POLITECNICO DI MILANO

Computing Infrastructures





Analytical techniques: open models

Danilo Ardagna

Credits: Ed Lazowska, Marco Gribaudo

Model Definition for Performance Studies

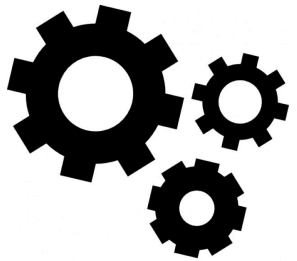
Existing system

System Design

Parameterize the Model

Performance Model

Create the Queuing Network



Evaluate the Model

Focus on single class

Focus on open models

Performance indices

Performance of open models

- We have characterized both open and closed queuing networks
- We have seen how we can find upper and lower bounds to throughput and response time with simple analytical expressions
- If system fulfills a set of requirements, it is possible to determine the performance of a queuing network analytically, using simple algorithms
- These types of systems are called *separable queuing networks*
- The computations can be carried on manually, or specific tools can be used

Performance can be computed starting from the average number of jobs found in the queue by a new arrival to a station

Assumption 1

- *Service center flow balance*: The number of arrivals at each service center is equal to the number of completions there

$$A_k = C_k$$

$$\lambda_k = X_k$$

Assumption 2

- *One step behaviour:* Only a single customer can move (arrive to or depart from a service center) at a time (no two jobs in the system can “change state” at exactly the same time)

Assumption 3

- *Routing homogeneity*: The proportion of time that a customer leaving center *j* proceeds directly to center *k* depends only on *j*, and *k*, and is independent of the number of customers currently at any of the centers, for all *j*, and *k* (i.e., routing is independent of the queue lengths at any center)

- A surprising result of separable models is that routing patterns are irrelevant to the performance measures at the system level (we will consider the routing to compute visits, but then we can neglect the routing)

- As long as nodes have the same demands, two queuing networks have the same system performance, independently of their topology

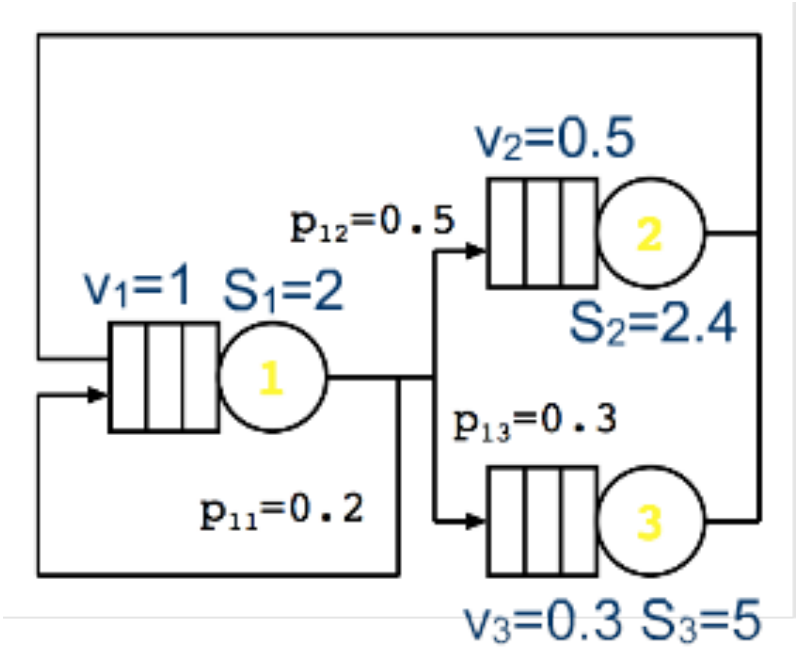
Assumption 3

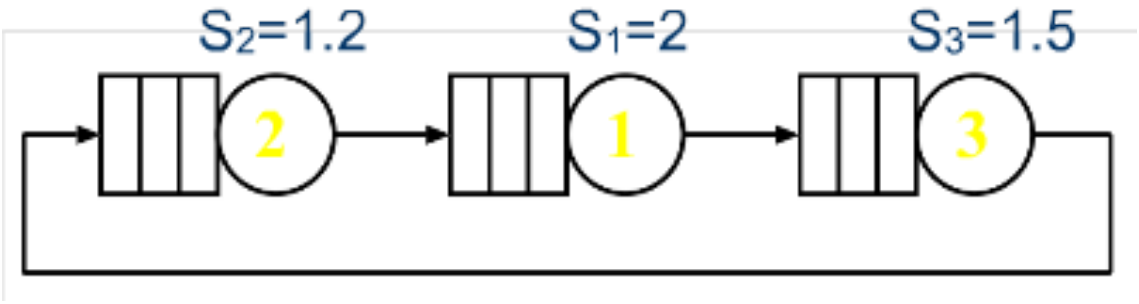
- Among the routing policies we have seen, the probabilistic routing fully satisfies the previous assumption

- Other routing policies, such as Join the Shortest Queue (JSQ) clearly violate this constraint, since the decision of the next station depends on the whole state of the system

Assumption 3

For example, these two networks have exactly the **same performance** in term of *system throughput*, *system response time*, *average number of jobs*, *utilizations* and *residence times*.



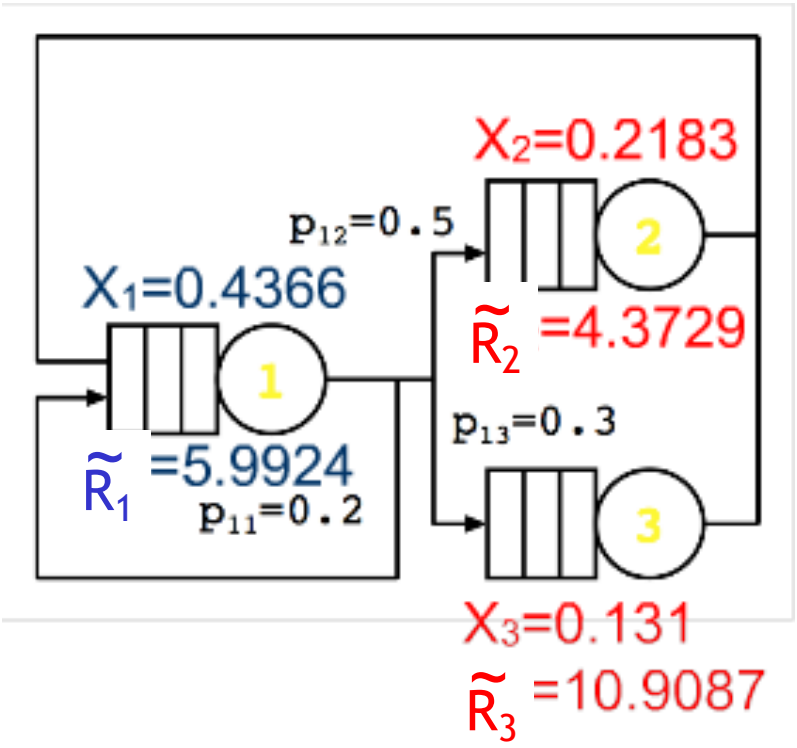


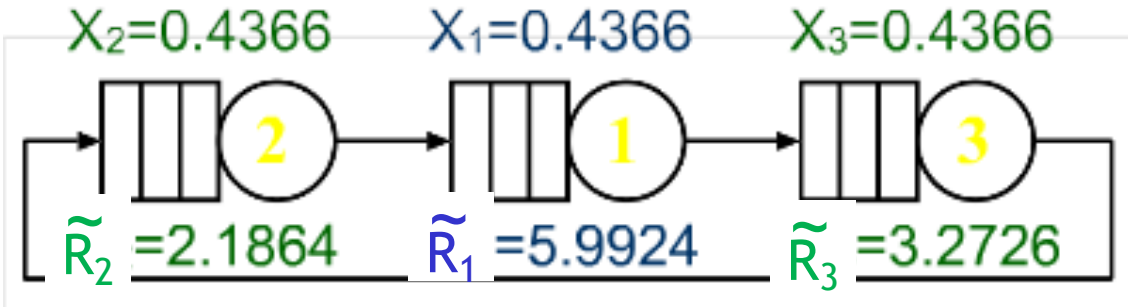
X=0.4366
R=11.4515

U_1=0.8732	U_2=0.5239	U_3=0.6549
N_1=2.6165	N_2=0.9546	N_3=1.4289
R_1=5.9925	R_2=2.1864	R_3=3.2726

Assumption 3

However, the two networks have **different** *throughputs* and average time for the single access (*response times)* at the single resources since stations are characterized by different visits





Assumption 3

- As we have seen, station throughput and response time depend on the visits and not only on the demands of the considered stations

$$X_k = v_k \cdot X$$

$$\widetilde{\mathrm{R}}_k = \frac{N_k}{X_k} = \frac{R_k}{v_k}$$

- To analyze a queuing network, we need its topology just to determine the visits and the demands of the different stations

- We can then perform the analysis using **only** such values

Assumption 4

- *Device homogeneity*: The rate of completions of jobs from a center may vary in an arbitrary manner with the number of jobs at that center, but otherwise may not depend on the number or placement of customers within the network

- This is a very tricky and important assumption that considers the combination of service time distributions, number of servers and queuing policies

- Processor sharing and infinite servers always fulfill the considered property

- FCFS queuing policy instead fulfills the previous property only for the exponential distribution (otherwise, the time depends on the placement of the job in the queue)

Assumption 5

- *Homogeneous external arrivals:* The rate of arrival of customers is independent of the number of the customers currently in the system or the placement of those customers

Assumption 6

- If we add also a sixth assumption (that reduces the scope of the fourth) solutions can be computed with simpler algorithms

- *Service time homogeneity:* The completion rate of jobs at each center, while it is busy, must be independent of the number of customers at that center, in addition to being independent of the number or placement of customers within the network

In this course, we will focus only on these cases

Assumptions general considerations

- Even if the previous assumptions seem very strict, they are valid for a large set of systems

- Moreover, for systems that do not fulfill the previous properties, the considered techniques can provide meaningful approximations

Jobs at the arrival

- To compute analytical solutions, there are two different techniques, depending on whether the considered system is open or closed

- In this course we focus only on open models, for which the technique is simpler, closed systems are considered by the more advanced *Performance Evaluation* course offered next semester

- Both techniques are based on the computation of the same property: *the average number of jobs found in the queue (**both** in **the queue** or **being served**) by a new arrival to a station*

Jobs at the arrival

- In both open or closed models we can compute all performance indices using Little's law and utilization law, starting from the residence time at each station

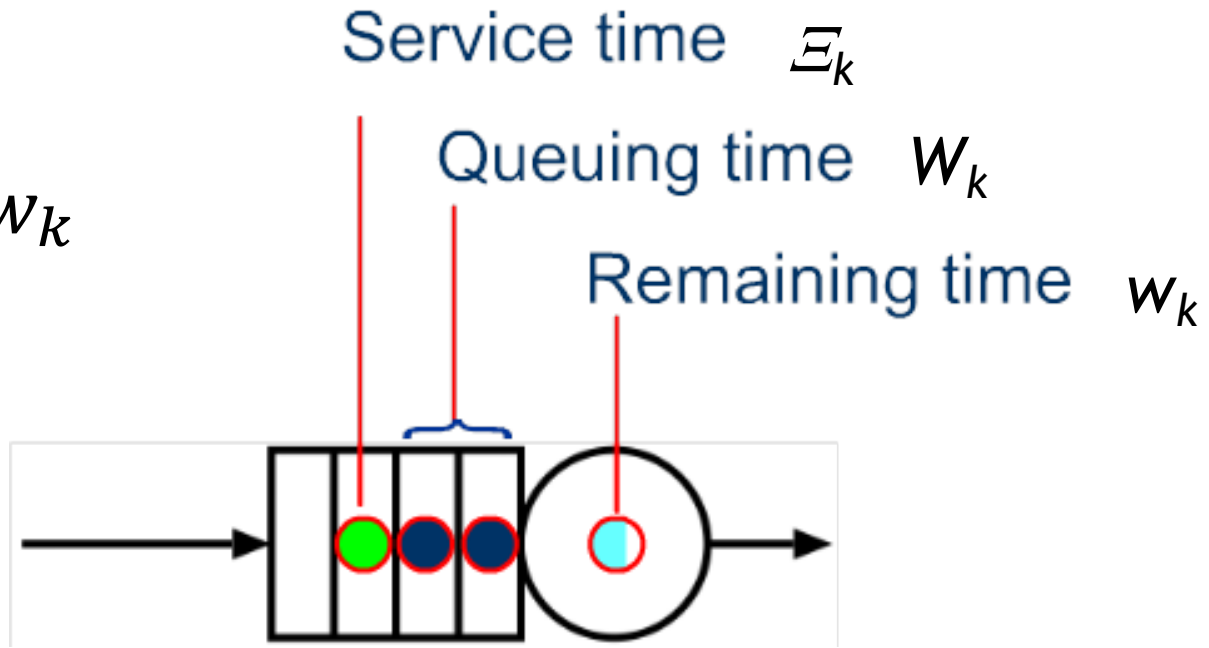
$$N = X \cdot R \qquad N_k = X \cdot R_k$$

$$U_k = X \cdot D_k$$

Jobs at the arrival

- Let us focus now on the time spent at node k, $\tilde{R}_k$, during one visit
- It is given by the sum of 3 components:
  - The service time of the arriving job Ξ_k
  - The queuing time W_k
  - The remaining service time w_k of the job found at the arrival

$$\tilde{R}_k = \Xi_k + W_k + w_k$$



Jobs at the arrival

- For infinite servers (i.e. delay stations), both the remaining service time w_k and the queuing time W_k are zero, since each job is served immediately
- $\tilde{R}_k$ is then identical to the average service time S_k

$$\Xi_k = S_k$$
$$W_k + w_k = 0$$
$$\tilde{R}_k = S_k$$

Jobs at the arrival

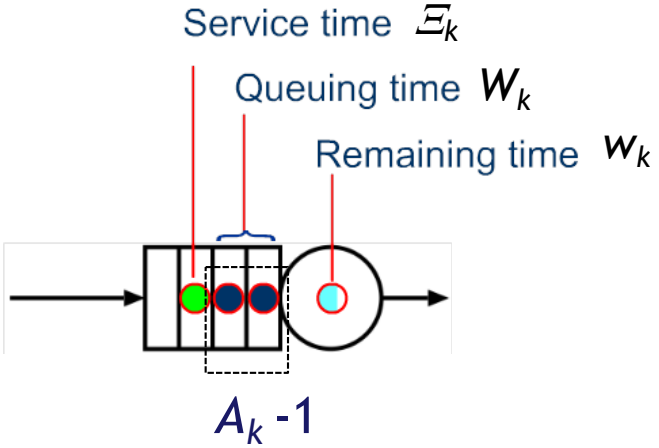
- In all the other cases, the service time homogeneity guarantees that the remaining service time is identical to the complete service time of one job

- In stations with exponential service time, this is guaranteed by the memory-less property of the exponential distribution

$$\Xi_k = S_k \qquad w_k = S_k$$

Jobs at the arrival

- The waiting delay can be expressed from the number A_k of costumers that an arriving job founds at the service center (both in queue and in service) at its arrival
- In particular we have:



$$\tilde{R}_k = \Xi_k + W_k + w_k$$

$$\Xi_k = S_k \qquad w_k = S_k$$

$$W_k = (A_k - 1) \cdot S_k$$

$$\tilde{R}_k = S_k + (A_k - 1) \cdot S_k + S_k = (1 + A_k) \cdot S_k$$

Jobs at the arrival

- In processor sharing systems the waiting time is also zero and the remaining time of the job in service is not relevant since jobs starts being served immediately

- However the slowdown caused by the other jobs is proportional to the length of the queue, and also in this case we have:

$$\tilde{R}_k = S_k + A_k \cdot S_k = (1 + A_k) \cdot S_k$$

Jobs at the arrival

- As a summary, for non-delay stations we have:

$$\tilde{R}_k = (1 + A_k) \cdot S_k$$

$$\mathrm{R}_k = v_k \cdot \tilde{R}_k = (1 + A_k) \cdot v_k \cdot S_k = (1 + A_k) \cdot D_k$$

- For delay stations instead we simply have:

$$\mathrm{R}_k = D_k$$

Jobs at the arrival

- In open models, the average number of jobs found in a service center at the arrival of a new costumer (both in queue and in service) is identical to the average number of jobs at the service center (both in queue and in service) :

$$A_k(\lambda) = N_k(\lambda)$$

$$\mathrm{R}_k(\lambda) = \big(1 + A_k(\lambda)\big) \cdot D_k = \big(1 + N_k(\lambda)\big) \cdot D_k$$

- $N_k(\lambda)$, which includes both jobs in queue and in service, is called queue length and also denoted with $Q_k(\lambda)$

Open models

- Using Little's law, we can express the queue length as function of the residence time and the arrival rate

$$\mathrm{R}_k(\lambda) = \left(1 + \mathrm{N}_k(\lambda)\right) \cdot D_k$$

$$N_k(\lambda) = \mathrm{X}(\lambda) \cdot \mathrm{R}_k(\lambda) = \lambda \cdot \mathrm{R}_k(\lambda)$$

$$\mathrm{R}_k(\lambda) = \left(1 + \lambda \cdot \mathrm{R}_k(\lambda)\right) \cdot D_k$$

Open models

- We can then rework the equation and apply the utilization law to find an expression for the residence time as function of the arrival rate (or of the utilization)

$$\mathrm{R}_k(\lambda) = D_k + \lambda \cdot \mathrm{R}_k(\lambda) \cdot D_k$$

Open models

- We can then rework the equation and apply the utilization law to find an expression for the residence time as function of the arrival rate (or of the utilization)

$$\mathrm{R}_k(\lambda) = D_k + \lambda \cdot \mathrm{R}_k(\lambda) \cdot D_k$$

$$(1 - \lambda \cdot D_k) \cdot \mathrm{R}_k(\lambda) = D_k$$

$$\mathrm{R}_k(\lambda) = \frac{D_k}{1 - \lambda \cdot D_k} = \frac{D_k}{1 - \mathrm{U}_k(\lambda)}$$

Open models

- Applying Little's law, we can compute the average number of jobs in a station and in the entire system

$$N_k(\lambda) = Q_k(\lambda)$$

Open models

- Applying Little's law, we can compute the average number of jobs in a station and in the entire system

$$N_k(\lambda) = Q_k(\lambda) = \lambda \cdot \mathrm{R}_k(\lambda) = \frac{\lambda \cdot D_k}{1 - \mathrm{U}_k(\lambda)} =$$

$$= \frac{\mathrm{U}_k(\lambda)}{1 - \mathrm{U}_k(\lambda)}$$

Open models

- Finally, starting from performance measures associated to the stations, we can derive the ones for the entire system:

Average number in system: $$N(\lambda) = \sum_{i=1}^{K} N_i(\lambda) = \sum_{i=1}^{K} Q_i(\lambda)$$

System response time: $$R(\lambda) = \sum_{i=1}^{K} R_i(\lambda)$$

Open models summary

Processing capacity: $\lambda_{sat} = 1 / D_{max}$

Throughput: $X(\lambda) = \lambda$

Utilization: $U_k(\lambda) = \lambda D_k$

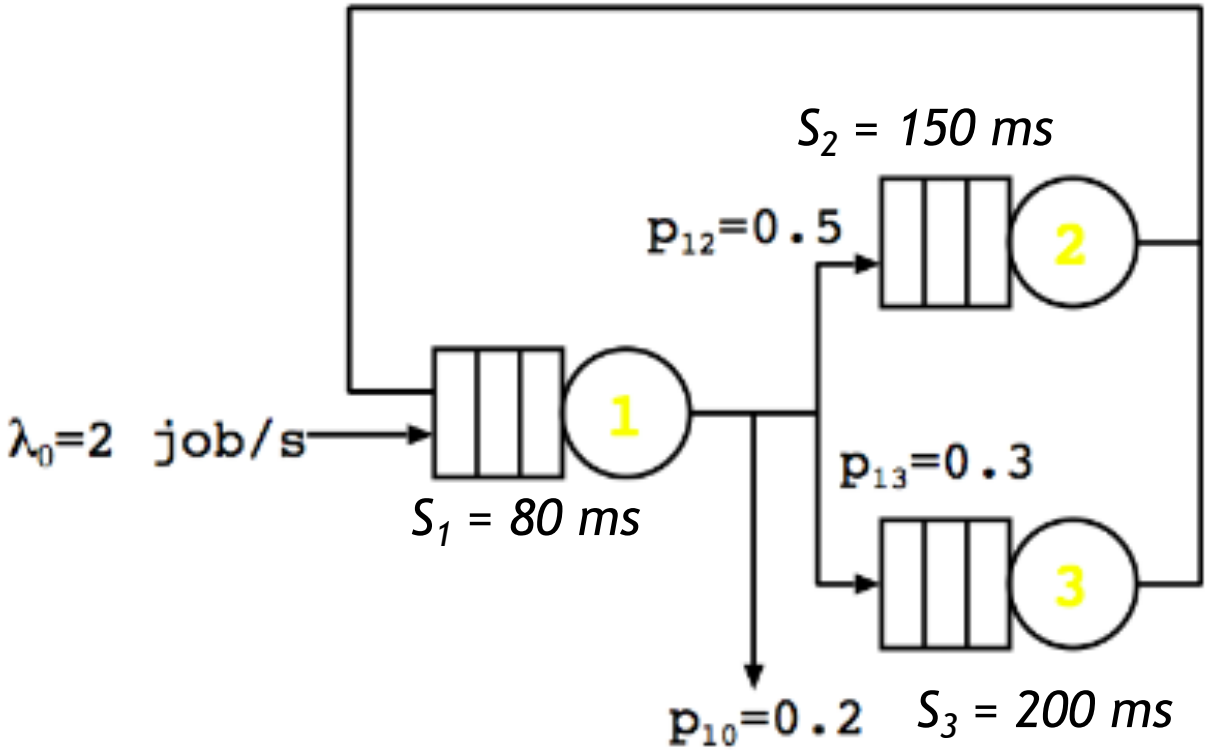
Residence time: $$R_k(\lambda) = \begin{cases} D_k & \text{Delay centers} \\ \dfrac{D_k}{1 - U_k(\lambda)} & \text{Queueing centers} \end{cases}$$

Queue length: $$N_k(\lambda) = Q_k(\lambda) = \lambda R_k(\lambda) = \begin{cases} U_k(\lambda) & \text{Delay centers} \\ \dfrac{U_k(\lambda)}{1 - U_k(\lambda)} & \text{Queueing centers} \end{cases}$$

Example: solution of an open model

Example

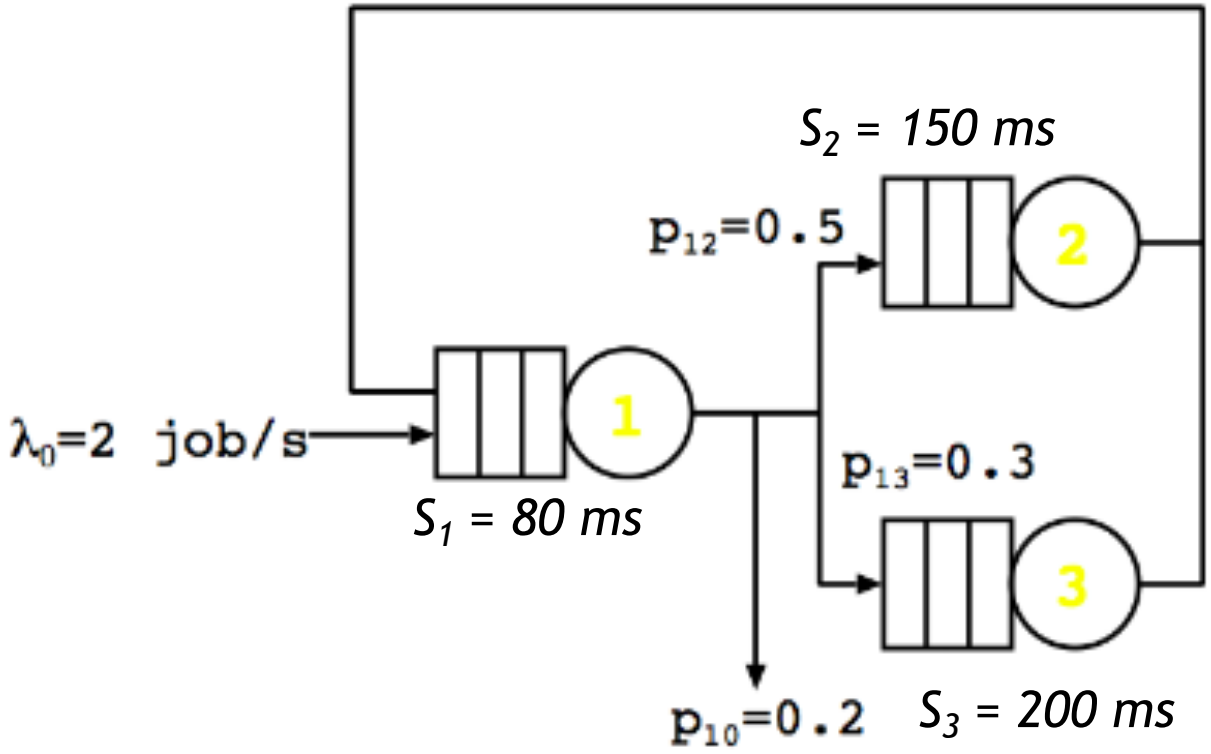
Compute the all the main performance indices for the following model:



Example: solution of an open model

Example

Compute the all the main performance indices for the following model:



$$\begin{cases} v_1 = 5 \\ v_2 = 2.5 \\ v_3 = 1.5 \end{cases}$$

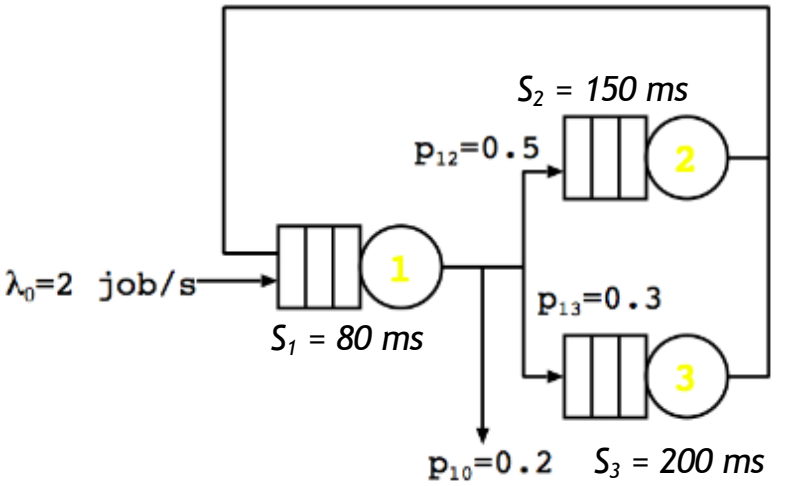
$$D_1 = 5 \cdot 80 = 0.4s$$
$$D_2 = 2.5 \cdot 150 = 0.375s$$
$$D_3 = 1.5 \cdot 200 = 0.3s$$

$$U_1 = 2 \cdot 0.4 = 0.8$$
$$U_2 = 2 \cdot 0.375 = 0.75$$
$$U_3 = 2 \cdot 0.3 = 0.6$$

$$X = \lambda = 2\ j/s$$

$$X_1 = \lambda_1 = 10\ j/s$$
$$X_2 = \lambda_2 = 5\ j/s$$
$$X_3 = \lambda_3 = 3\ j/s$$

Example: solution of an open model



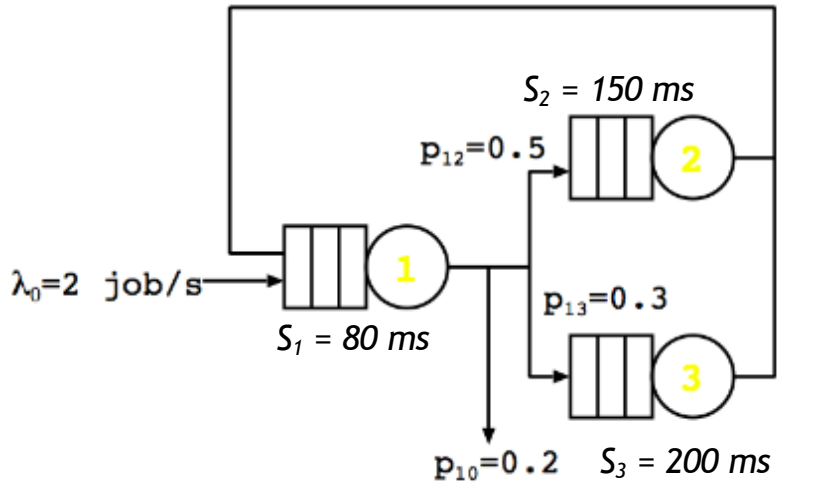
Example

Compute the all the main performance indices for the following model:

$$D_1 = 5 \cdot 80 = 0.4s$$
$$D_2 = 2.5 \cdot 150 = 0.375s$$
$$D_3 = 1.5 \cdot 200 = 0.3s$$

$$U_1 = 2 \cdot 0.4 = 0.8$$
$$U_2 = 2 \cdot 0.375 = 0.75$$
$$U_3 = 2 \cdot 0.3 = 0.6$$

Example: solution of an open model



Example

Compute the all the main performance indices for the following model:

$$D_1 = 5 \cdot 80 = 0.4s$$
$$D_2 = 2.5 \cdot 150 = 0.375s$$
$$D_3 = 1.5 \cdot 200 = 0.3s$$

$$U_1 = 2 \cdot 0.4 = 0.8$$
$$U_2 = 2 \cdot 0.375 = 0.75$$
$$U_3 = 2 \cdot 0.3 = 0.6$$

$$R_1 = \frac{0.4}{1-0.8} = 2s \qquad N_1 = \frac{0.8}{1-0.8} = 4$$

$$R_2 = \frac{0.375}{1-0.75} = 1.5s \qquad N_2 = \frac{0.75}{1-0.75} = 3$$

$$R_3 = \frac{0.3}{1-0.6} = 0.75s \qquad N_3 = \frac{0.6}{1-0.6} = 1.5$$

$$\tilde{R}_1 = \frac{0.08}{1-0.8} = 0.4s$$

$$\tilde{R}_2 = \frac{0.15}{1-0.75} = 0.6s$$

$$\tilde{R}_3 = \frac{0.2}{1-0.6} = 0.5s$$

$$R = 2 + 1.5 + 0.75 = 4.25s$$

$$N = 4 + 3 + 1.5 = 8.5$$