuesta en un modelo formulado como sistema de colas.
- Consideremos el siguiente sistema de colas.



- La saturación se alcanza cuando en una de las cola se alcanza un 100% de utilización.

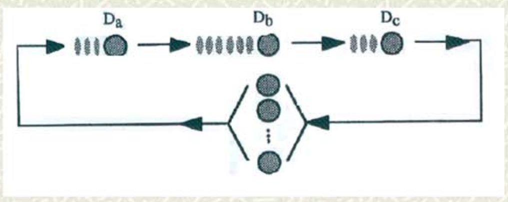
$$U_{\max} = 1 = X_{Sat} \times D_{\max} \quad \Rightarrow \quad X_{Sat} = \frac{1}{D_{\max}} = \frac{1}{D_b} = \frac{1}{3.0} = 0.33\, Trans/s$$

- Throughput sin congestión. No hay congestión cuando no hay requerimientos esperando en las colas . El tiempo de respuesta es igual al tiempo de servicio:

$$R = \frac{N}{X(N)} - Z \quad \Rightarrow \quad D_a + D_b + D_c = \frac{N}{X_{sc}(N)} - Z \quad \Rightarrow \quad X_{sc}(N) = \frac{N}{D_a + D_b + D_c - Z}$$

Notas:

Cotas y cuello de botella



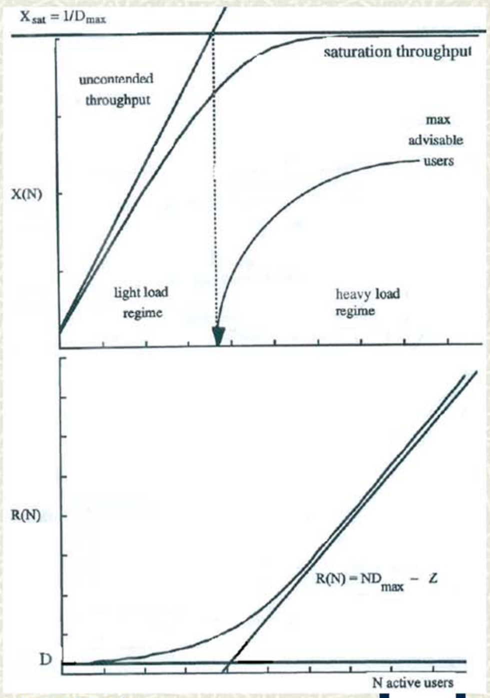
- Mejor tiempo de respuesta:

$$R_{\min} = D_a + D_b + D_c$$

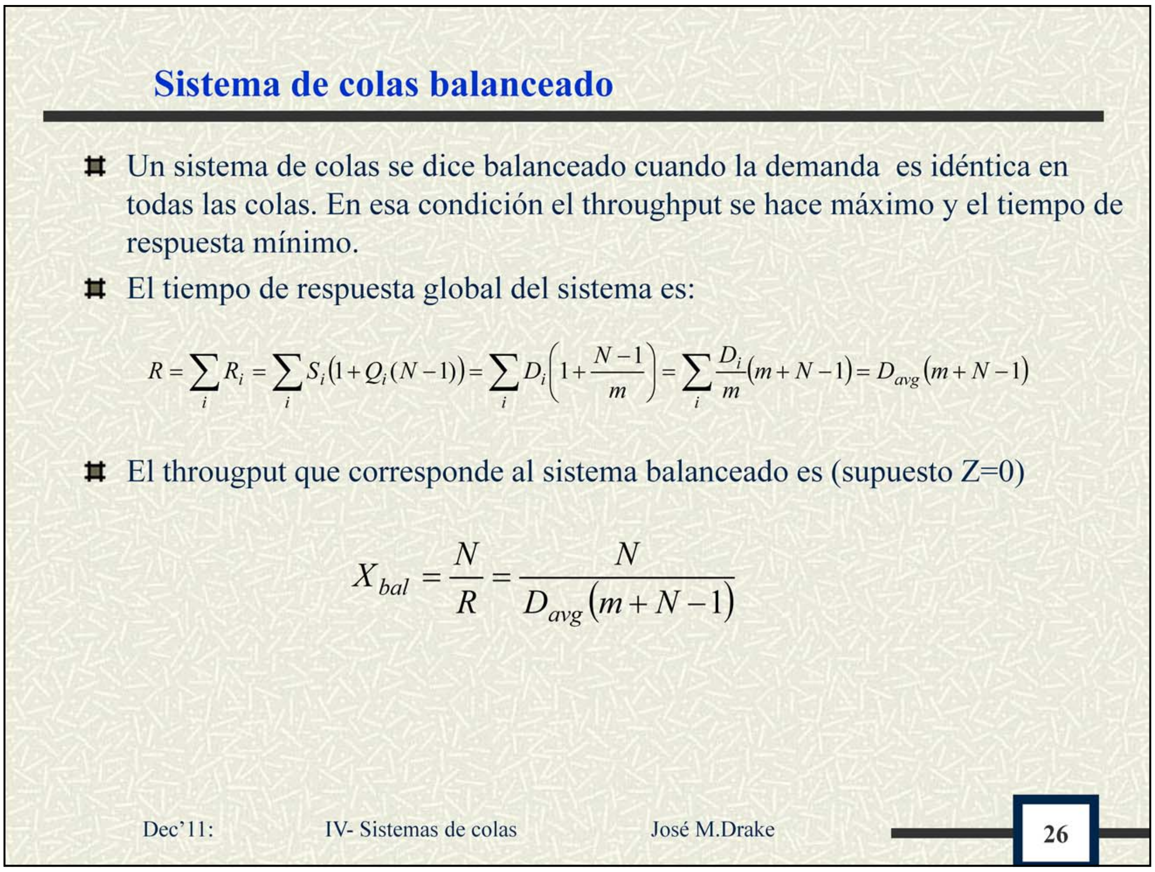
- El peor tiempo de respuesta R= ∞

- El tiempo de respuesta bajo saturación X=1/D_b:

$$R = \frac{N}{X_{sat}} - Z \quad => \quad R = N \times D_b - Z$$



Notas:



Sistema de colas balanceado

- Un sistema de colas se dice balanceado cuando la demanda es idéntica en todas las colas. En esa condición el throughput se hace máximo y el tiempo de respuesta mínimo.
- El tiempo de respuesta global del sistema es:

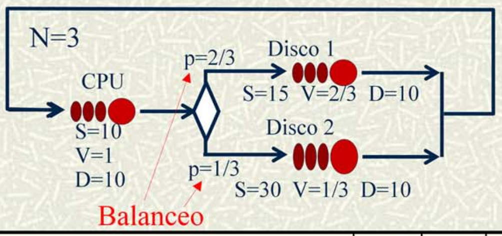
$$R = \sum_i R_i = \sum_i S_i\left(1 + Q_i(N-1)\right) = \sum_i D_i\left(1 + \frac{N-1}{m}\right) = \sum_i \frac{D_i}{m}(m+N-1) = D_{avg}(m+N-1)$$

- El througput que corresponde al sistema balanceado es (supuesto Z=0)

$$X_{bal} = \frac{N}{R} = \frac{N}{D_{avg}(m+N-1)}$$

Notas:

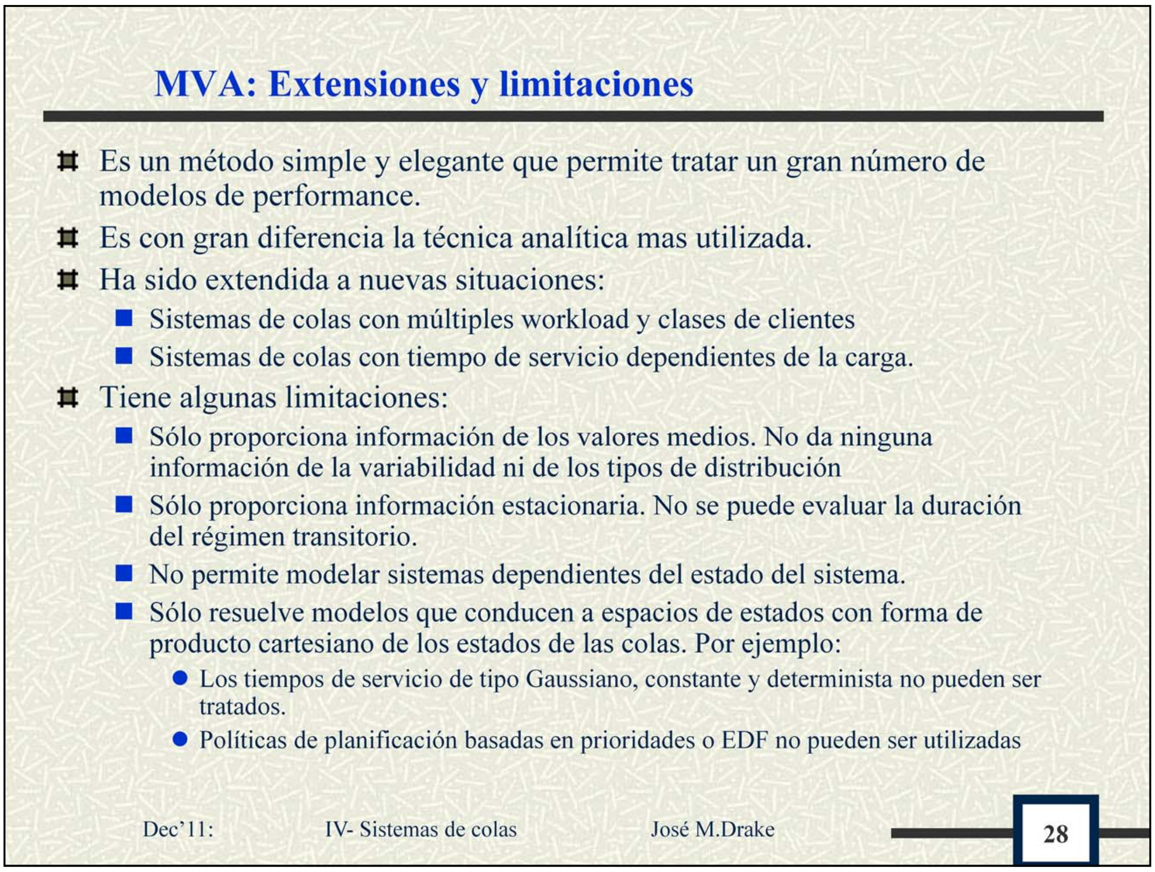
Ejemplo de sistema balanceado



$$X_o = \frac{n}{D_{avg}(K+n-1)} = \frac{10}{10\times(3+10-1)} = \frac{1}{12} = 0.8333$$

Iteración n		9				10			
Cola		1	2	3		1	2	3	
Tiempo servico	S_i	10	15	30					
Tasa de visita	V_i	1	0,7	0,3					
Tasa de demanda	D_i	10	10	10					
Tiempo respuesta $R'_i{}^{(n)}$						40,0000	40,0000	40,0000	
Tiempo respuesta global $R_o^{(n)}$						120,0000			
Throughput final $X_o^{(n)}$						0,0833			
Throughput de cada cola $X_i^{(n)}$						0,0833	0,0556	0,0278	
Utilización de cada cola $U_i^{(n)}$						0,8333	0,8333	0,8333	
Ocupación media por cola $q_i^{(n)}$		3	3	3		3,3333	3,3333	3,3333	

Notas:


MVA: Extensiones y limitaciones

- Es un método simple y elegante que permite tratar un gran número de modelos de performance.
- Es con gran diferencia la técnica analítica mas utilizada.
- Ha sido extendida a nuevas situaciones:
  - Sistemas de colas con múltiples workload y clases de clientes
  - Sistemas de colas con tiempo de servicio dependientes de la carga.
- Tiene algunas limitaciones:
  - Sólo proporciona información de los valores medios. No da ninguna información de la variabilidad ni de los tipos de distribución
  - Sólo proporciona información estacionaria. No se puede evaluar la duración del régimen transitorio.
  - No permite modelar sistemas dependientes del estado del sistema.
  - Sólo resuelve modelos que conducen a espacios de estados con forma de producto cartesiano de los estados de las colas. Por ejemplo:
    - Los tiempos de servicio de tipo Gaussiano, constante y determinista no pueden ser tratados.
    - Políticas de planificación basadas en prioridades o EDF no pueden ser utilizadas

Dec'11: IV- Sistemas de colas José M.Drake 28

Notas:

The basic MVA algorithm is quite powerful and elegant. It is applicable across a wide set of performance models. It has been the focus of much research and several extensions have been developed. These include:

- Multi-class networks
- Networks with load dependent servers
- Networks with open and closed classes of customers

Several approximation techniques have also be adapted to MVA to model systems having non-product form features, including first-come-first-serve and priority multi-class networks. The extension of MVA to product form, load-independent, multi-class networks is found in references.

However, even with its widespread applicability, there are limitations and shortcomings surrounding MVA. For example:

•MVA does not provide the steady state probabilities of individual system states. Thus, if it were important to know the probability that there were, say, less than five customers at a device in order to meet some QoS criteria, MVA would not be helpful. It would be necessary to revert to solving the global balance equations. As its name implies, MVA only provides the mean values of various performance metrics, not the associated distributions.

•MVA does not provide transient analysis information. For instance, if it were necessary to know how long it would take the system to recover from a temporary overload in one sector of the system and return to "steady state" behavior, MVA would be inadequate.

•MVA does not model state dependent behavior. For example, consider the modeling of a simple routing protocol, with two paths between a particular source and destination, where the customers (i.e., message packets) select the path least busy (i.e., dependent on the particular system state). Although a Markovian model can be easily constructed and solved for this situation, MVA is of little use.

•MVA solves product form networks. As a result, MVA is not directly applicable to non-product form situations. These restrictions exclude certain device service distributions (e.g., Gaussian, uniform, constant), certain device queuing disciplines (e.g., priority, multi-class FCFS), and certain device loading strategies (e.g., shortest queue routing, deterministic routing).