

Performance Evaluation and Applications

 POLITECNICO DI MILANO

Multi-class Queueing Networks



Motivating example

An E-commerce storage server, when handling $\lambda = 20 \text{ req/s}$, shows a 50% utilization. After a while, the workload increases 20%, reaching a total of $\lambda = 24 \text{ req/s}$. The administrator would expect to experience an utilization of 60%: the system however becomes unstable!

What is happening? Why the system has become unstable?

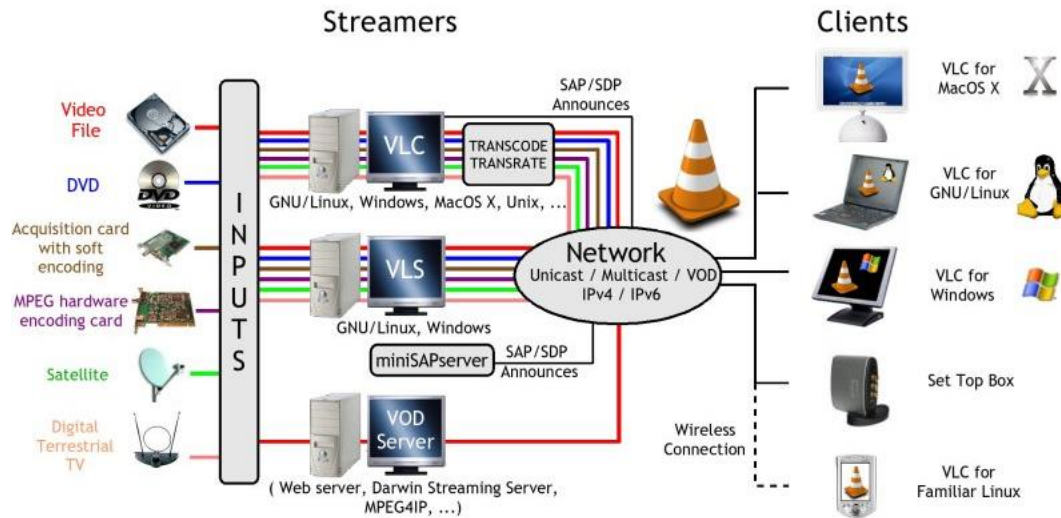
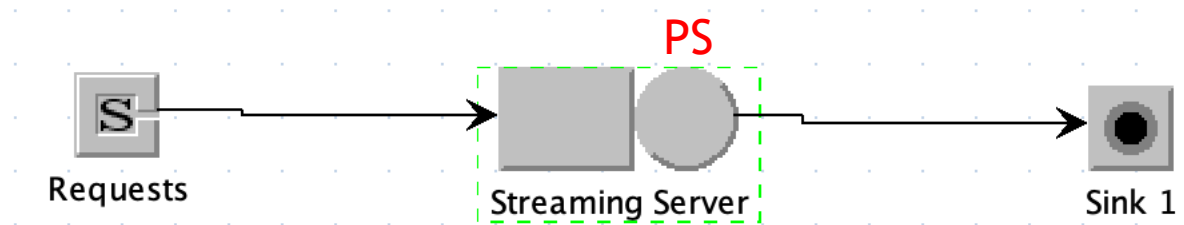




Motivation

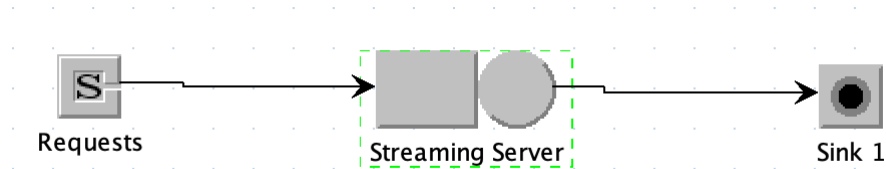
3

Let us consider a simple streaming server system. It can be modelled with a $M/G/1/PS$ queue (*a separable model*).

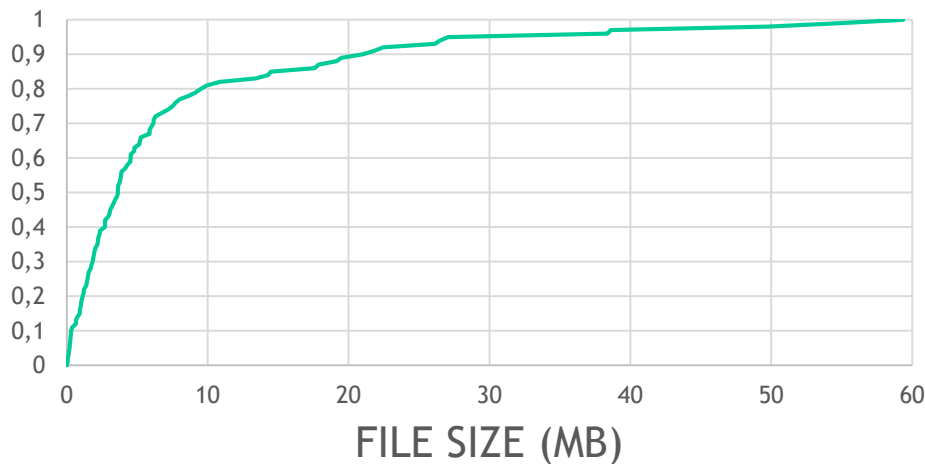




We measure its main performance indices, and we guess its average service time and current workload.



CONTENT SIZE DISTRIBUTION



$$U = 0.356$$

$$X = 5 \text{ file / s}$$

$$R = 116 \text{ ms}$$

$$N = 0.562 \text{ files}$$

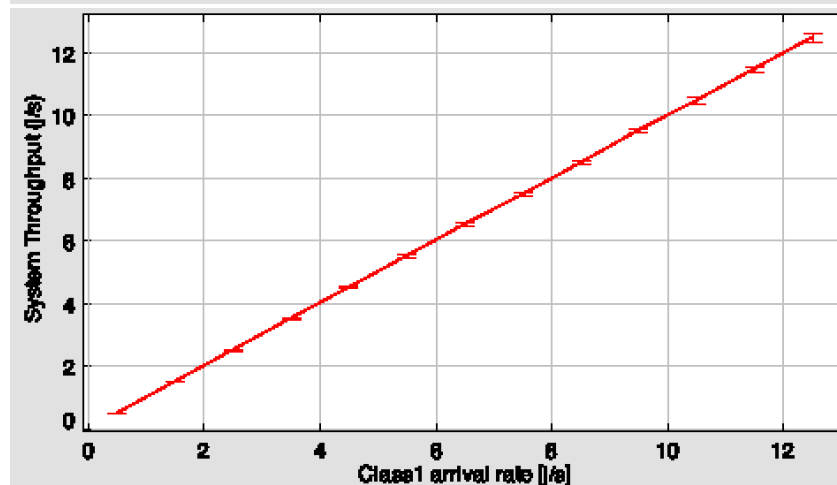
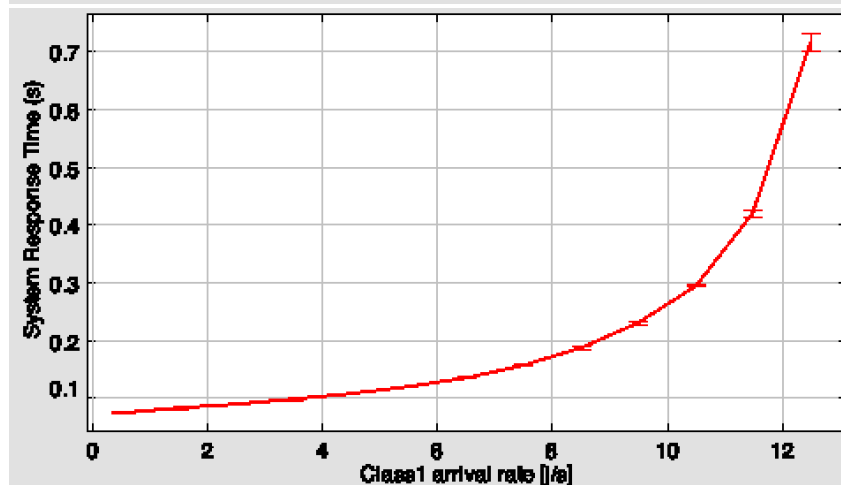
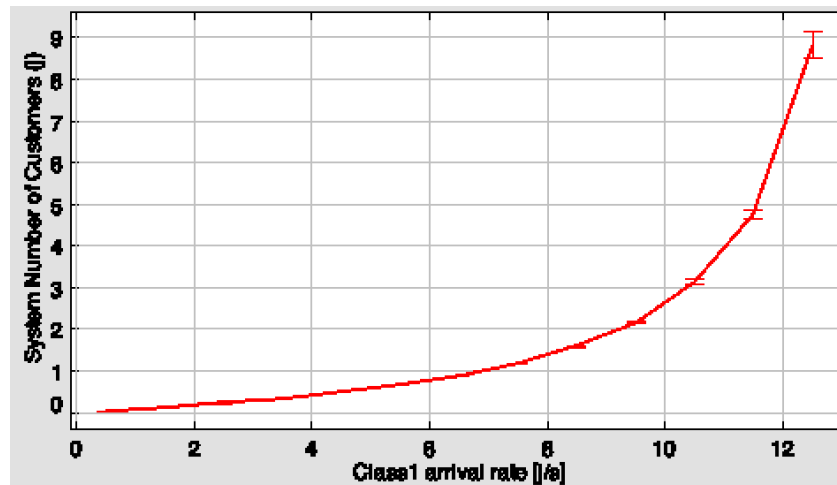
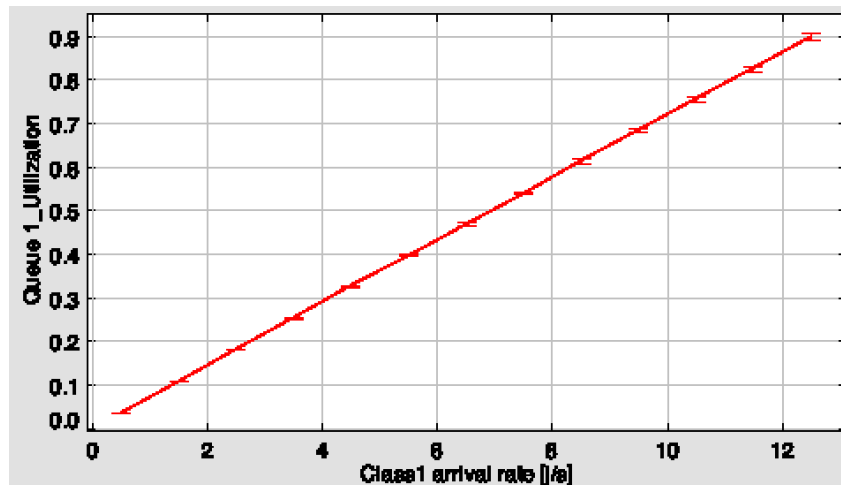
Average file size: 7.2 MB

Average network speed: 100 MB/s

$$\rightarrow S = 72 \text{ ms}$$

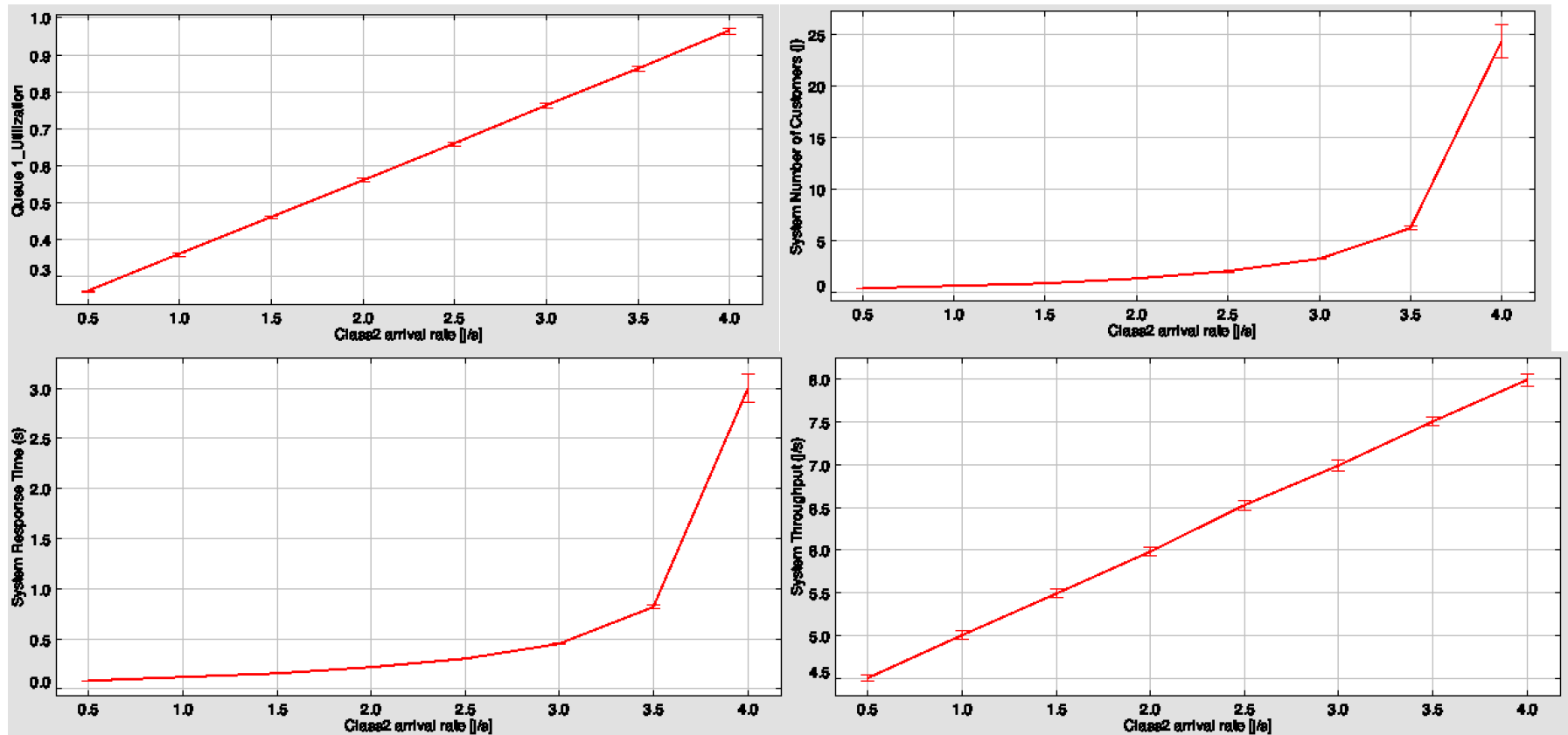


With a model, we try to infer its performance varying the workload (arrival rate).



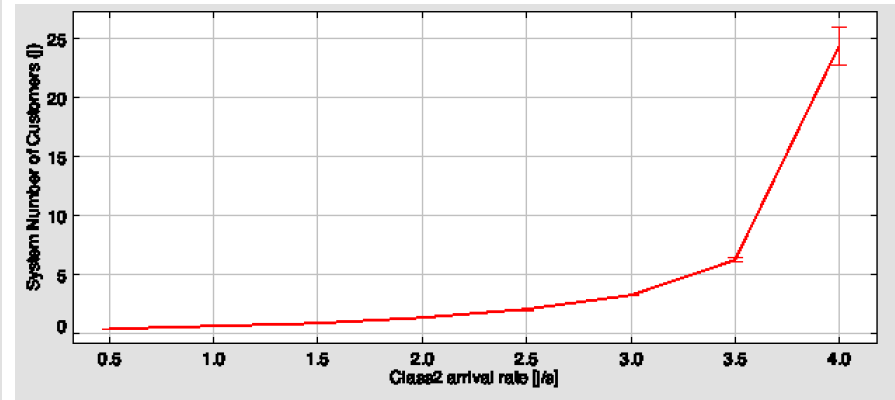
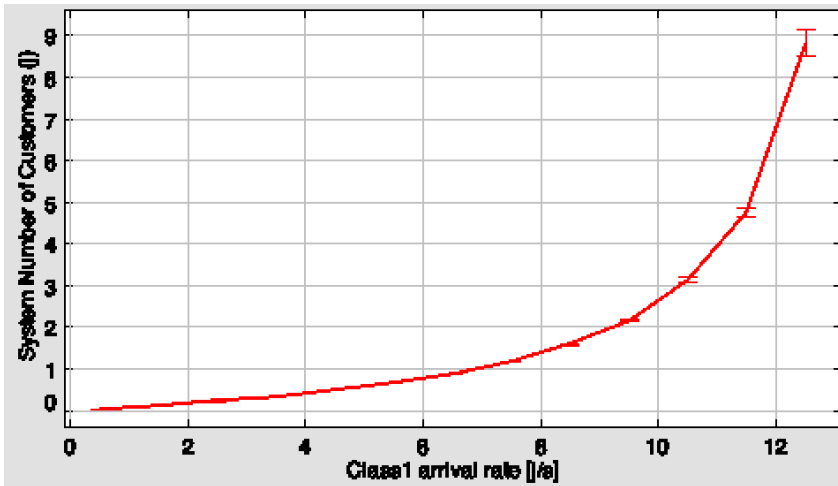


During the validation, we see that the system evolves in a different way, becoming unstable much earlier than expected.

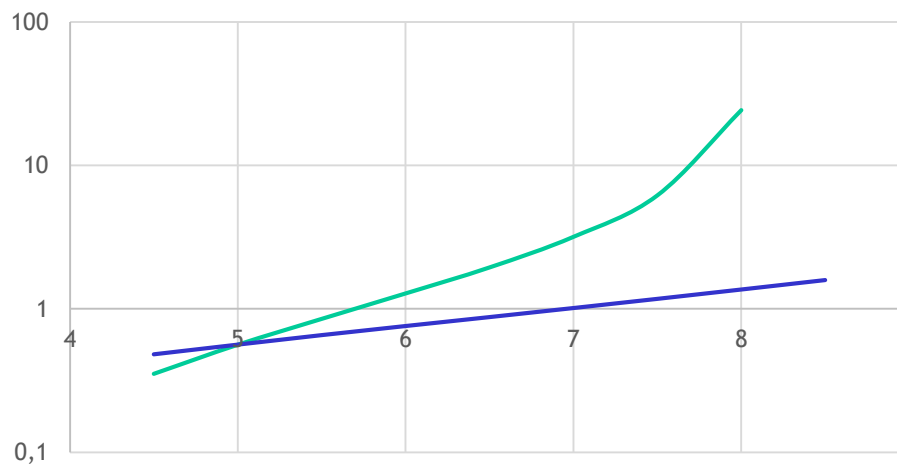




Let's compare more closely the average number of jobs



— Real — Model

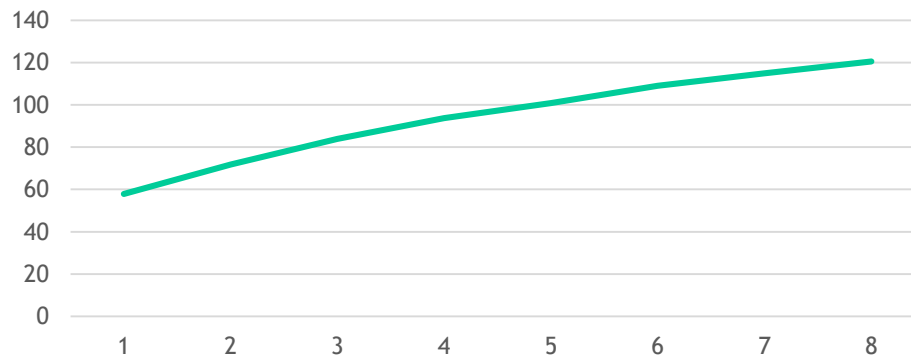
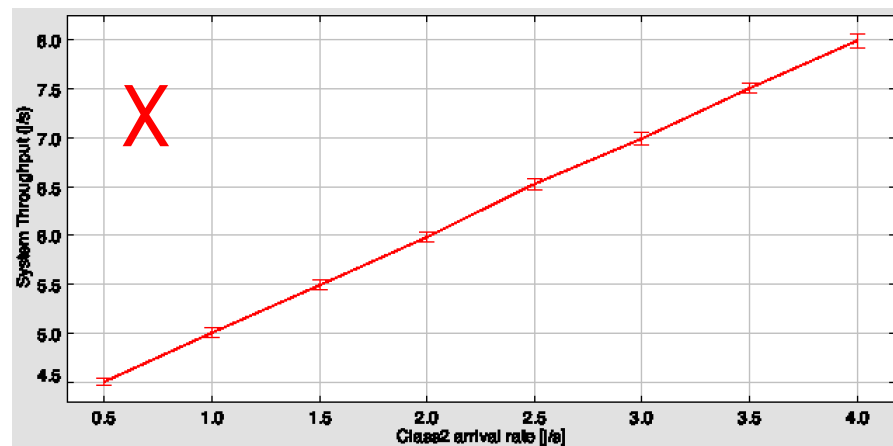
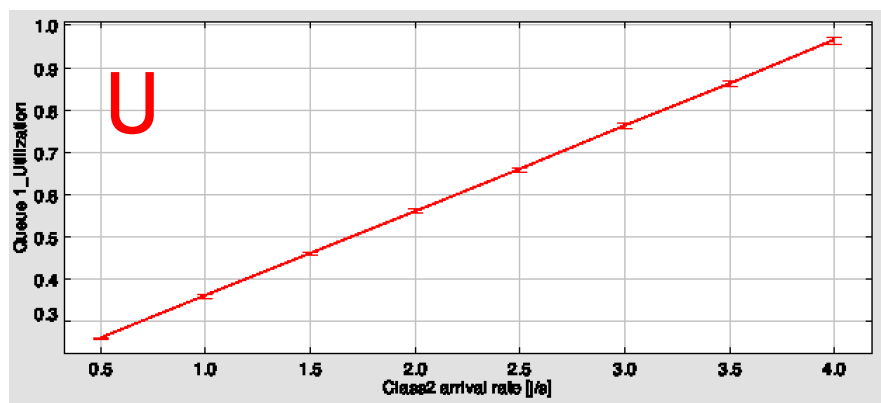


The model overestimates the response time for smaller arrival rates, and underestimates it for higher workloads.



With a closer look, it seems that the average service time is increasing with the workload.

$$U = X S \quad S = U / X$$



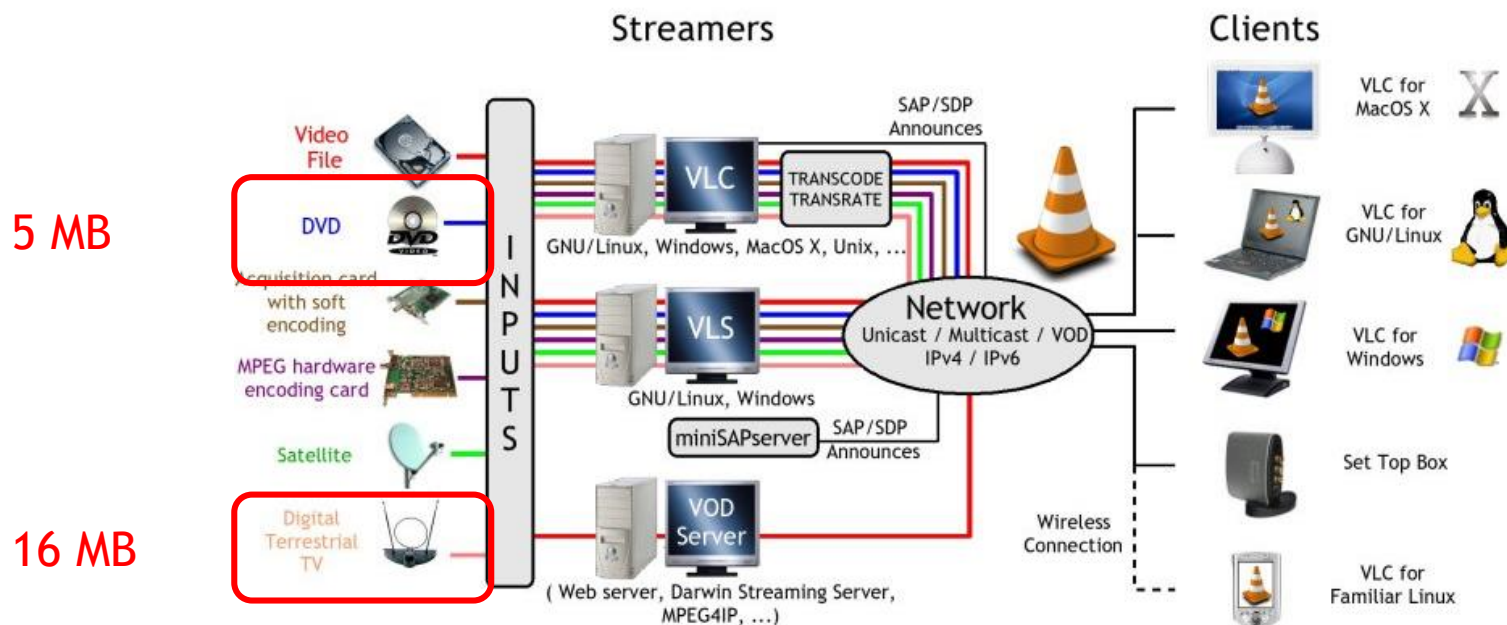
$$S = U / X$$



Motivation

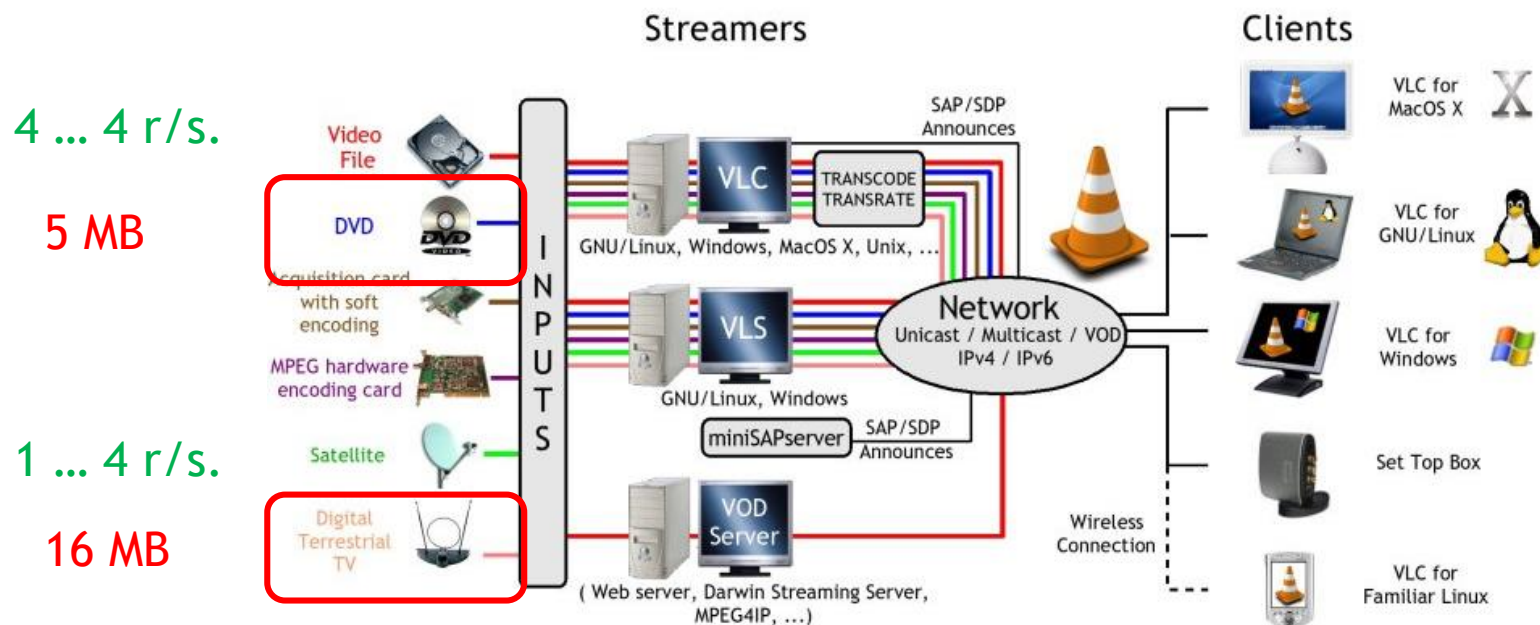
9

This however is caused by a simple fact: the workload is divided into two different types of requests, that are characterized by very different service times.





This workload does not increase uniformly between these two types of requests: in particular, it seems that the only type of requests growing are the longer ones.





Multi-class models

Multi-class models distinguish the type of jobs circulating in the system and allow job-dependent behaviors.

Jobs are partitioned into *classes*: all the jobs belonging to the same class are characterized by the same properties.

However, each class has different parameters and is characterized by an independent behavior with respect to the others.



Multi-class models

As for single-class systems, *open models* are characterized by external arrivals of jobs belonging to different classes, and *Closed models* are networks where each class is characterized by a fixed number of jobs that circulates inside the system, and possibly a corresponding different reference station and think time.

Multi-class systems also allows *Mixed models*, in which some classes are populated by jobs coming from external arrivals, and some others are characterized by a fixed population circulating in the network.

Multi-class models

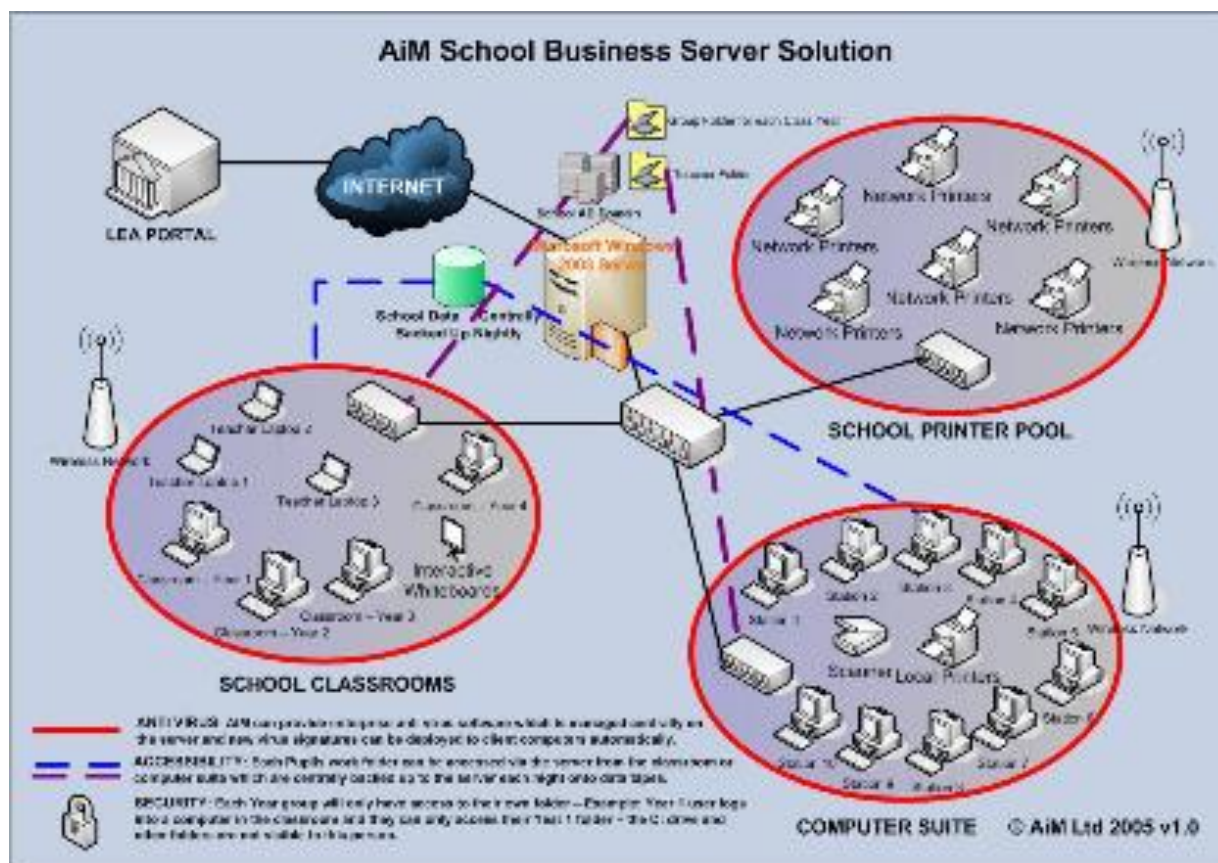
For example, a dynamic web page, offering two types of services (e.g. searching a catalog, and looking at images of a product) can be studied with an *open multi-class model*.

Classes corresponds to the different types of services, and the resources to the tiers of the application.



Multi-class models

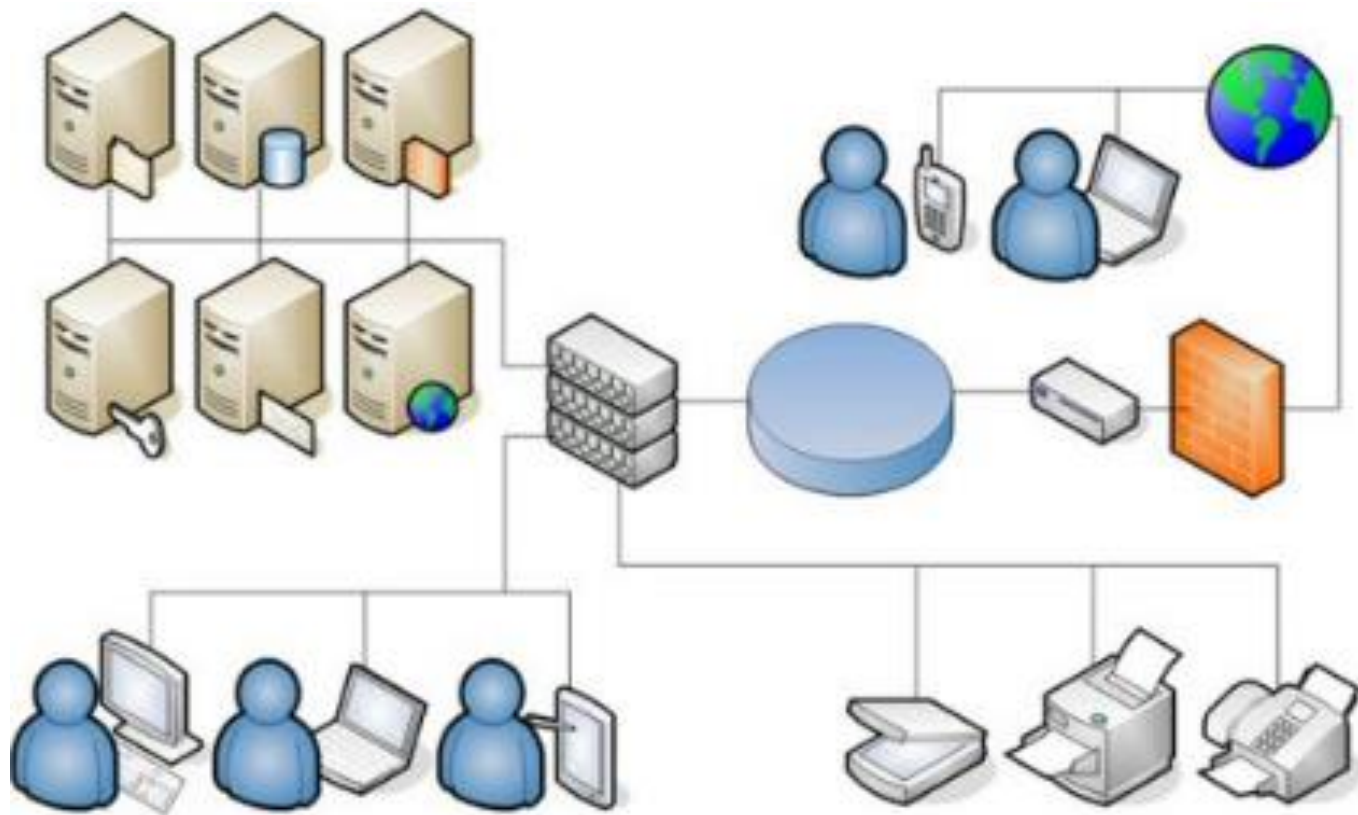
The information system of a school is an example of *closed multi-class model*. Different closed classes of customers accounts for students, professors, and administrative staff. Resources represent disks, PCs, web portals and so on.





Multi-class models

An example of *mixed multi-class model* is the information system of company that hosts both the intranet (used by the employees - a closed class) and the web server that handles external requests (an open class).





Analytical solutions

Analytical solutions are available for multi-class queuing networks, but extra assumptions with respect to single class models are required.

Processor sharing and LCFS with preemptive resume queuing disciplines can be analyzed without additional requirements.

FCFS, although being the most natural queuing policy, it cannot be generally considered for analytical solution of multi-class models: it can be used only if all the classes have the same average service time. Different classes, however, can be characterized by a different number of visits: this allows the models to have different demands at the various resources for the different classes.



Analytical solutions

Multi-class models are parameterized by:

- λ_c - Arrival rate for class c (open classes)
- N_c - Population size for class c (closed classes)
- D_{kc} - Demand for class c at resource k
(note that demands also include think times in time sharing models)

If the full set of performance measures is required (i.e. station throughput and station average response time), at least one of the following parameters is also needed:

- S_{kc} - Average service time for class c at resource k
- V_{kc} - Visits to resource k for class c
- p_{ikc} - Routing probability for class c jobs to join resource k after finishing their service at resource i .



Analytical solutions

Visits (both in open and closed systems) should be computed *per class*, each involving the solution of a specific set of traffic equations.

Here we have $\lambda_c = \sum_k \lambda_{IN[k],c}$.

Closed classes might also have a different reference station $ref(c)$.

$$\left\{ \begin{array}{l} v_{kc} = \frac{\lambda_{IN[k],c}}{\lambda_c} + \sum_{i=1}^K v_{ic} \cdot p_{ikc} \\ \dots \end{array} \right. \quad \text{if } c \text{ is Open}$$

$$\left\{ \begin{array}{l} v_{kc} = \sum_{i=1}^K v_{ic} \cdot p_{ikc} \\ v_{ref(c)} = 1 \end{array} \right. \quad \forall k \neq ref(c) \quad \text{if } c \text{ is Closed}$$



Analytical solutions

Connection between global throughput, local throughput, service demand, average service time, response time, residence time and visits, for a given class c , remains the same as in single class models.

$$D_{kc} = v_{kc} \cdot S_{kc}$$

$$R_{kc} = v_{kc} \cdot \Phi_{kc}$$

$$X_{kc} = v_{kc} \cdot X_c = v_{kc} \cdot \lambda_c \text{ (for open classes)}$$



Analytical solutions

Utilization law and *Little's law* are valid for each combination of class c and resource k .

$$U_{kc} = X_{kc} \cdot S_{kc} = X_c \cdot D_{kc}$$

$$N_{kc} = X_{kc} \cdot \Phi_{kc} = X_c \cdot R_{kc}$$



Analytical solutions

Summing up the utilization, the throughput and the average number of jobs for all classes at a station k , we can compute the corresponding station-wise metrics.

$$U_k = \sum_c U_{kc}$$

$$X_k = \sum_c X_{kc}$$

$$N_k = \sum_c N_{kc}$$



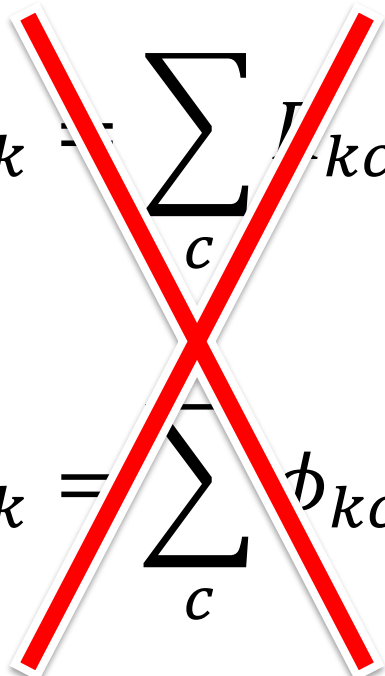
Little's law is also valid for the entire station k .

$$N_k = X_k \cdot \Phi_k = X \cdot R_k$$



Analytical solutions

Residence and response times for a resource k require extra care since they do not simply sum up.


$$R_k = \sum_c J_{kc}$$
$$\phi_k = \sum_c \phi_{kc}$$



Analytical solutions

To obtain the correct expression, we must apply Little's law for the entire resource k , and per class c .

$$\begin{aligned} N_{kc} &= X_{kc} \cdot \Phi_{kc} = X_c \cdot R_{kc} \\ N_k &= X_k \cdot \Phi_k = X \cdot R_k \end{aligned} \quad N_k = \sum_c N_{kc}$$

$$R_k = \frac{N_k}{X} = \frac{\sum_c N_{kc}}{X} = \sum_c \frac{X_c}{X} R_{kc}$$

$$\phi_k = \frac{N_k}{X_k} = \frac{\sum_c N_{kc}}{X_k} = \sum_c \frac{X_{kc}}{X_k} \phi_{kc}$$



Analytical solutions

Per class c , system-wise performance indices can then be computed by summing up the corresponding metrics computed relevant to class c for each station k .

$$N_c = \sum_k N_{kc}$$

$$R_c = \sum_k R_{kc}$$

Note that Little's law is also valid per class c with the system-wise measures.

$$N_c = X_c \cdot R_c$$



Analytical solutions

Aggregate performance metrics (*system-wise, not class dependent*) can be then obtained by summing up the performance metrics per resource k or per class c .

$$X = \sum_c X_c$$

$$N = \sum_k N_k = \sum_c \sum_k N_{kc} = \sum_c N_c$$

$$R = \sum_k R_k = \sum_c \sum_k \frac{X_c}{X} R_{kc} = \sum_c \frac{X_c}{X} R_c$$



Open models: stability condition

The stability condition for open models with multi-class workloads, accounts for the different arrival rates of the different classes:

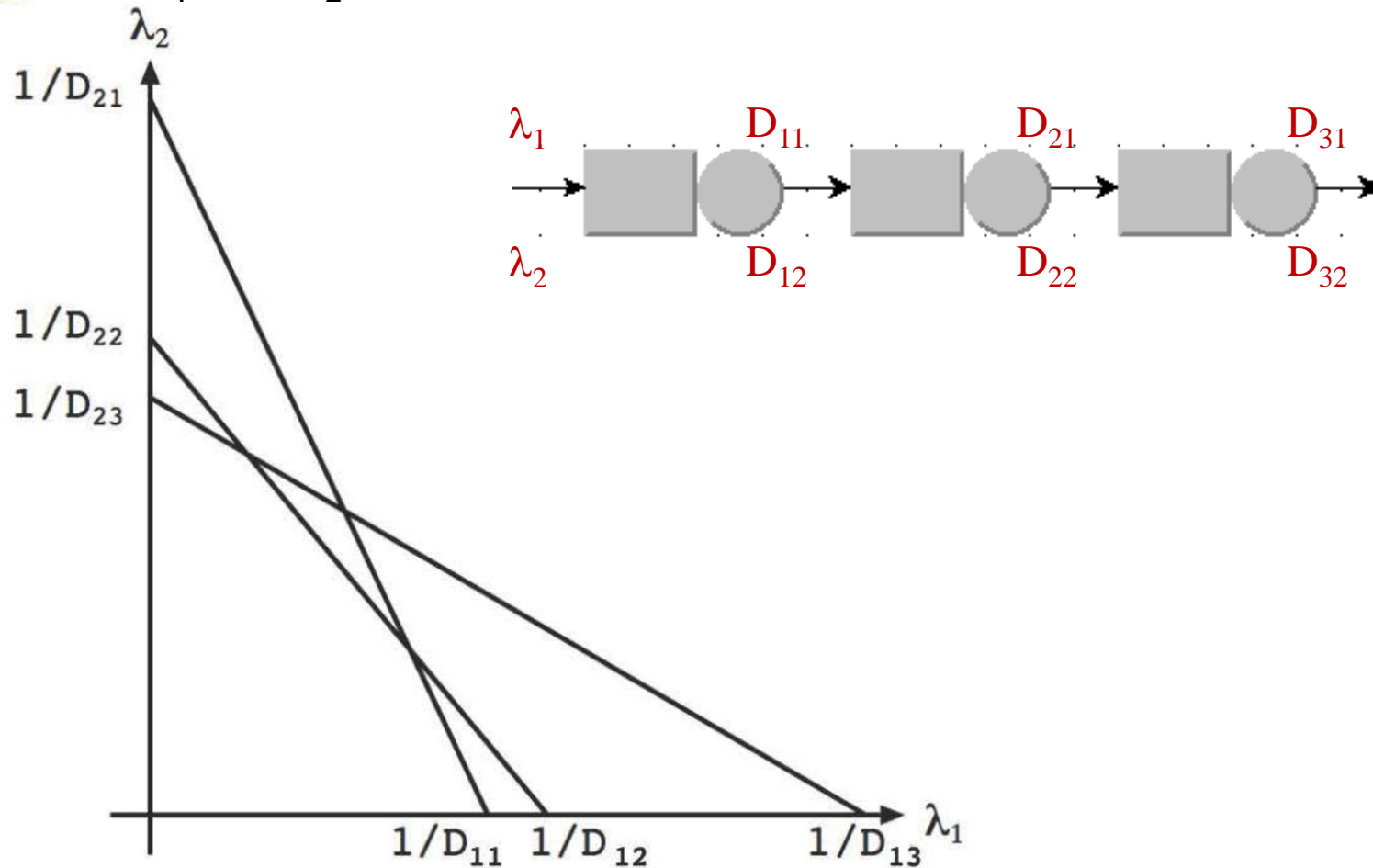
$$\max_k U_k = \max_k \left\{ \sum_c U_{kc} \right\} = \max_k \left\{ \sum_c X_c \cdot D_{kc} \right\} < 1$$

$$\max_k \left\{ \sum_c \lambda_c \cdot D_{kc} \right\} < 1$$



Open models: stability condition

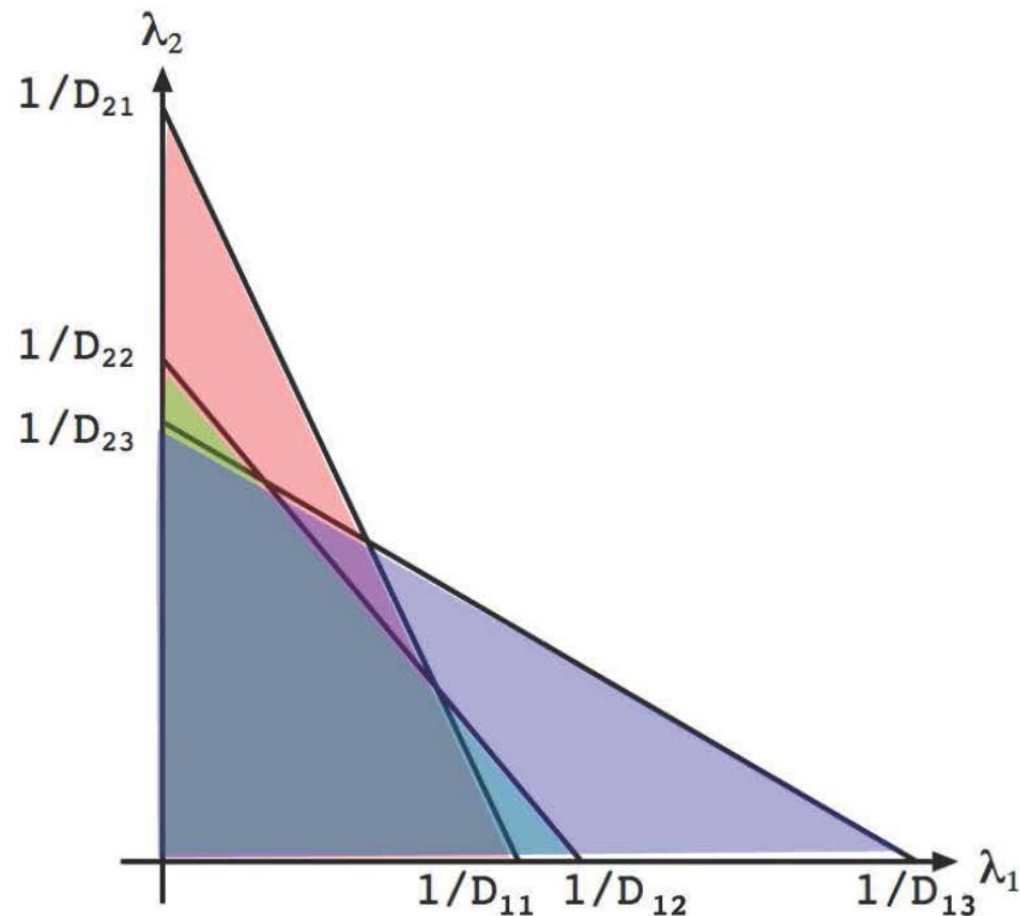
Note that the arrival rates for the various classes that satisfies the previous constraint correspond to different regions of a plane on the λ_1 and λ_2 axes.





Open models: stability condition

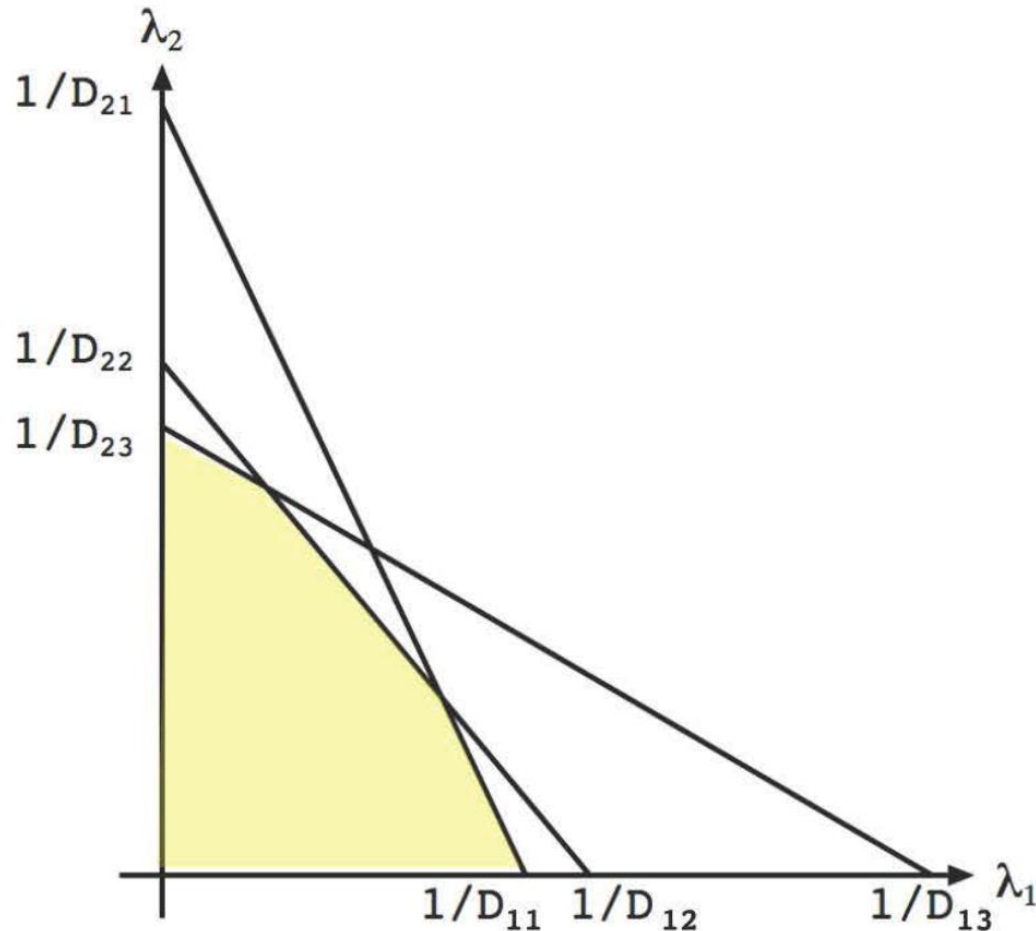
Each resource determines a different area on the plane.





Open models: stability condition

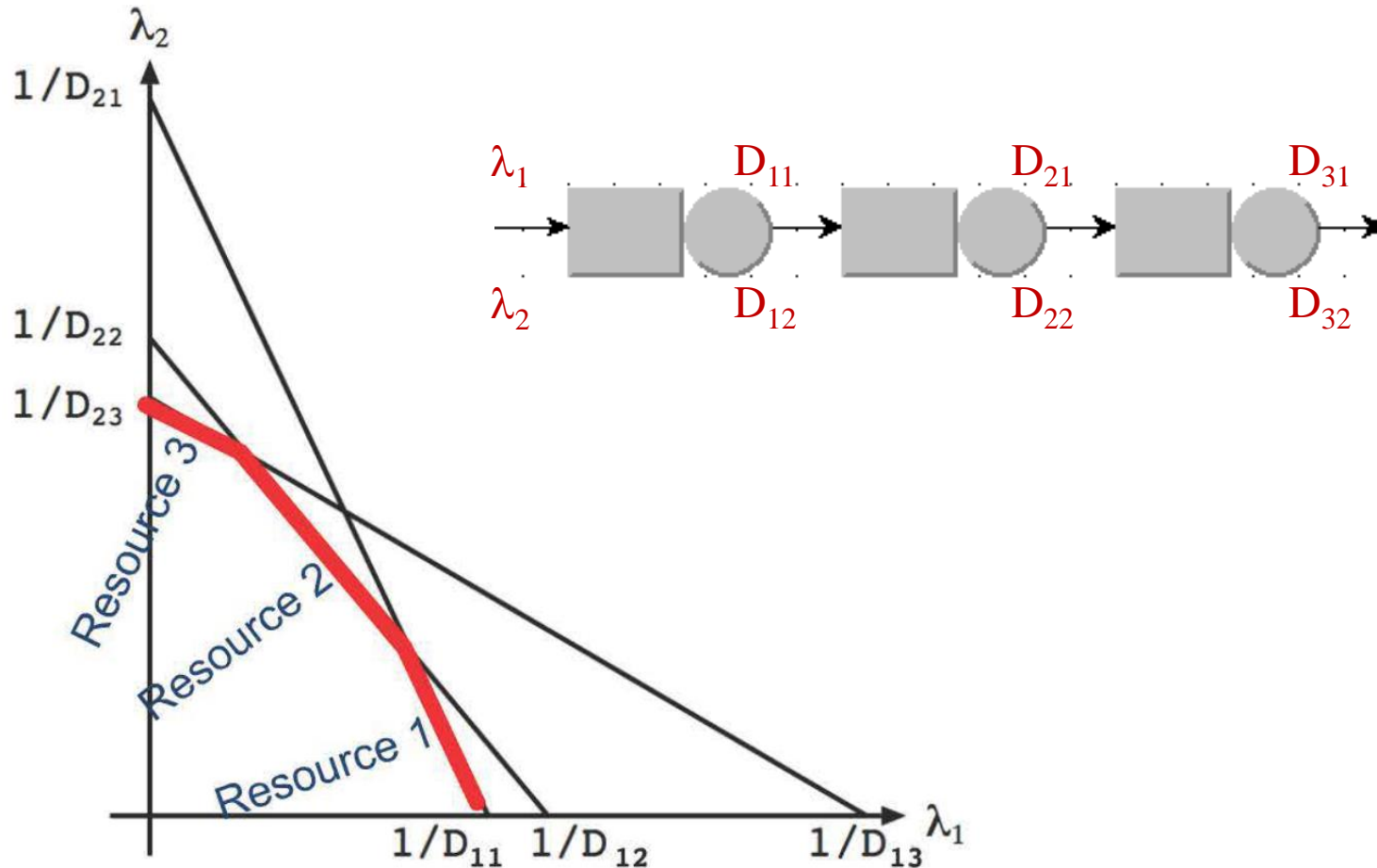
Only values for the arrival rates λ_1 and λ_2 that satisfy all the constraints are valid.





Open models: stability condition

The borders of the regions identify a different bottleneck for the system.





Analysis of Motivating Example

With a more detailed analysis, the system administrator has found that there are two very different types of requests, characterized by the following service times and arrival rates:

$$\lambda_A = 18 \text{ req/s}, \lambda_B = 2 \text{ req/s}, D_A = 10 \text{ ms}, D_B = 160 \text{ ms}.$$
$$U = 18 \cdot 0.01 + 2 \cdot 0.16 = 0.5$$

The increase in traffic experienced, if split among the two classes, is the following:

$$\lambda_A = 18 \text{ req/s}, \lambda_B = 6 \text{ req/s}$$

This clearly violates the stability conditions, since:

$$U = 18 \cdot 0.01 + 6 \cdot 0.16 = 1.14$$