Politecnico di Milano
Dip. Elettronica e Informazione
Milan, Italy

Imperial College London

Quantitative System Evaluation with Java Modelling Tools

Giuliano Casale
Imperial College London
g.casale@imperial.ac.uk

Giuseppe Serazzi
Politecnico di Milano
giuseppe.serazzi@polimi.it

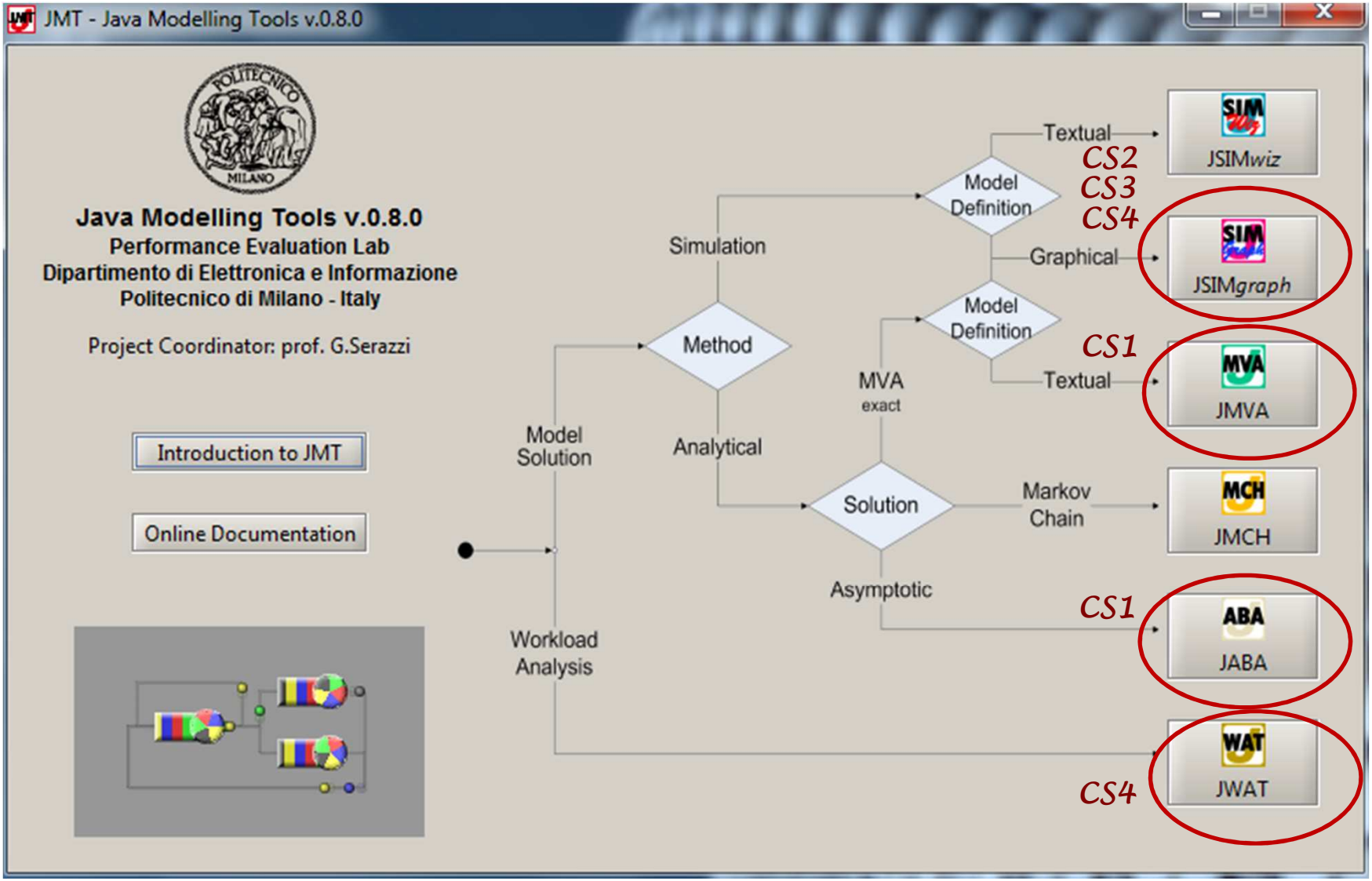
Tutorial – ICPE 2011

tutorial outline

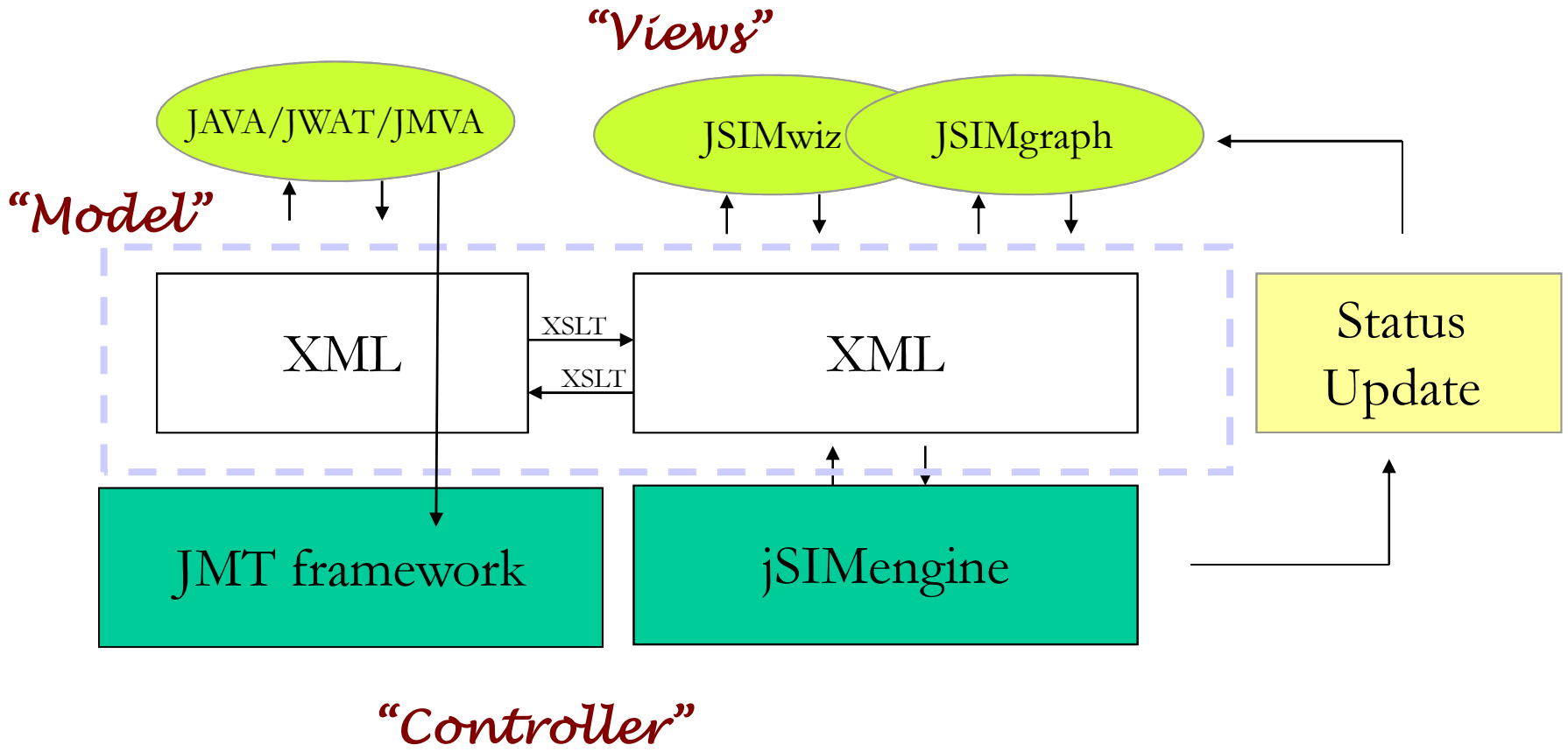
- overview of Java Modelling Tools (http://jmt.sf.net)

- case study 1 (CS1): bottlenecks identification, performance evaluation, optimal load
- case study 2 (CS2): model with multiple exit paths
- case study 3 (CS3): resource contention
- case study 4 (CS4): multi-tier applications, web services

Java Modelling Tools (http://jmt.sf.net)



architecture



software development

- JMT is open source, Java code and ANT build scripts at http://jmt.sourceforge.net/Download.html
- size: ~4,000 classes; 21MB code; 174,805 lines
- subversion
  svn co https://jmt.svn.sourceforge.net/svnroot/jmt jmt
- source tree

trunk (root also for help, examples, license information, ...)
- src
  - jmt
    - analytical (jMVA algorithms)
    - commandline (command line wrappers)
    - common (shared utilities)
    - engine (main algorithms & data structures)
    - framework (misc utilities)
    - gui (graphical user interfaces)
    - jmarkov (JMCH)
    - test (application testing)

core algorithms - jMVA

Mean Value Analysis (MVA) algorithm (e.g., [Lazowska et al., 1984])

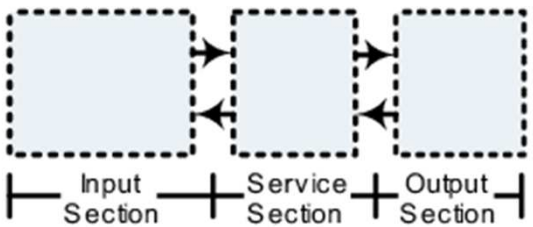
- fast solution of product-form queueing networks
- open models: efficient solution in all cases
- closed models: efficient for models with up to 4-5 classes

Product-form queueing networks solvable by MVA

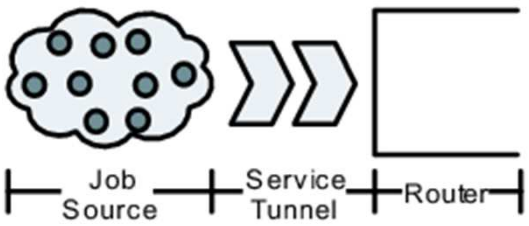
- PS/FCFS/LCFS/IS scheduling
- Identical mean service times for multiclass FCFS
- Mixed models (open + closed), load-dependent
- Service at a queue does not depend on state of other queues
- No blocking, finite buffers, priorities
- Some theoretical extensions exist, not implemented in jMVA

core algorithms – jSIMengine: simulation

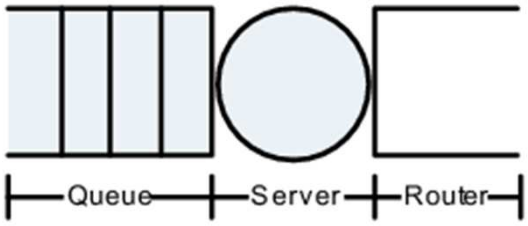
- components in the simulation are defined by 3 sections



component sections

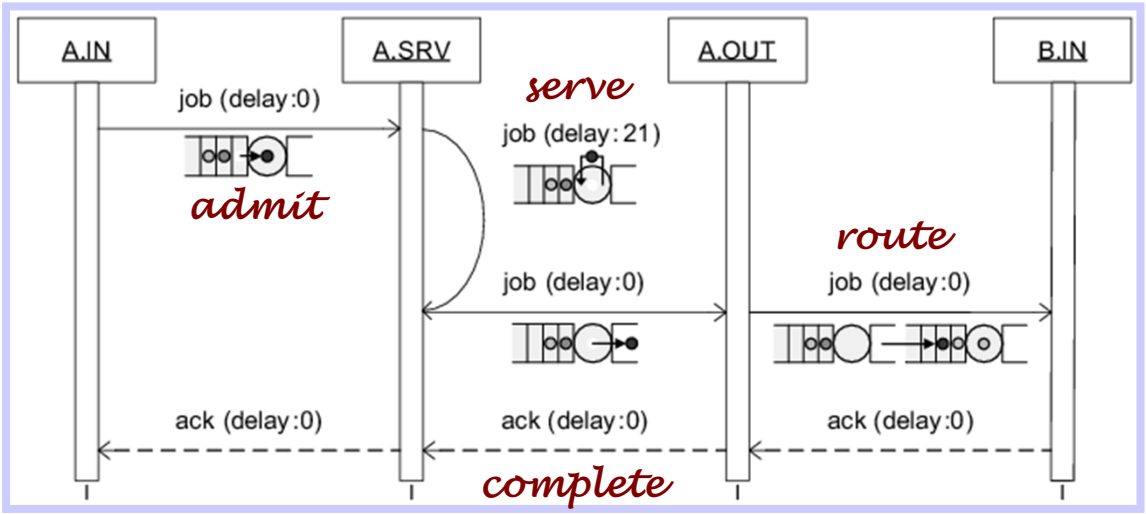


external arrivals (open class)



queueing station

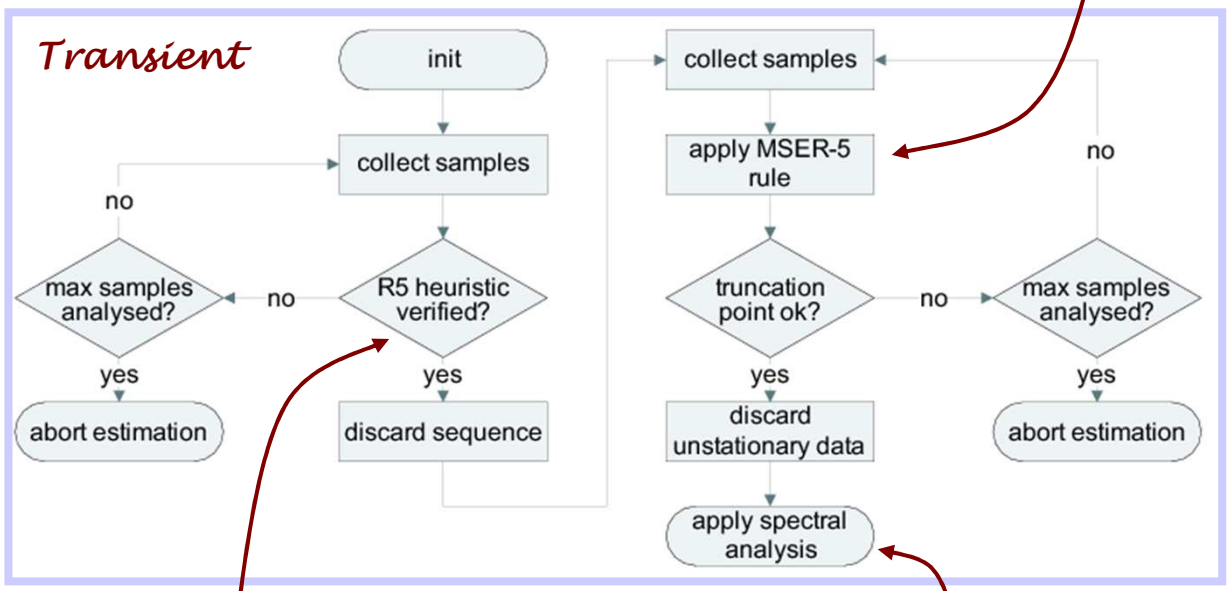
- discrete-event simulation engine



core algorithms – jSIMengine: statistical analysis

- transient filtering flowchart

[Spratt, *M.S. Thesis*, 1998]



[Pawlikowski, *CSUR*, 1990]

[Heidelberger&Welch, *CACM*, 1981]

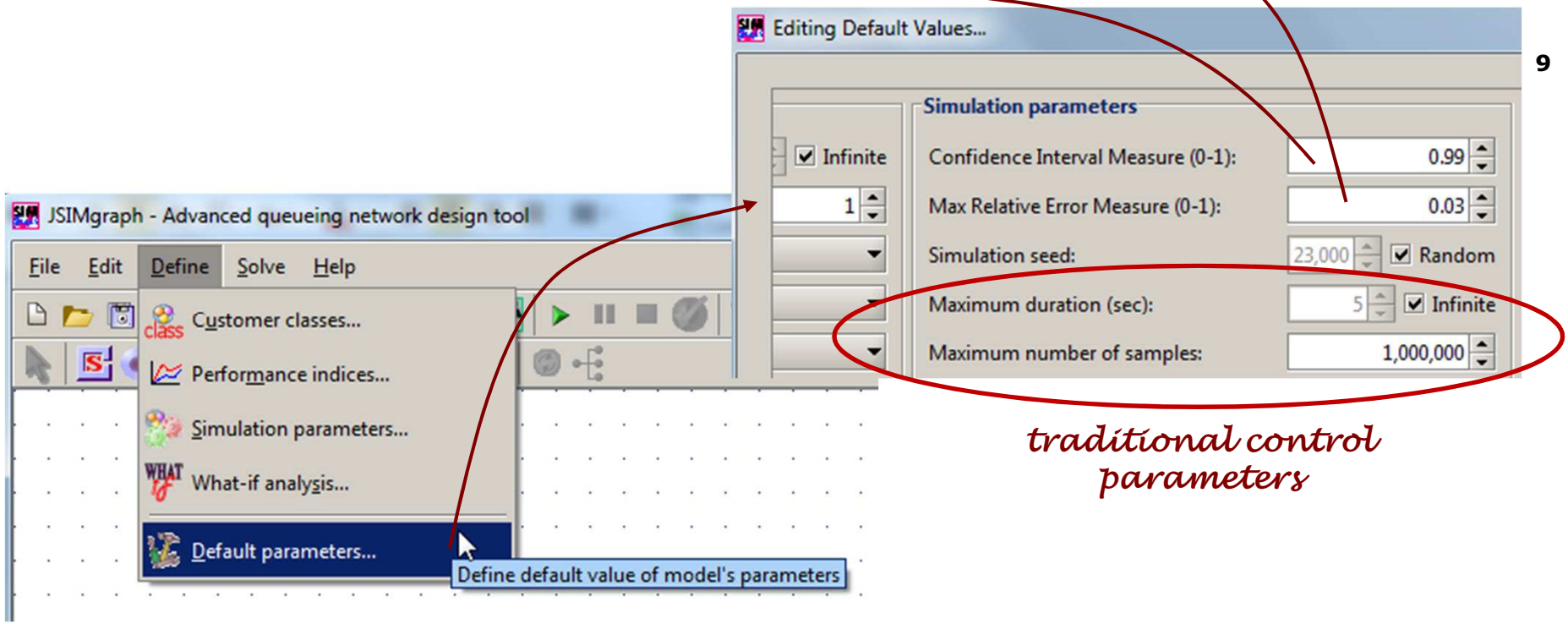
core algorithms – jSIMengine: simulation stop

- simulation stops automatically

$$1-\alpha \leq P[\text{relative error} \leq \frac{\varepsilon}{1-\varepsilon}]$$

maximum relative error

confidence level



traditional control parameters

Politecnico di Milano
Dip. Elettronica e Informazione
Milan, Italy

Imperial College London

CASE STUDY 1:

Bottlenecks identification
Performance evaluation
Optimal load

closed model
multiclass workload

JABA + JMVA

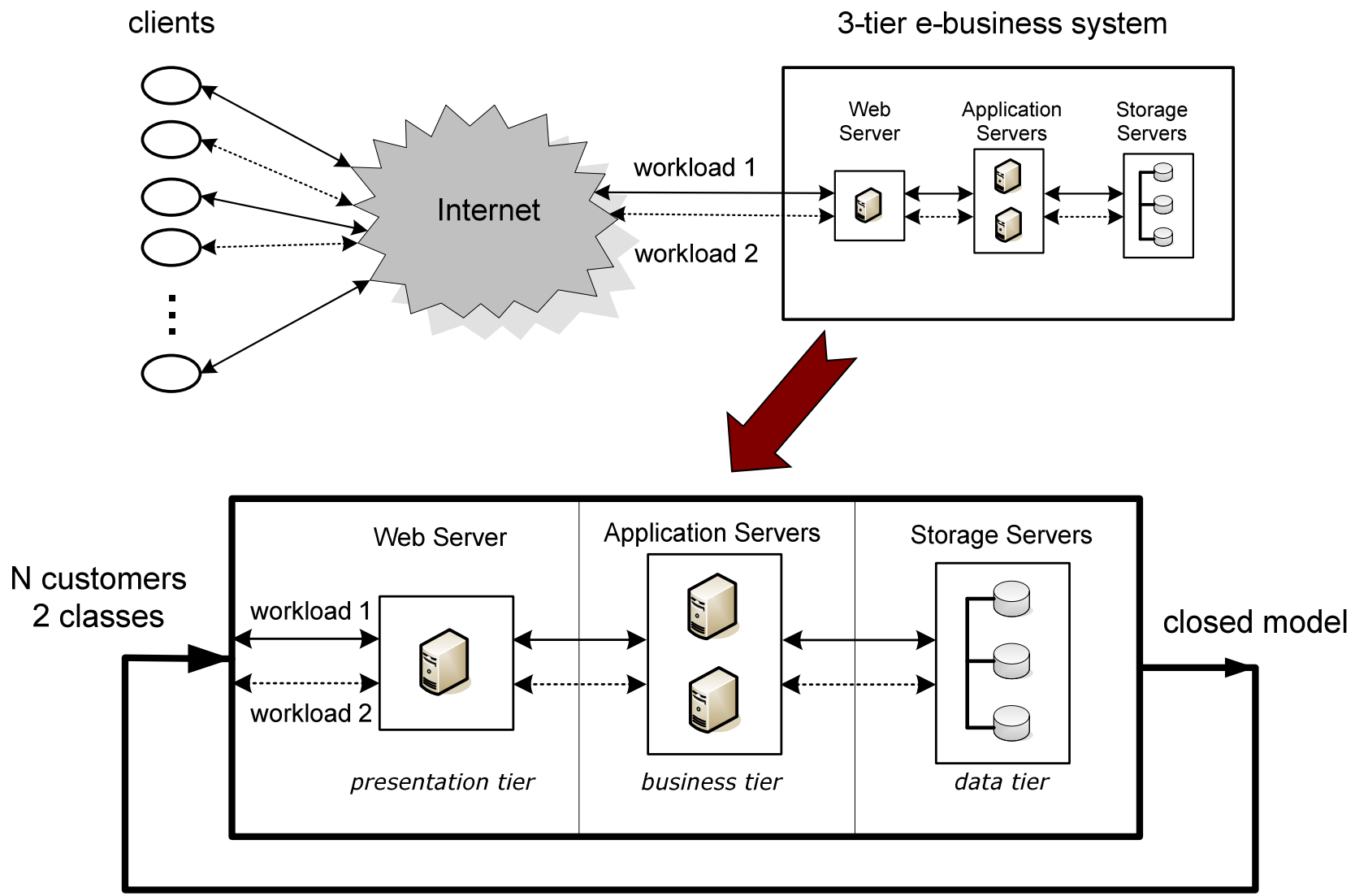
Outline

- objectives
- system topology
- bottlenecks detection and common saturation sectors
- performance evaluation
- optimal loading

characteristics of the system

- e-business services: a variety of activities, among them information retrieval and display, data processing and updating (mainly data intensive) are the most important ones
- two classes of requests with different resource loads and performance requirements
- presentation tier: light load (less demanding than that of the other two tiers)
- application tier: business logic computations
- data tier: store and fetch DB data (search, upload, download)
- to reduce the number of parameters (and to simplify obtaining their values) we have choosen to parameterize the model in term of global loads L_i, i.e., service demands D_i

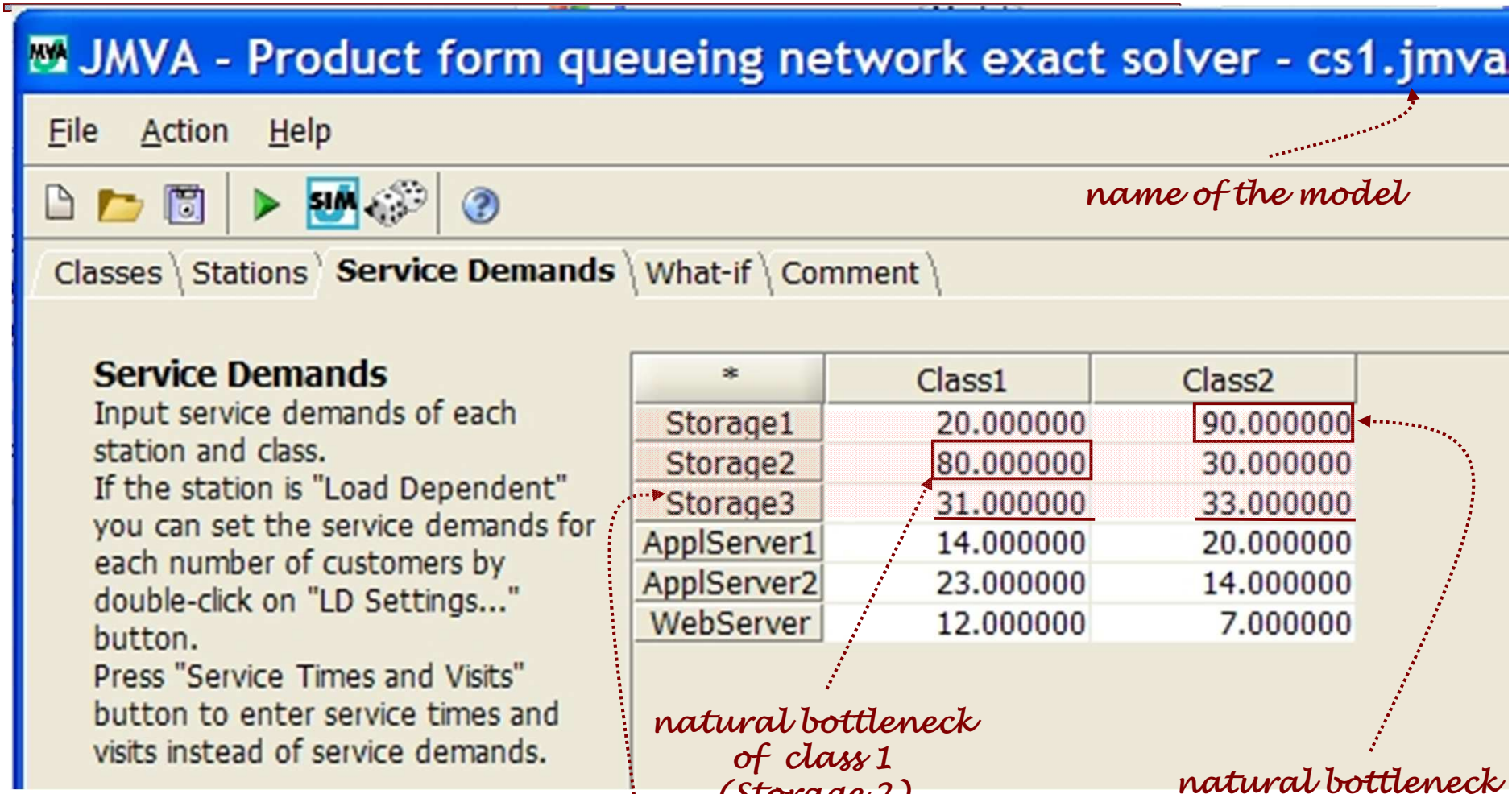
topology of a 3-tier enterprise system



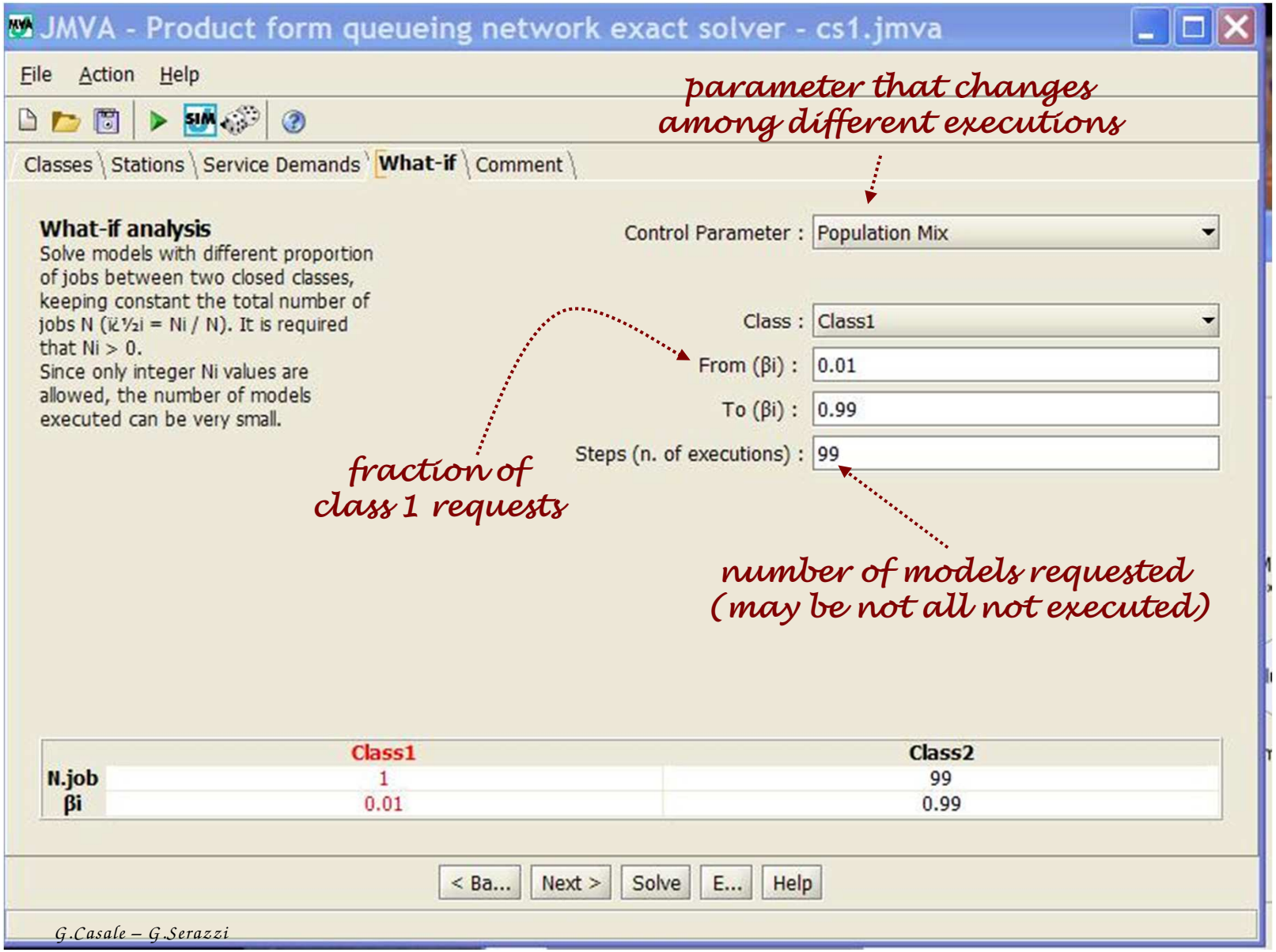
workload parameters

- resource **L**oadings matrix: Service **D**emands, i resources, r classes $D_{ir} = V_{ir} * S_{ir}$
- global number of customers: N=100
- system population: $\underline{N}=\{N_1,N_2\}$ $\{1,99\}\rightarrow\{99,1\}$
- population mix: $\underline{\beta}=\{\beta_1,\beta_2\}$, fraction of jobs per class,
- $\underline{\beta}$ variable: study of the optimal load (optimal mix)
- asymptotic behavior: $\underline{\beta}$ constant, N increasing

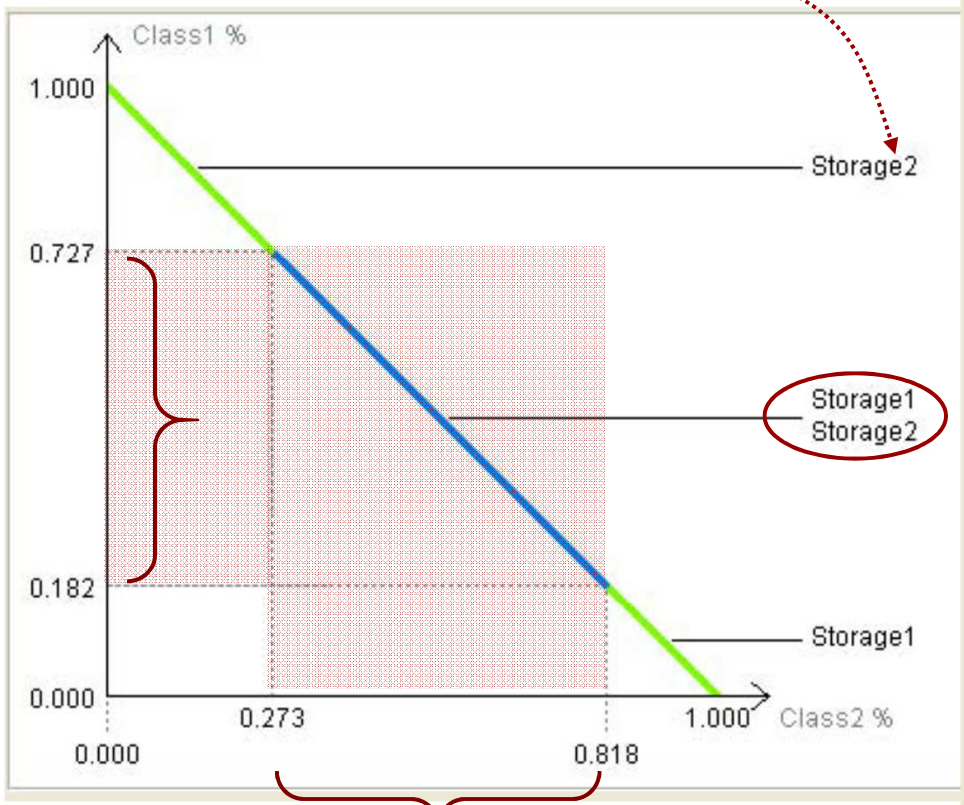
Service Demands (resource Loadings)



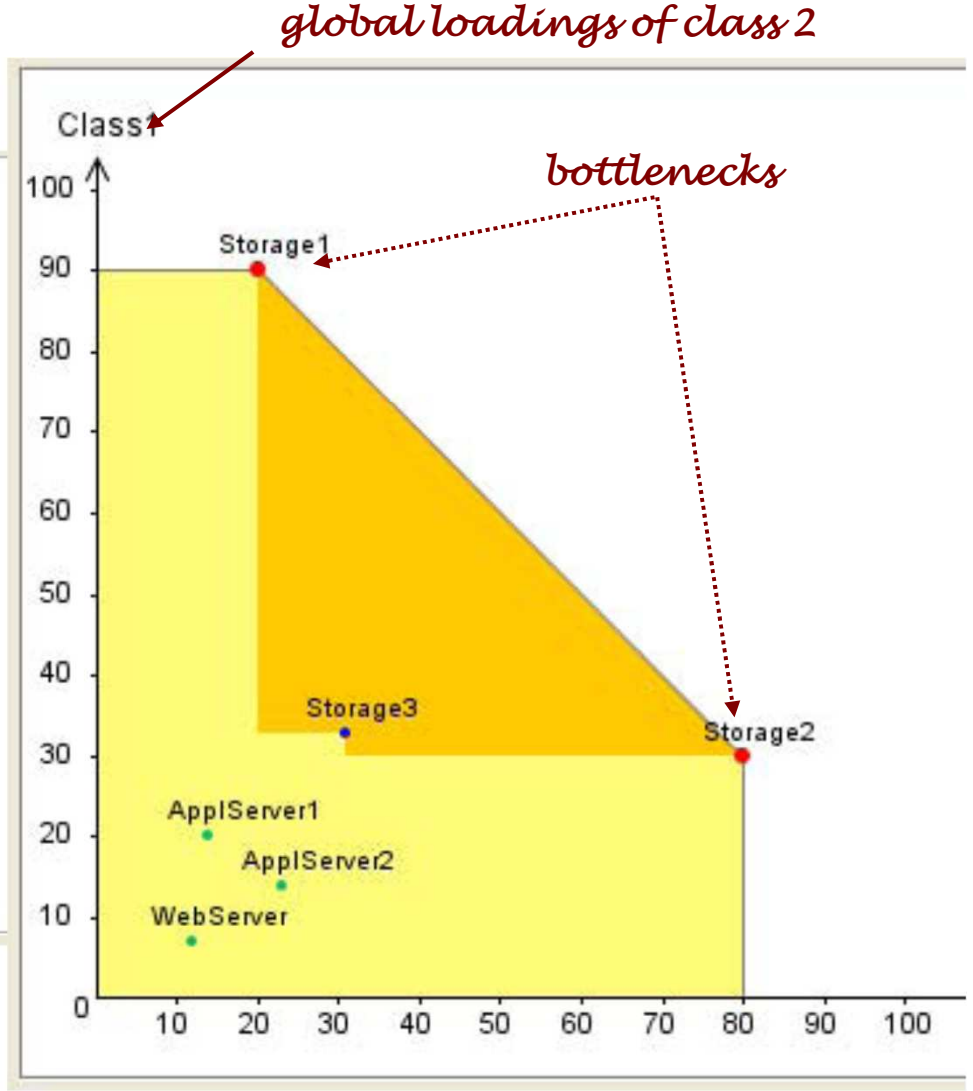
What-if analysis (JMVA with multiple executions)



Bottlenecks switching (JABA asymptotic analysis)

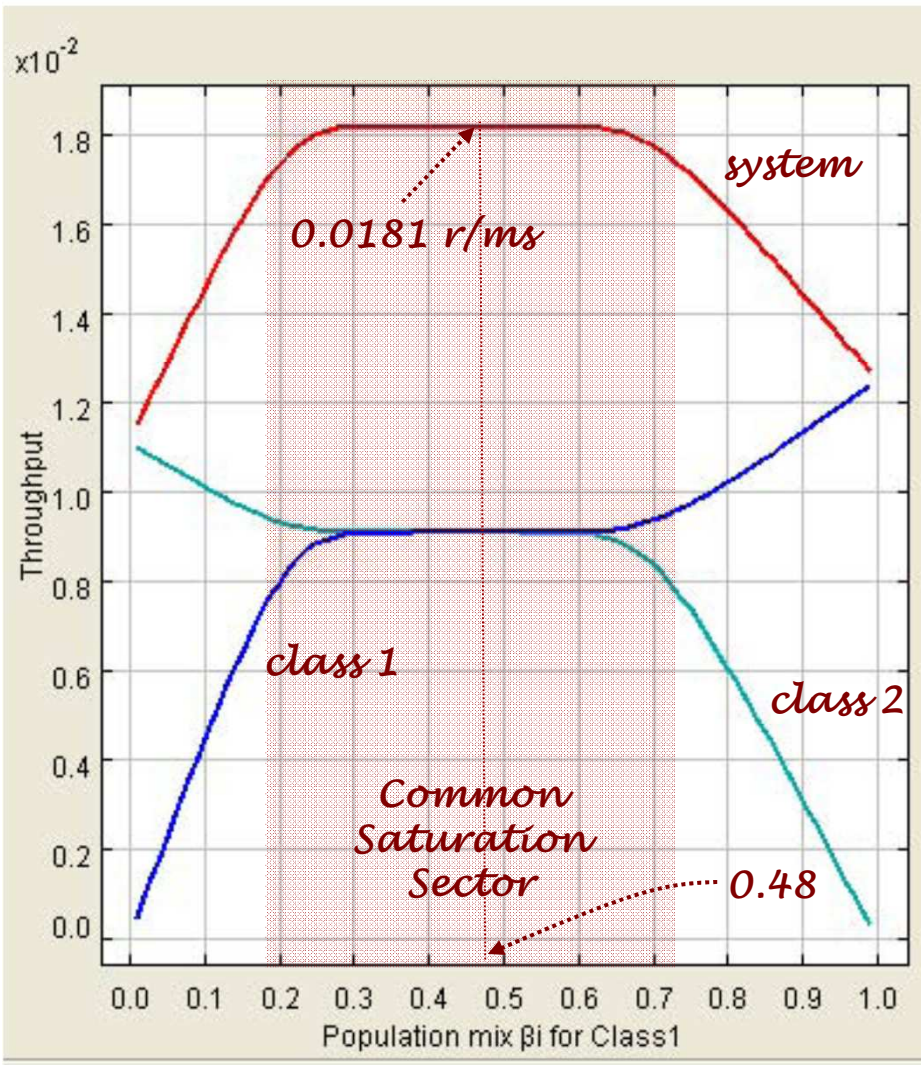


fraction of class 2 jobs that saturate two resources concurrently (Common Saturation Sector)

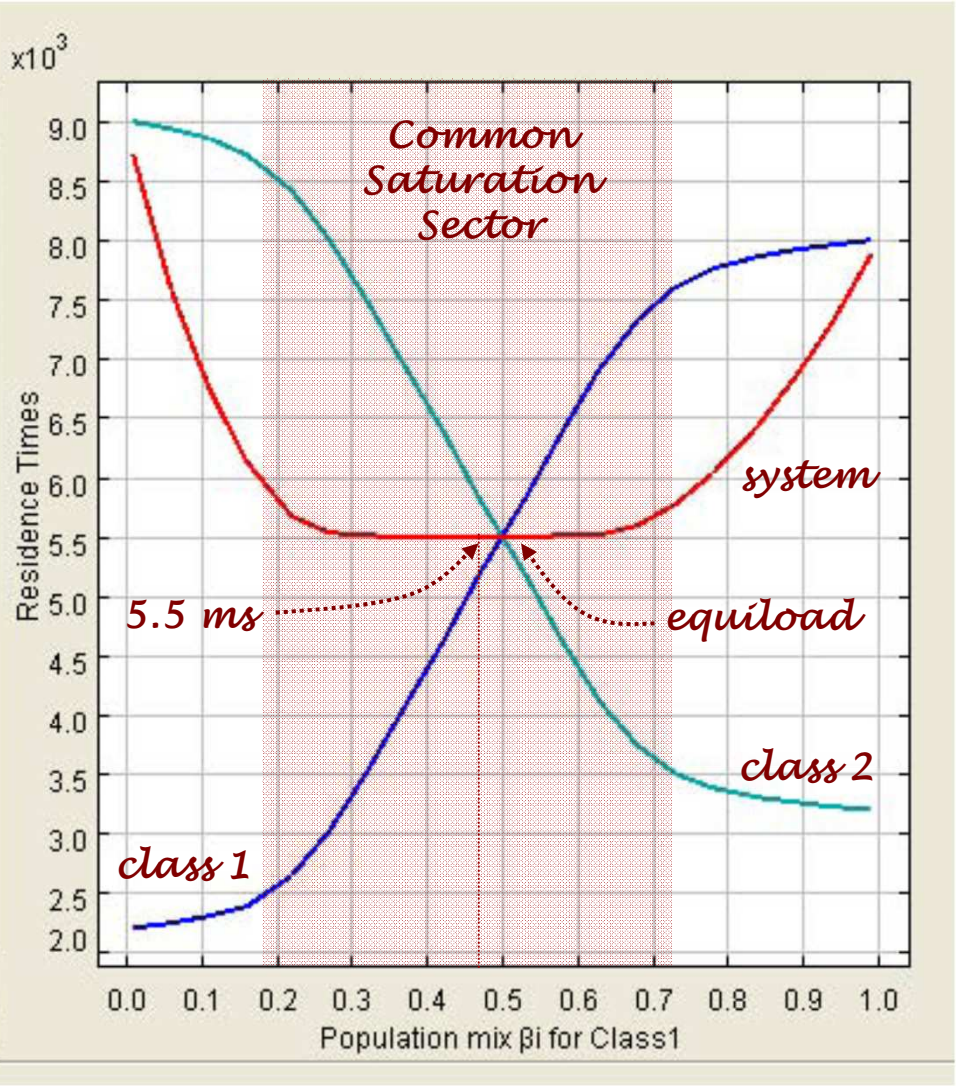


global loadings of class 1

throughput and Response time {N=1,99}-{99,1}, JMVA

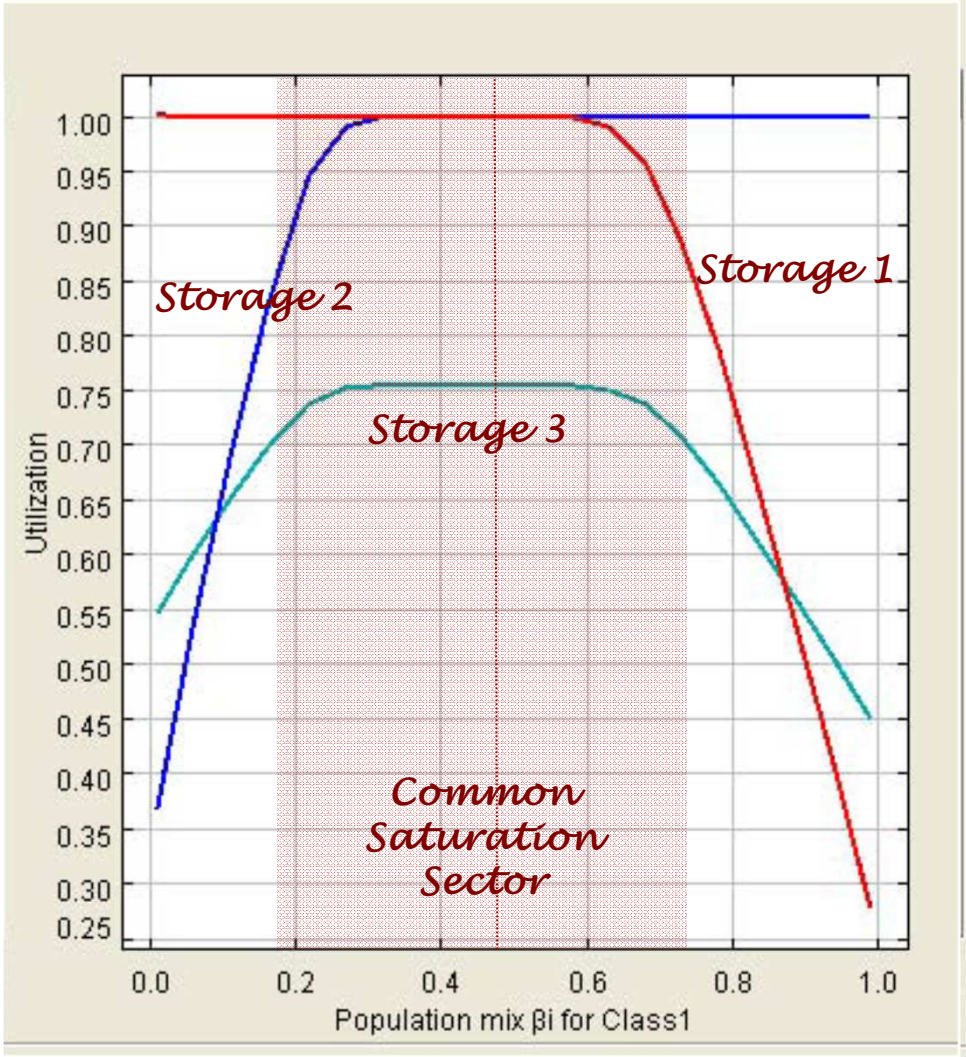


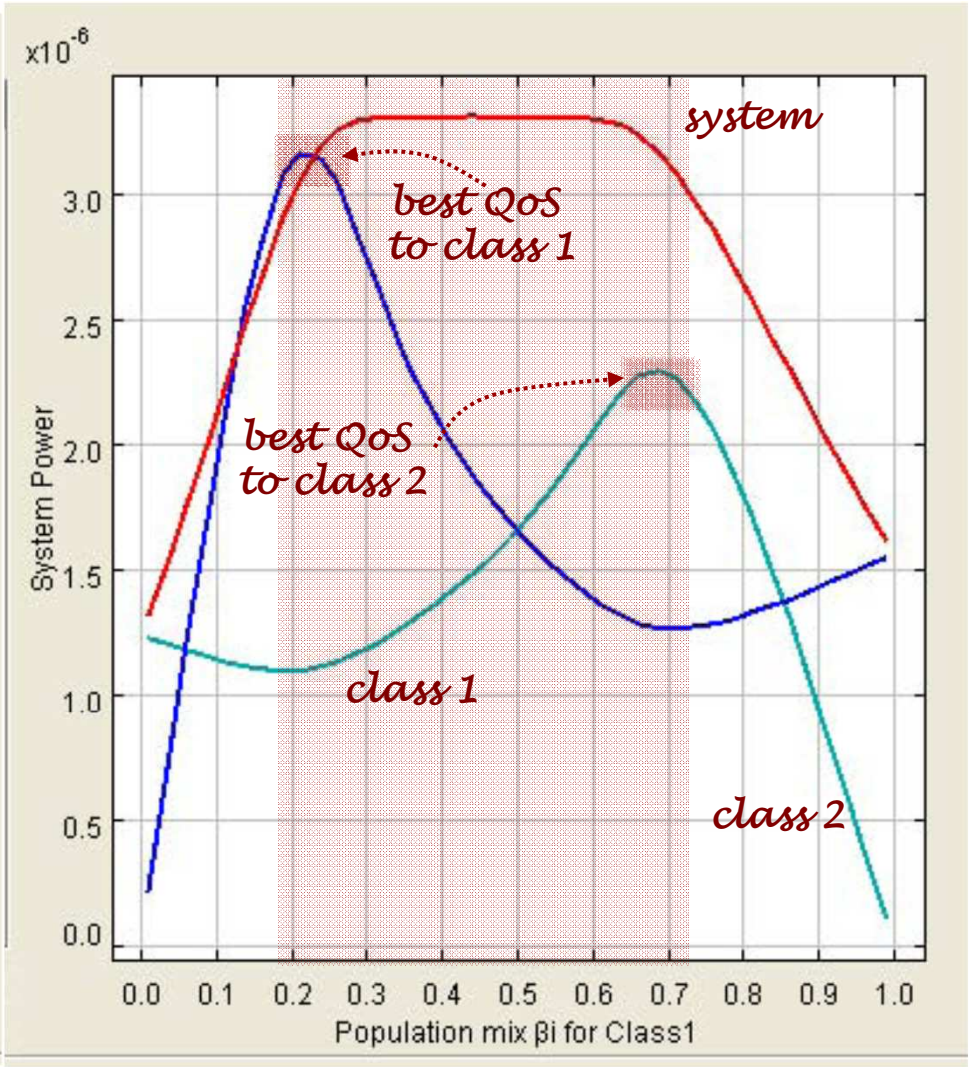
throughput X



Response times

Utilizations and Power {N=1,99}–{99,1}



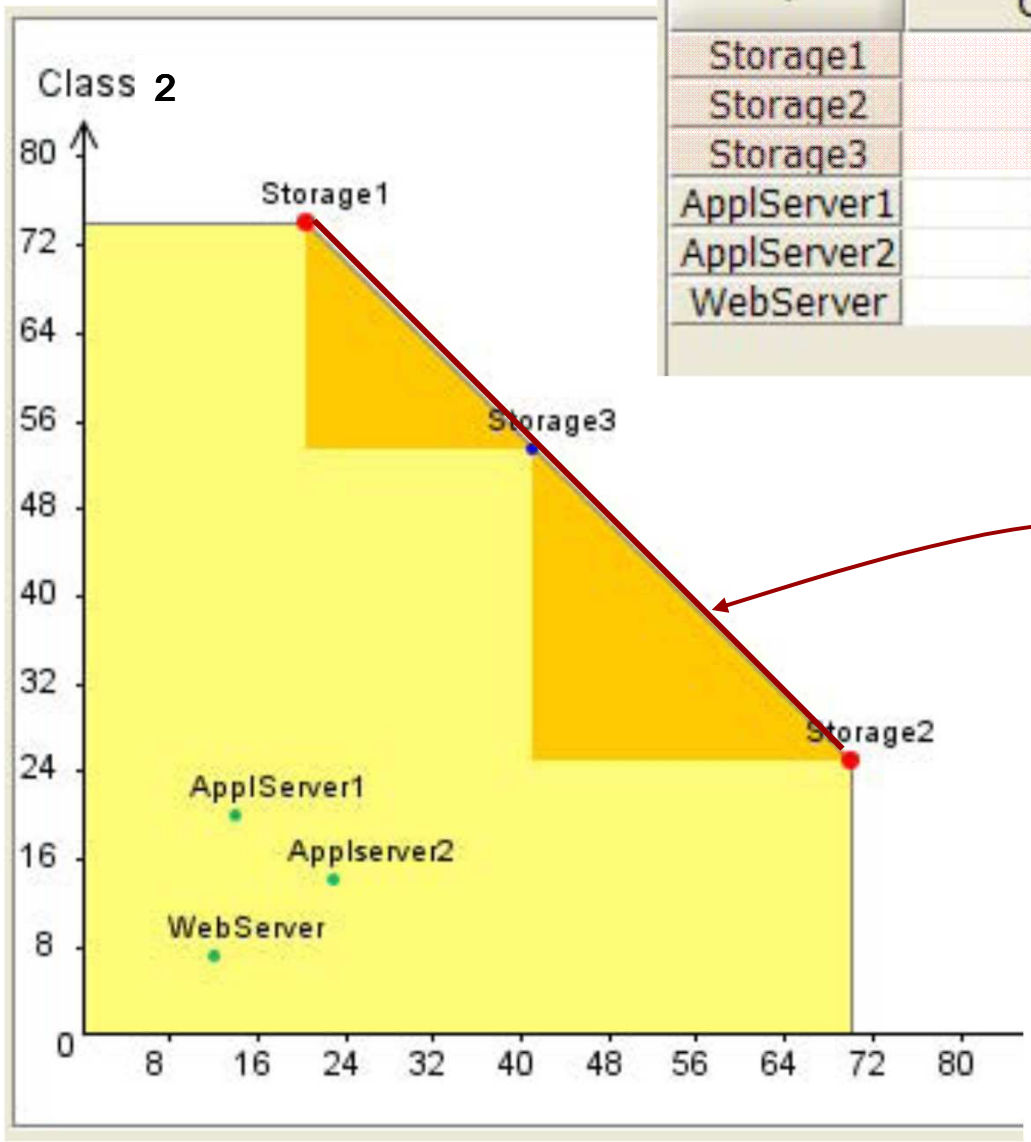


Utilizations

Power (X/R)

optimized load: service demands and bottlenecks

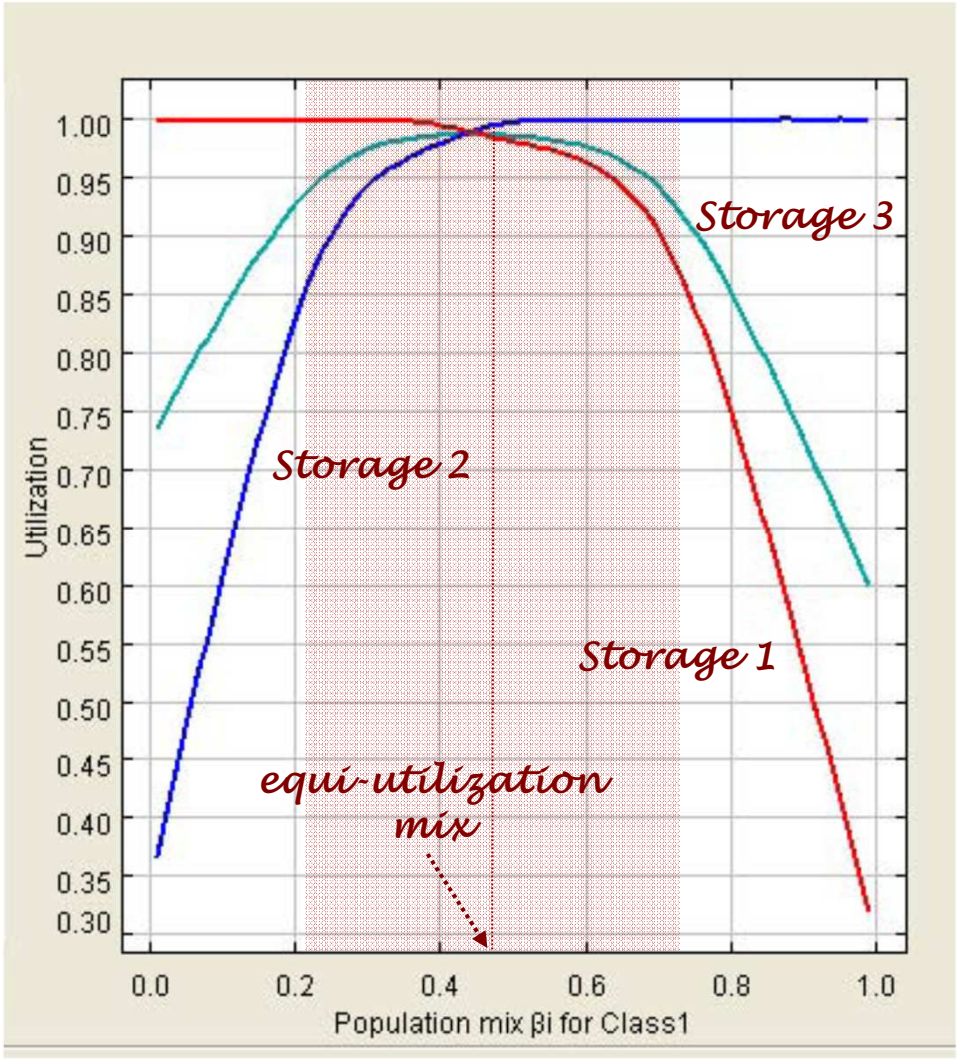
*	Class1	Class2	
Storage1	20.500000	74.000000	94.5
Storage2	70.000000	25.000000	95
Storage3	41.000000	53.500000	94.5
ApplServer1	14.000000	20.000000	
ApplServer2	23.000000	14.000000	
WebServer	12.000000	7.000000	



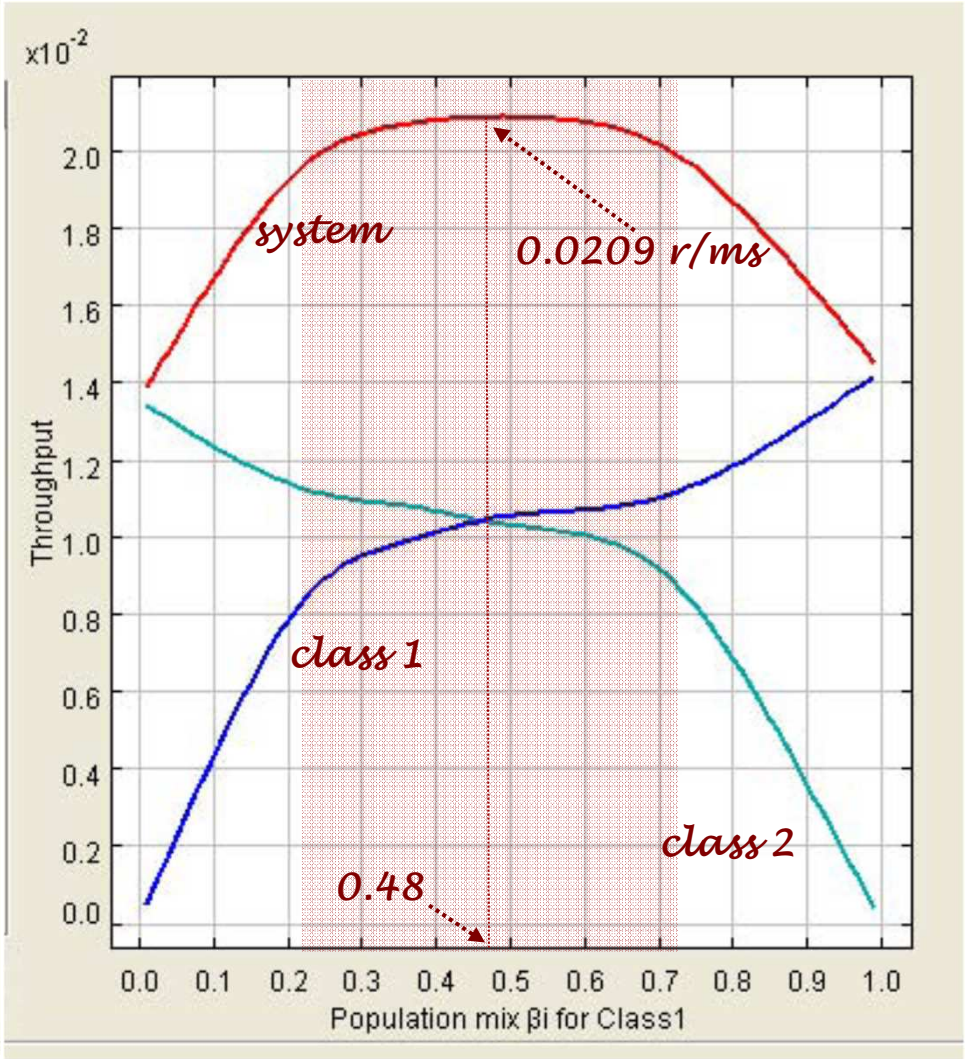
multiple bottlenecks
equi-utilization line

Class 1

optimized load: U and X

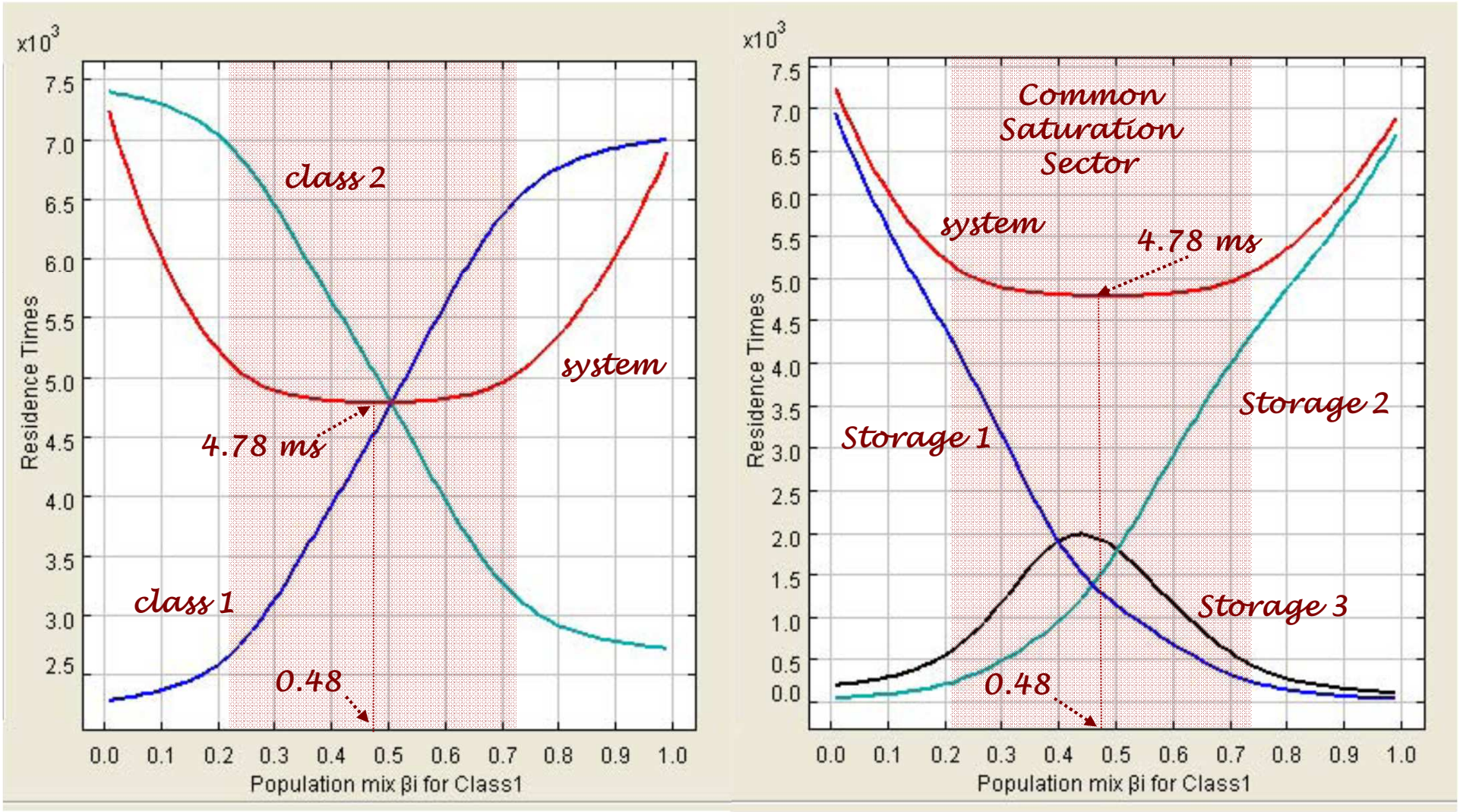


Utilizations



throughput X

optimized load: Response times and Residence times



Response times

Residence times

CASE STUDY 2: model with multiple exit paths

open model
single class workload
different routing policies

JSIMgraph

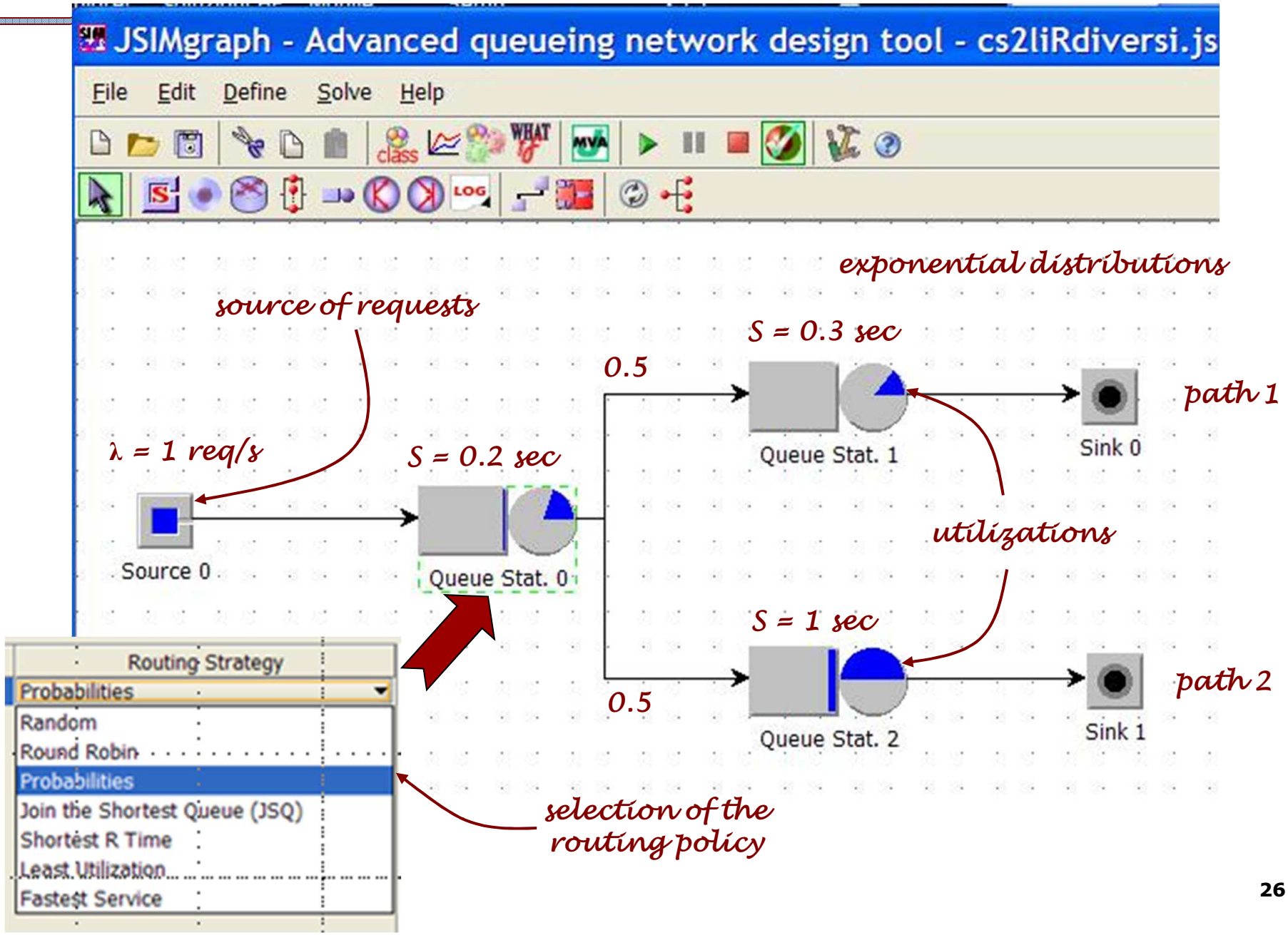
Outline

- objectives
- system topology
- what-if analysis
- performance with “probabilistic” routing
- performance with “least utilization” routing
- performance with “Joint the Shortest Queue” routing

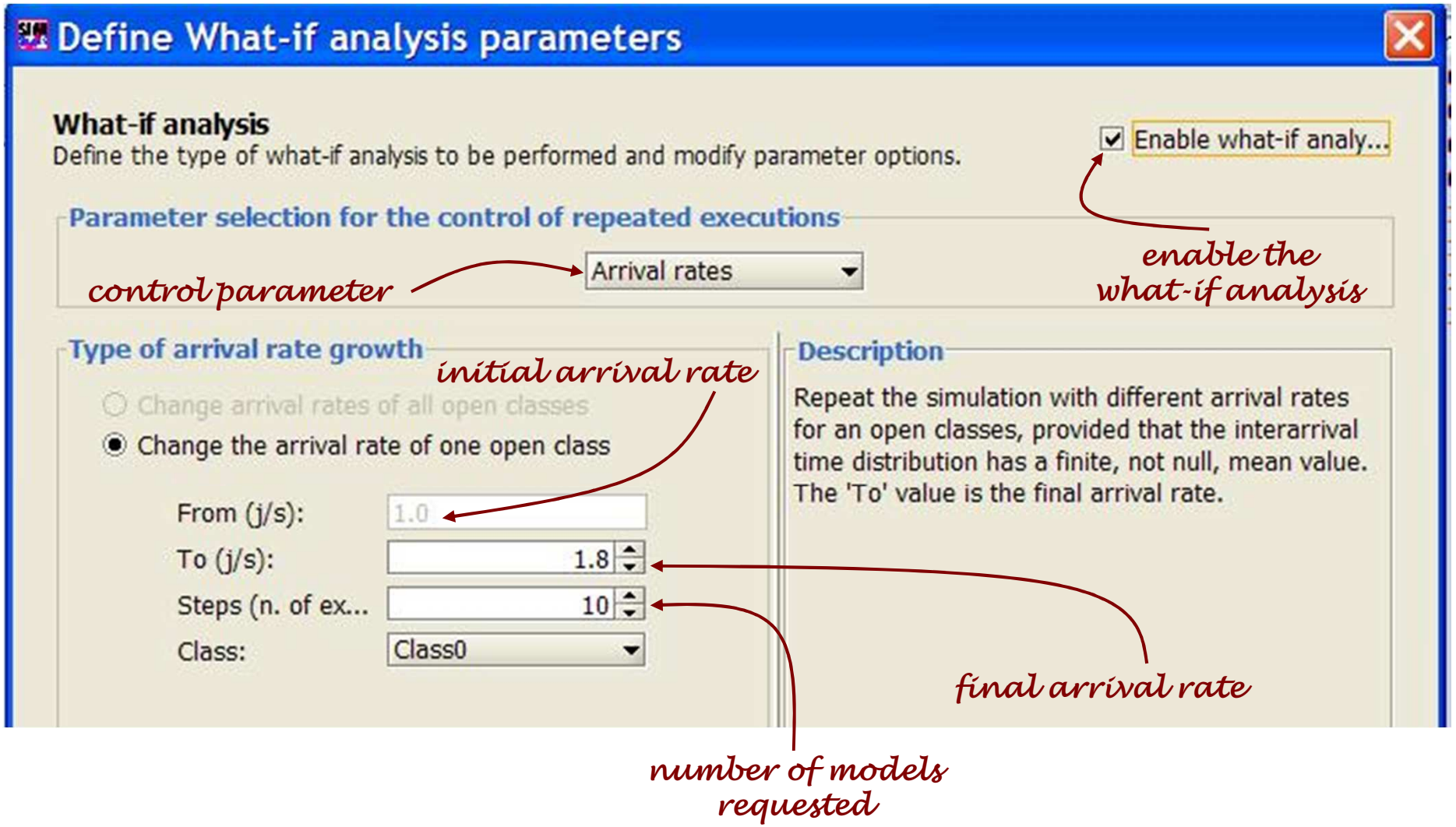
objectives

- fallacies in using the index system response time also in single class models
- open model with multiple exit paths (sinks), e.g., drops, alternative processing, multi-core, load balancing, clouds, ...
- differencies between response time per sink and system res ponse time
- impact on performance of different routing policies

system topology

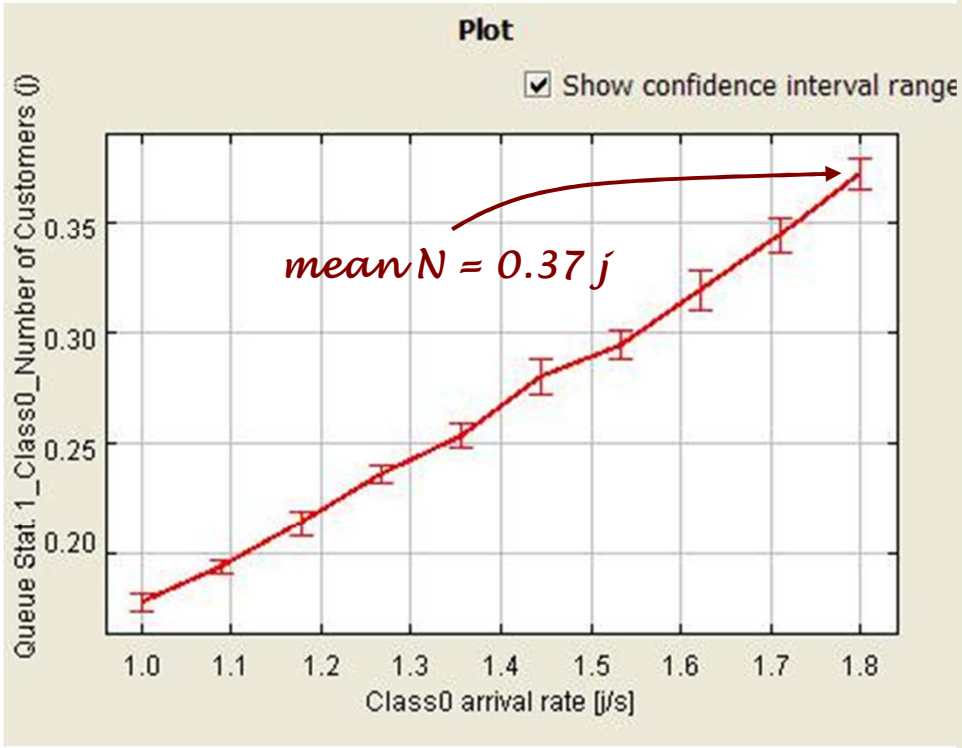


What-if analysis settings

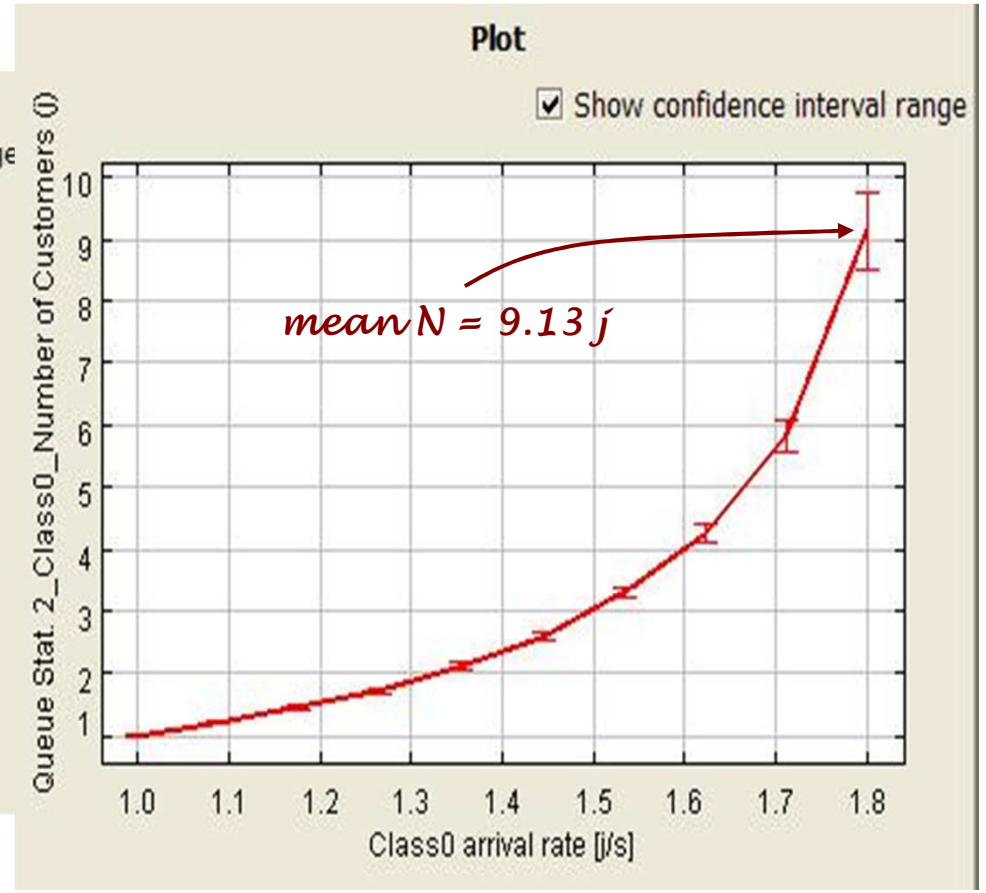


n. of customers N in the two paths (prob. routing)

path 1

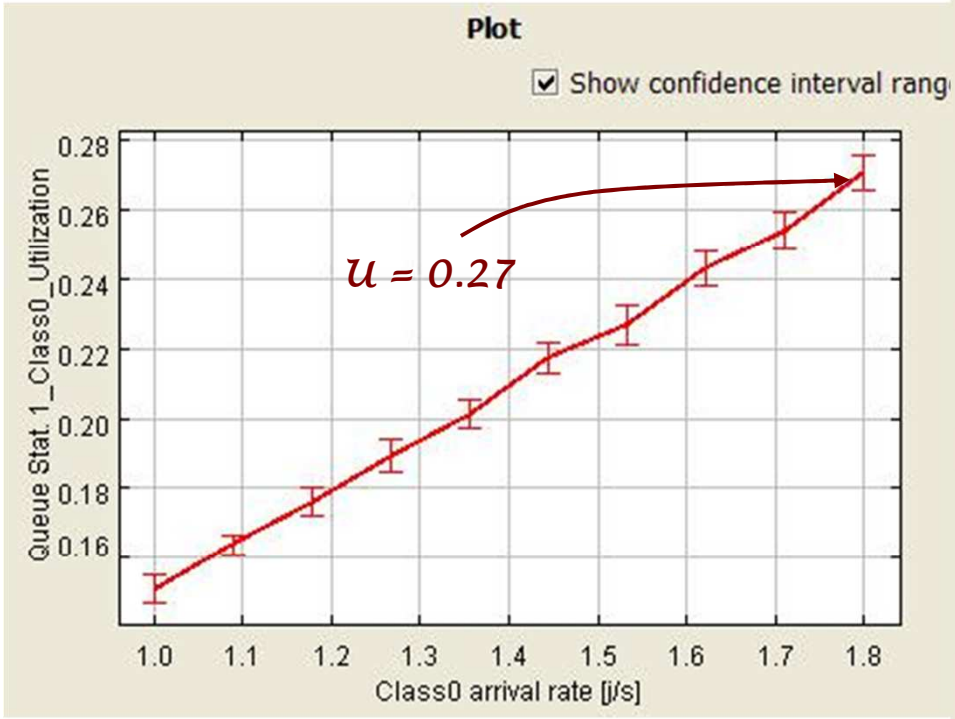


path 2

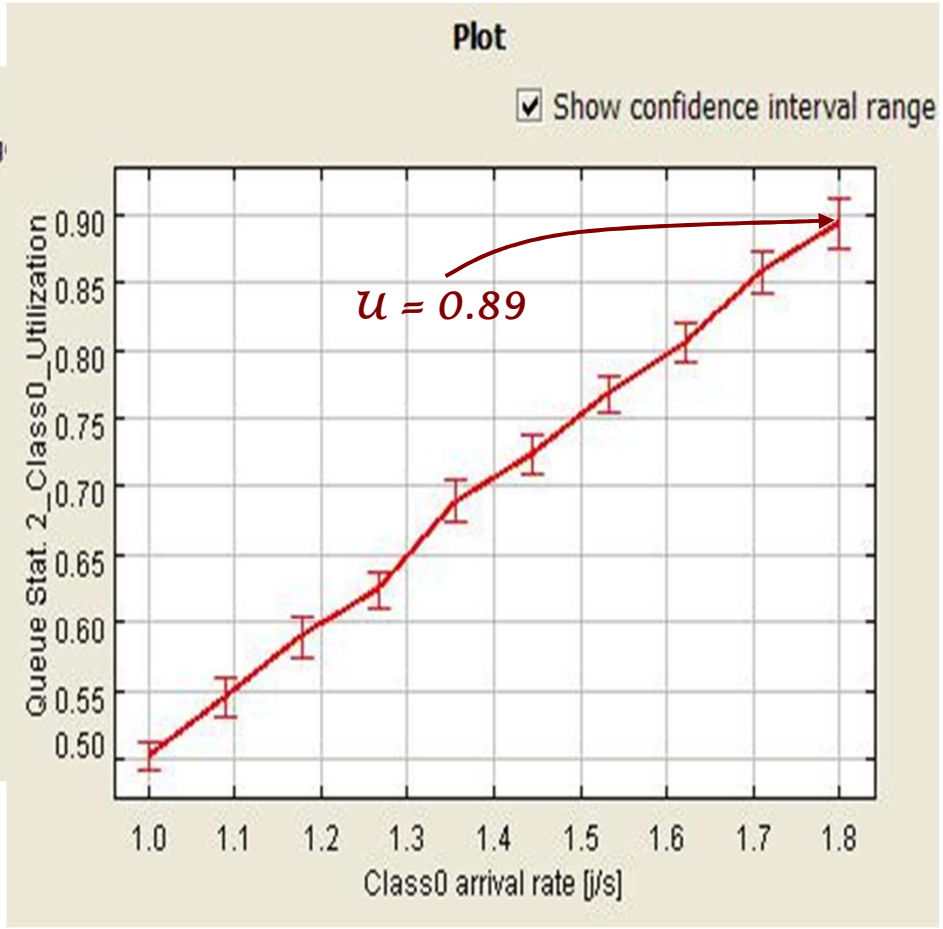


Utilizations (per path) with prob. routing

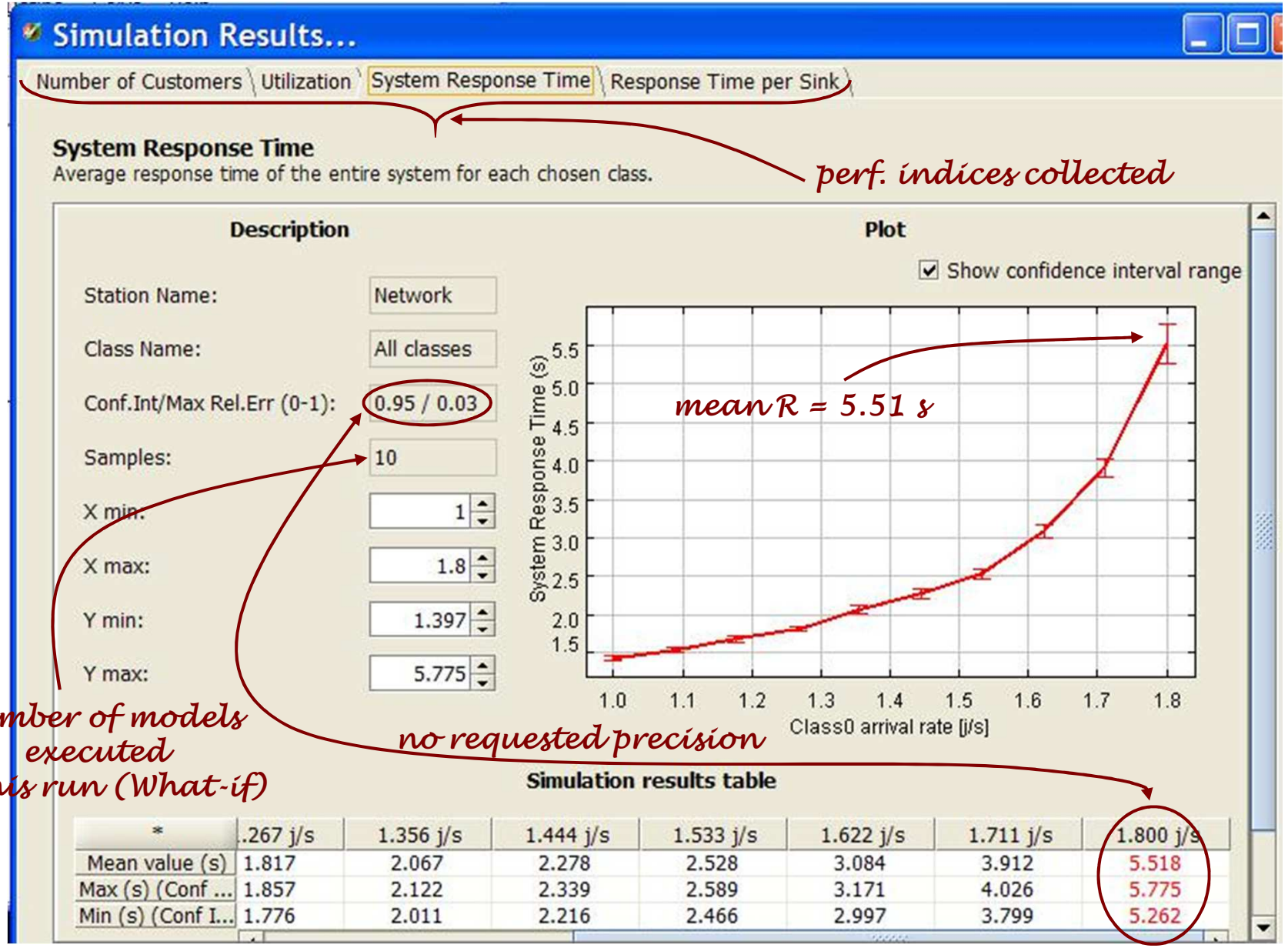
path 1



path 2

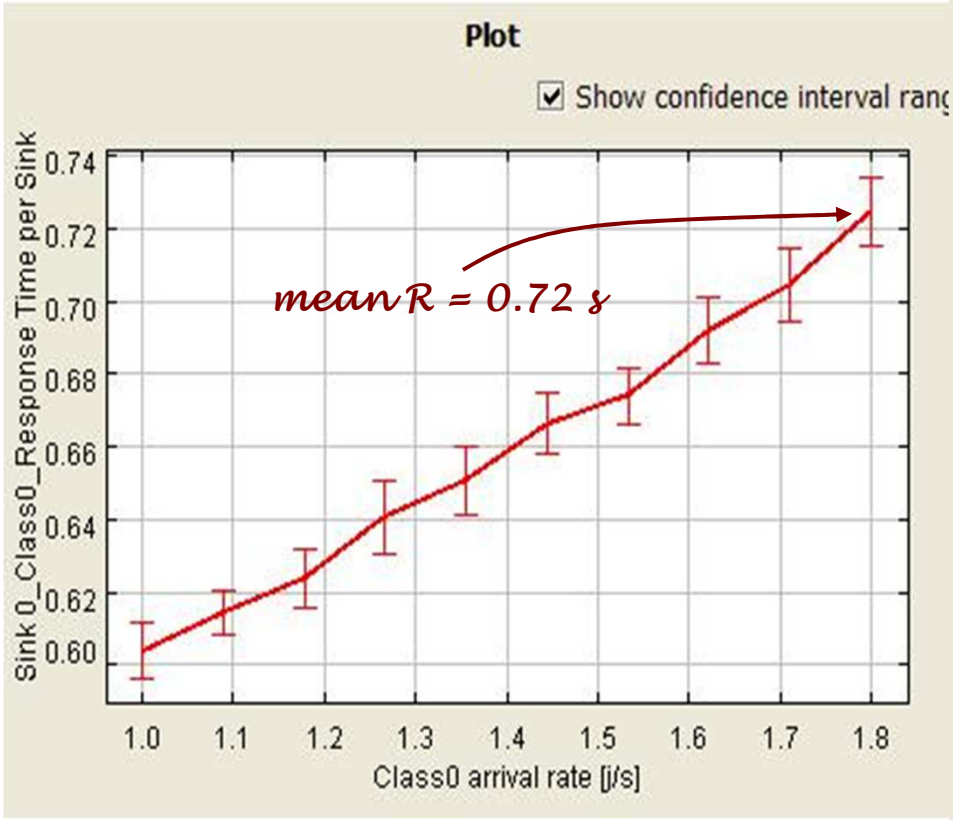


system Response time (prob. routing)

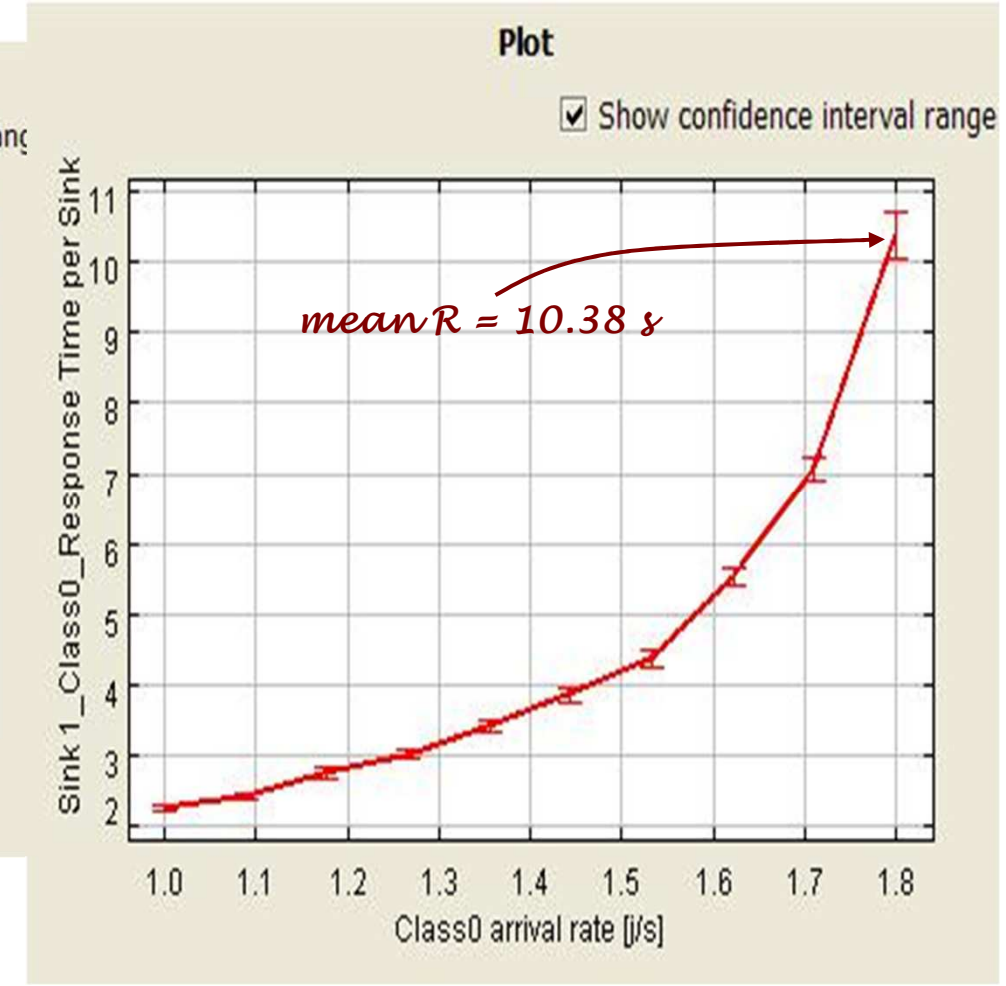


Response time per path (prob. routing)

path 1



path 2

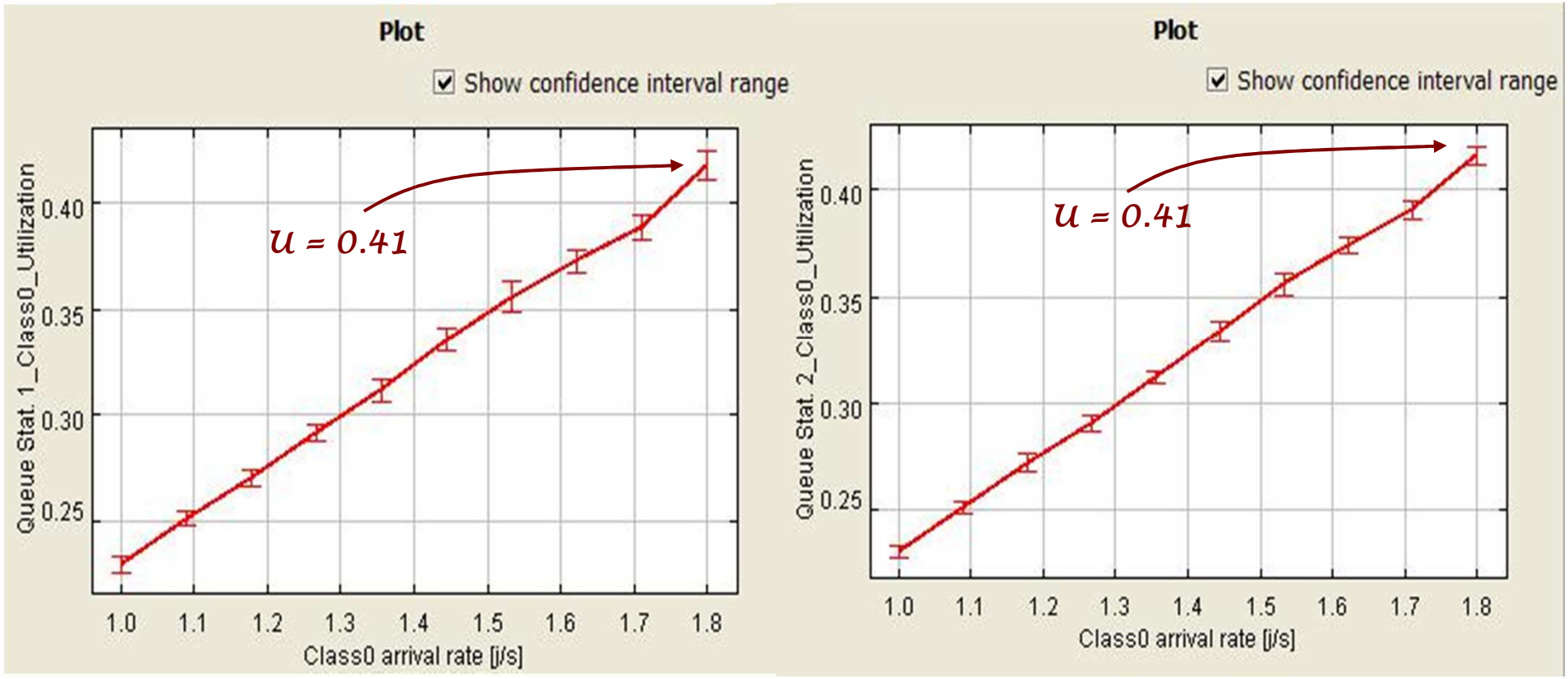


system response time R = 5.5 sec

Utilizations with "least utilization" routing

path 1

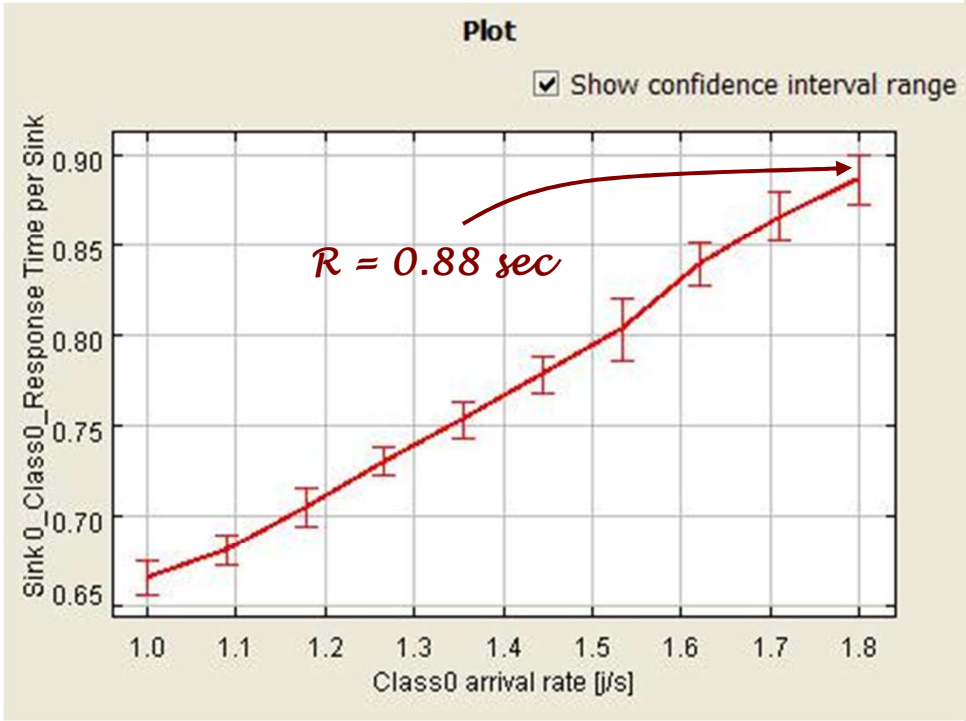
path 2



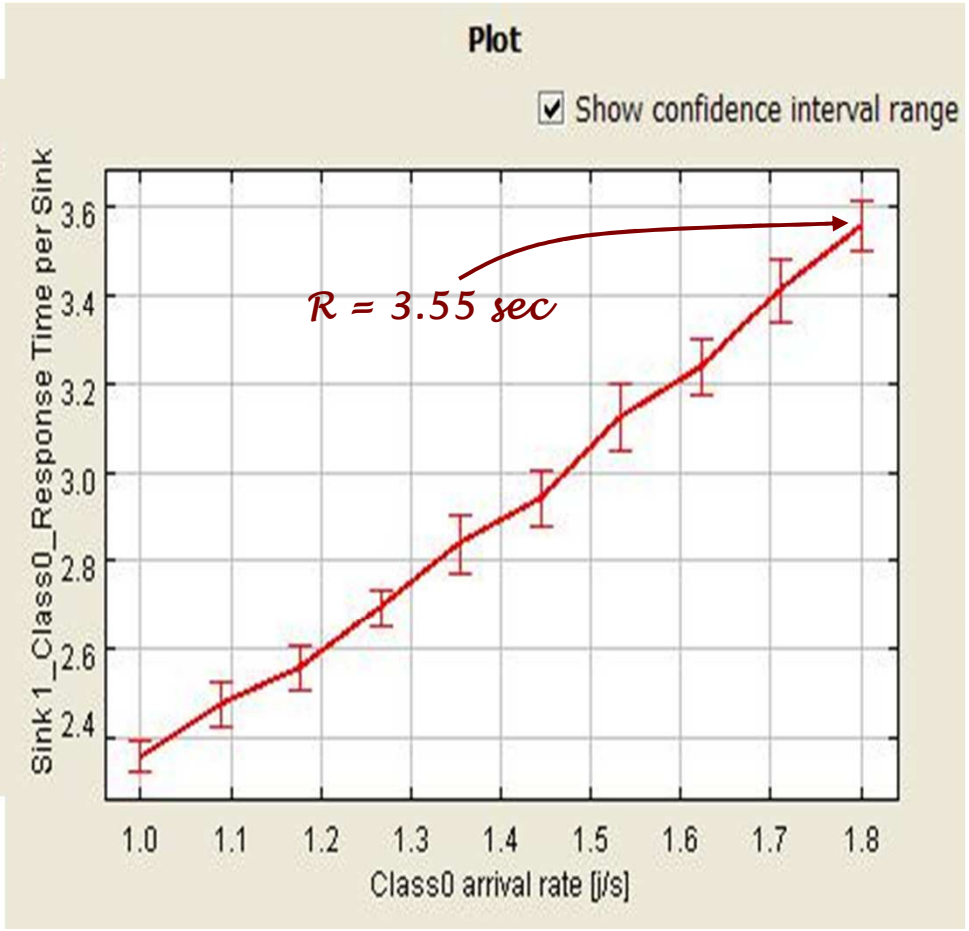
utilizations well balanced

Response times with "least utilization" routing

path 1



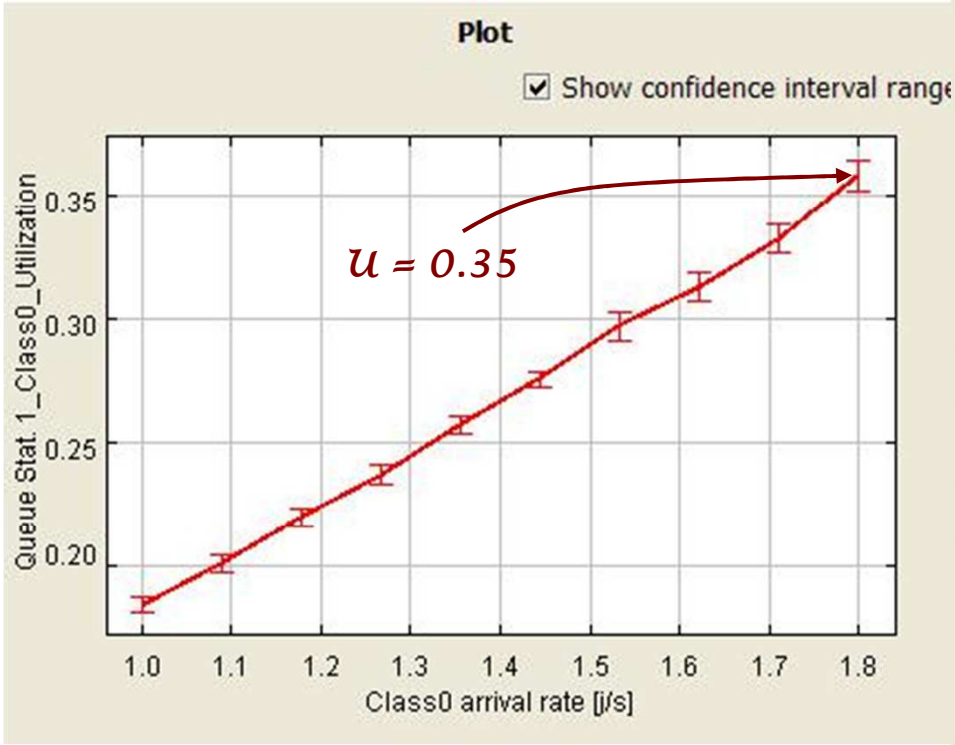
path 2



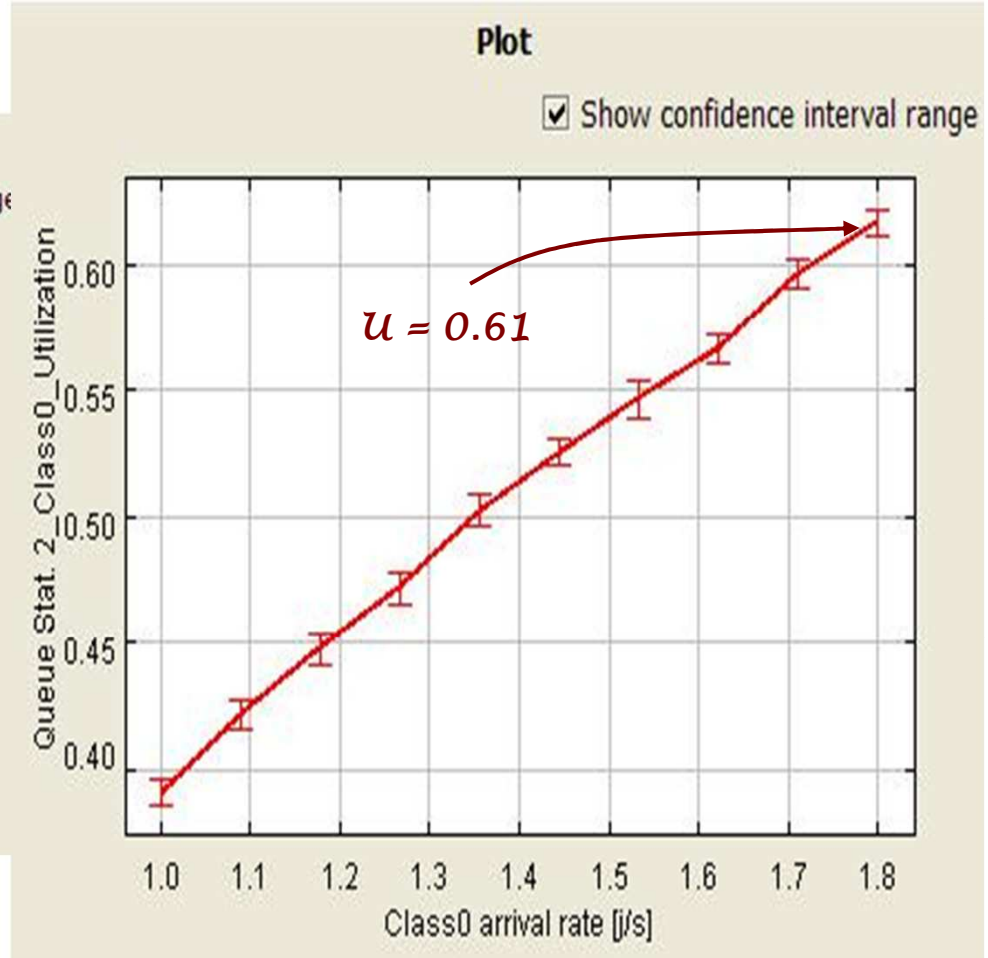
system response time R = 1.5 sec

Utilizations with "Joint the Shortest Queue" routing

path 1

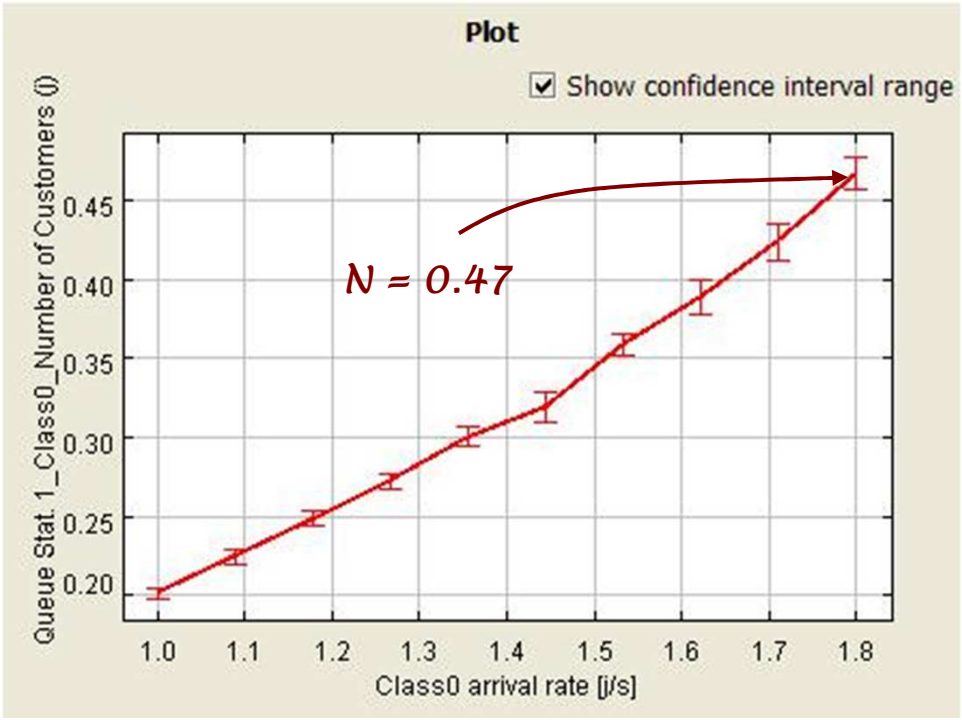


path 2

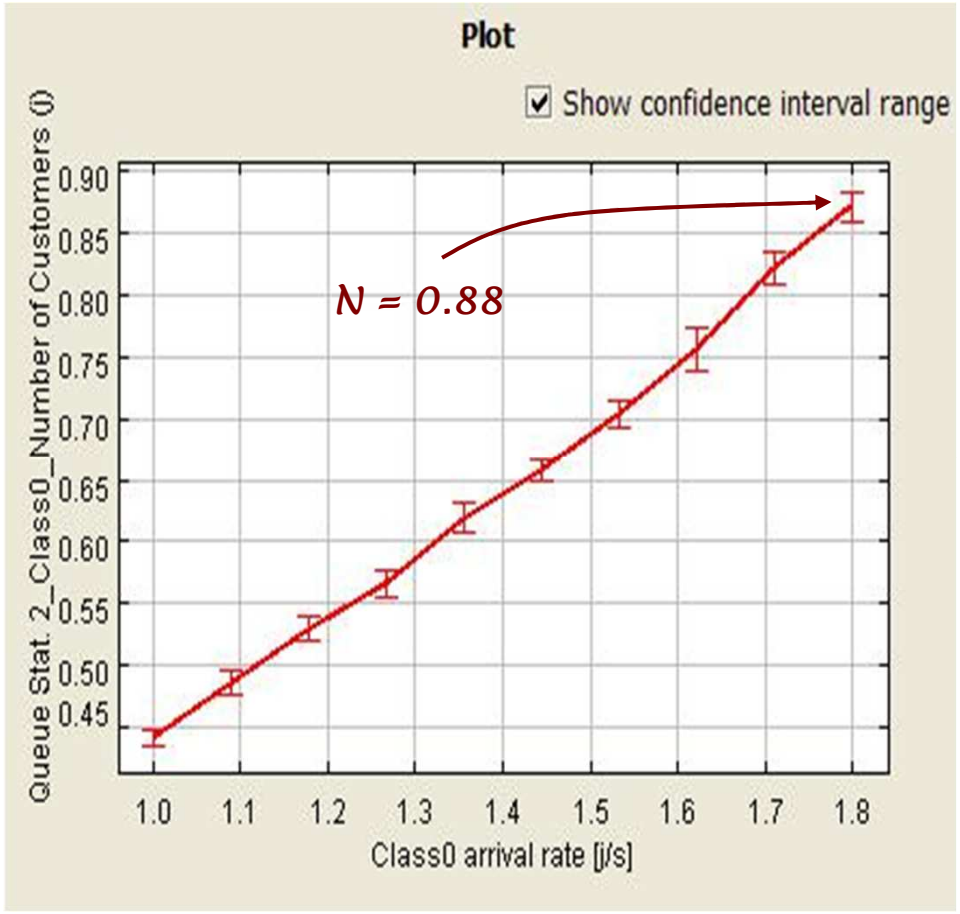


N of customers with JSQ routing

path 1

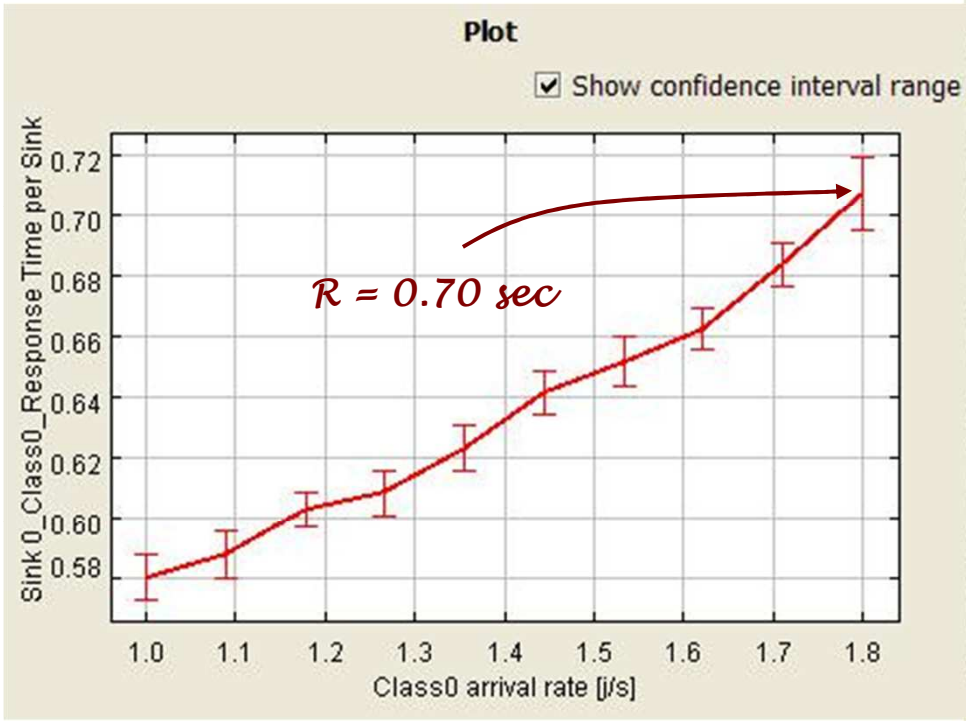


path 2

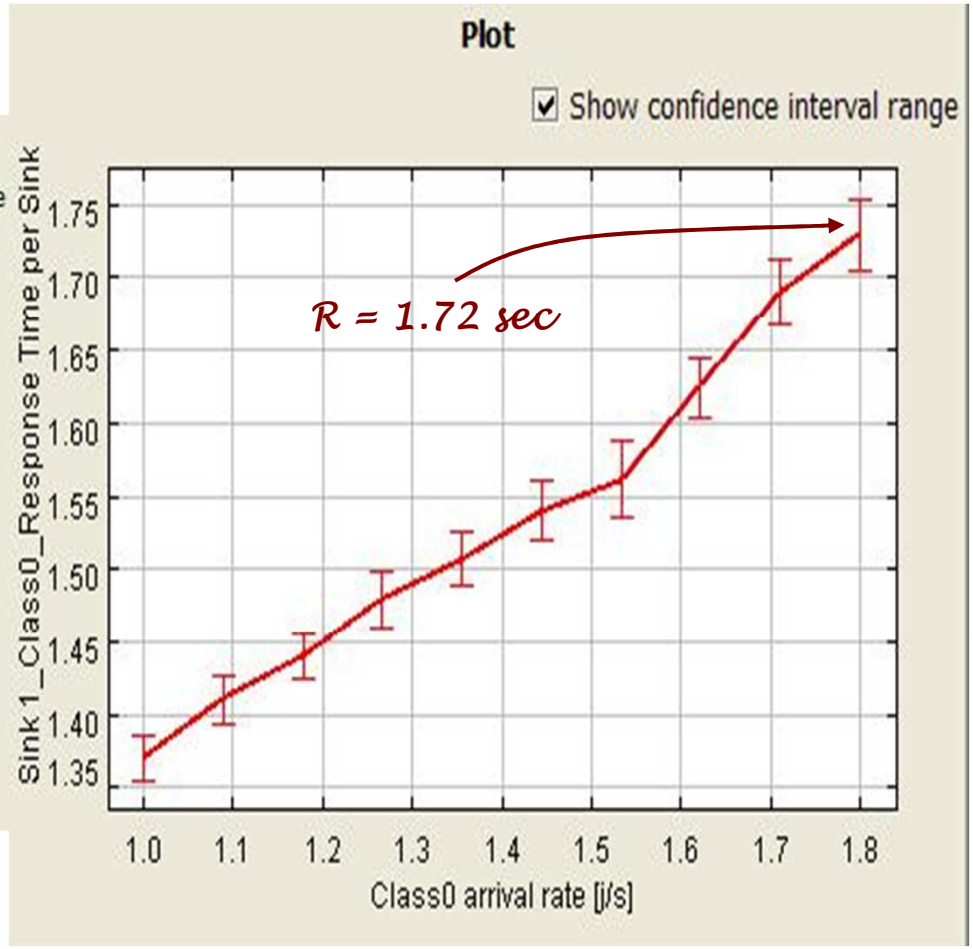


Response times with JSQ routing

path 1



path 2



system response time R = 1.05 sec

CASE STUDY 3
Resource Contention
(use of Finite Capacity Regions - FCR)

contention of components
hardware: I/O devices, memory, servers, ...
software: threads, locks, semaphores, ...
bandwidth

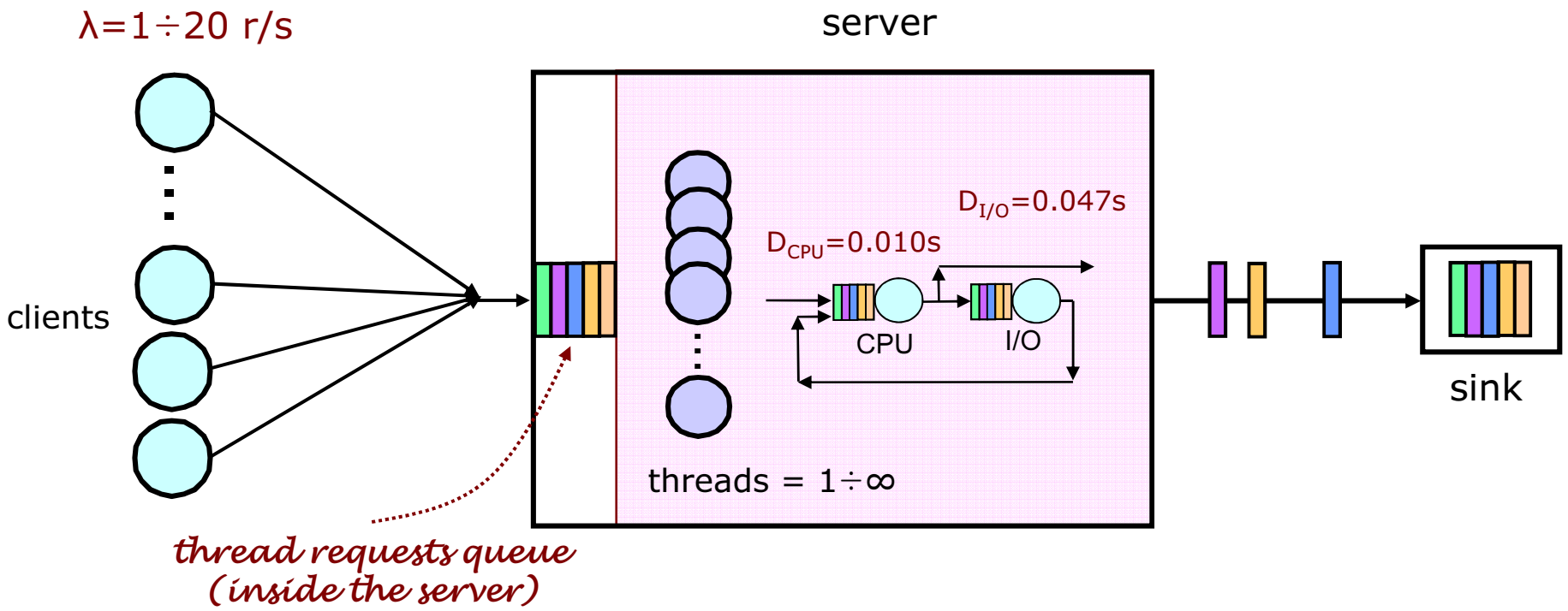
open model
single class workload

JSIMgraph

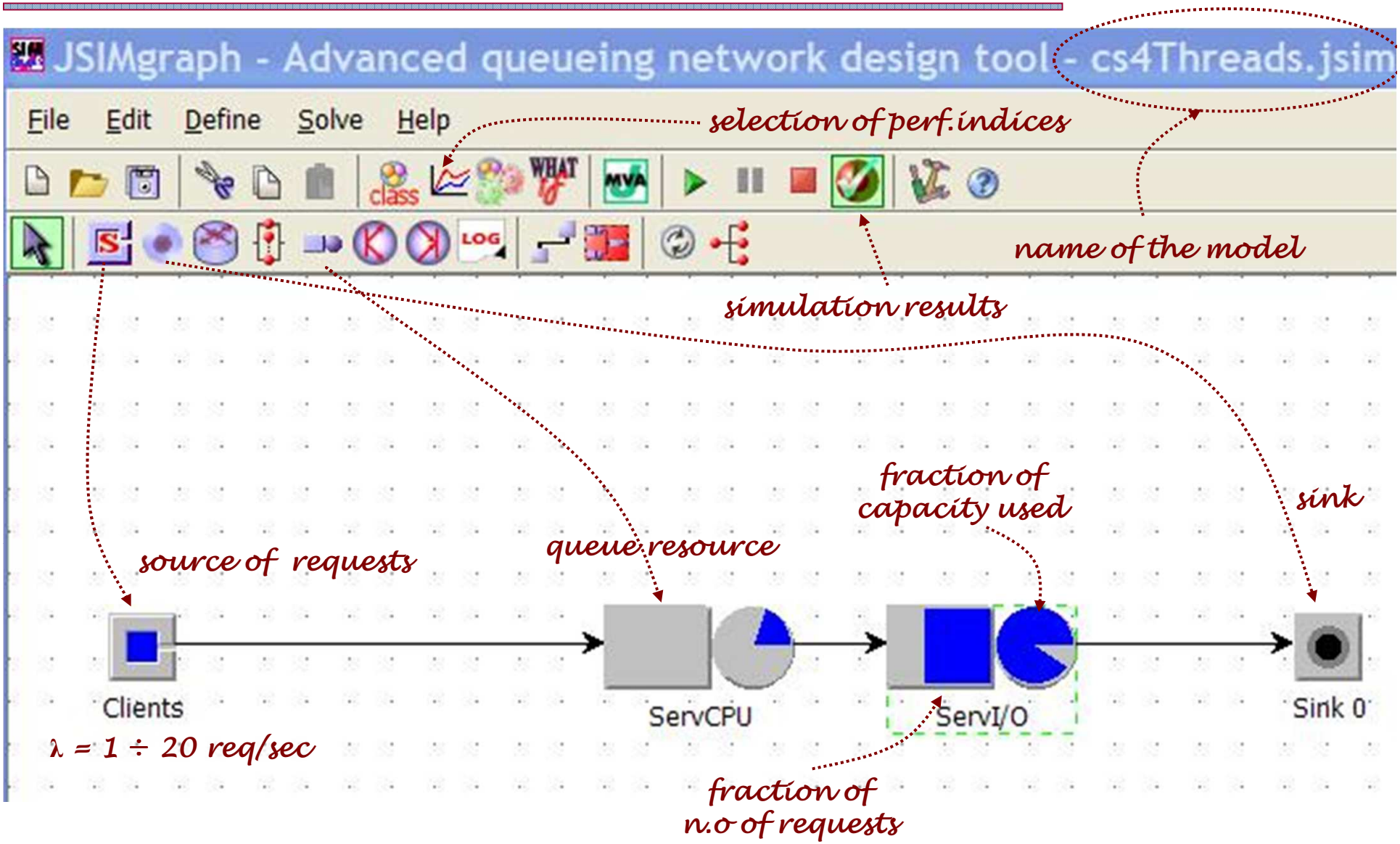
modeling contention

- fixed number of hw/sw components (threads, db locks, semaphores, ...)
- clients compete for the available component free
- request execution time: wait time for the next free component + wait time for the hardware resources (CPU, I/O, ...) + execution time
- request interarrival times exponentially distributed
- payload of different sizes (exponentially distributed)
- evaluate the execution time of requests when the number of clients ranges from 1 to 20 and the number of components ranges from 1 to 10 (∞), evaluate the drop rate and the wait time in queue for the next available component
- implement several models with different level of completeness

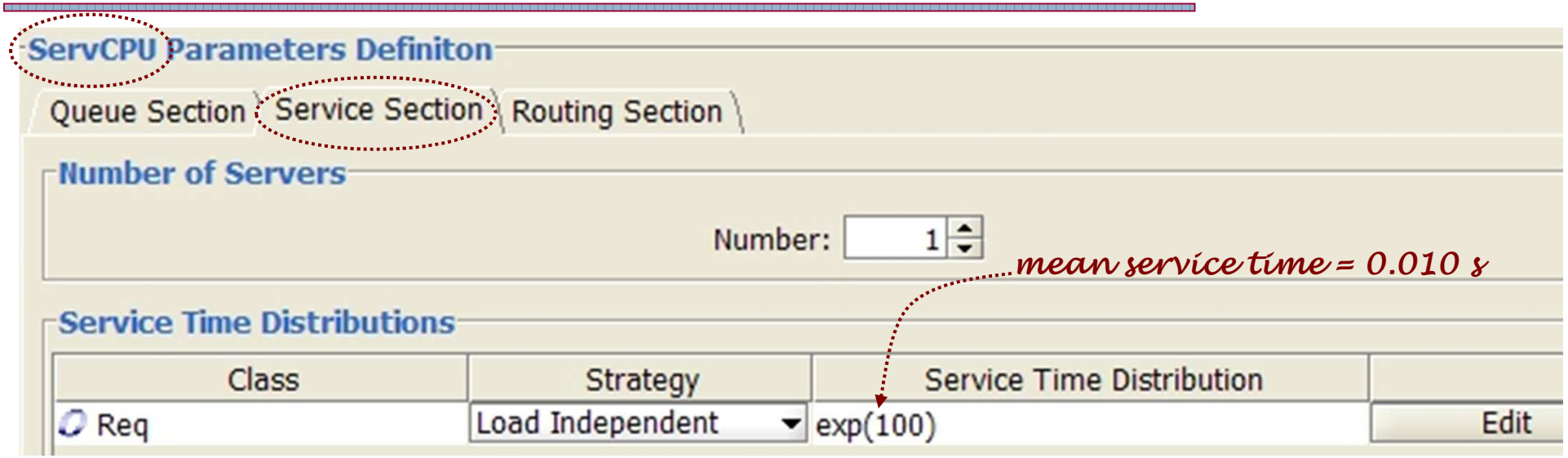
threads (resource hw/sw) contention (simple model)

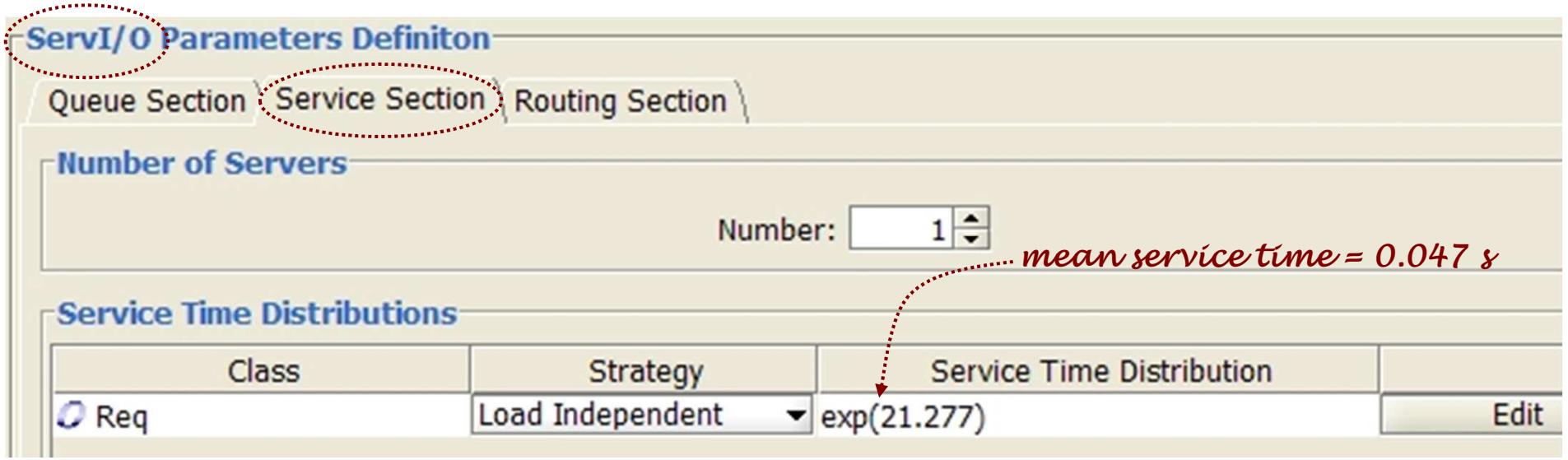


model definition (unlimited threads and queue size)

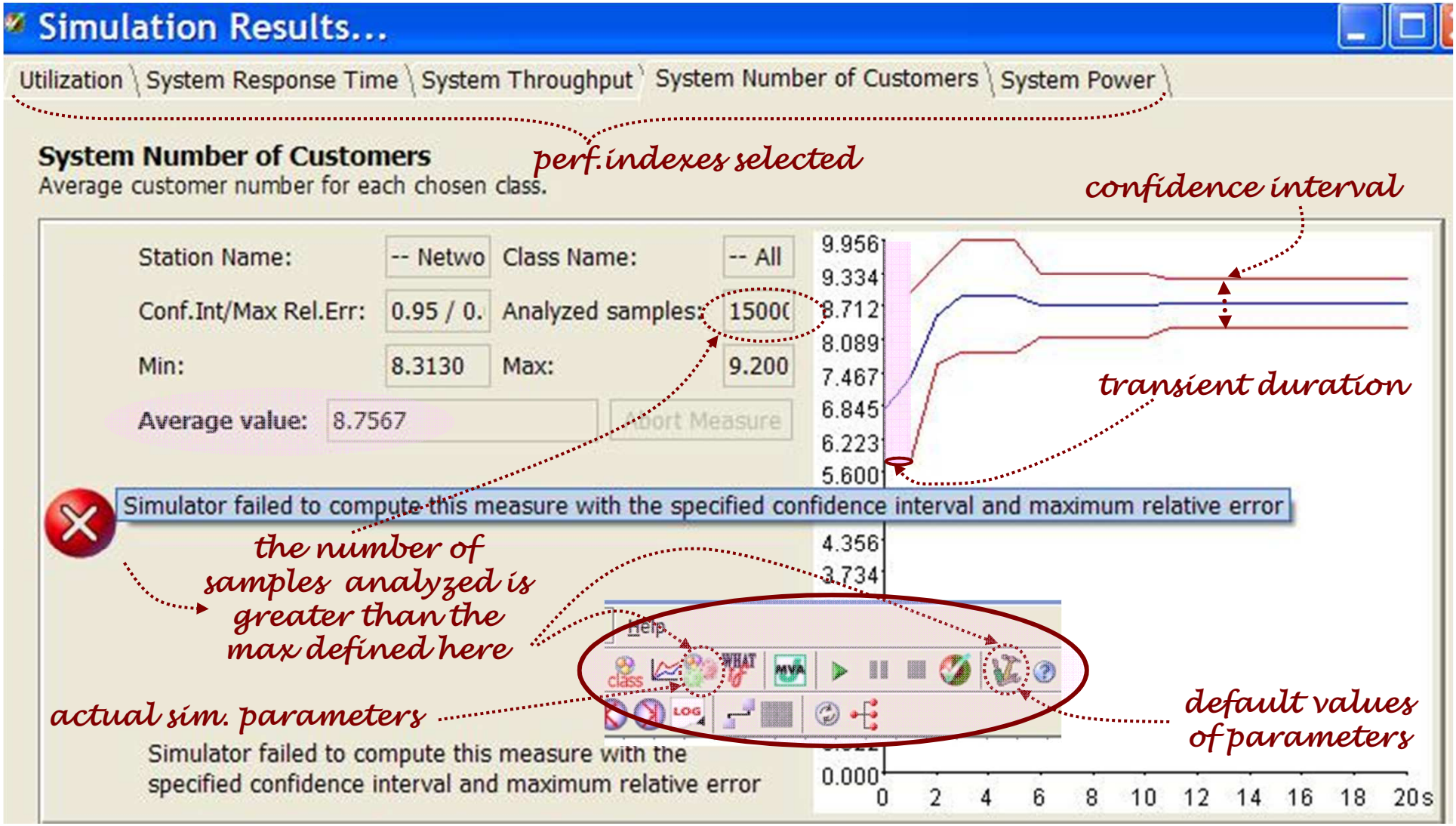


input parameters (service demands)

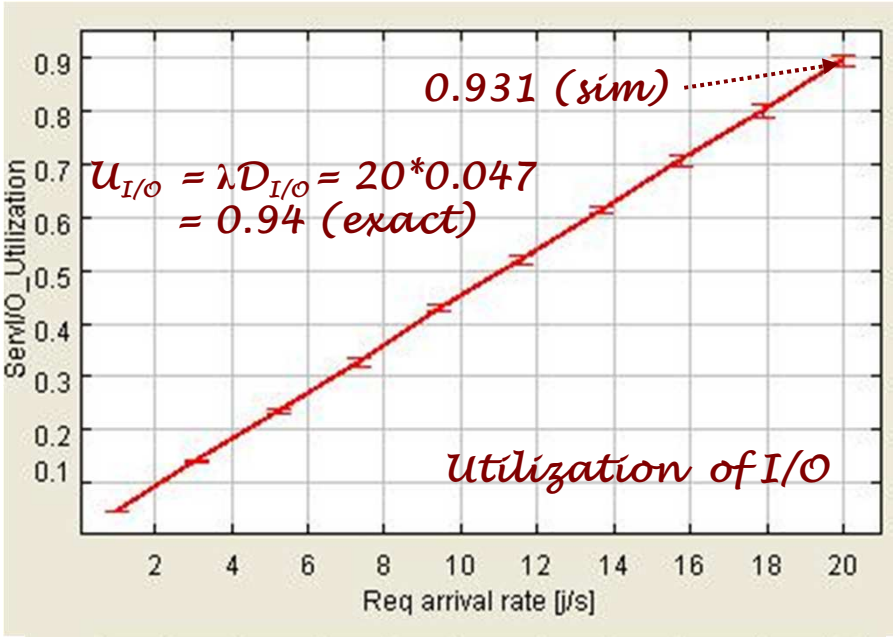


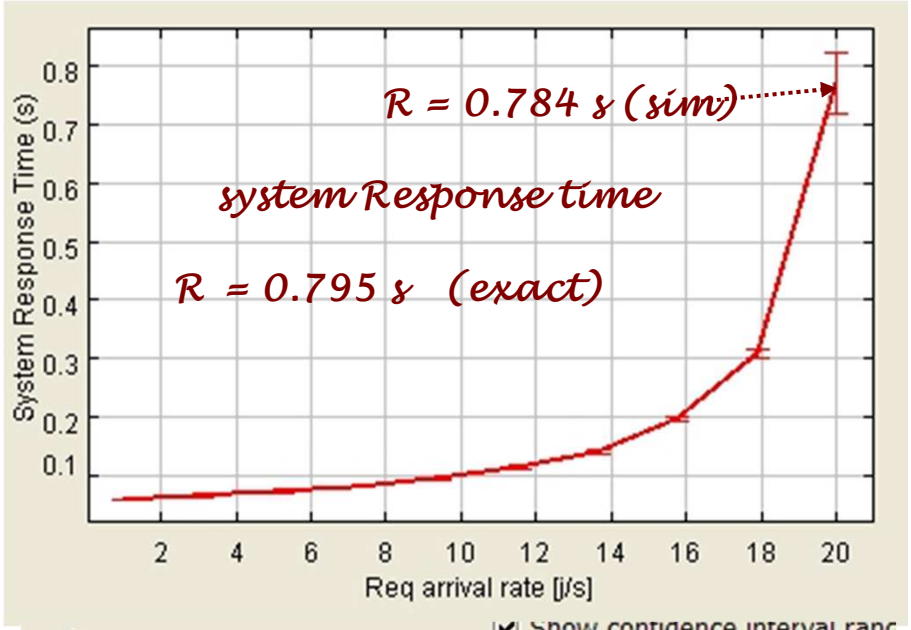


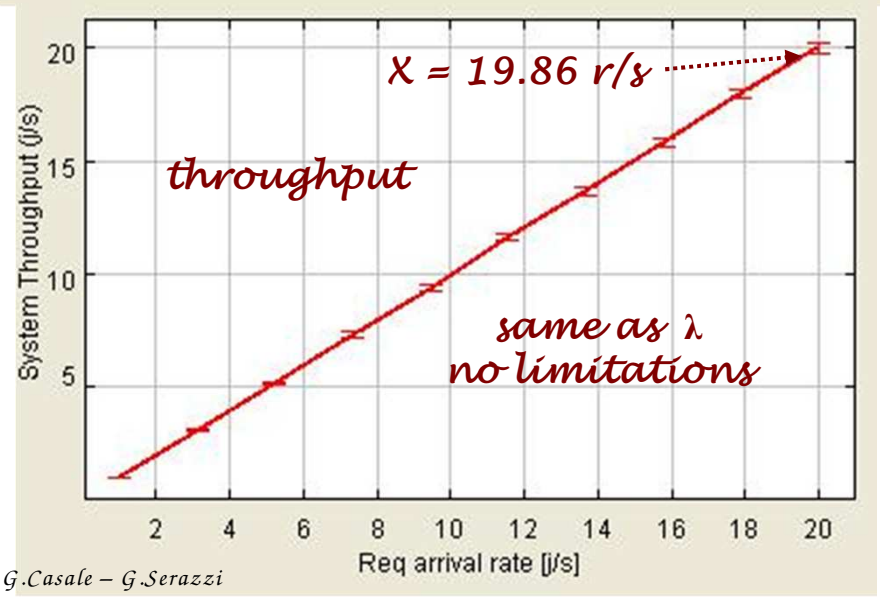
system Response time (λ=20 req/sec)

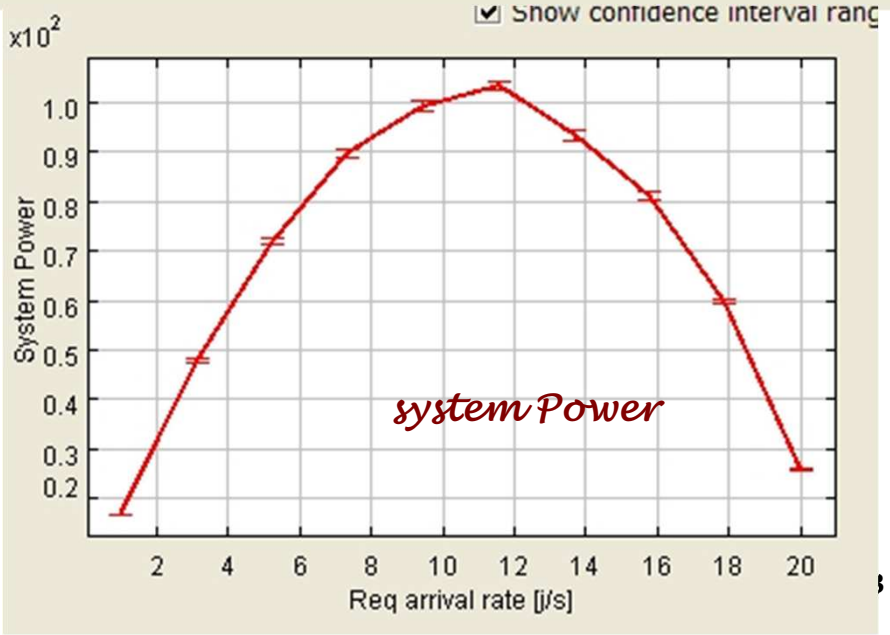


λ=1÷20 req/s, unlimited threads & queue size (JSIMgraph)

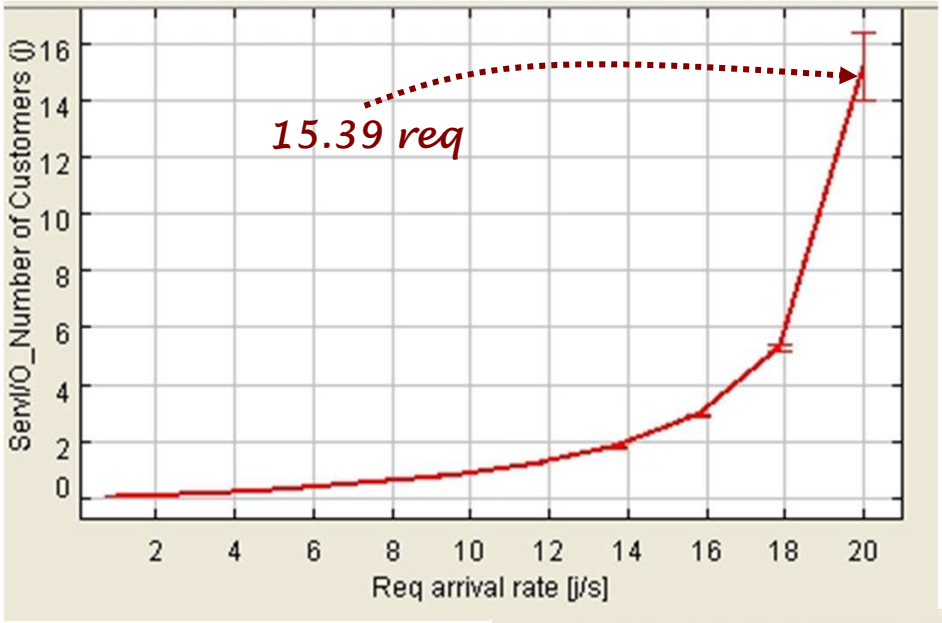


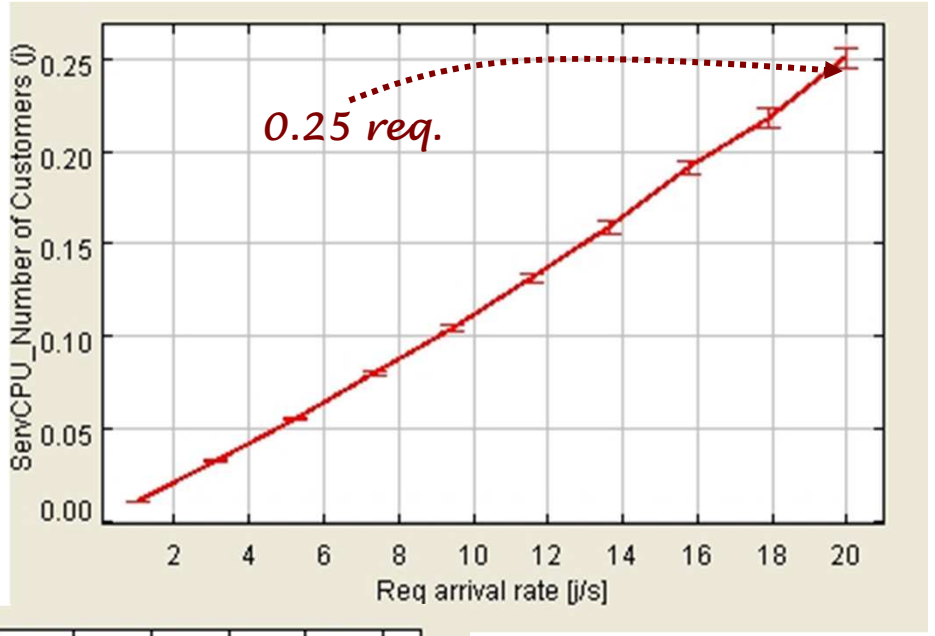


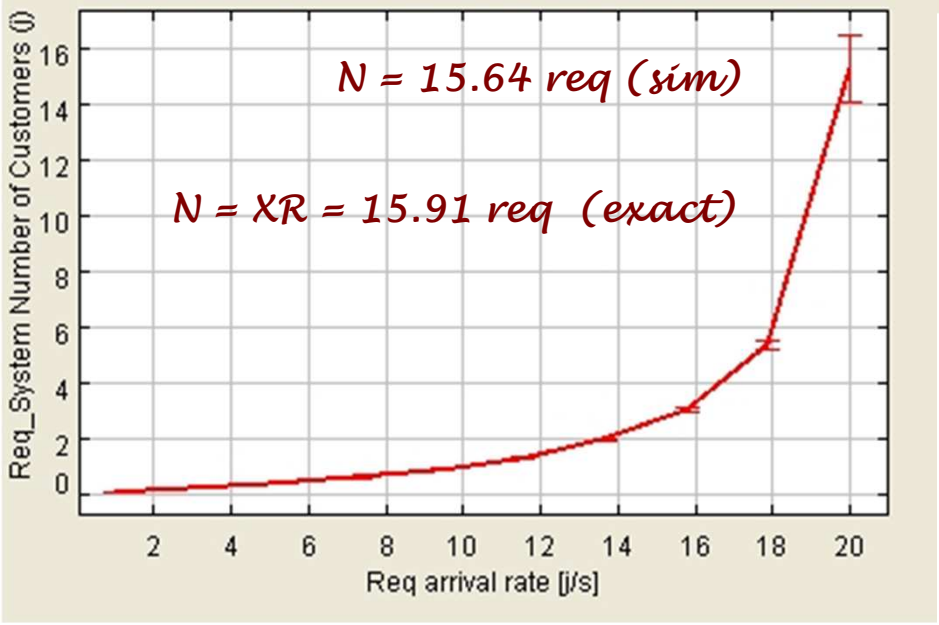




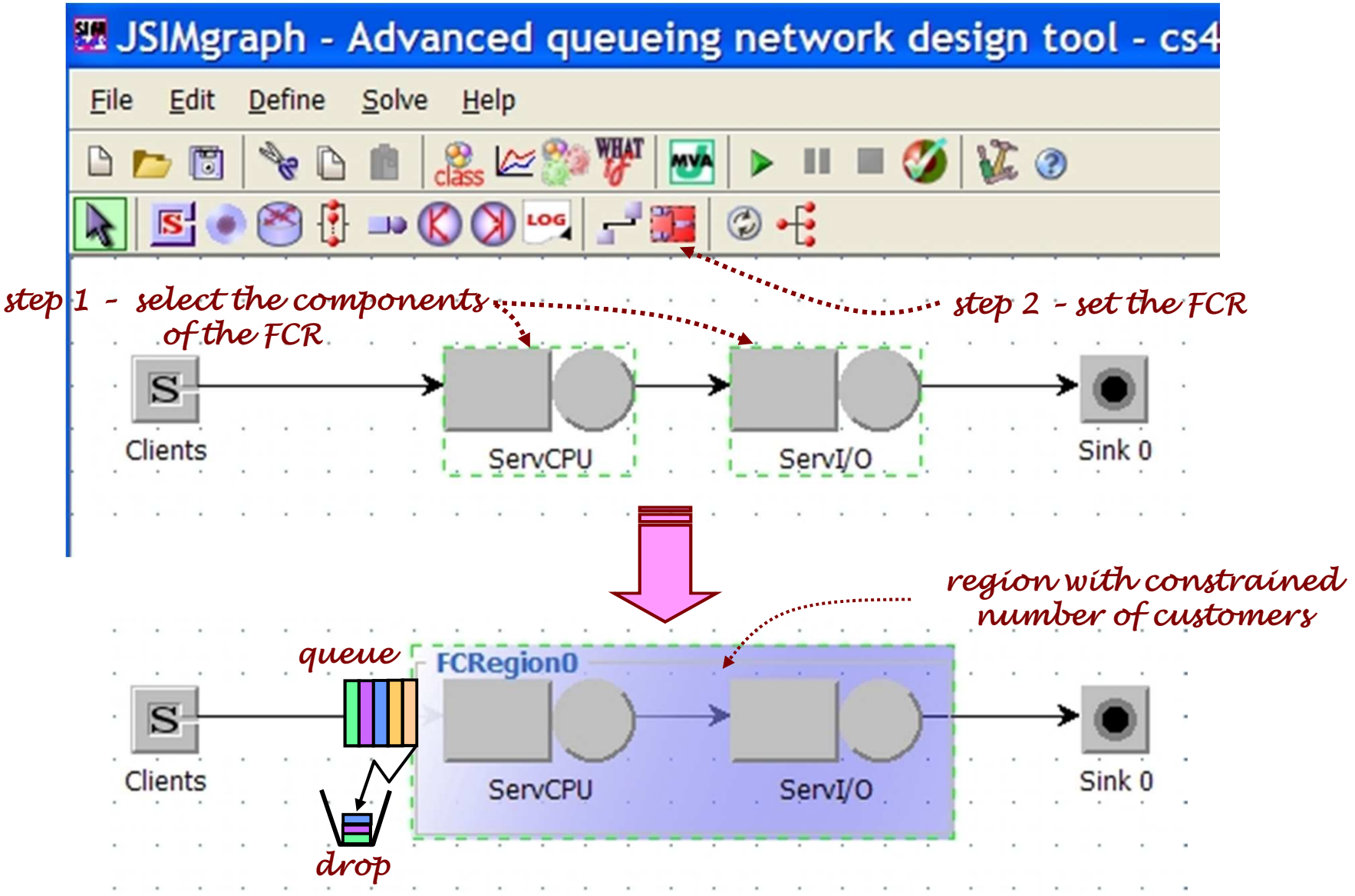
Number of requests (unlimited threads & queue size)



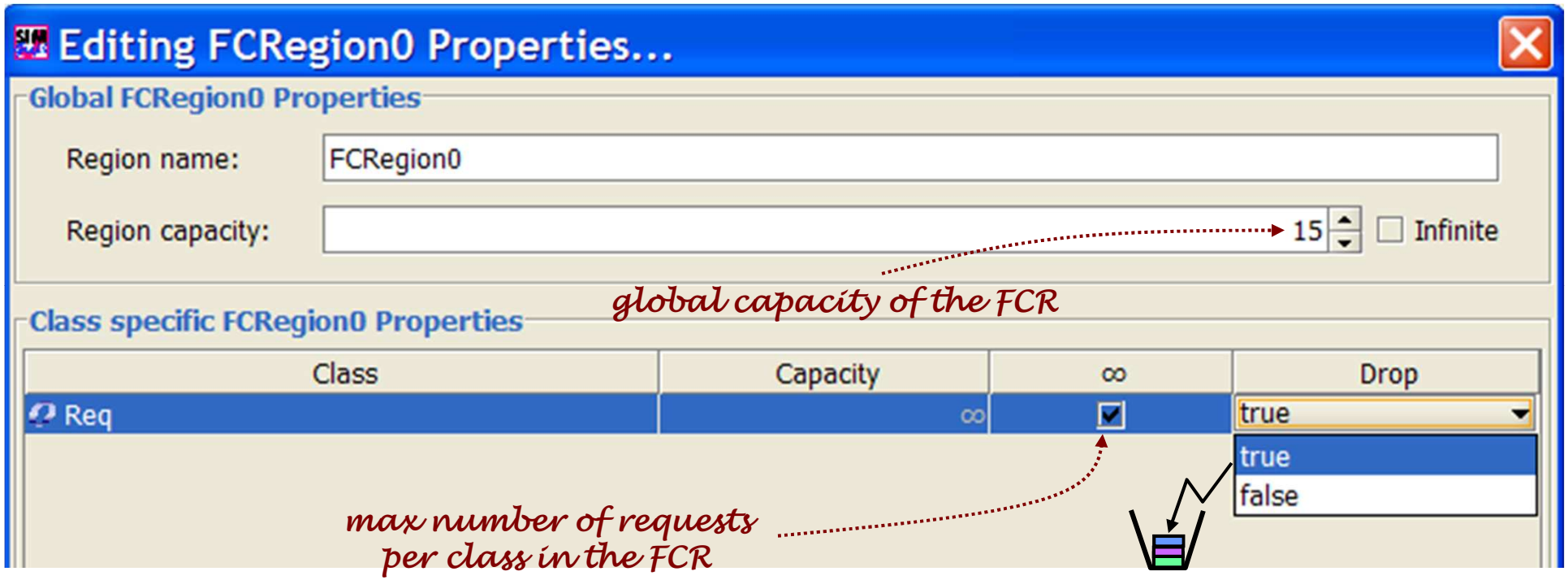




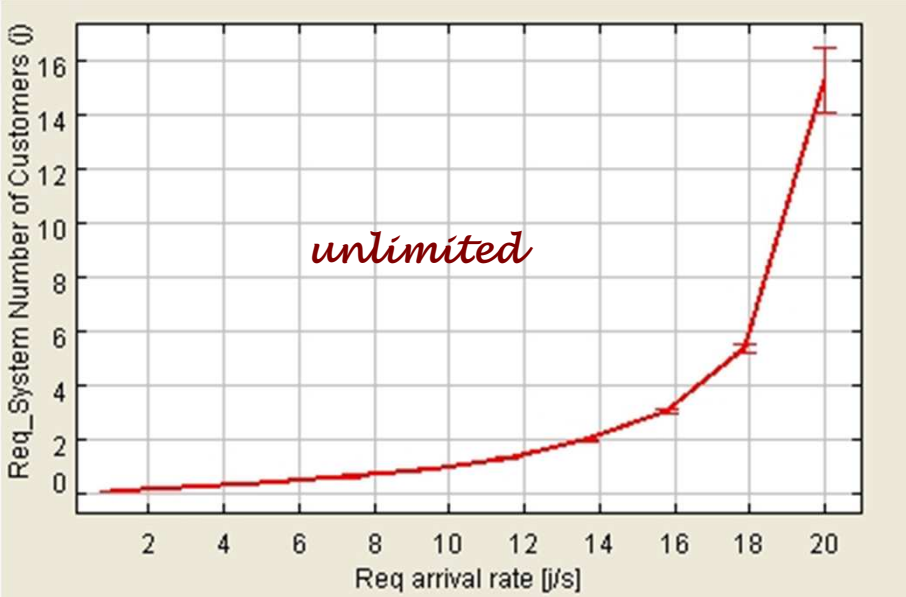
set of a Finite Capacity Region – FCR

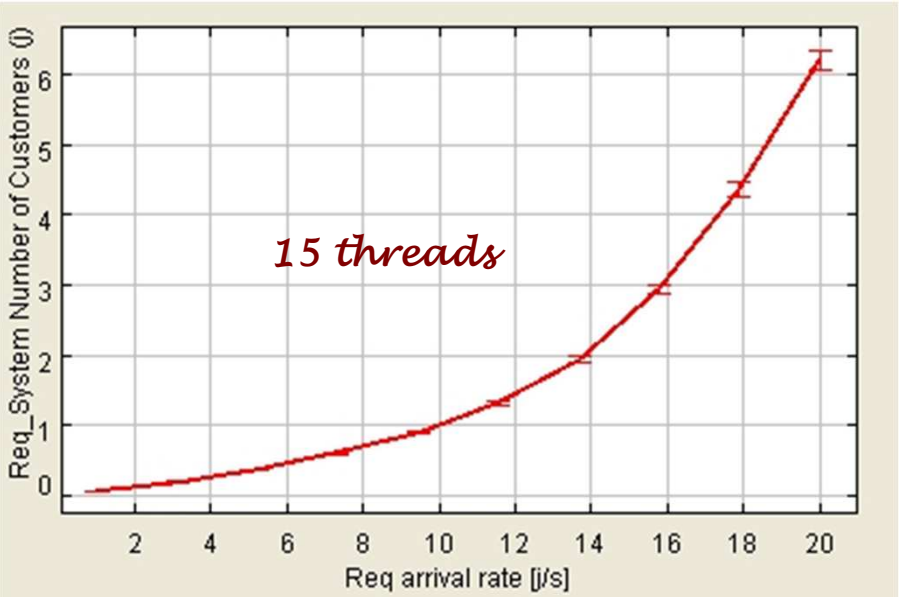


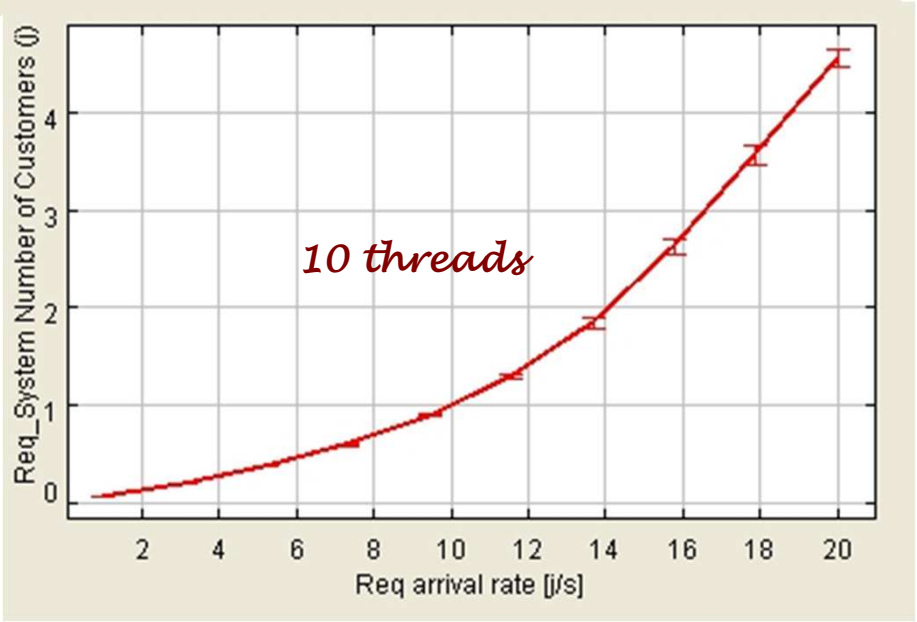
FCR parameters

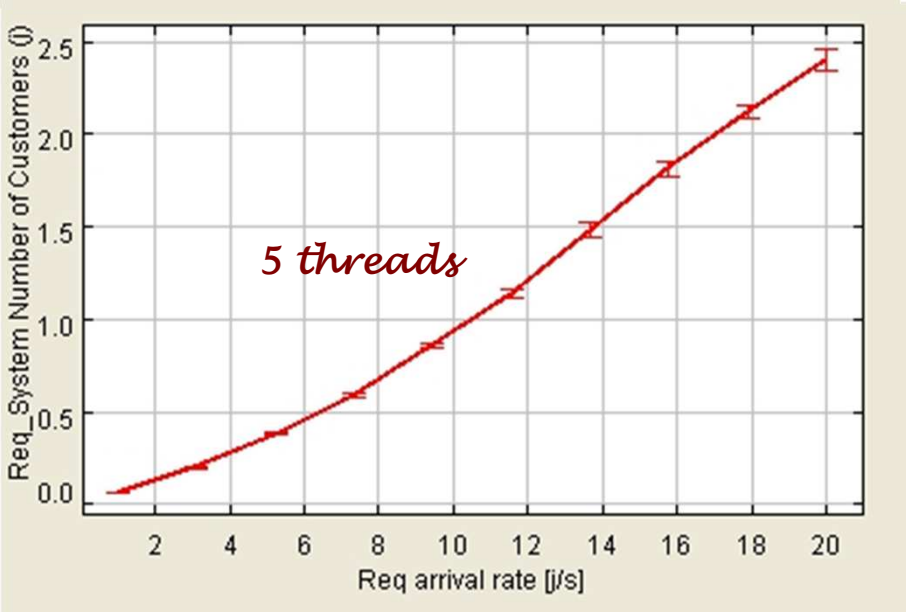


system Number of requests (limited n. threads and drop)

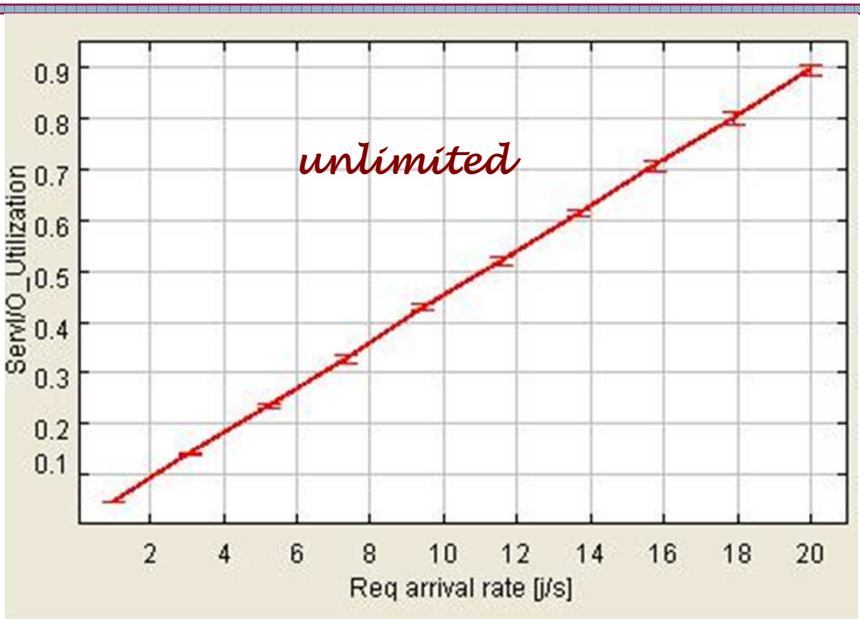


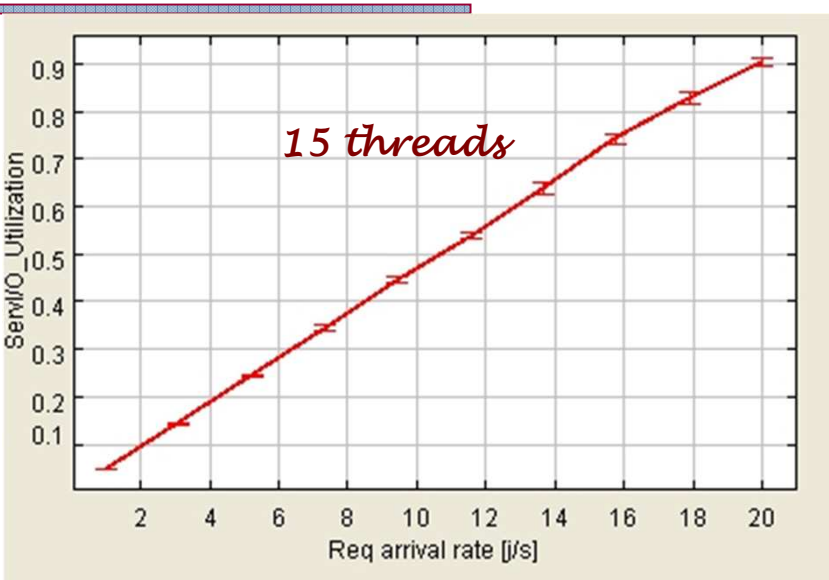


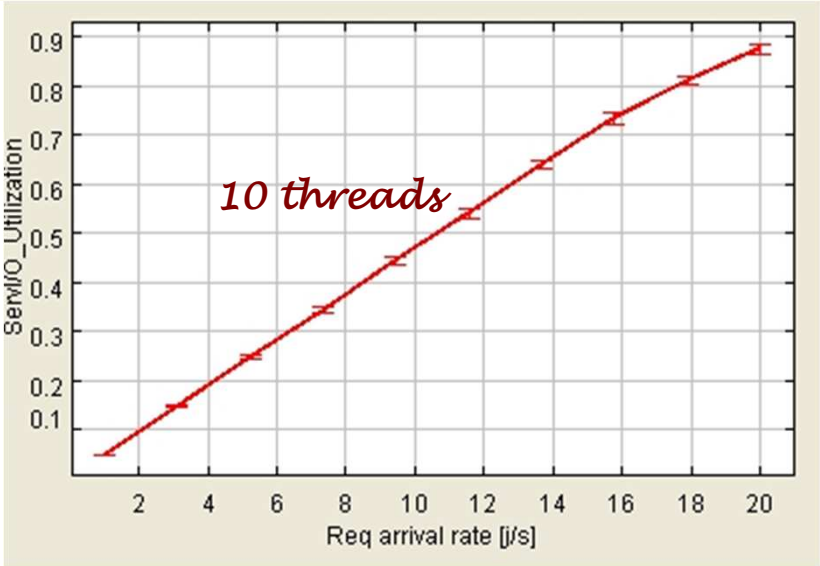


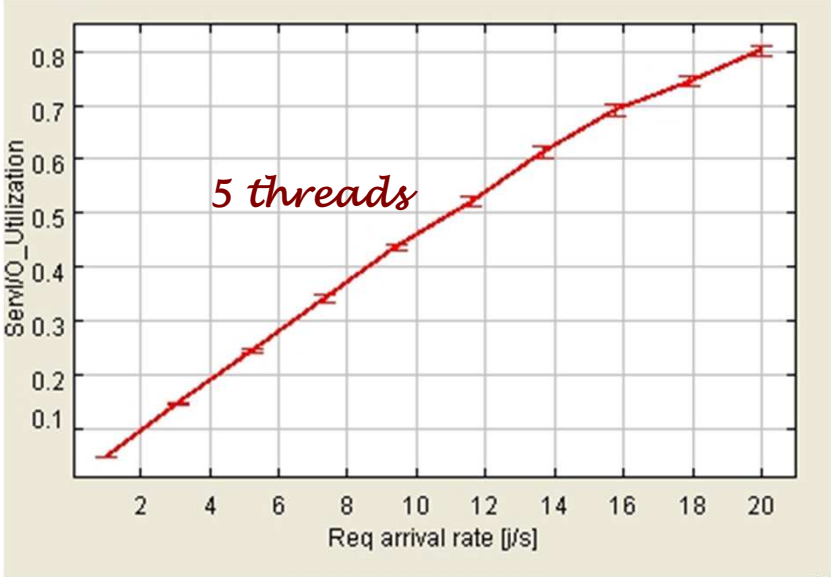


Utilization of I/O server (limited n. threads and drop)

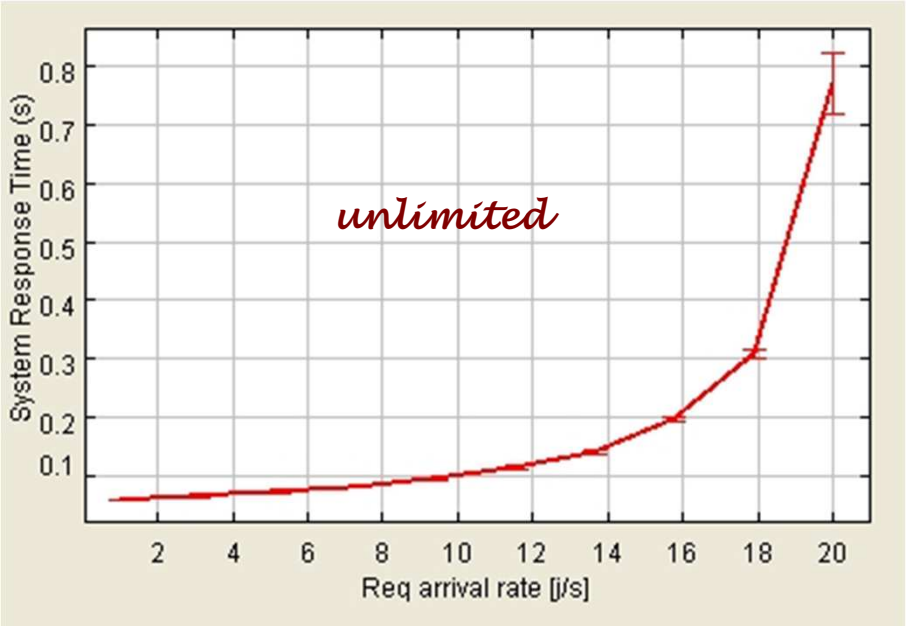


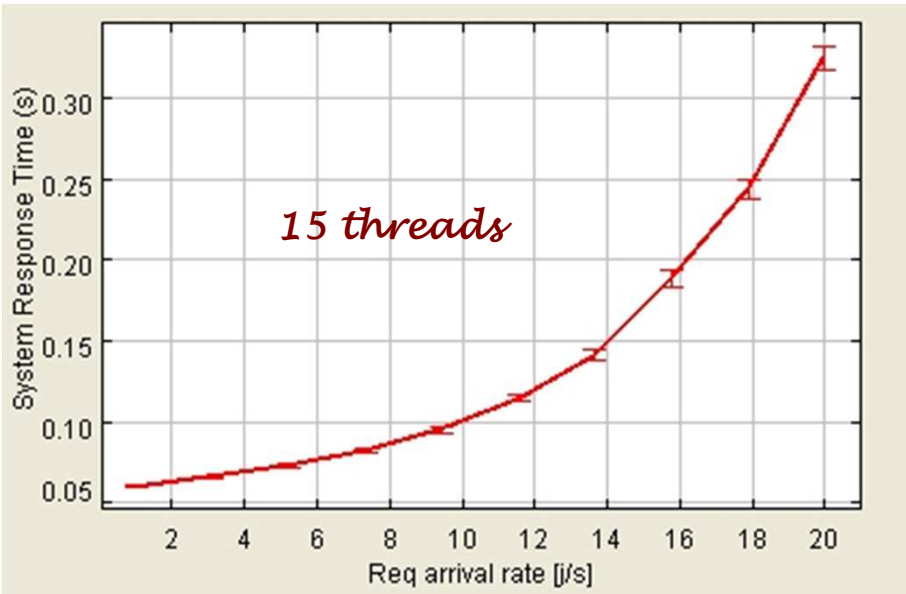


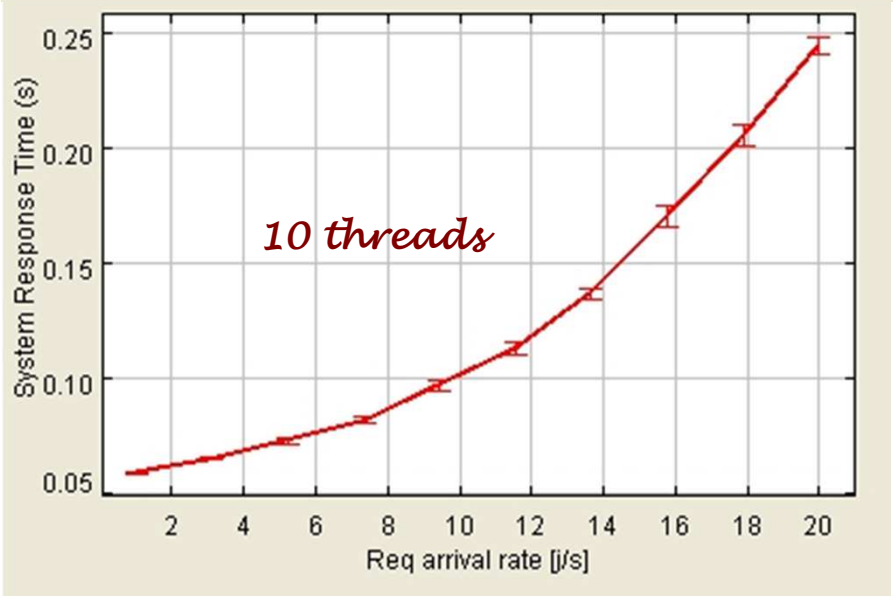




system Response time (limited n. threads and drop)

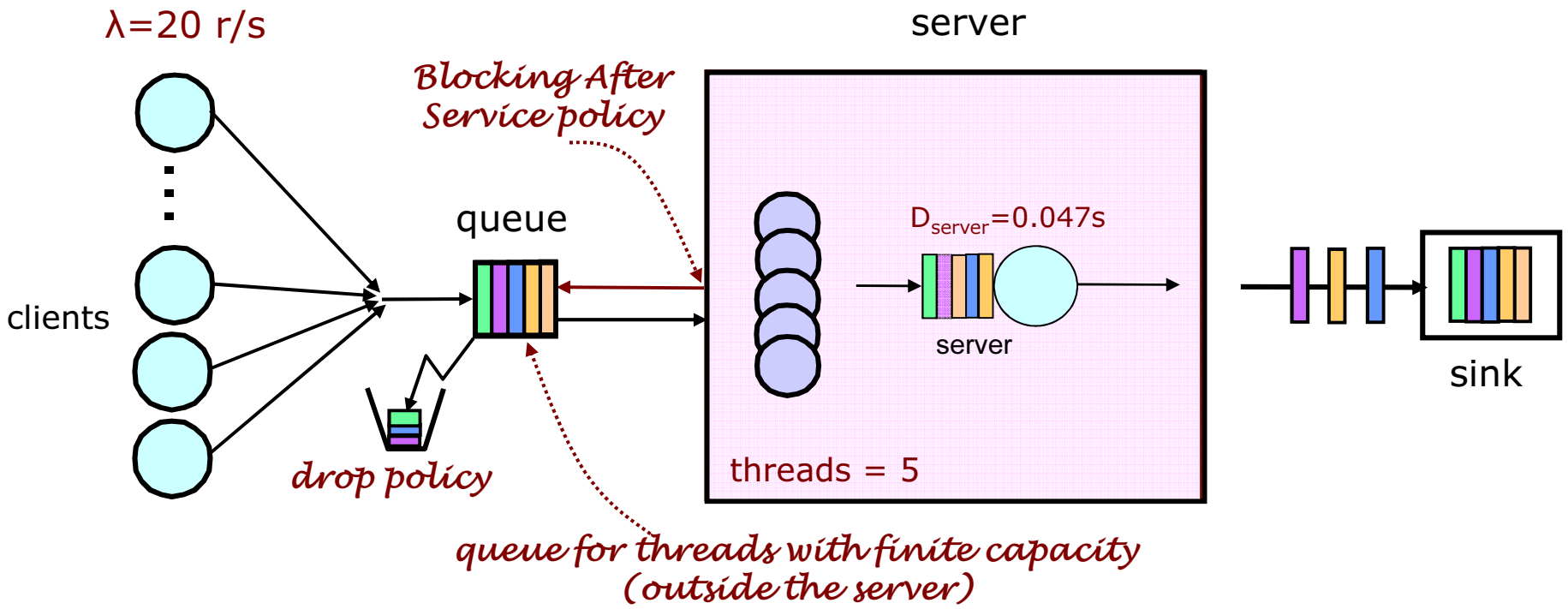






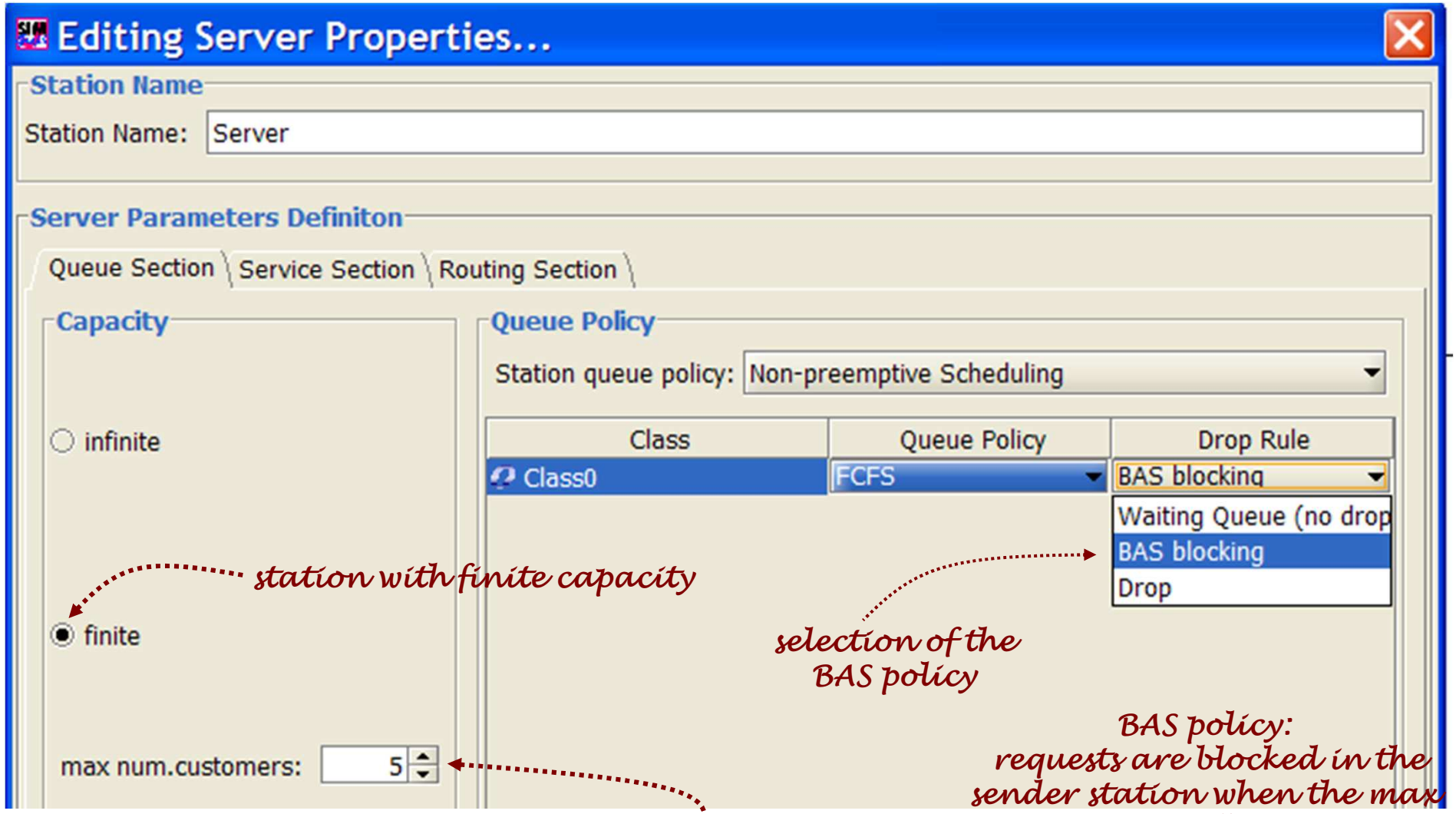


external finite queue for limited threads

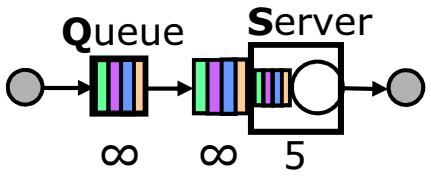
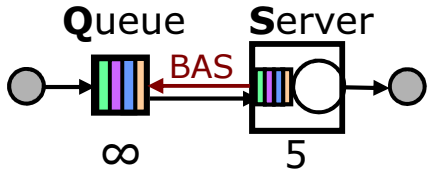
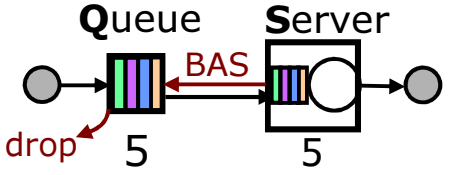
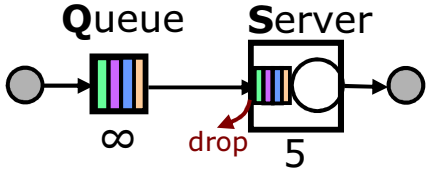


- the queue for threads is limited (e.g., to limit the number of connections in case of denial of service attack, to guarantee a negotiated response time for the accepted requests, ...)
- the requests arriving when the queue is full are rejected (drop policy)
- the number of threads is limited and the requests are queued in a resource different from the server (load balancer, firewall, ...)
- evaluate the combination of different admission policies

set Block After Service (BAS) blocking policy



different admission policies for Queue and Server

λ=20 req/s	N	R	U	X	Drop	Queue and Server stations
Qsize= ∞ Q Ser=5, queue S	0 16.11	0 0.77	0 0.95	20.06	0	
Qsize= ∞ Q Ser=5, BAS S	11.03 4.77	0.53 0.24	0 0.923	19.82	0	
Qsize=5 drop Q Ser=5, BAS S	0.94 3.82	0.05 0.20	0 0.88	18.76	1.14	
Qsize= ∞ Q Ser=5, drop S	0 2.34	0 0.136	0 0.812	17.16	2.866	

CASE STUDY 4

Multi-Tier Applications and Web Services (Worker Threads, Workflows, Logging, Distributions)

closed models
single class and multiclass workloads
fork-join

JSIMgraph+JWAT

performance evaluation of a multi-tier application

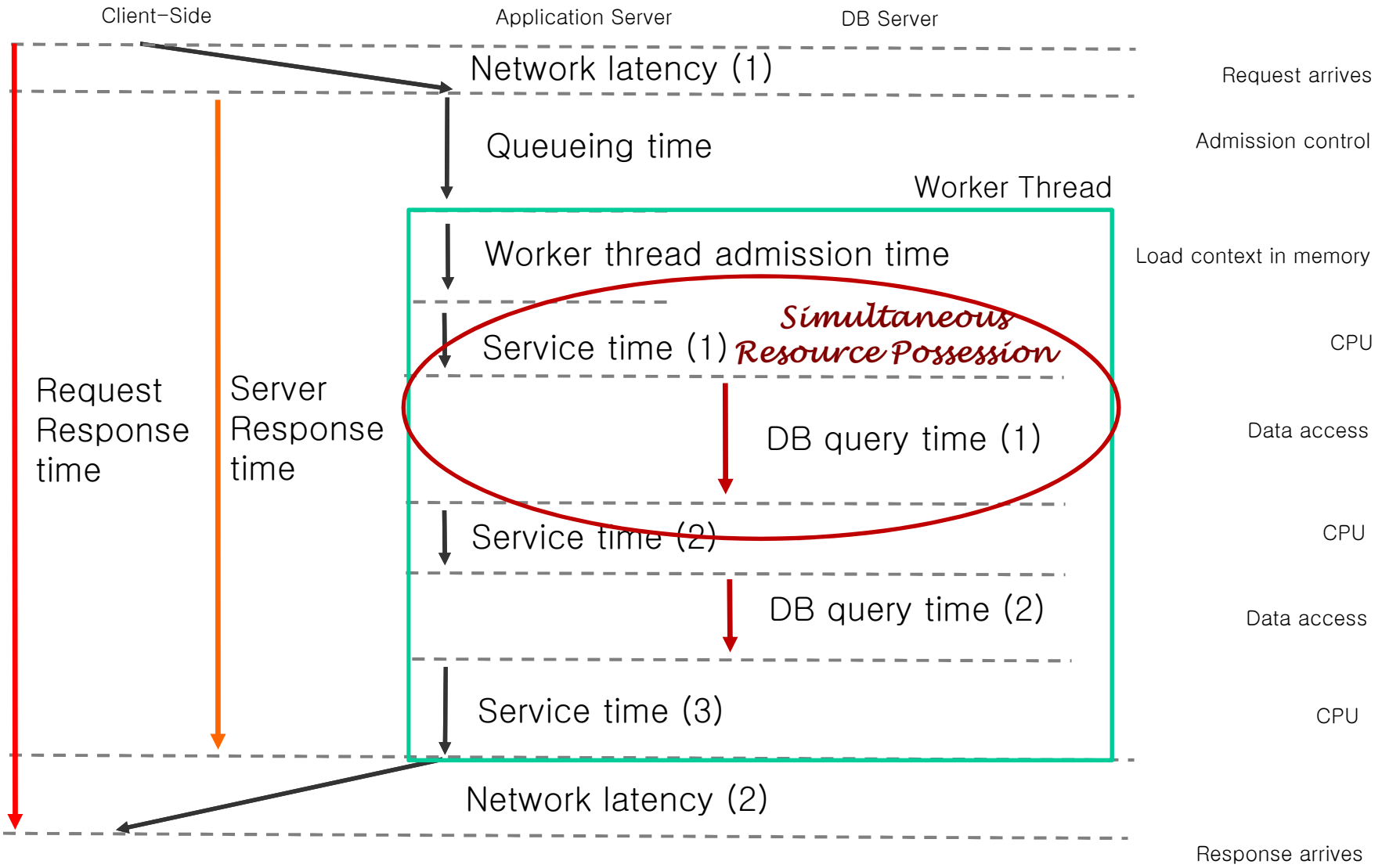
- multi-tier application serves a transactional workload which requires processing by an application server (AS) and by a database (DB)
- the AS serves requests using a fixed set of worker threads
- requests waiting for a worker thread are queued by the admission control system

- utilization measurements available for the AS and for the DB
  - know both for AS and DB the average service time S
  - e.g., linear regression estimate

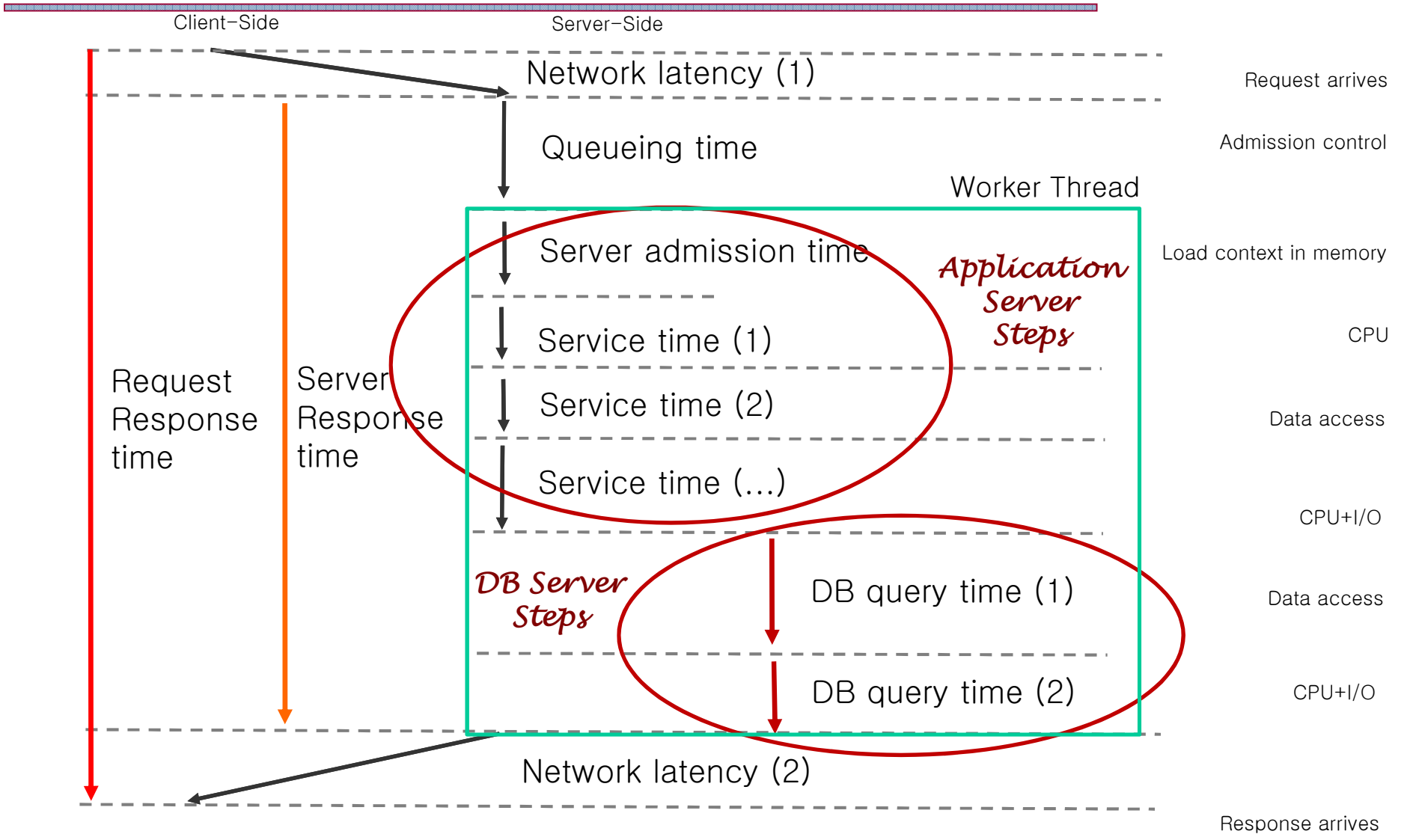
$$U=SX+Y,\ U = \text{utilization},\ X = \text{throughput},\ Y = \text{noise}$$

- evaluate response time for increasing worker threads

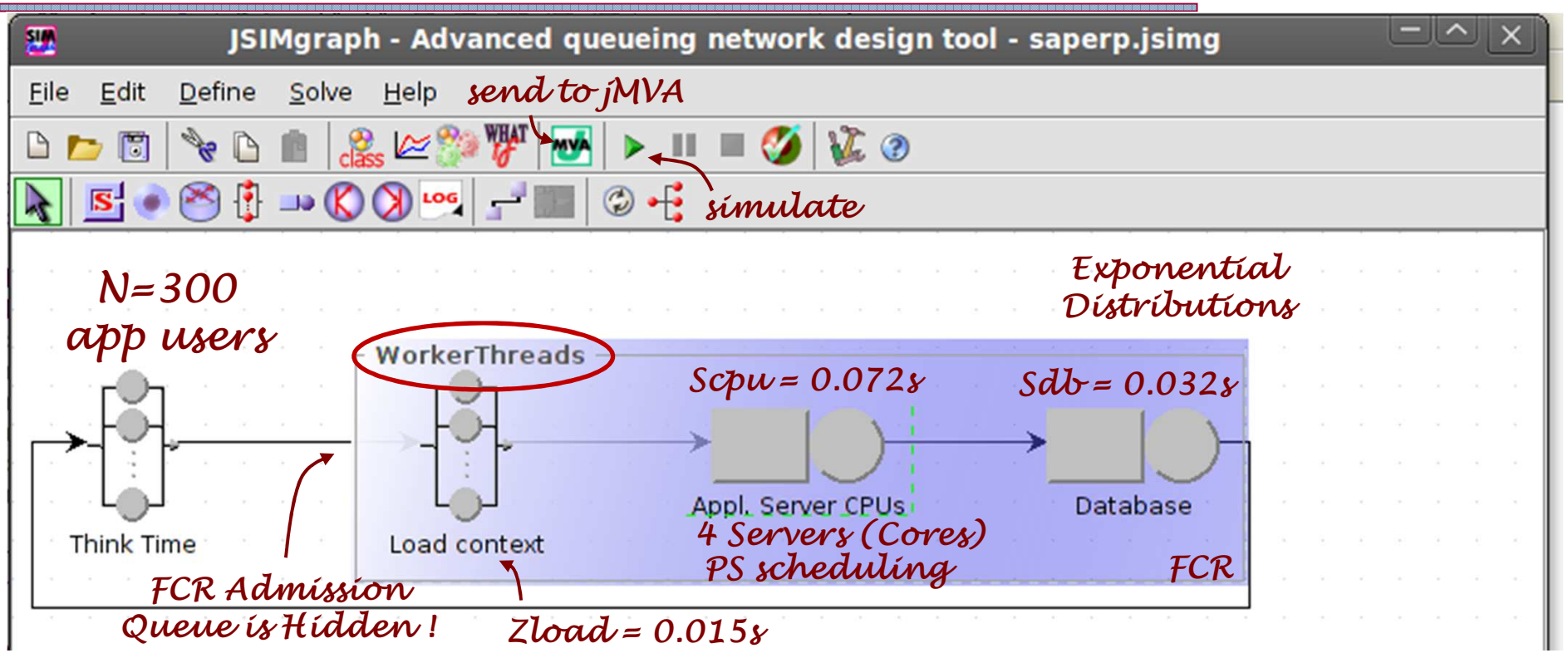
transaction lifecycle

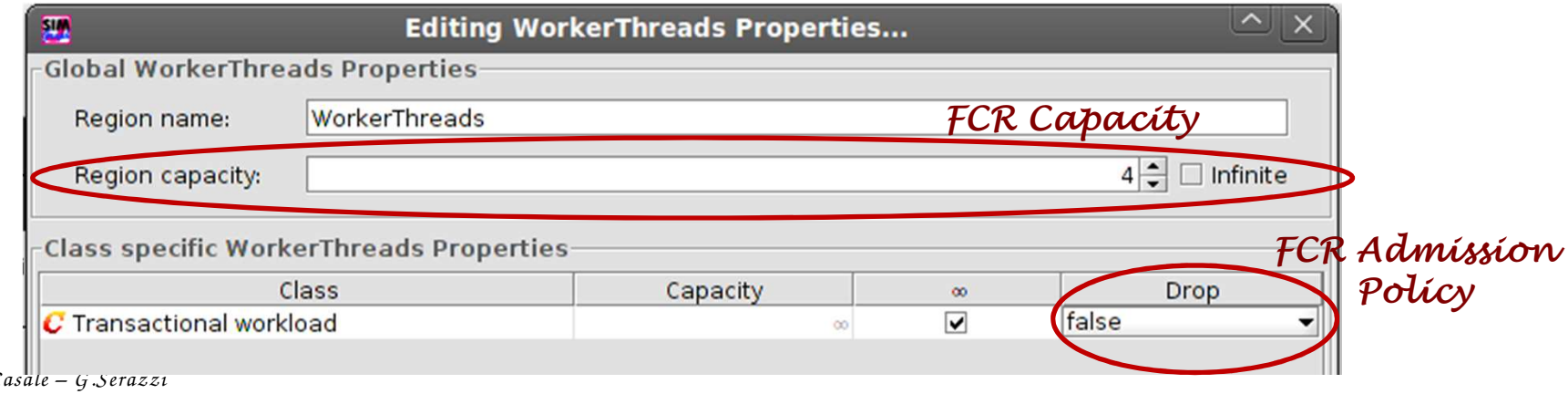


modelling abstraction (easier to define and study)

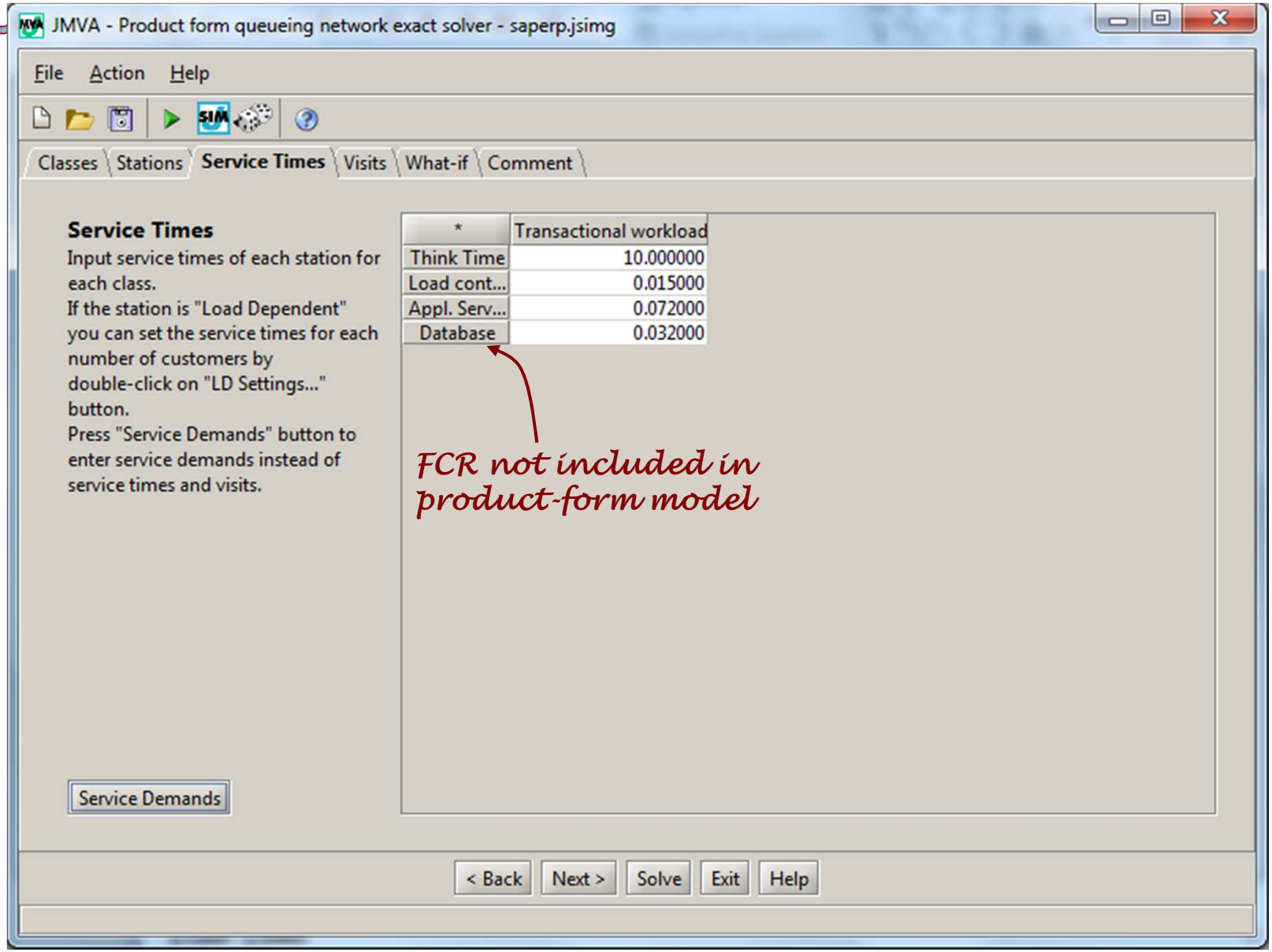


modelling multi-tier applications

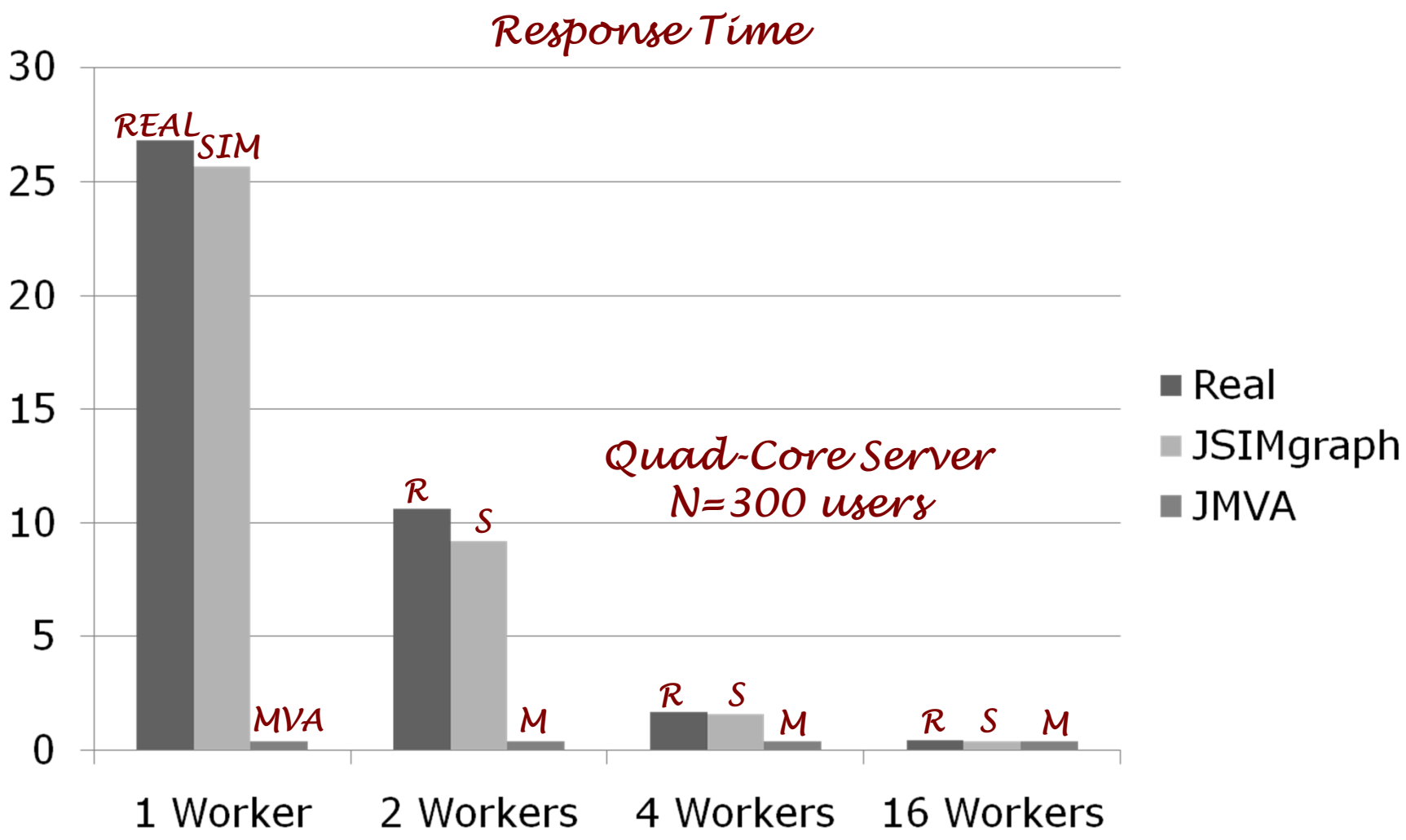




simulation vs jMVA model



SAP Business Suite [Li, Casale, Ellahi; ICPE 2010]

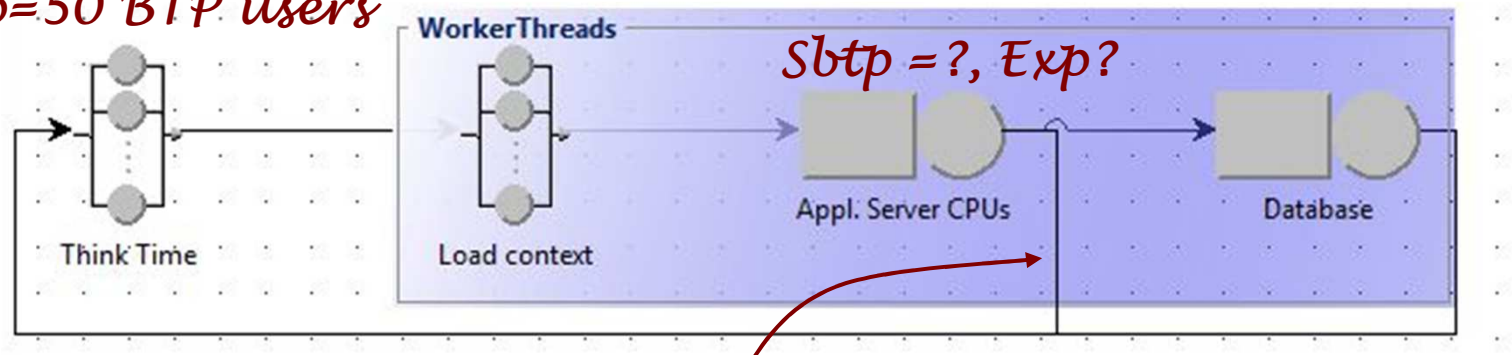


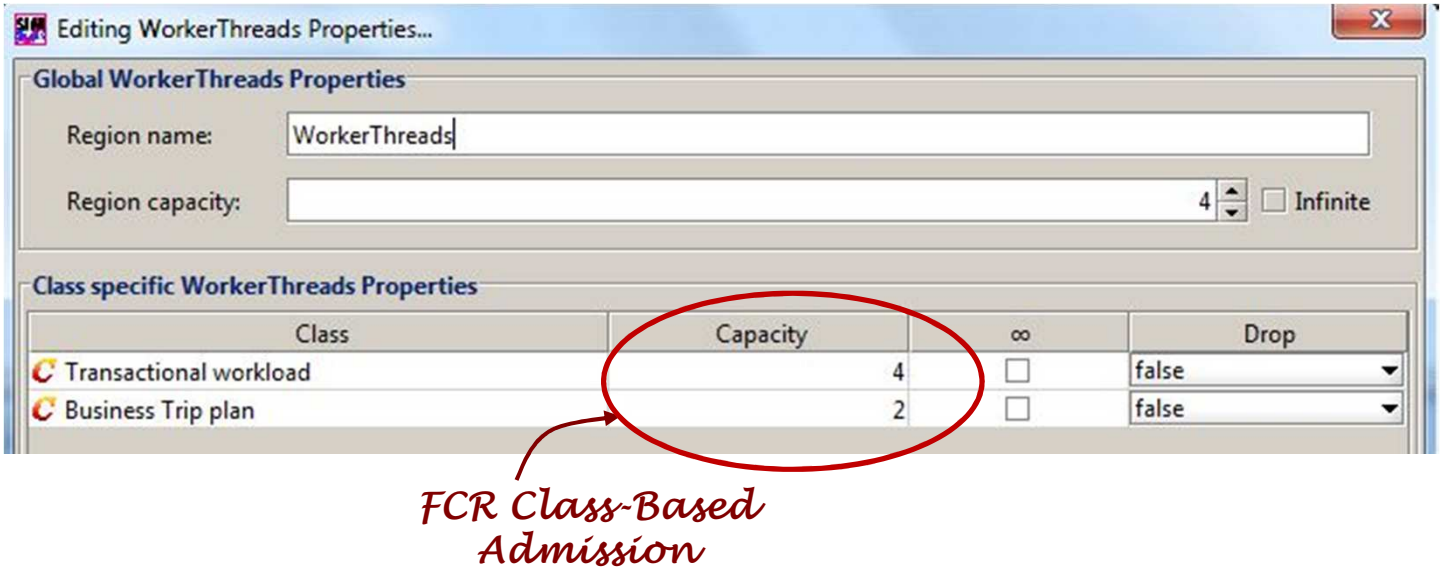
what-if analysis – adding a web service class

- some requests now access the service composition engine of the multi-tier application to create a business travel plan
- services are composed on the fly from external providers (travel agencies, flight booking service) according to a workflow
- worker thread remains busy for the entire duration of the web service workflow
- evaluate end-to-end response time for each class

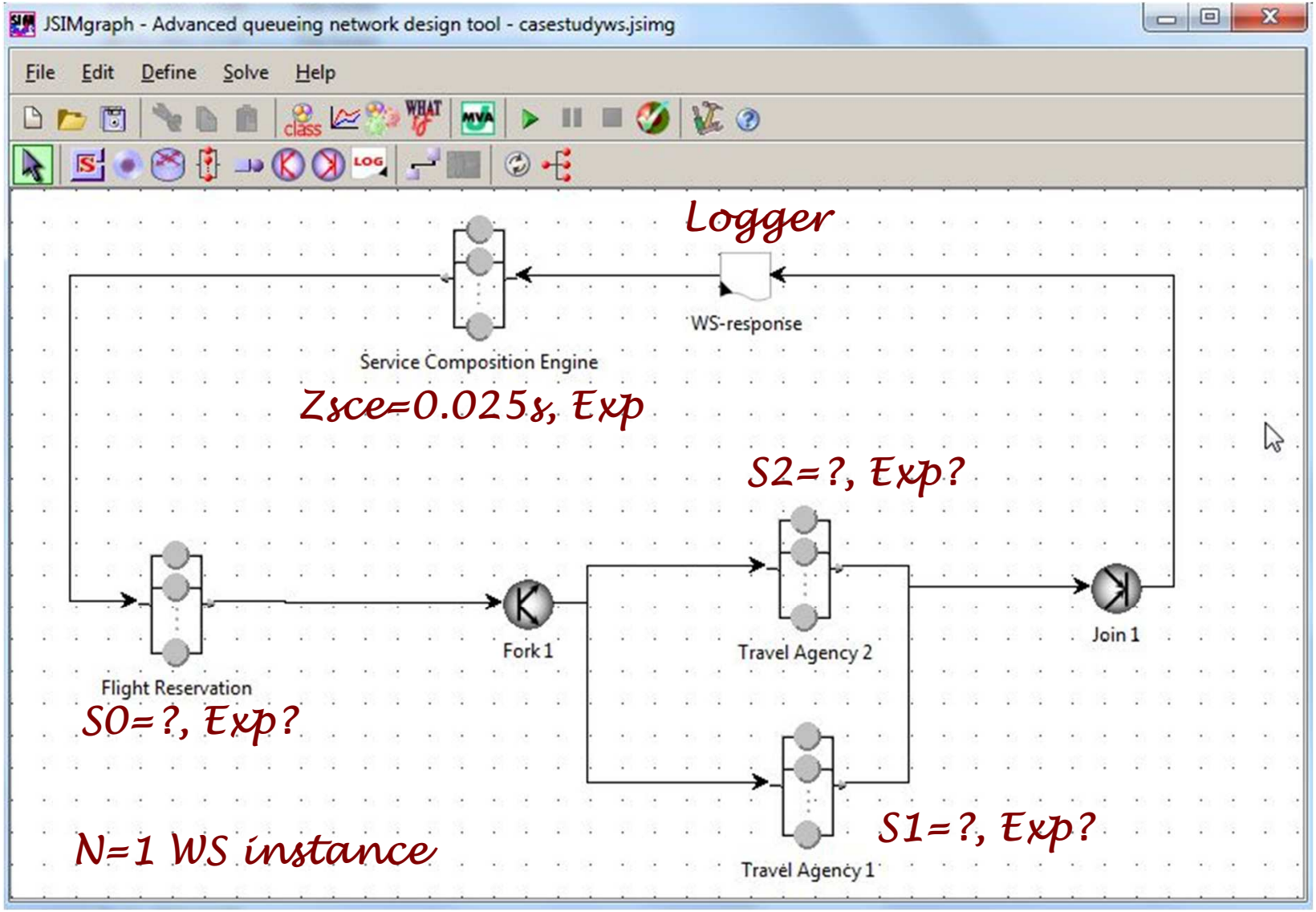
business trip planning (BTP) web service

N=300 app users
Nbtp=50 BTP users

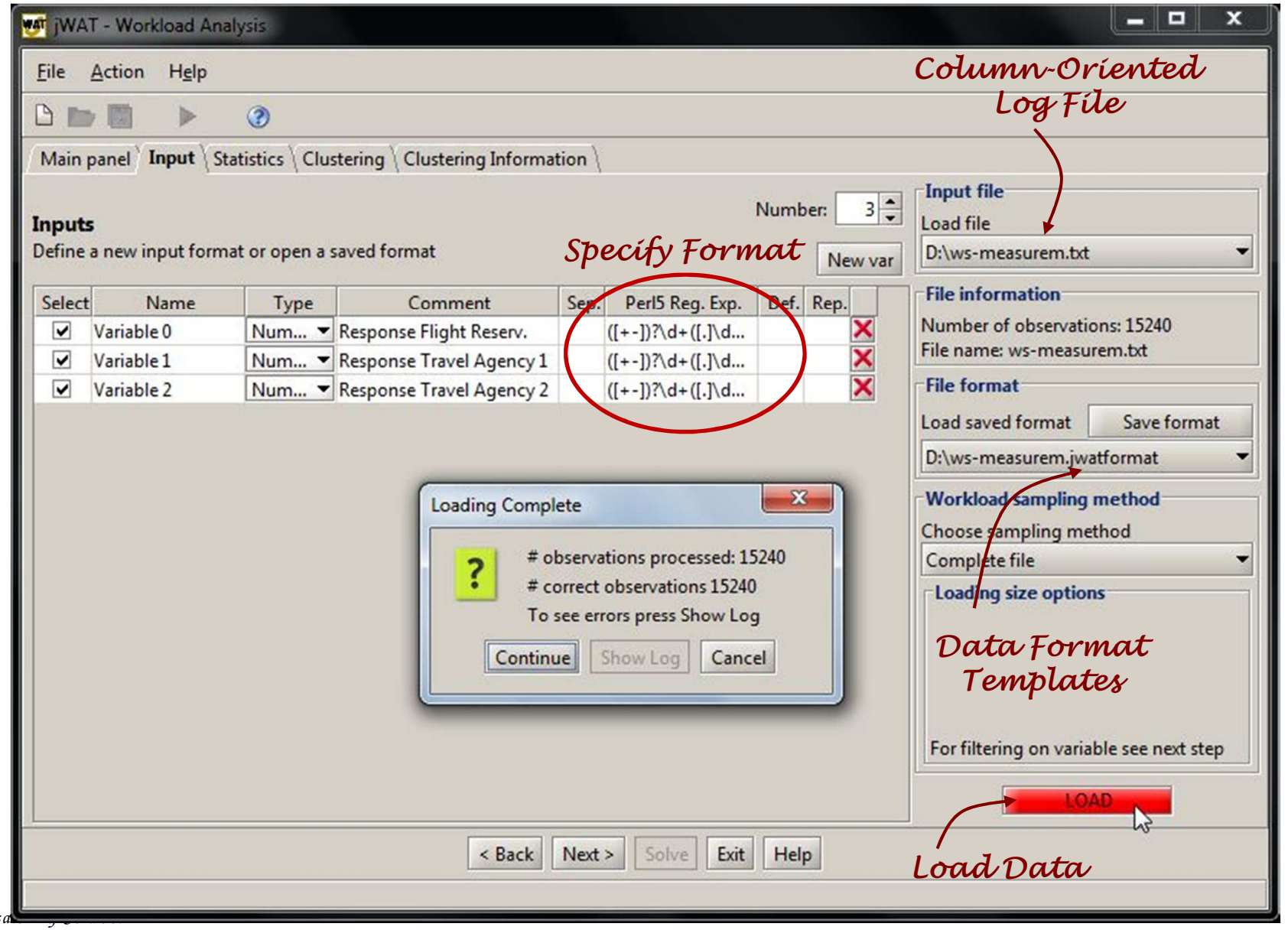




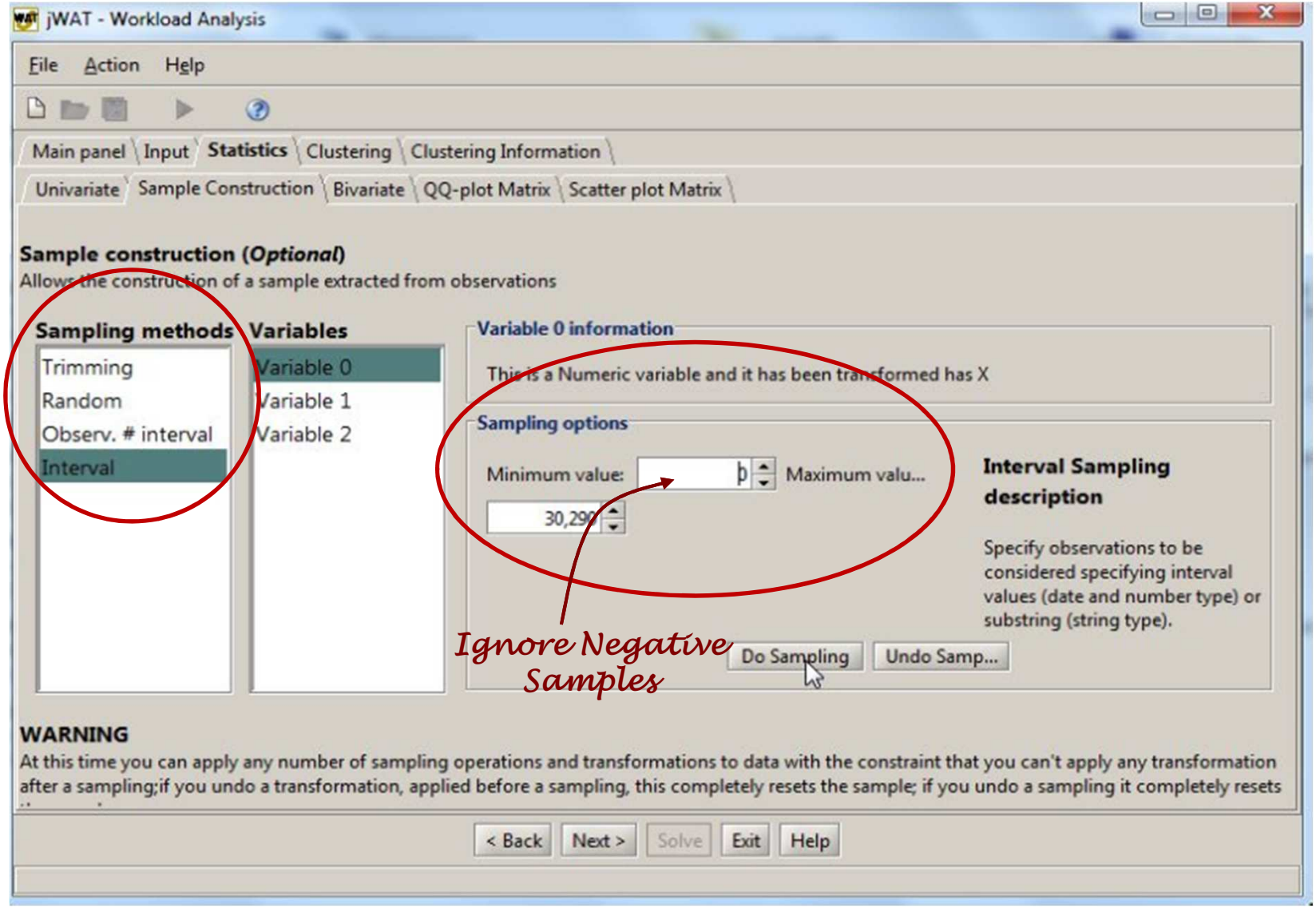
BTP web service sub-model



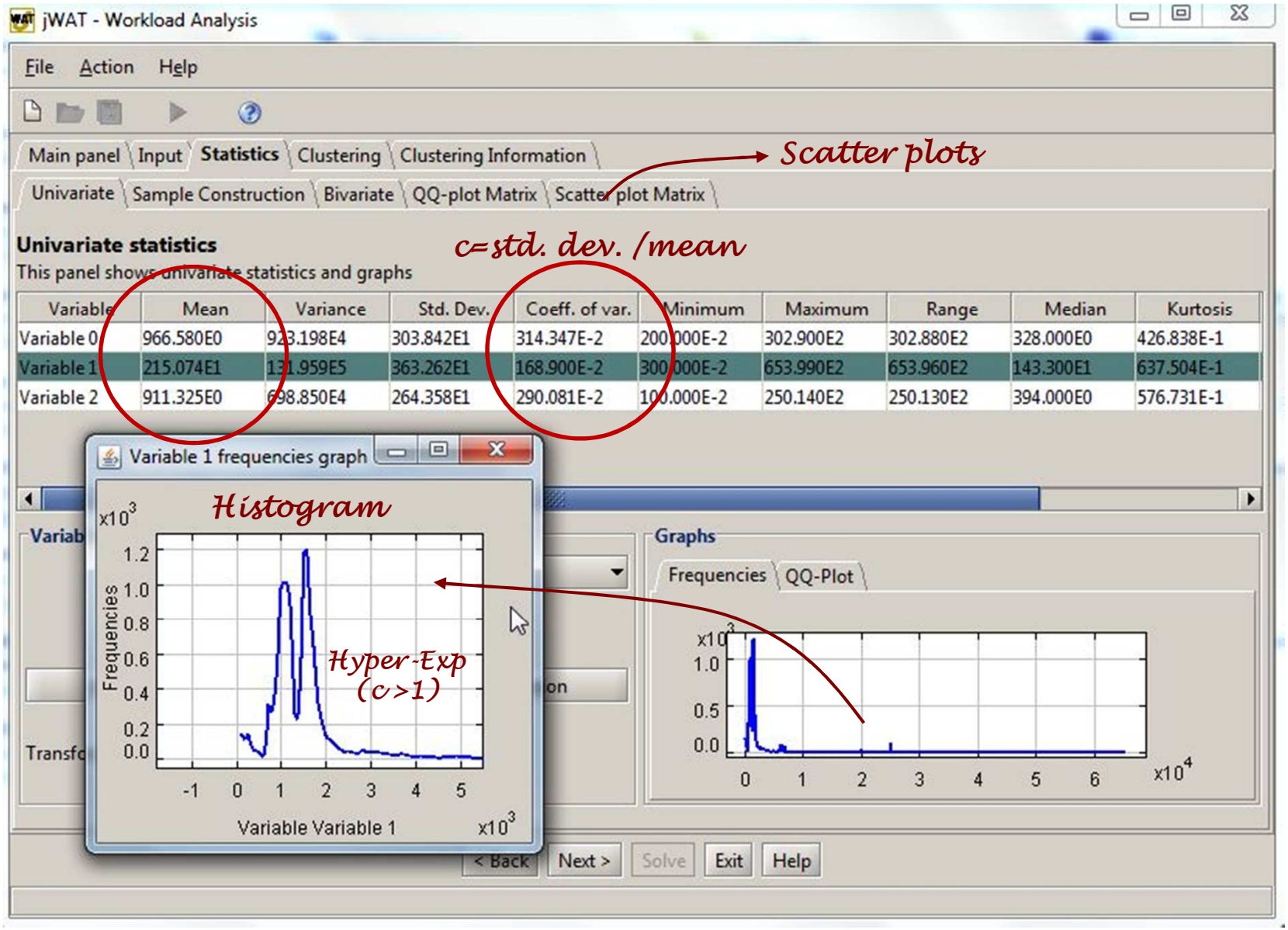
jWAT – Workload Analysis Tool



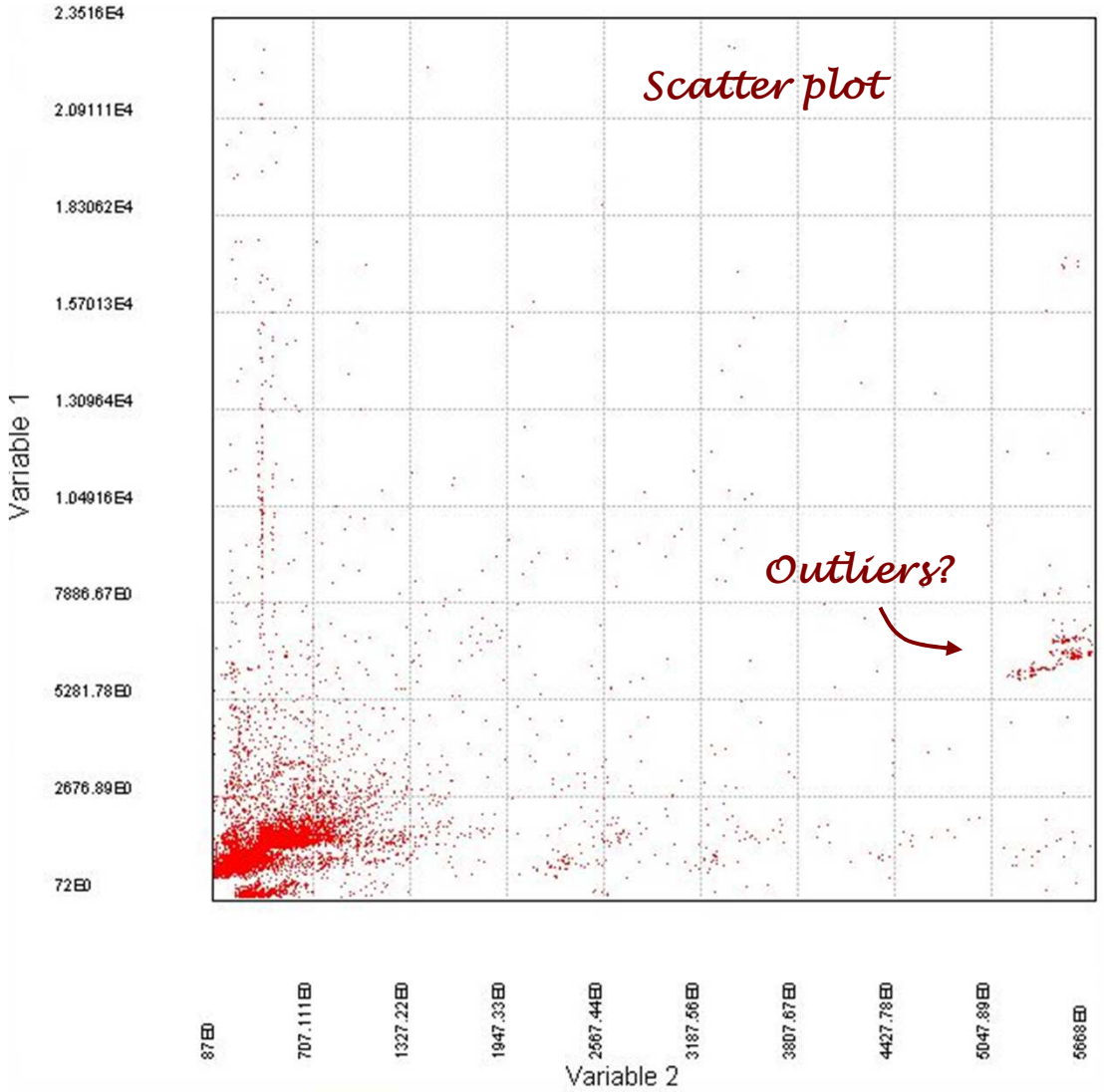
jWAT – data filtering



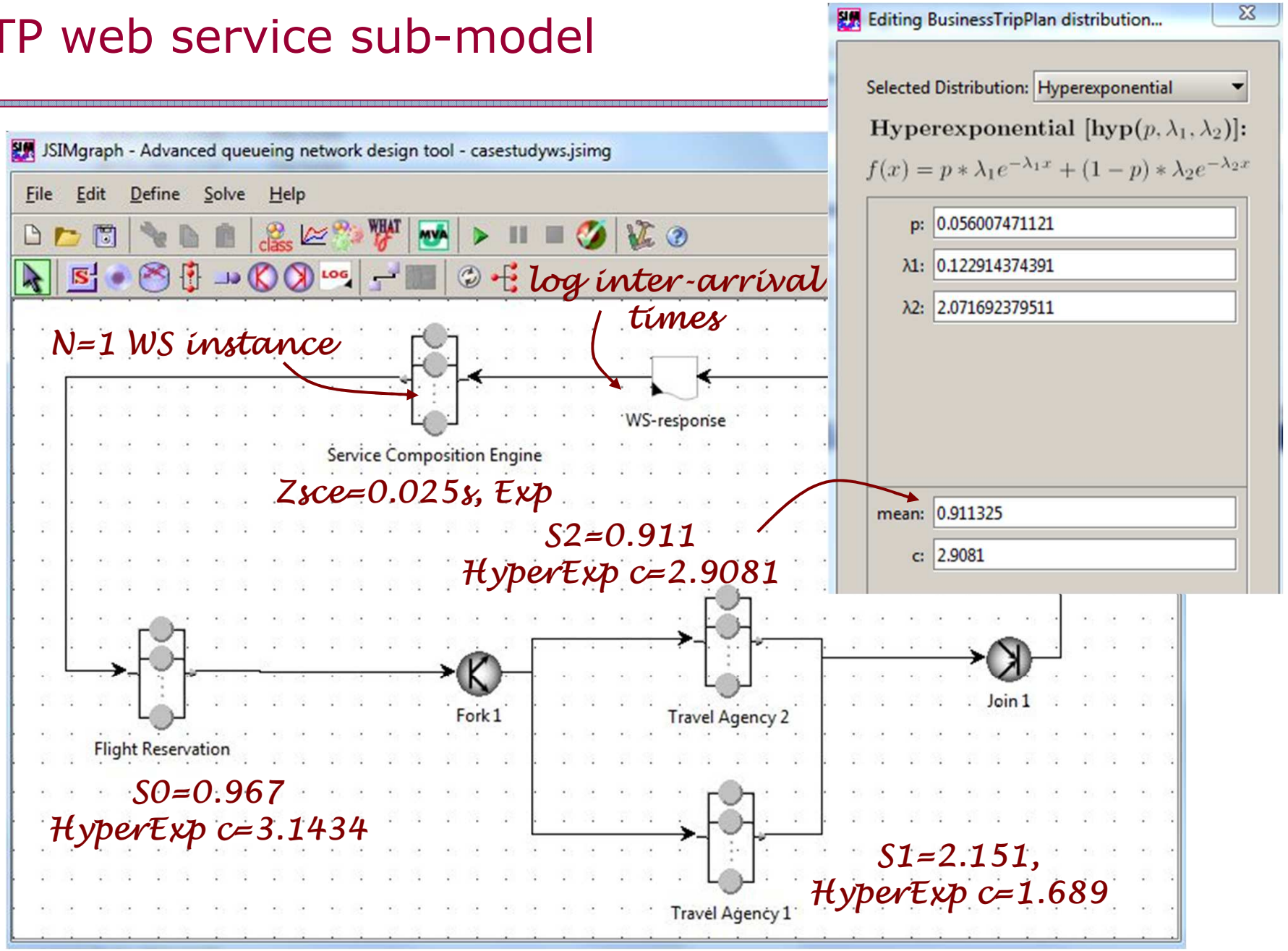
jWAT – descriptive statistics



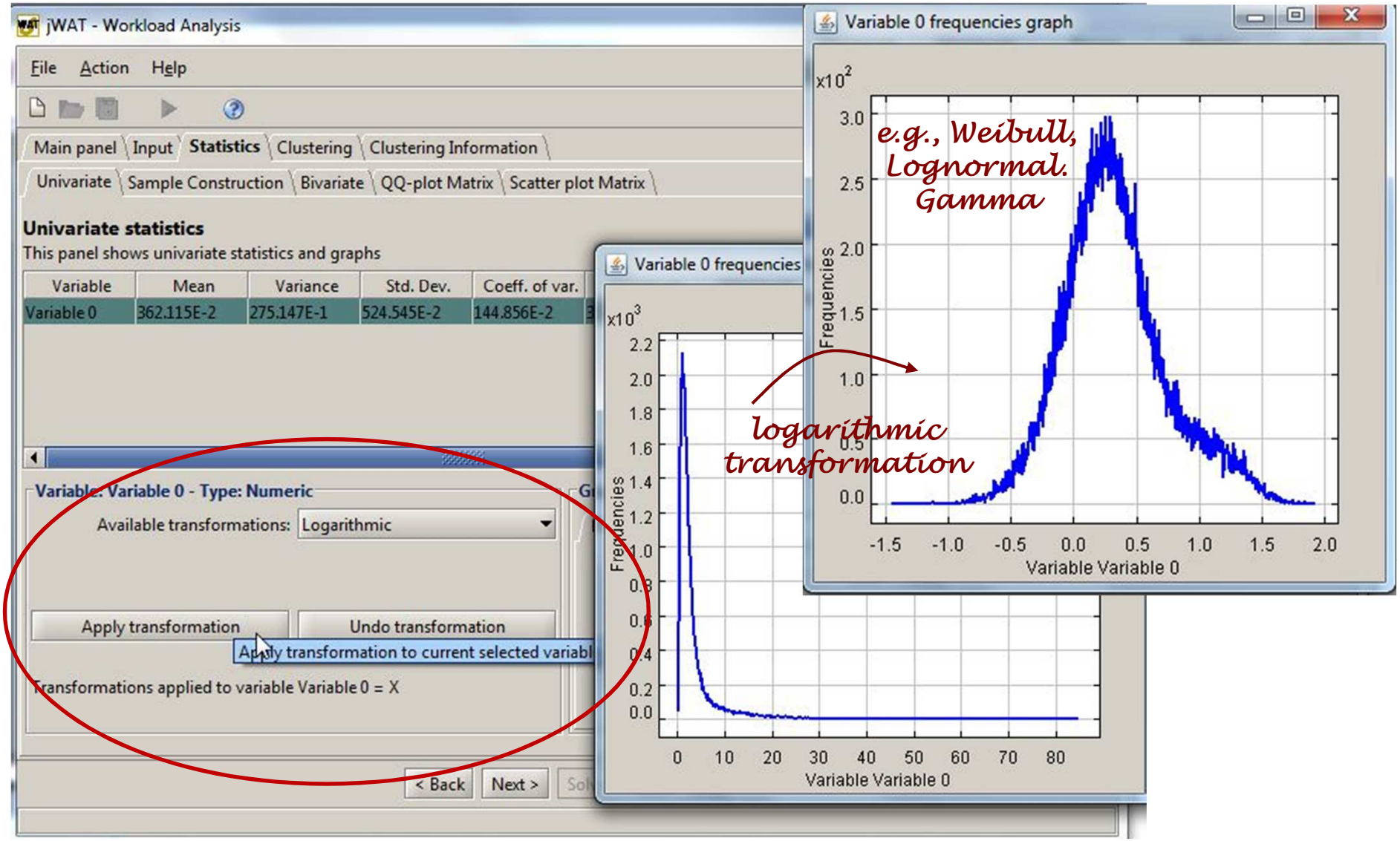
jWAT – scatter plot



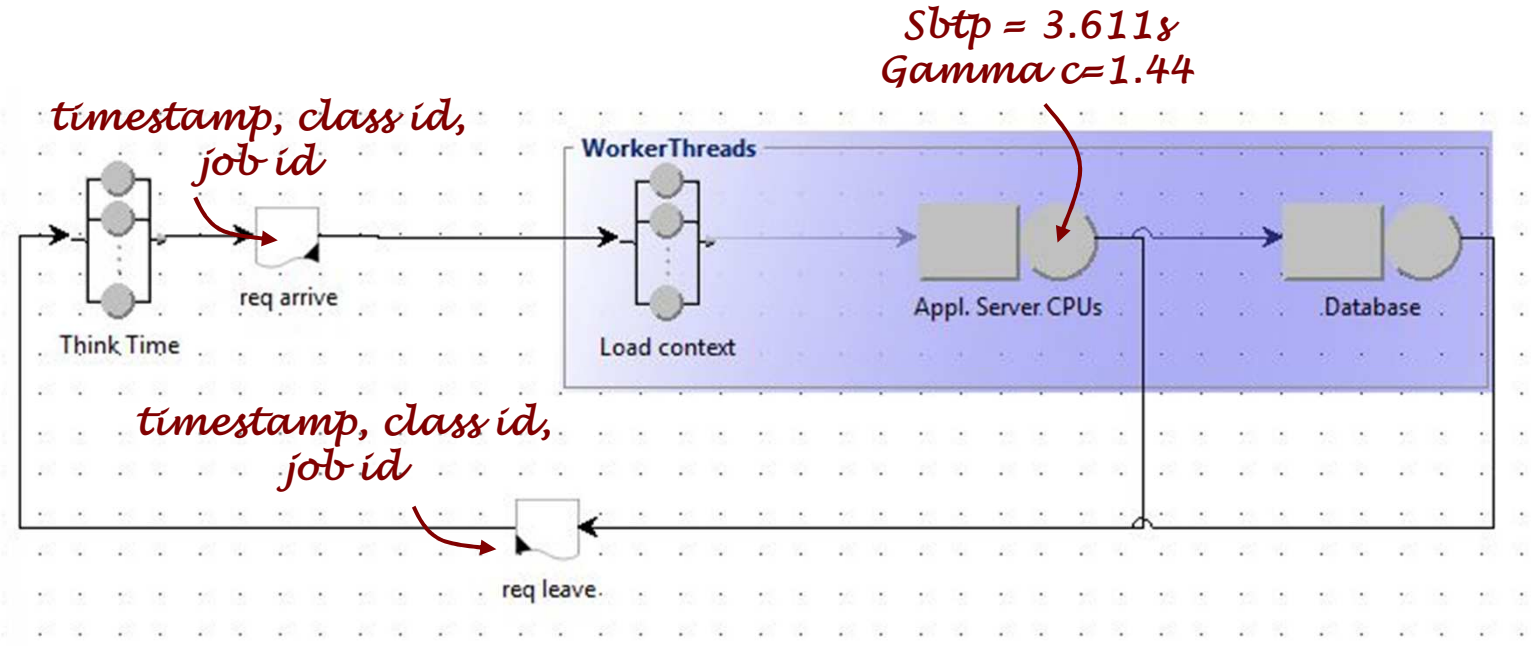
BTP web service sub-model



BTP response times



response time distribution – logger components



global.csv

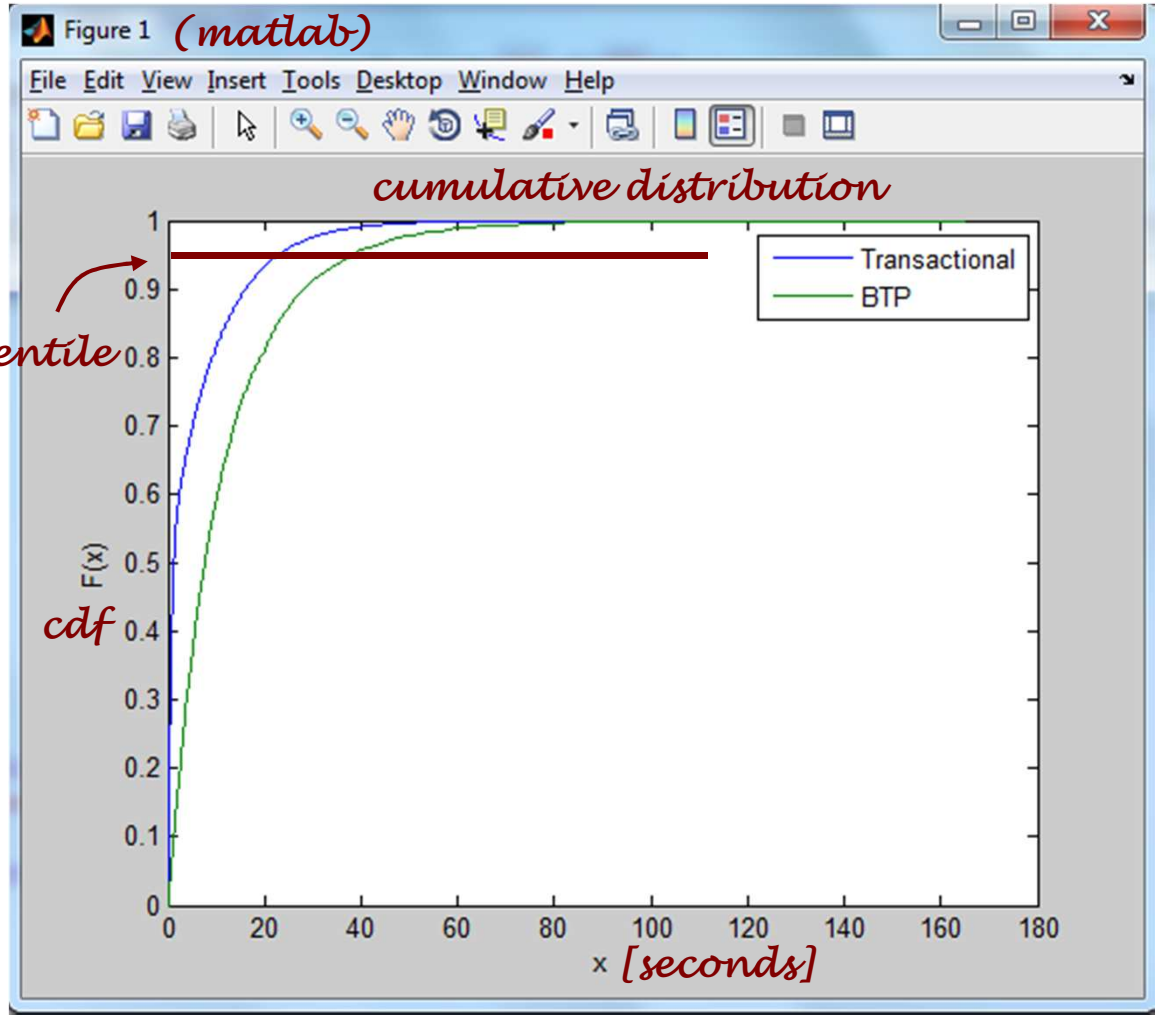
job id (same throughout simulation)

```
LOGGERNAME;TIMESTAMP;JOB_ID;CLASS_ID;INTERARRIVAL_SAMECLASS;INTERARRIVAL_ANYCLASS;SIMUL_START_TIME
req arrive;0.009420010041266342;253625;Transactional workload;;;
req leave;0.021755765433481?;253625;Transactional workload;;;
req arrive;0.031032734664243056;253498;Transactional workload;;;
req leave;0.049159953329098l4;253498;Transactional workload;;;
req arrive;0.07727161520772474;253542;Transactional workload;;;
```

logger id

job class

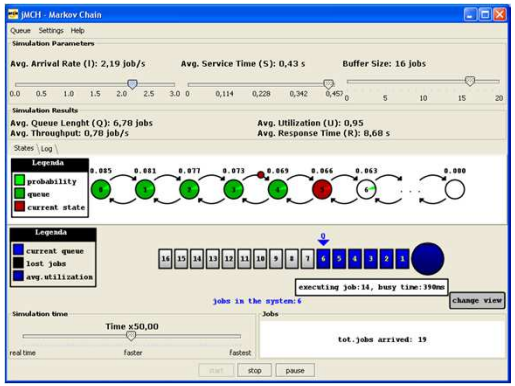
response time distribution analysis



CONCLUSION

Final remarks

- Analysis with Java Modelling Tools (http://jmt.sf.net)
  - Queueing network simulation
  - Bottlenecks identification
  - Workload analysis
  - Mean value analysis
  - ...
- JMT-Based examples and exercises (http://perflib.net)
- Topics not covered by this tutorial
  - jMCH
  - Burstiness analysis
  - Trace-driven simulation
  - ...
- JMT discussion forum: http://sourceforge.net/forum/?group_id=163838



References

- *G.Casale, G.Serazzi. Quantitative System Evaluation with Java Modelling Tools (Tutorial).* in Proc. of ACM/SPEC ICPE 2011 (companion paper).
- *M.Bertoli, G.Casale, G.Serazzi. User-Friendly Approach to Capacity Planning Studies with Java Modelling Tools,* in Proc. of SIMUTOOLS 2009.
- *M.Bertoli, G.Casale, G.Serazzi. JMT - Performance Engineering Tools for System Modeling.* ACM Perf. Eval. Rev., 36(4), 2009
- *M.Bertoli, G.Casale, G.Serazzi. The JMT Simulator for Performance Evaluation of Non Product-Form Queueing Networks,* in Proc. of SCS Annual Simulation Symposium 2007, 3-10, Norfolk, VA, Mar 2007.
- *M.Bertoli, G.Casale, G.Serazzi. Java Modelling Tools: an Open Source Suite for Queueing Network Modelling and Workload Analysis,* in Proc. of QEST 2006, 119-120, Sep 2006.
- E.Lazowska, J.Zahorjan, G.S.Graham, K.C.Sevcik, *Quantitative System Performance: Computer System Analysis Using Queueing Network Models*, Prentice-Hall, 1994.
- K.Pawlikowski: Steady-State Simulation of Queuing Processes: A *Survey* of Problems and Solutions. *ACM Comput. Surv.* 22(2): 123-170, 1990.
- P.Heidelberger and P.D.Welch. A spectral method for confidence interval generation and run length control in simulations. *Comm. ACM.* 24, 233-245, 1981.
- S.C.Spratt. Heuristics for the startup problem. *M.S. Thesis*, Department of Systems Engineering, University of Virginia, 1998.

Contact us!

g.casale@imperial.ac.uk
giuseppe.serazzi@polimi.it