Autonomic Computing

Daniel A. Menascé
Department of Computer Science
George Mason University

Credits

- My former Ph.D. students:
  - Mohamed Bennani, Oracle
  - Ron Dodge, West Point

Motivation for AC

- "…main obstacle to further progress in IT is a looming software complexity crisis." (from an IBM manifesto, Oct. 2001).
  - Tens of millions of lines of code
  - Skilled IT professionals required to install, configure, tune, and maintain.
  - Need to integrate many heterogeneous systems
  - Limit of human capacity being achieved

Motivation for AC (cont'd)

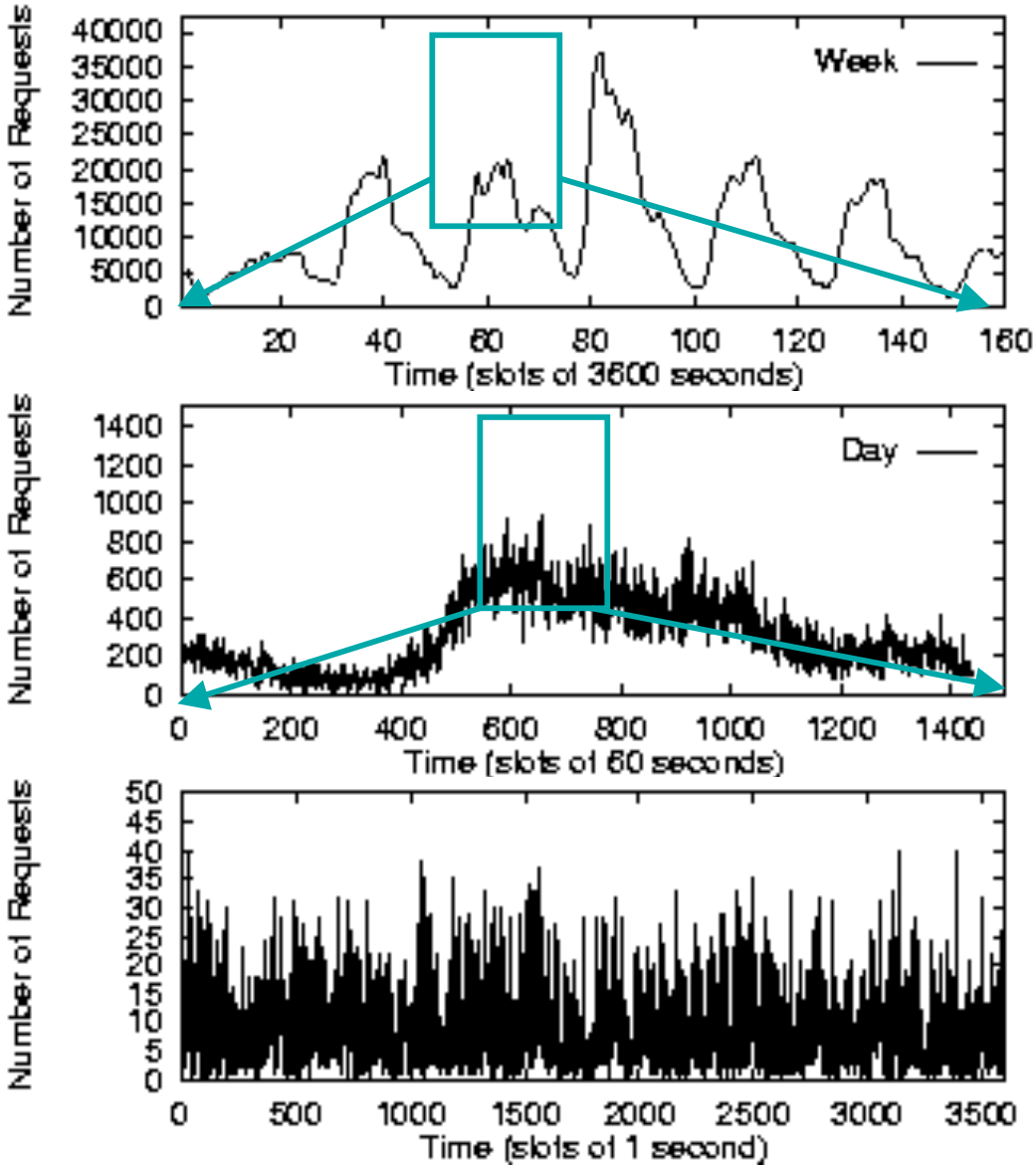
- Harder to anticipate interactions between components at design time:
  - Need to defer decisions to run time
- Computer systems are becoming too massive, complex, to be managed even by the most skilled IT professionals
- The workload and environment conditions tend to change very rapidly with time

Multi-scale time workload variation of a Web Server

3600 sec

60 sec

1 sec



Self-Optimizing Systems

- Complex middleware and database systems have a very large number of configurable parameters.

Web Server (IIS 5.0)	**Application Server (Tomcat 4.1)**	**Database Server (SQL Server 7.0)**
HTTP KeepAlive	acceptCount	Cursor Threshold
Application Protection Level	minProcessors	Fill Factor
Connection Timeout	maxProcessors	Locks
Number of Connections		Max Worker Threads
Logging Location		Min Memory Per Query
Resource Indexing		Network Packet Size
Performance Tuning Level		Priority Boost
Application Optimization		Recovery Interval
MemCacheSize		Set Working Set Size
MaxCachedFileSize		Max Server Memory
ListenBacklog		Min Server Memory
MaxPoolThreads		User Connections
worker.ajp13.cachesize		

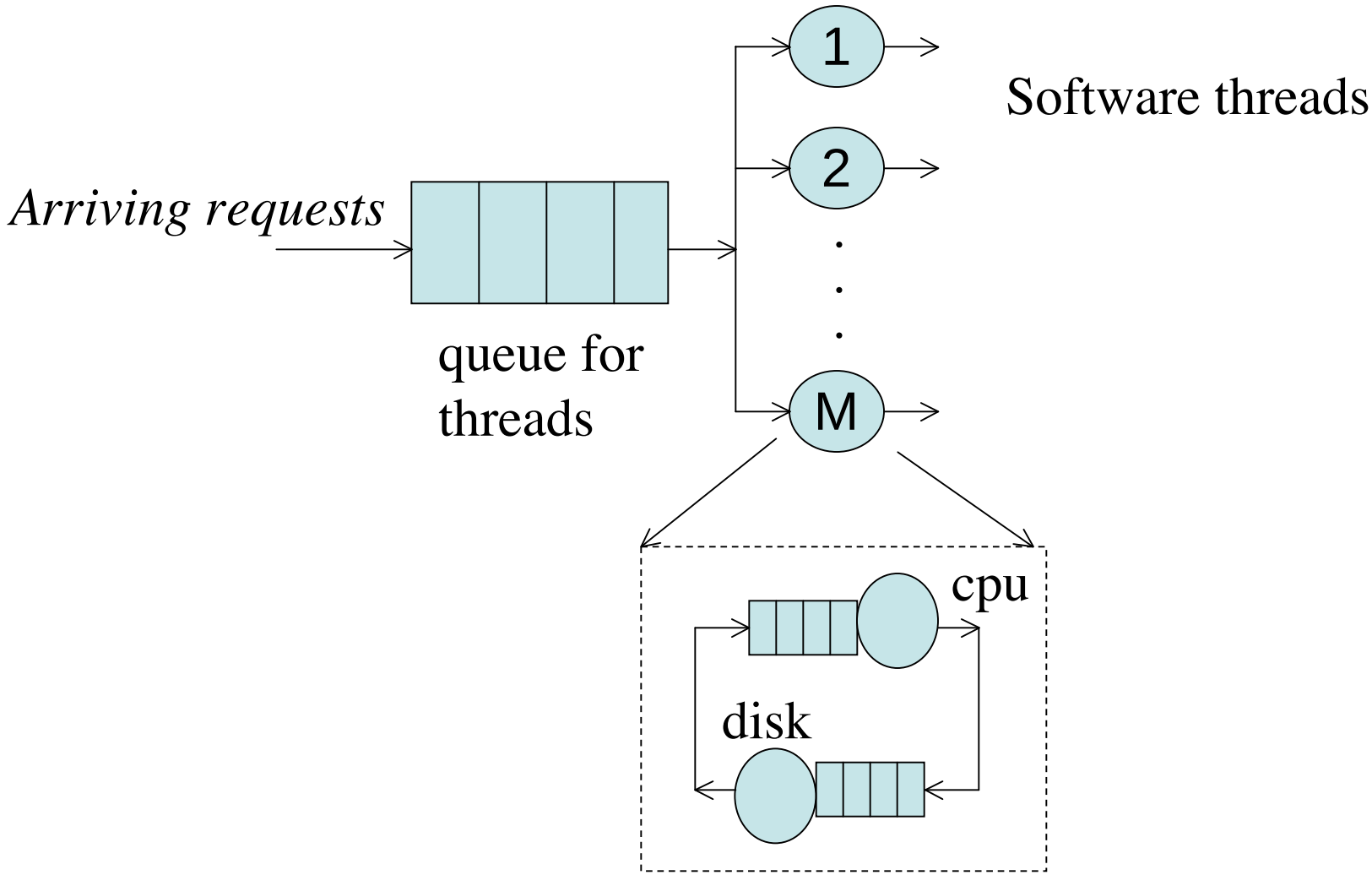
Autonomic Computing

- System that can manage themselves given high-level objectives.
  - High-level objectives can be expressed in term of service-level objectives or utility functions.
- Autonomic computing is inspired in the human autonomic nervous system:
  - "The autonomic nervous system consists of sensory neurons and motor neurons that run between the central nervous system and various internal organs such as the: heart, lungs, viscera, glands. It is responsible for monitoring conditions in the internal environment and bringing about appropriate changes in them. The contraction of both smooth muscle and cardiac muscle is controlled by motor neurons of the autonomic system."
    http://users.rcn.com/jkimball.ma.ultranet/BiologyPages/P/PNS.html
  - The autonomic nervous system functions in an involuntary, reflexive manner.

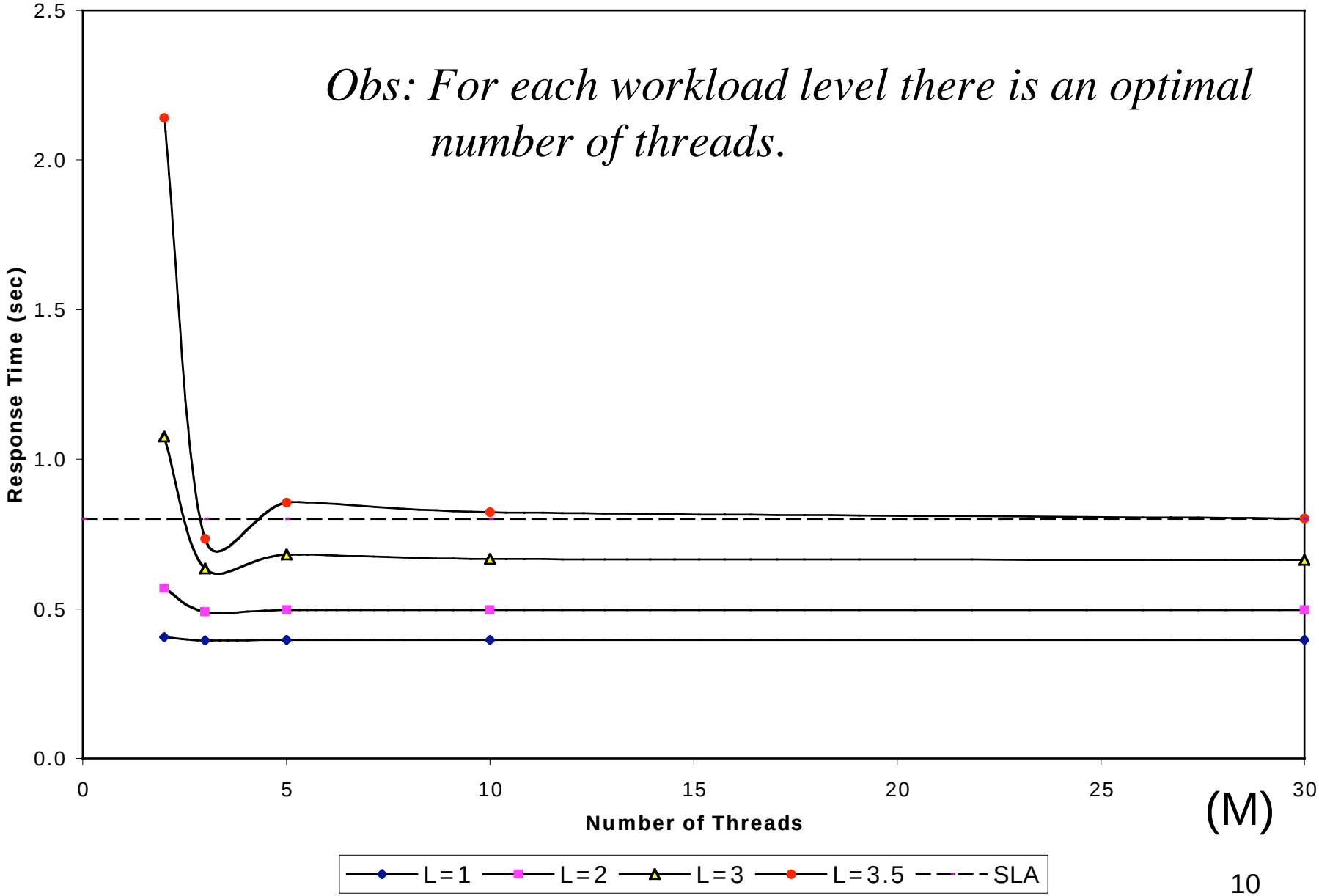
Autonomic Systems

- Self-managing
  - Self-configuring
  - Self-optimizing
  - Self-healing
  - Self-protecting
- Self-* systems

Self-optimizing systems: motivation

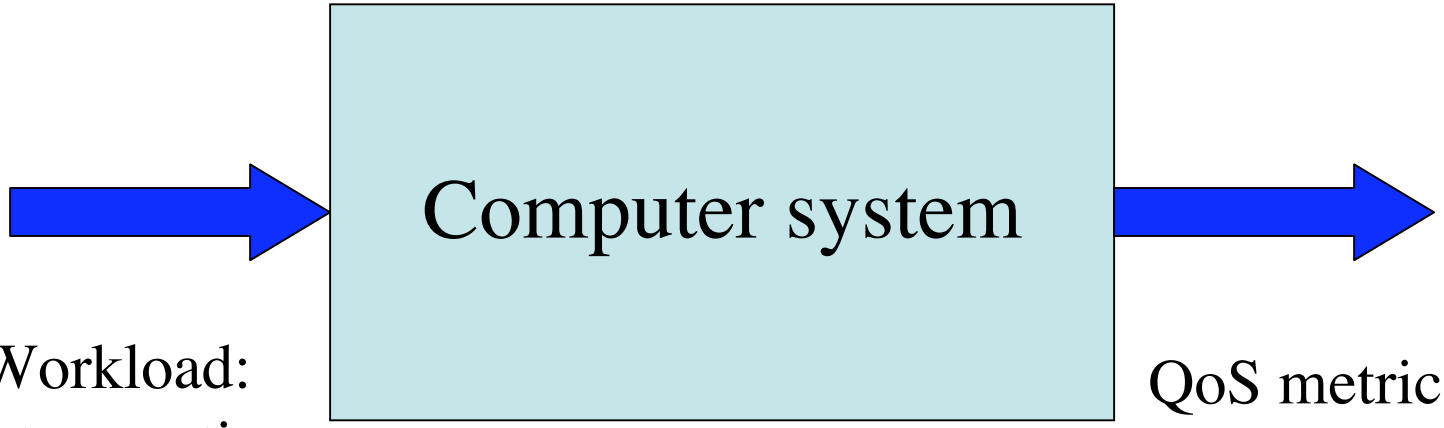


Q: How does the response time vary with the number of software threads, M?



(M)

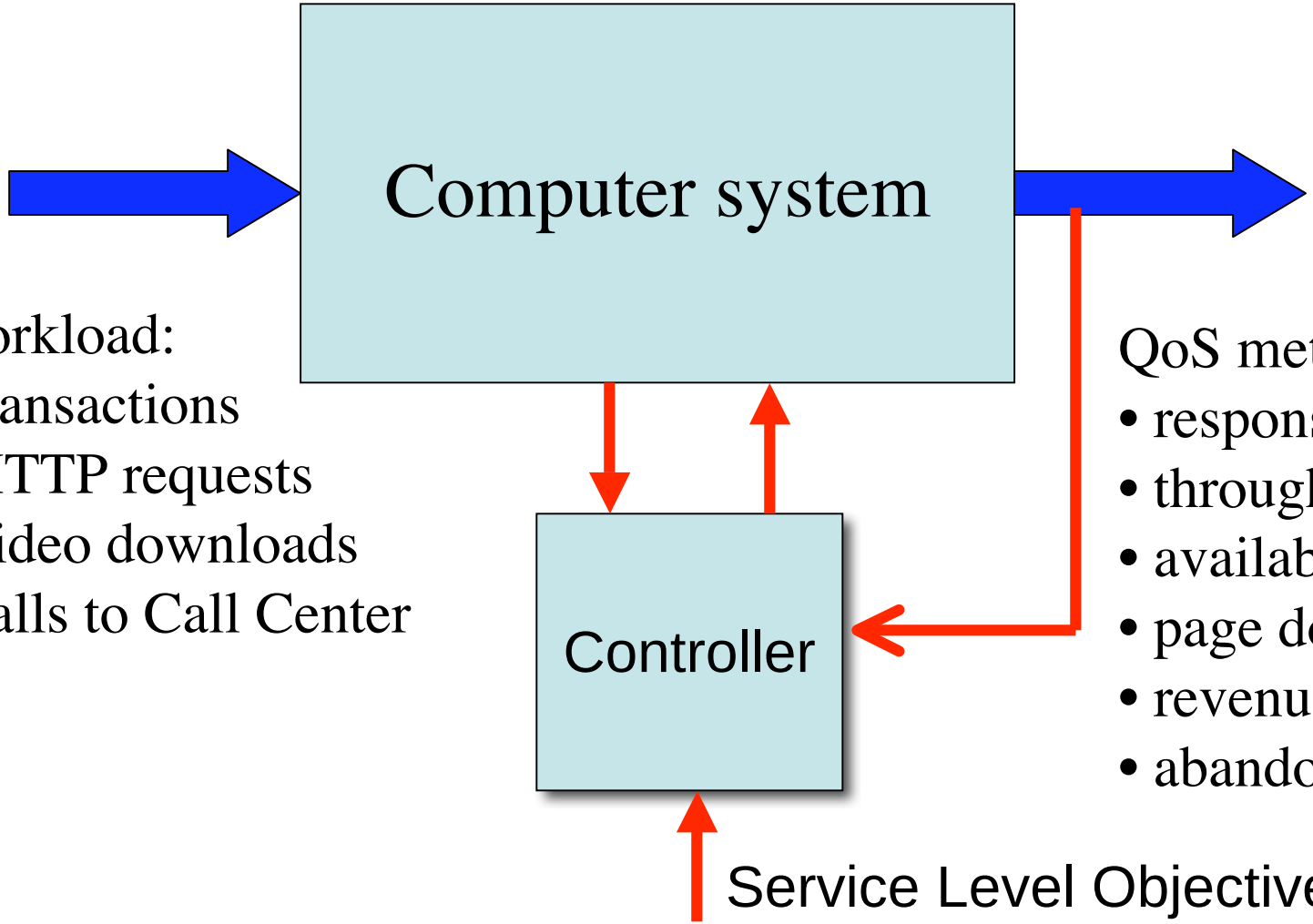
Workload and QoS metrics



Workload:
- transactions
- HTTP requests
- video downloads
- calls to Call Center

QoS metrics:
- response time
- throughput
- availability
- page download time
- revenue throughput
- abandonment rate

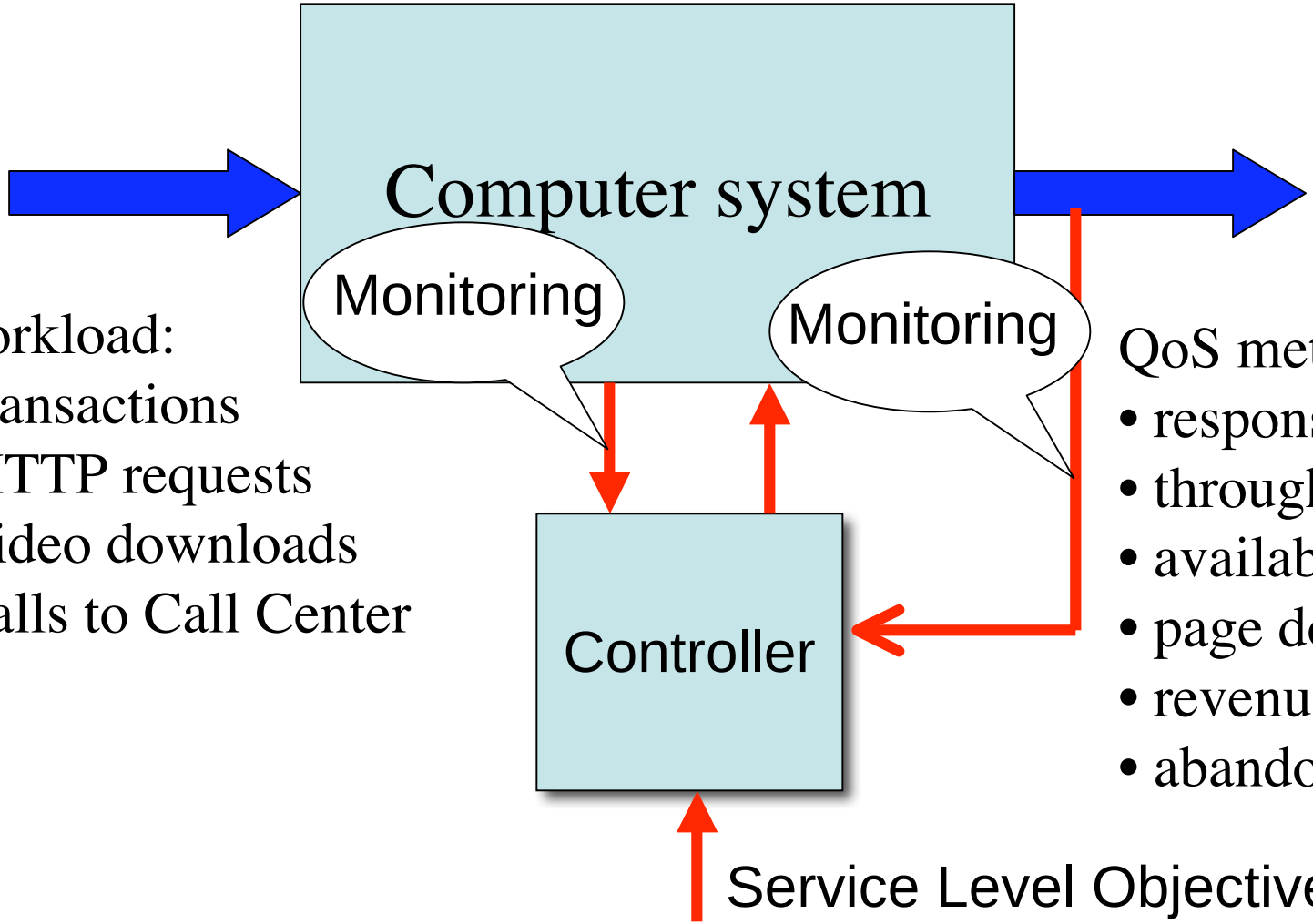
Self-optimizing System



Workload:
- transactions
- HTTP requests
- video downloads
- calls to Call Center

QoS metrics:
- response time
- throughput
- availability
- page download time
- revenue throughput
- abandonment rate

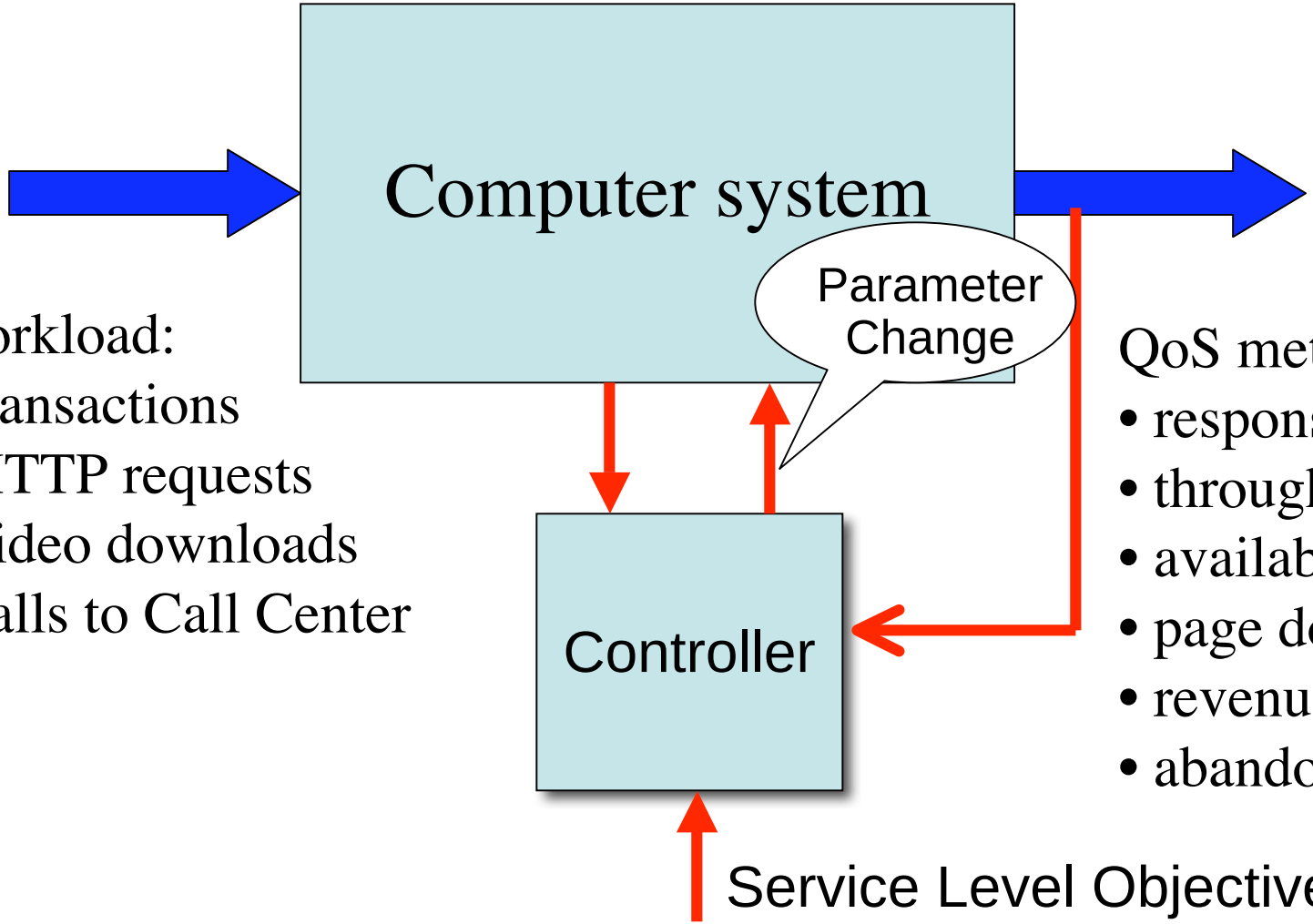
Self-optimizing System



Workload:
- transactions
- HTTP requests
- video downloads
- calls to Call Center

QoS metrics:
- response time
- throughput
- availability
- page download time
- revenue throughput
- abandonment rate

Self-optimizing System



Workload:
- transactions
- HTTP requests
- video downloads
- calls to Call Center

QoS metrics:
- response time
- throughput
- availability
- page download time
- revenue throughput
- abandonment rate

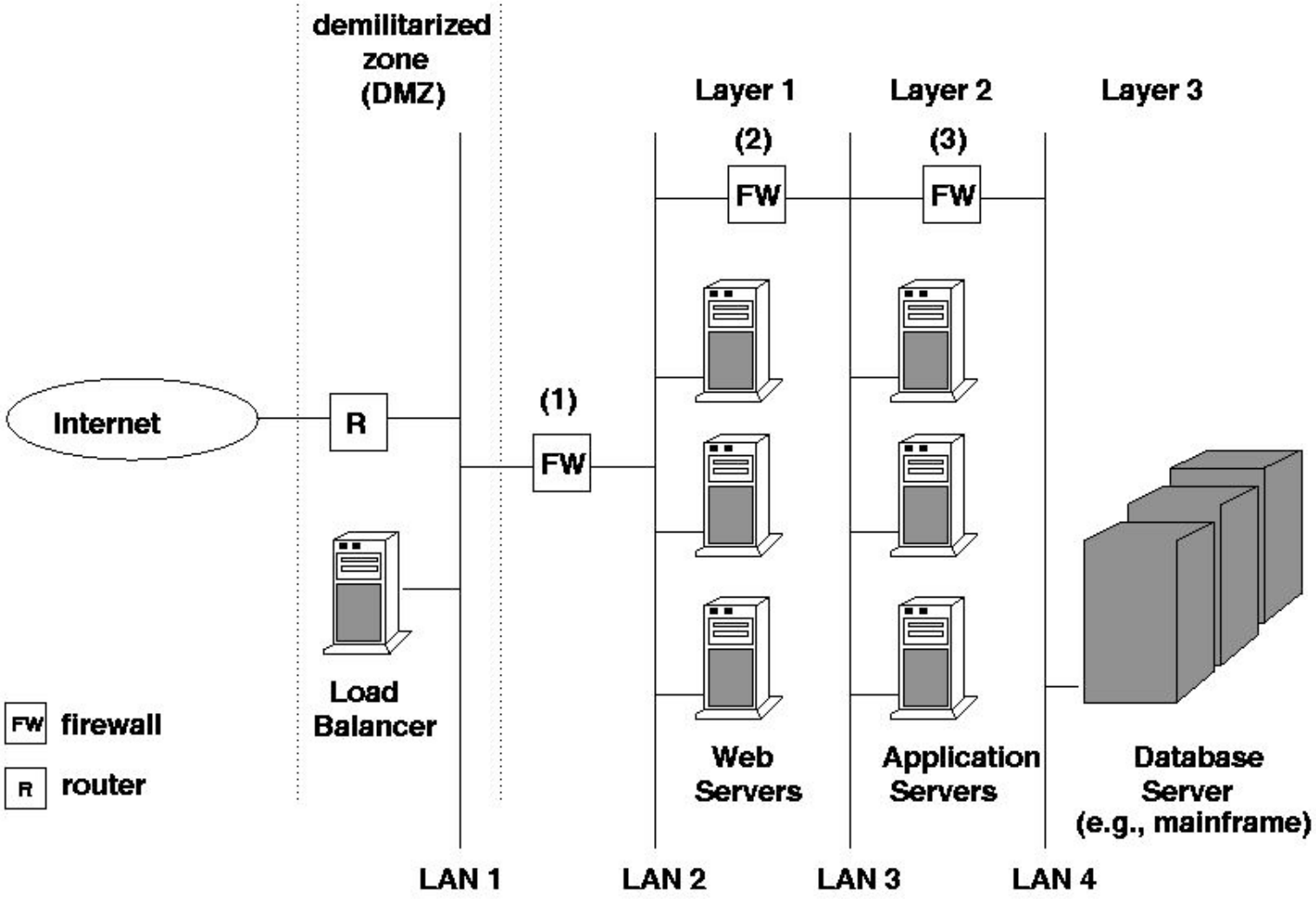
Examples of Autonomic Computing Systems

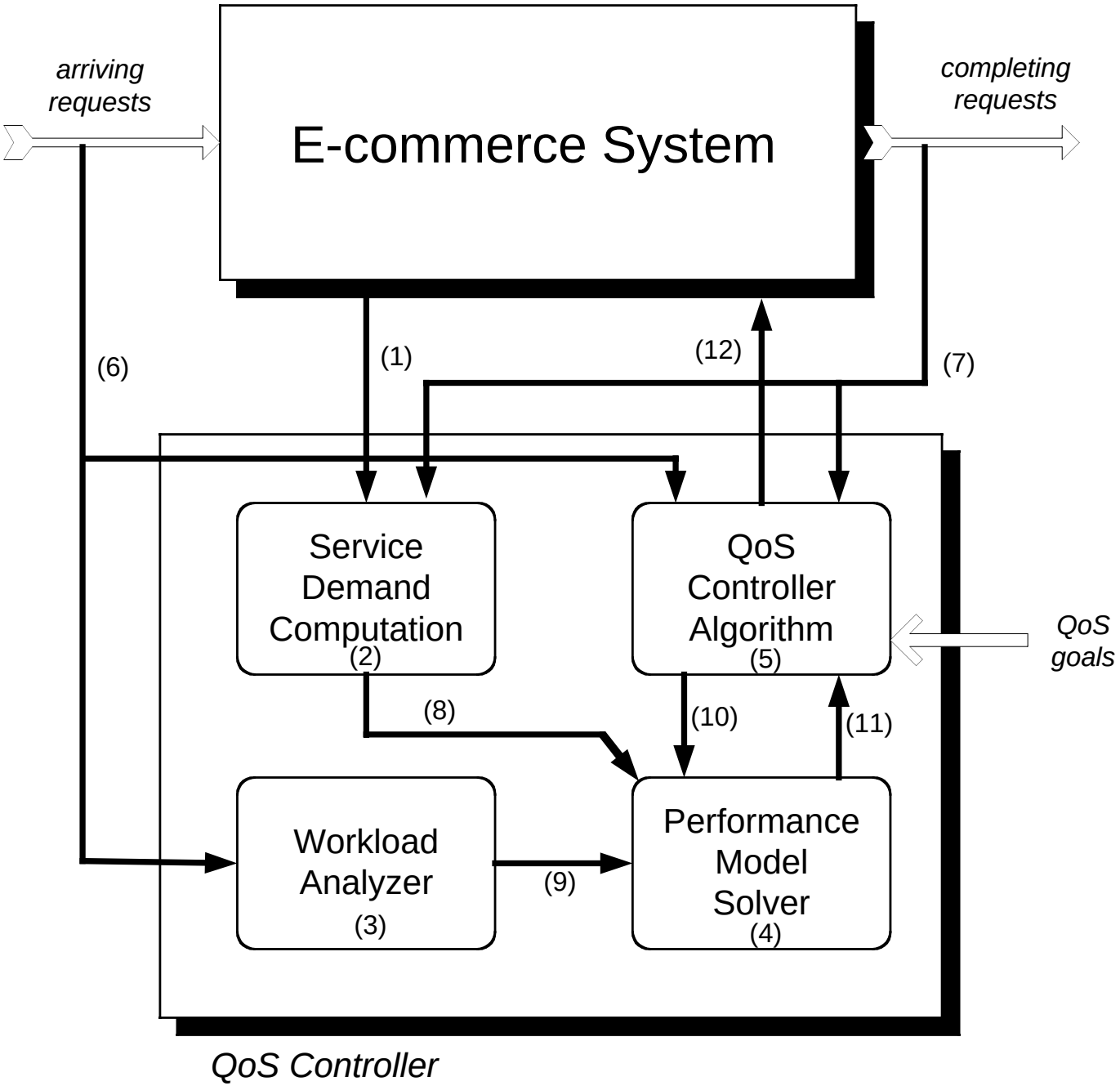
- E-commerce system
- Internet Data Center
- Virtualized Environment

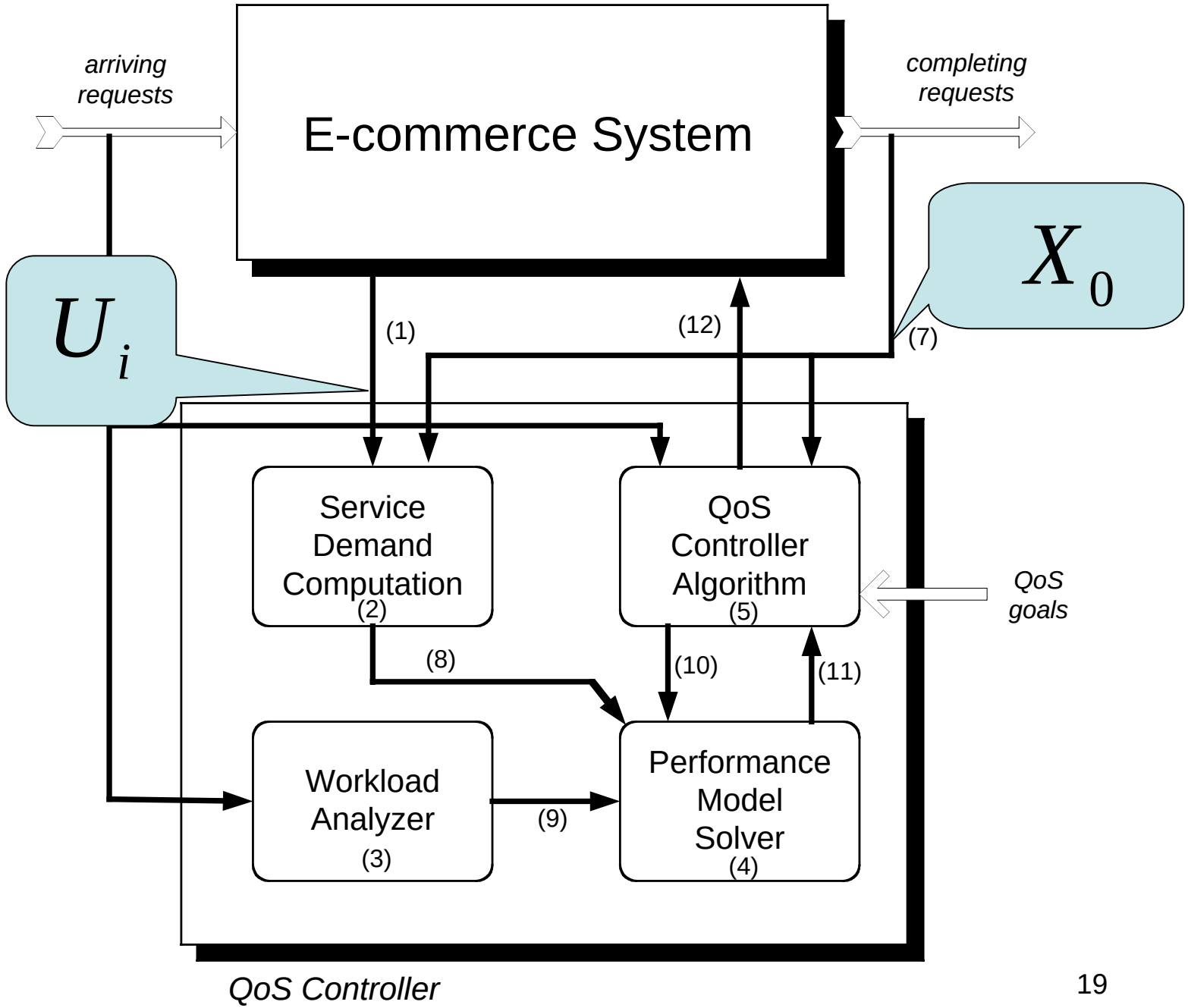
Examples of Autonomic Computing Systems

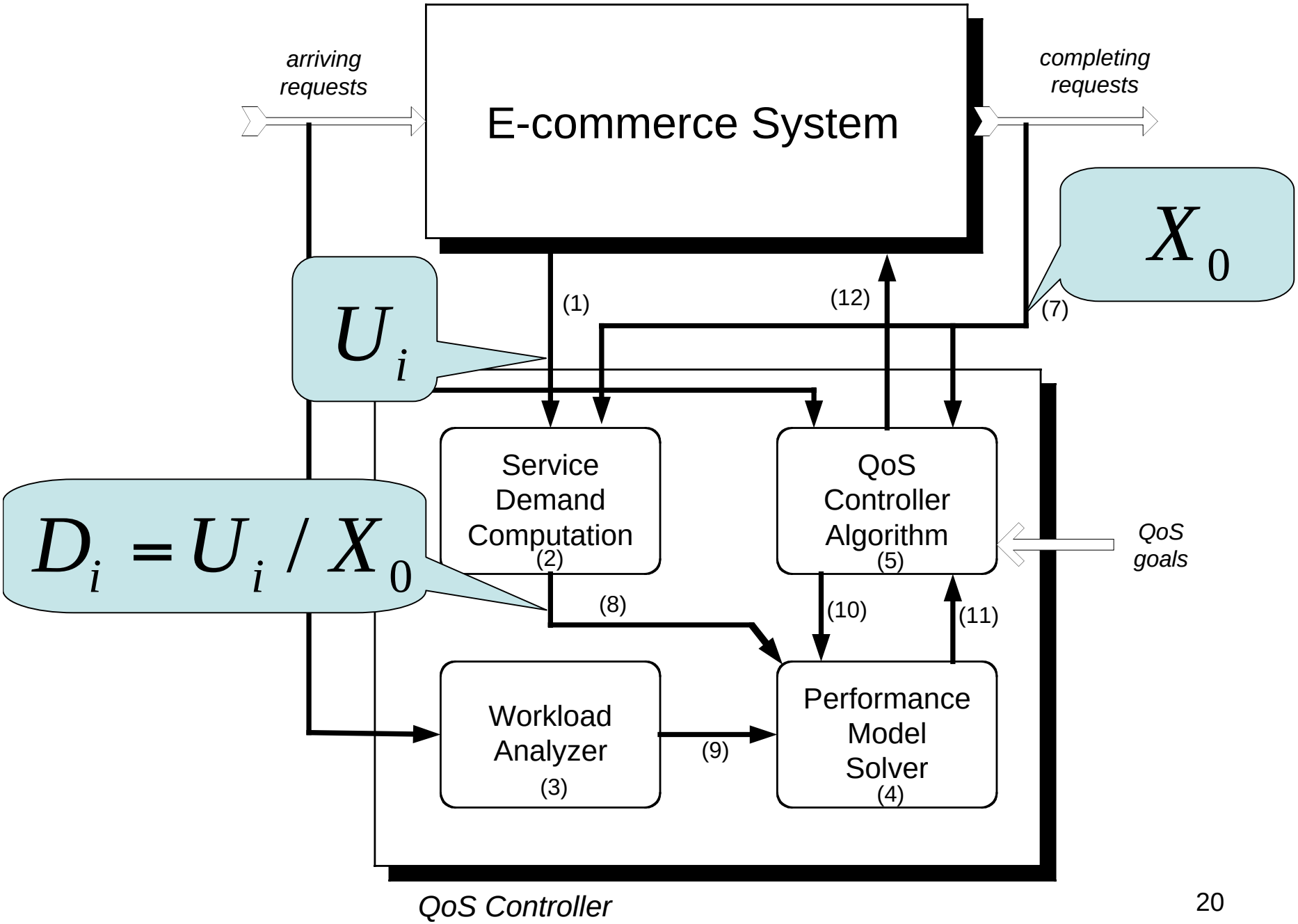
- E-commerce system
- Internet Data Center
- Virtualized Environment

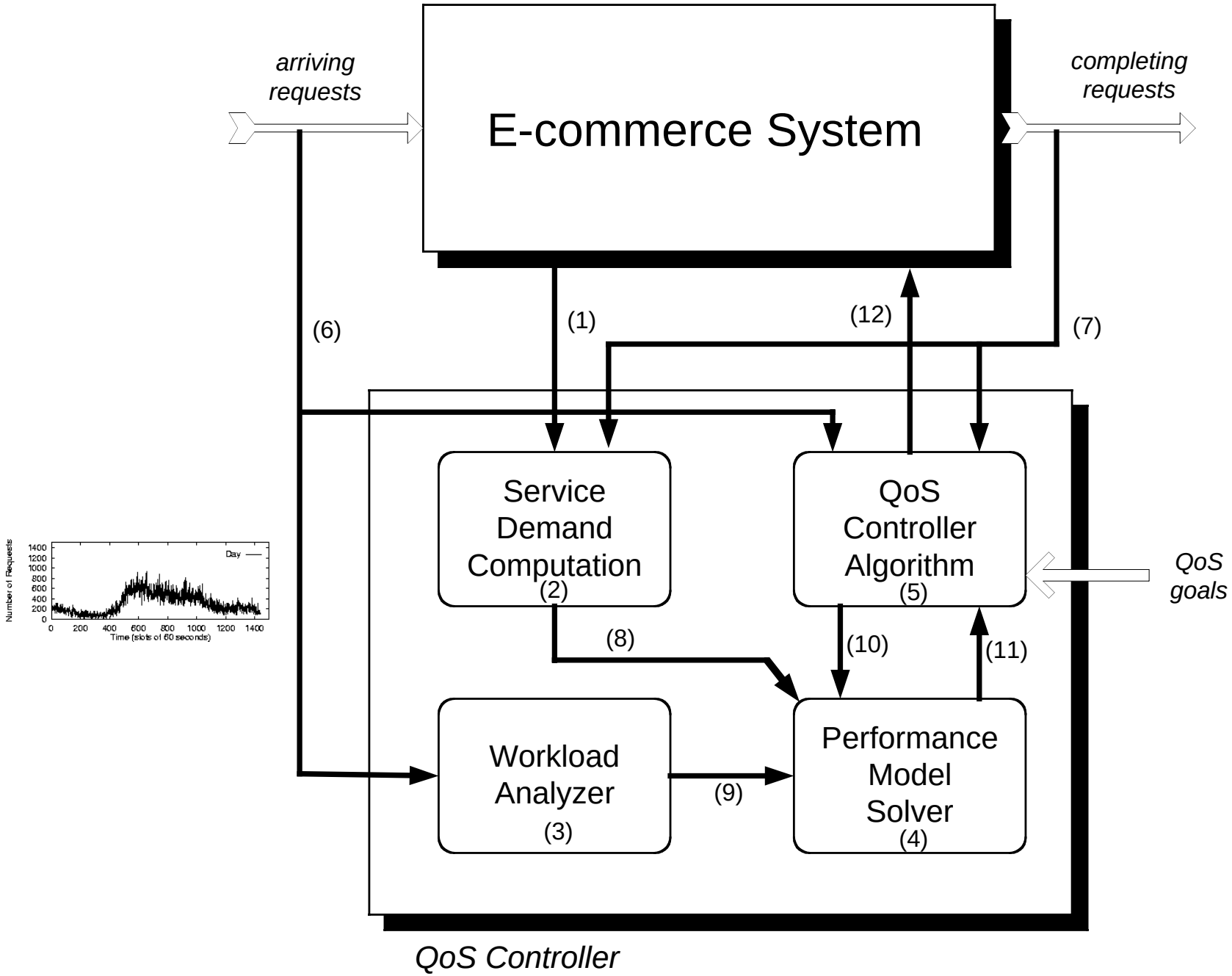
Multi-tier Architecture

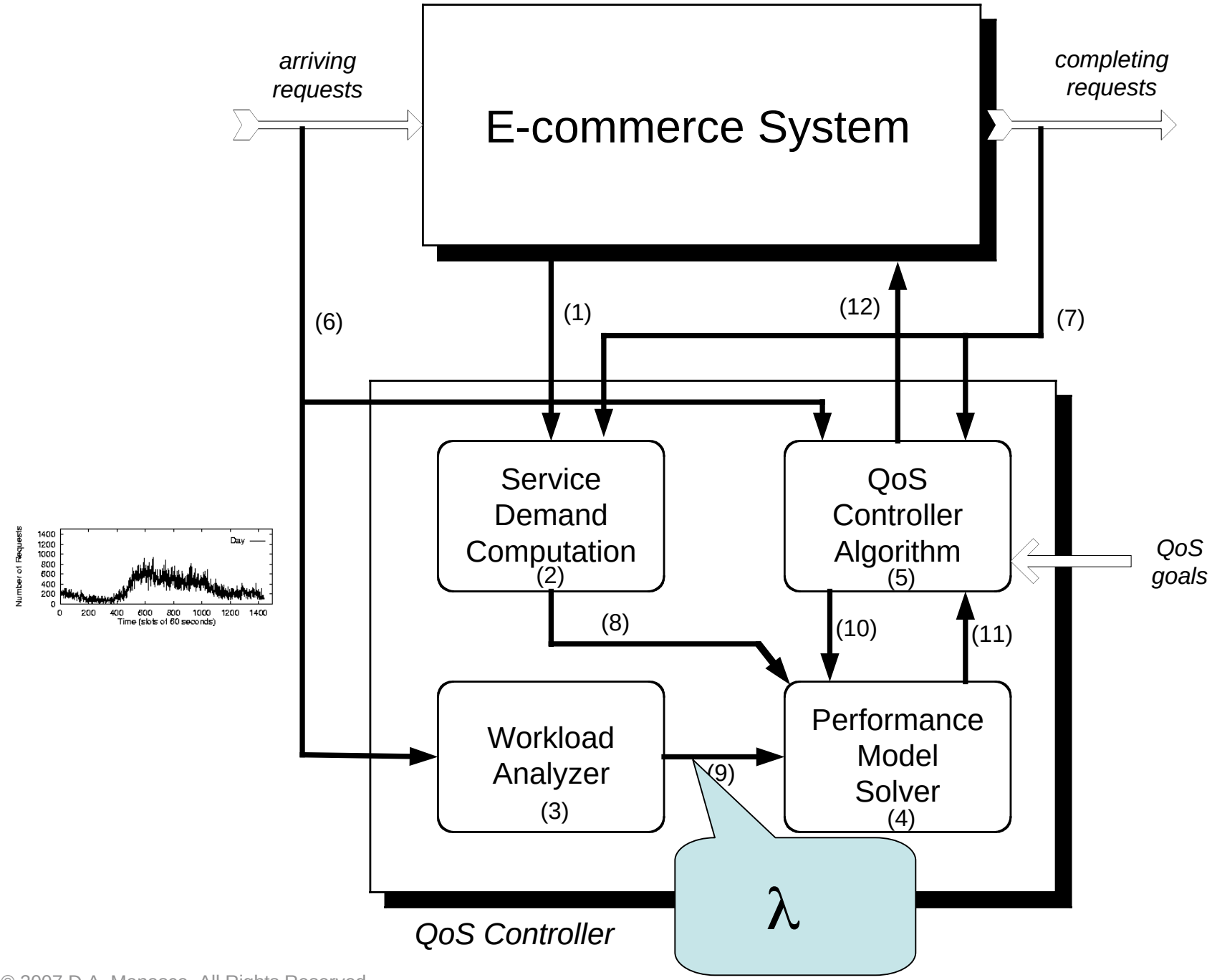


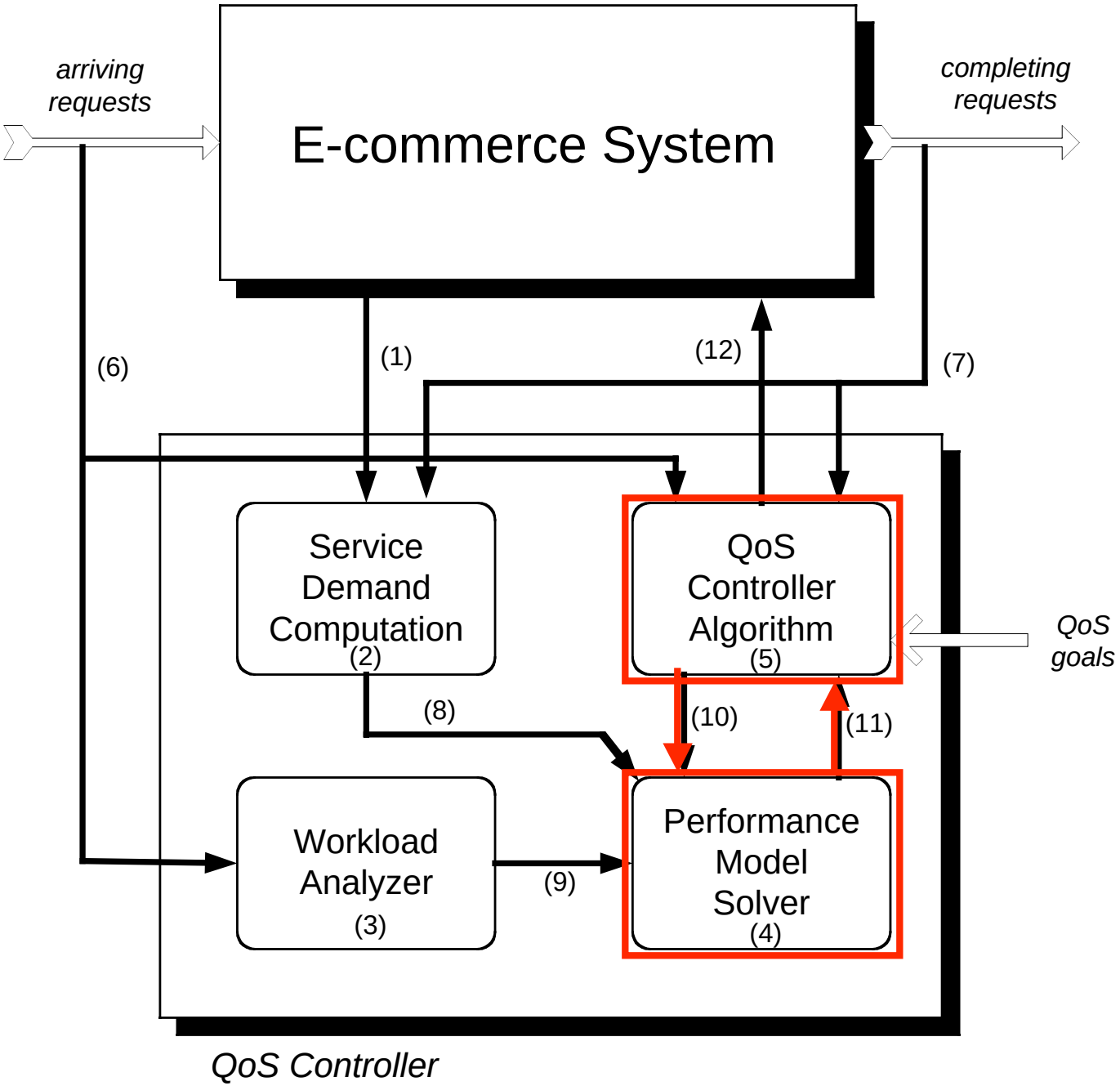




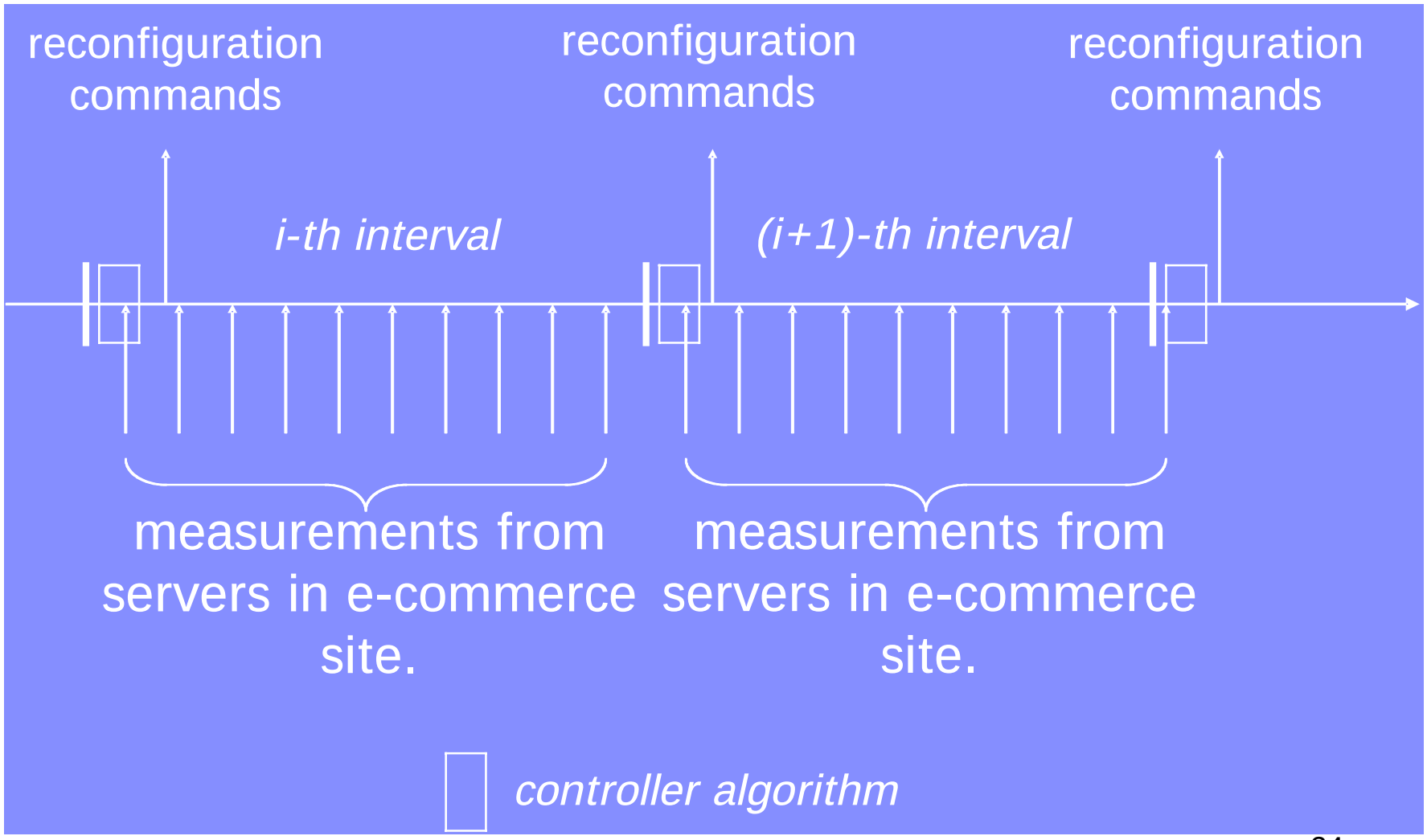




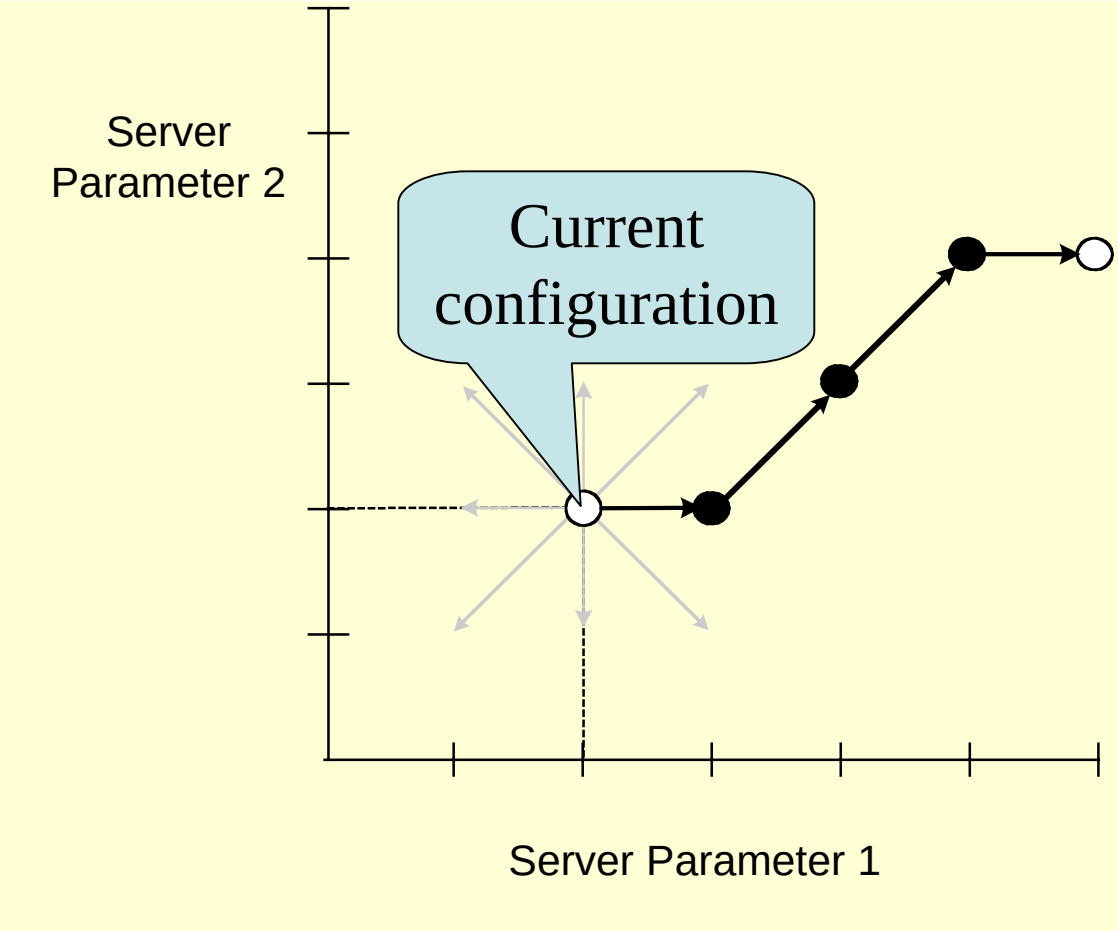




Controller Interval

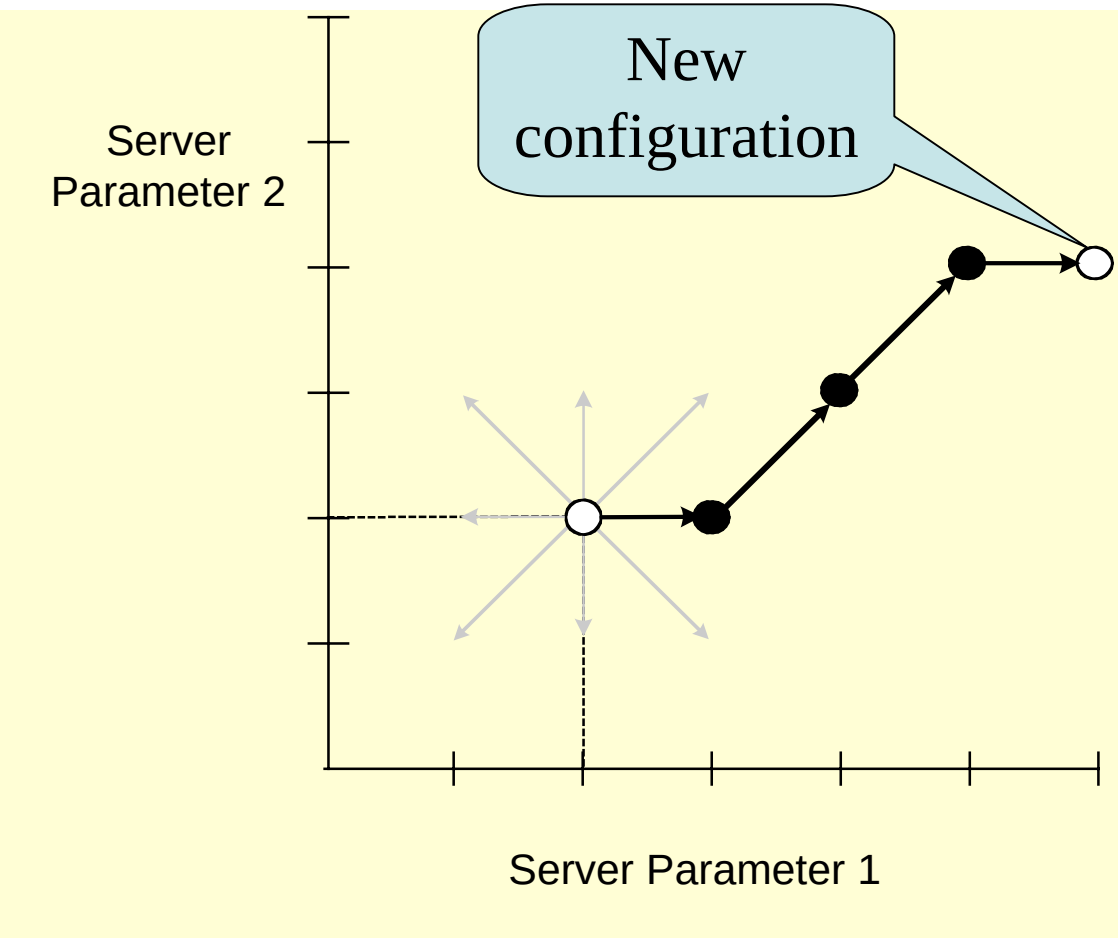


Self-Optimization Approach



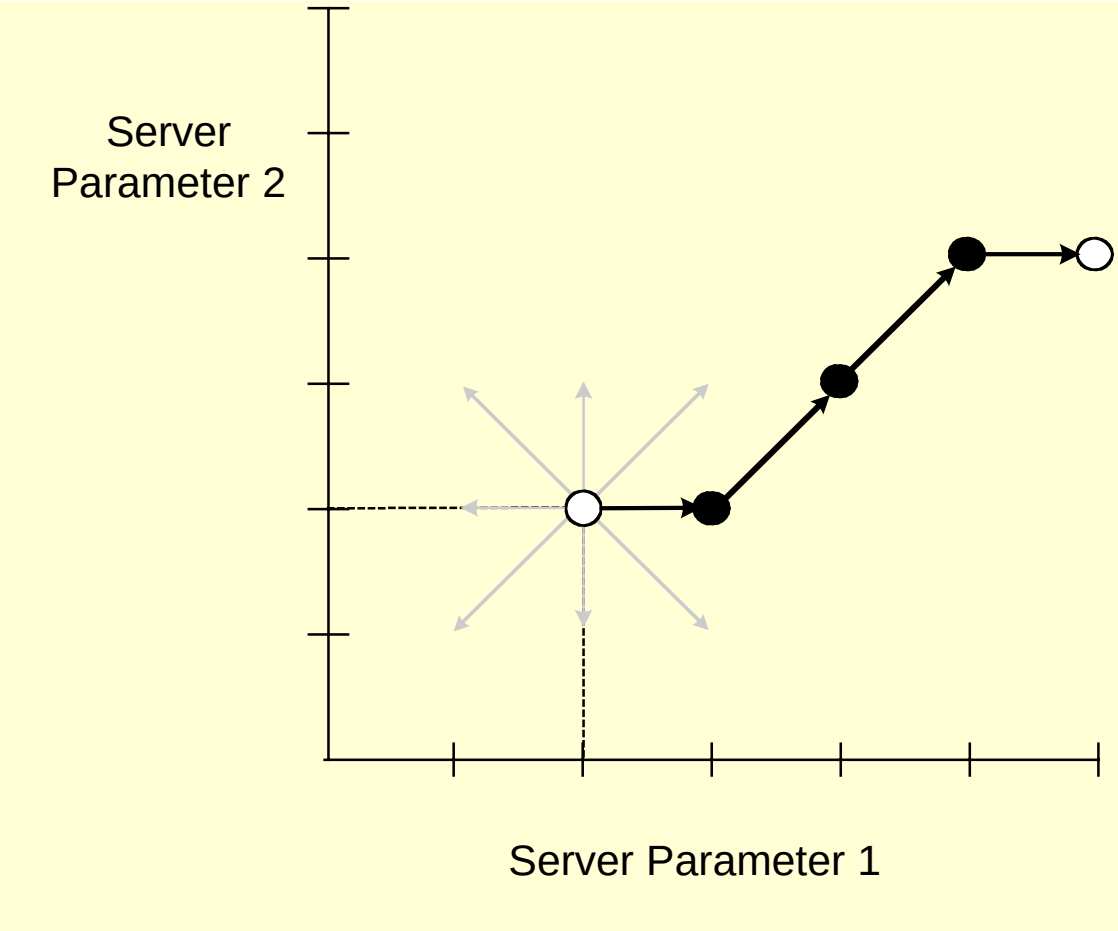
- The space of configuration points is searched using combinatorial search techniques.
- Each point has a QoS value computed through an analytic performance model.

Self-Optimization Approach



- The space of configuration points is searched using combinatorial search techniques.
- Each point has a QoS value computed through an analytic performance model.

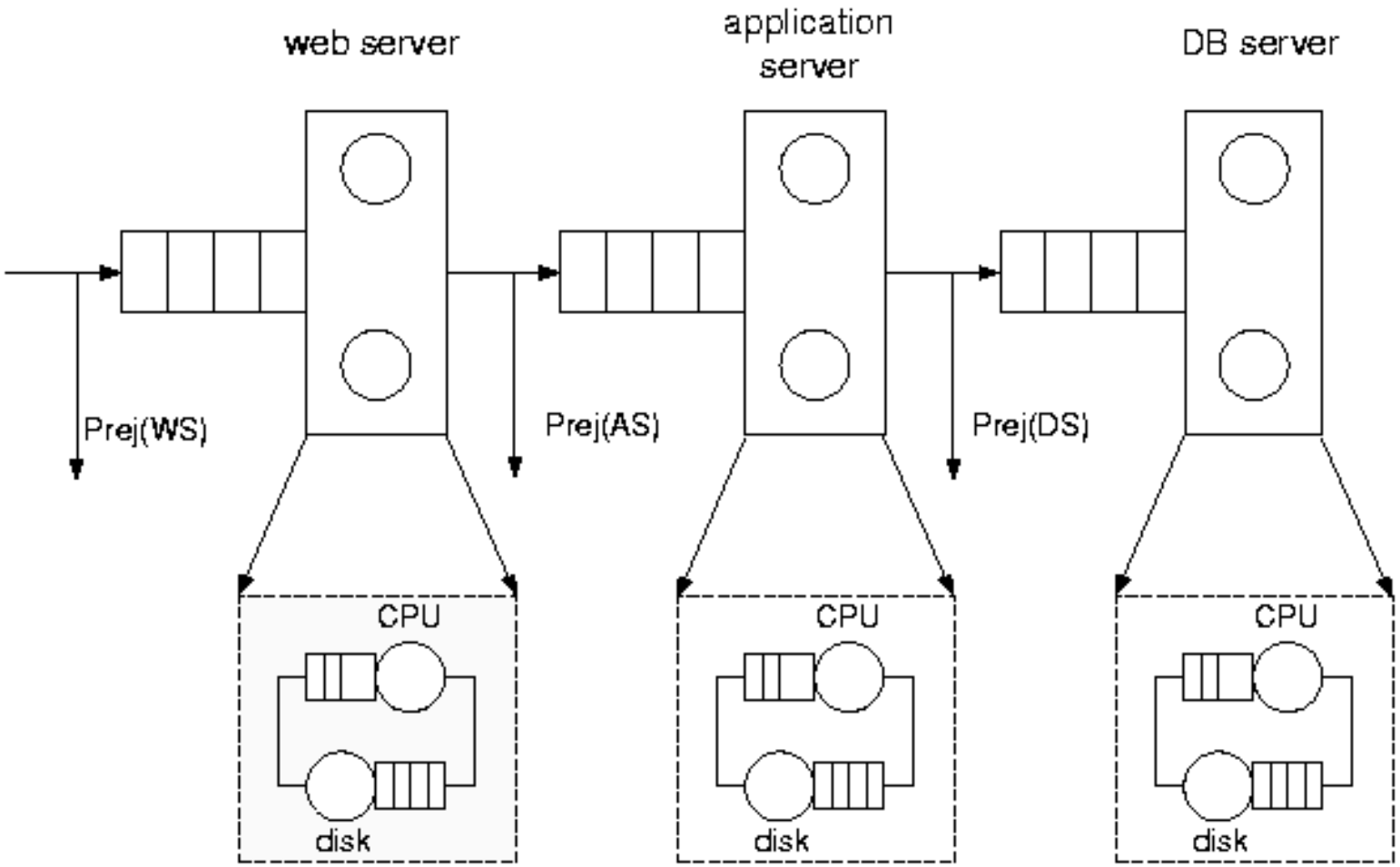
Self-Optimization Approach



- The space of configuration points is searched using a combinatorial search technique.
- Each point has a QoS value computed through an analytic performance model.

$$QoS = f(\vec{W}, c_1, c_2, \ldots, c_m)$$

A Queuing Model is Used to Compute QoS Values



Combined QoS Metric

$$QoS = w_R \times \Delta QoS_R + w_P \times \Delta QoS_P + w_X \times \Delta QoS_X$$

w_R, w_P , and w_X are relative weights that indicate the relative importance of response time, throughput, and probability of rejection.

QoS Metric

$$QoS = w_R \times \Delta QoS_R + w_P \times \Delta QoS_P + w_X \times \Delta QoS_X$$

w_R, w_P , and w_X are relative weights that indicate the relative importance of response time, throughput, and probability of rejection.

⇨ ΔQoS_R ΔQoS_P ΔQoS_X are relative deviations of the response time, throughput, and probability of rejection metrics with respect to their desired levels.

QoS Metric

$$QoS = w_R \times \Delta QoS_R + w_P \times \Delta QoS_P + w_X \times \Delta QoS_X$$

w_R, w_P , and w_X are relative weights that indicate the relative importance of response time, throughput, and probability of rejection.

ΔQoS_R, ΔQoS_P, and ΔQoS_X are relative deviations of the response time, throughput, and probability of rejection metrics with respect to their desired levels.

⇨ The QoS metric is a dimensionless number in the interval [-1, 1].

QoS Metric

$$QoS = w_R \times \Delta QoS_R + w_P \times \Delta QoS_P + w_X \times \Delta QoS_X$$

w_R, w_P , and w_X are relative weights that indicate the relative importance of response time, throughput, and probability of rejection.

ΔQoS_R, ΔQoS_P, and ΔQoS_X are relative deviations of the response time, throughput, and probability of rejection metrics with respect to their desired levels.

The QoS metric is a dimensionless number in the interval [-1, 1].

⇨ If all metrics meet or exceed their QoS targets, QoS $\geq$ 0.

Response Time Deviation

$$\Delta QoS_R = \frac{R_{\max} - R_{measured}}{\max(R_{\max}, R_{measured})}$$

- = 0 if the response time meets its target.
- > 0 if the response time exceeds its target.
- < 0 if the response time does not meet its target.

- $\Delta QoS_R \le 1 - (\sum_{i=1}^{K} D_i) / R_{\max} < 1$
- $-1 < -(1 - R_{\max} / R_{measured}) \le \Delta QoS_R$

Probability of Rejection Deviation

$$\Delta QoS_P = \frac{P_{\max} - P_{measured}}{\max(P_{\max}, P_{measured})}$$

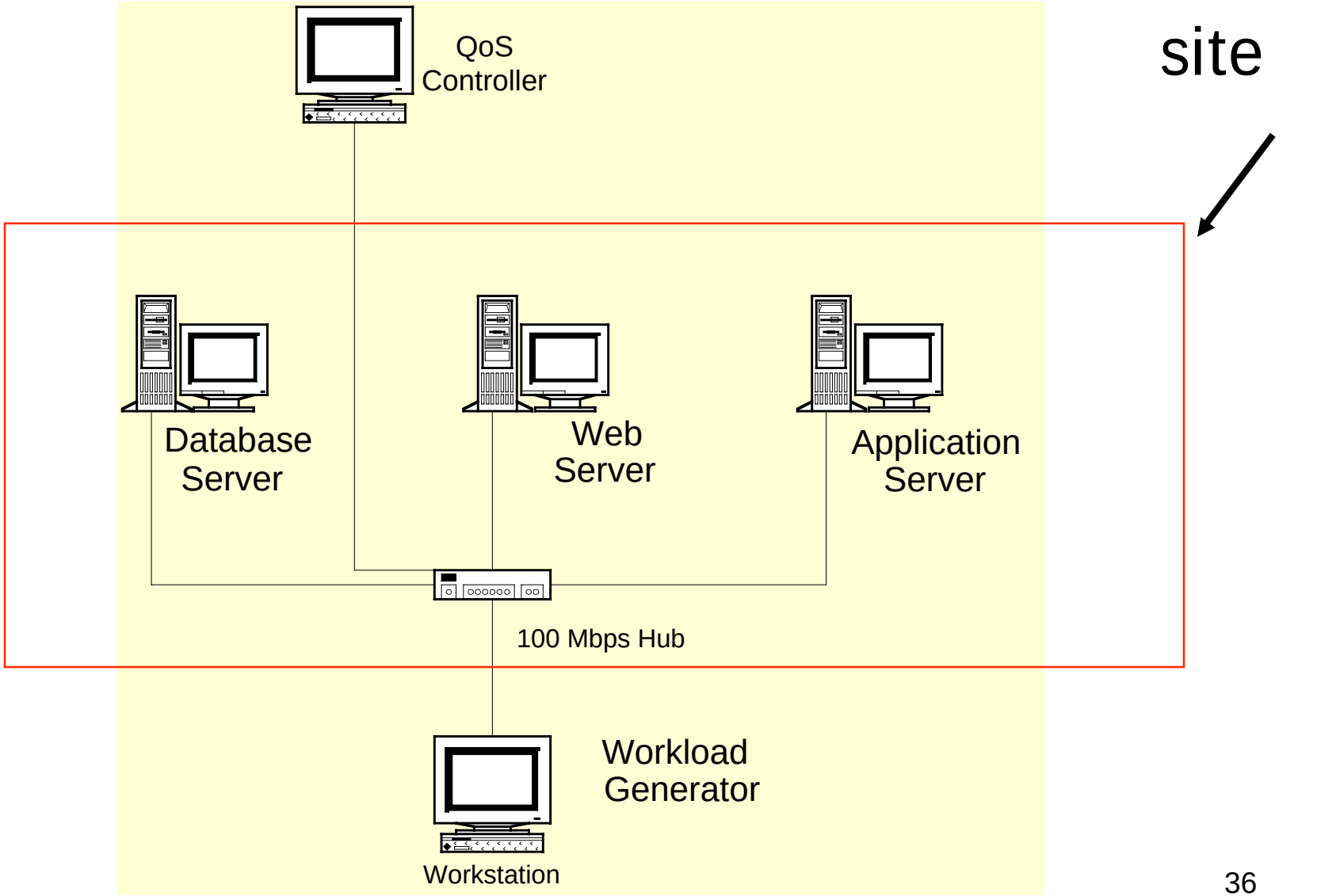
- = 0 if the probability of rejection meets its target.
- > 0 and ≤ 1 if the probability of rejection exceeds its target.
- < 0 and ≥ -1 if the probability of rejection does not meet its target.

Throughput Deviation

$$\Delta QoS_X = \frac{X_{measured} - X^*_{\min}}{\max(X_{measured}, X^*_{\min})}$$

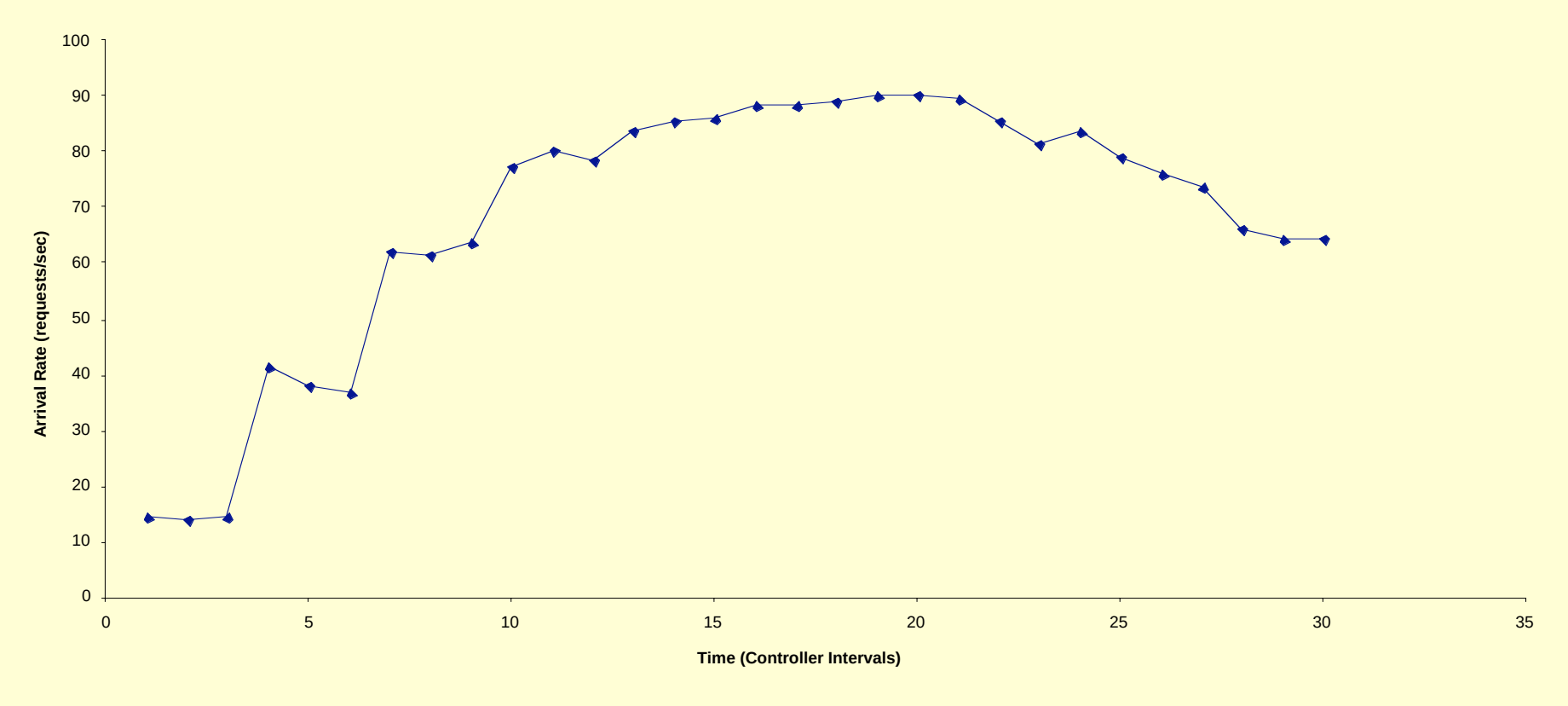
- $X^*_{\min} = \min(\lambda, X_{\min})$
- = 0 if the throughput meets its target.
- > 0 and < 1 if the throughput exceeds its target.
- < 0 and > -1 if the throughput does not meet its target.

Prototype Configuration

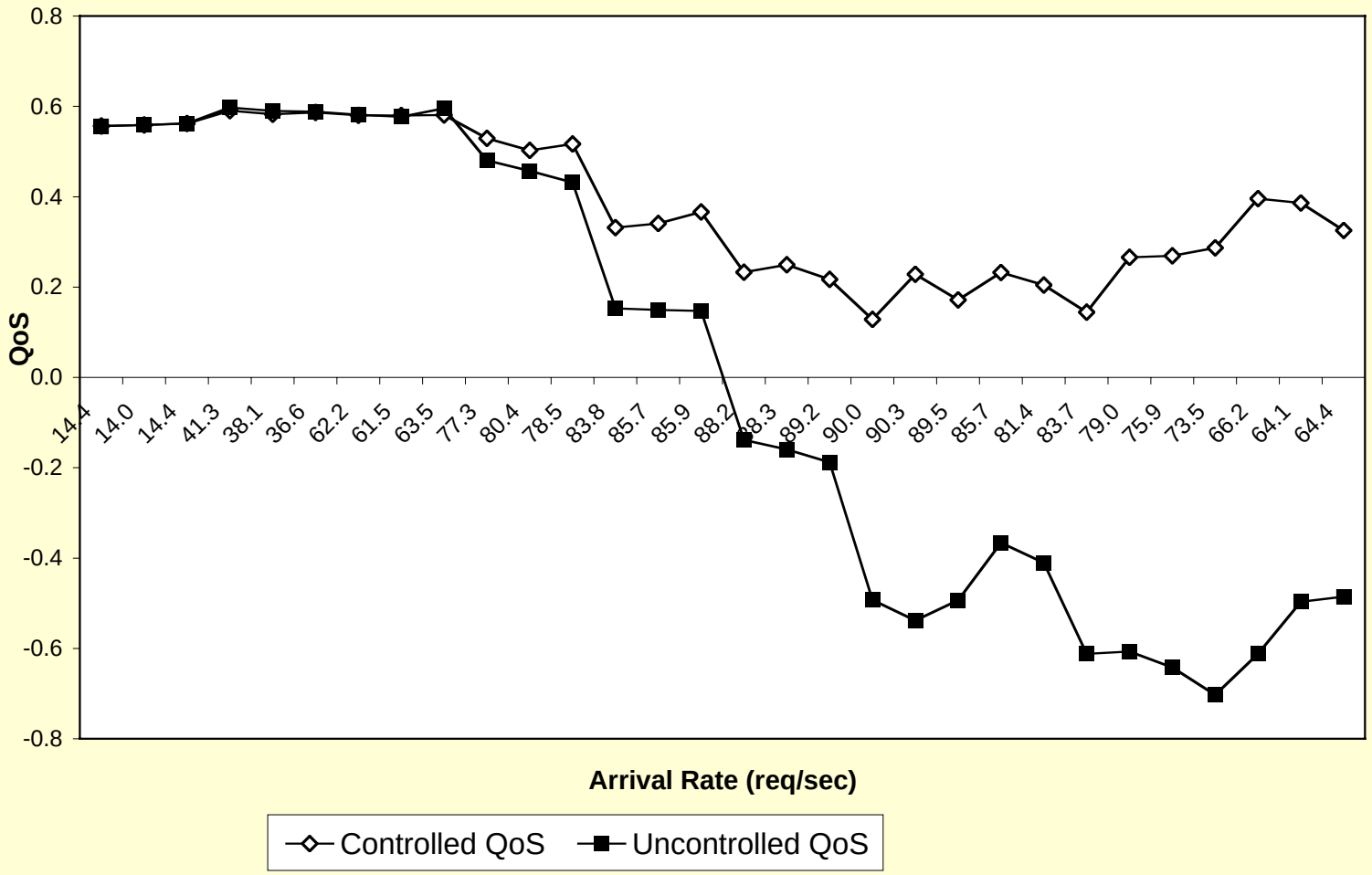


Experiment Results

Arrival rate

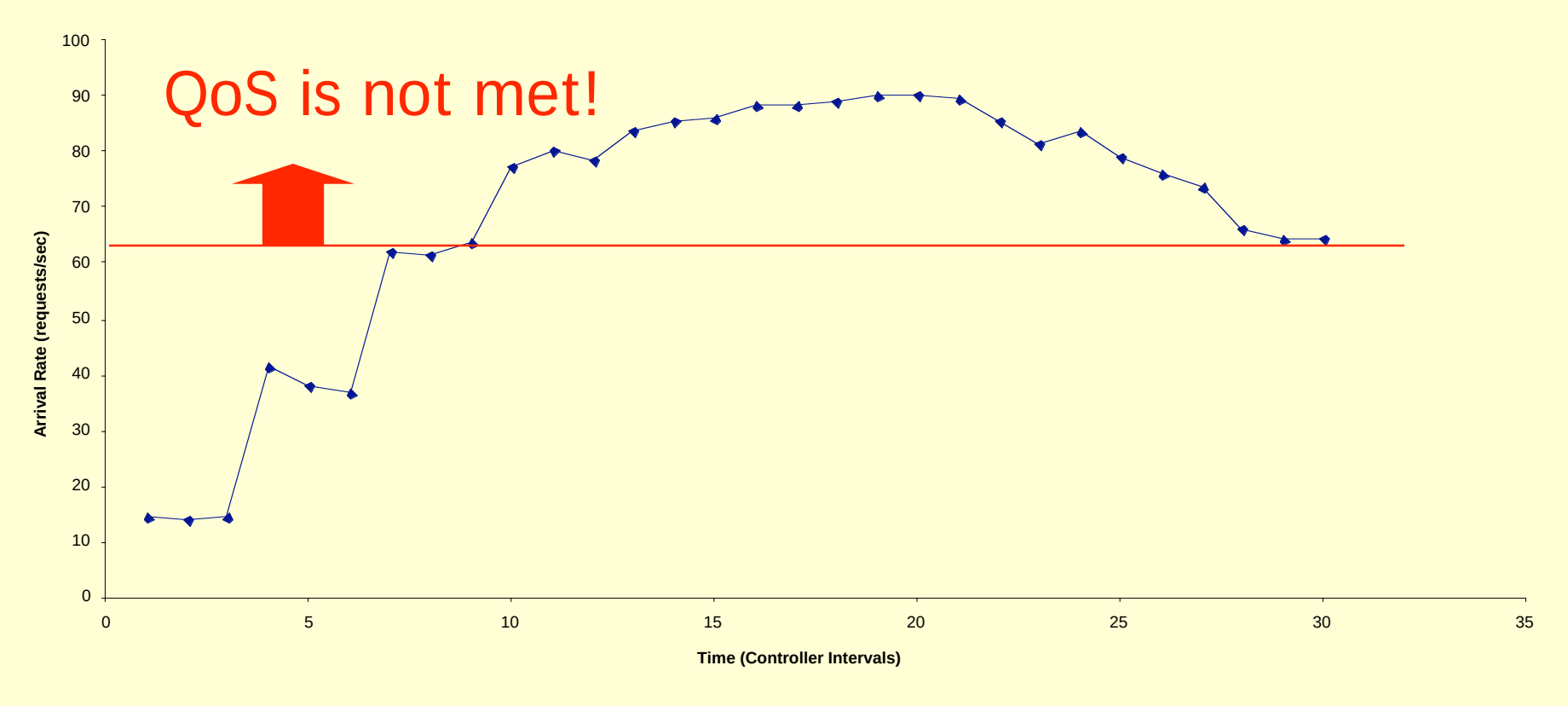


Results of QoS Controller



Experiment Results

Arrival rate



Examples of Autonomic Computing Systems

- E-commerce system
- Internet Data Center
- Virtualized Environment

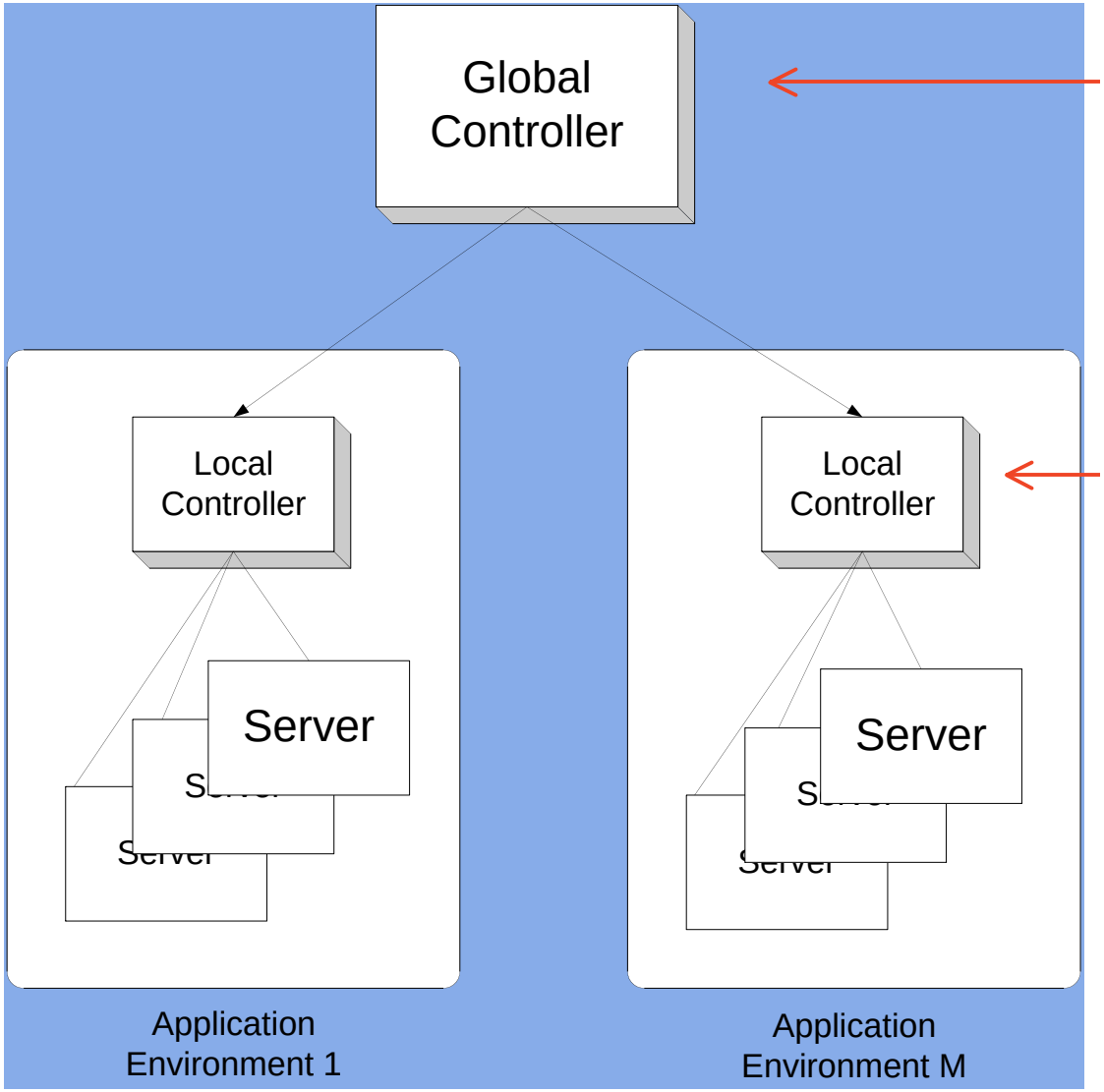
Motivation for Dynamic Resource Allocation in Data Centers

- Large data centers host several application environments (AEs)
- AEs run customers' applications (either online transactions or batch processes) on assigned servers (resources)
- AEs are subject to workloads that exhibit wide and unpredictable variations in their intensities
- Resources need to be redeployed dynamically among AEs to optimize some global utility function

Dynamic Resource Allocation Problem (cont'd)

- The data center has the following characteristics:
  - M Application Environments (batch and online)
  - A fixed number, N, of physical resources (servers)
  - All servers have the same installed software
  - A server can be assigned to any AE

Dynamic Resource Allocation
Two-level Controllers



Decides how many servers to assign to each AE.

Implements Global Controller's decisions.

Dynamic Resource Allocation Problem – Utility Functions

- The global controller uses a global utility function, Ug, to assess the adherence of the overall data center performance to desired service levels (SLAs)

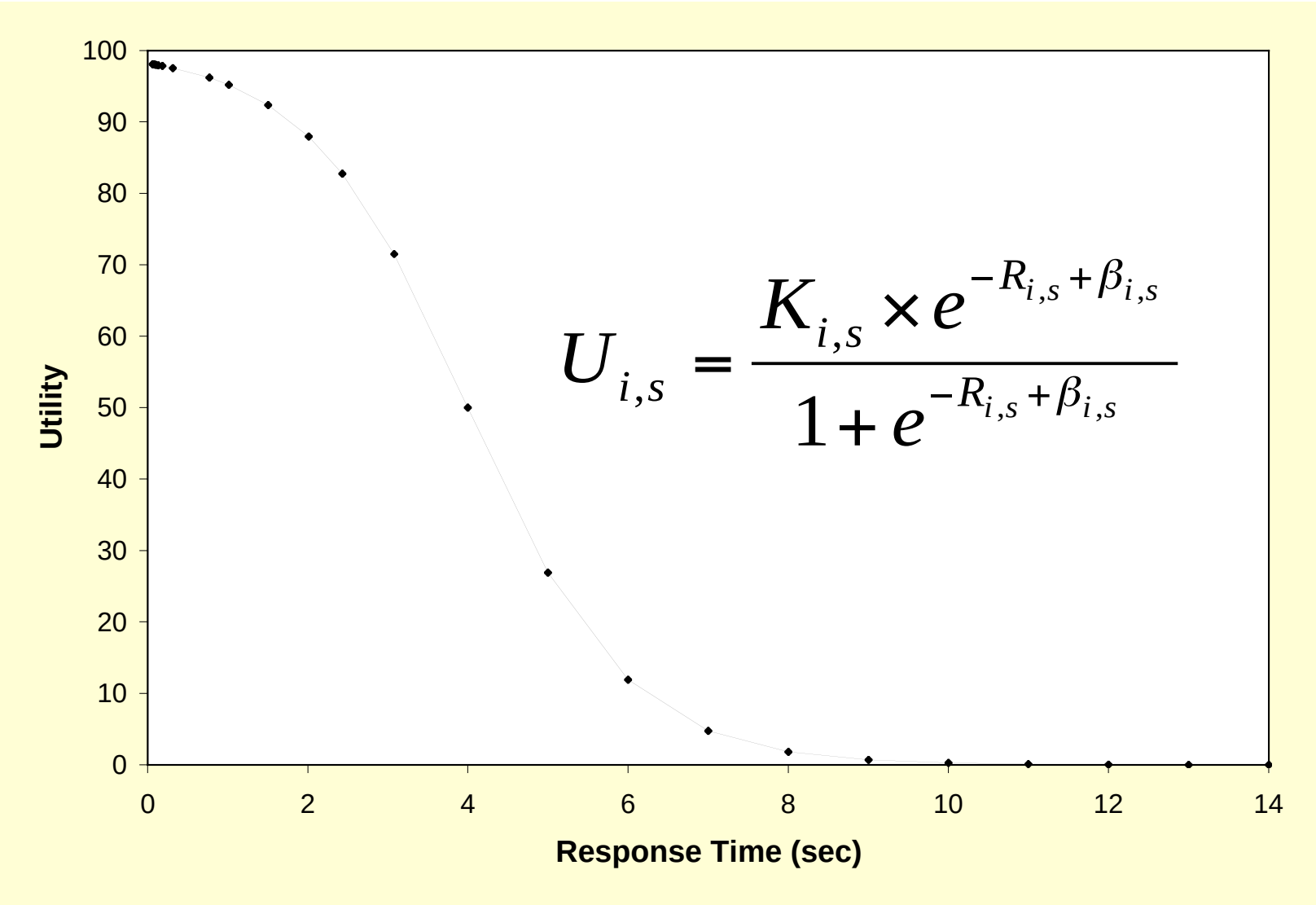
$$U_g = h(U_1, \ldots, U_M)$$

$$U_i = \sum_{s=1}^{S_i} a_{i,s} \times U_{i,s}$$

$$0 < a_{i,s} < 1$$

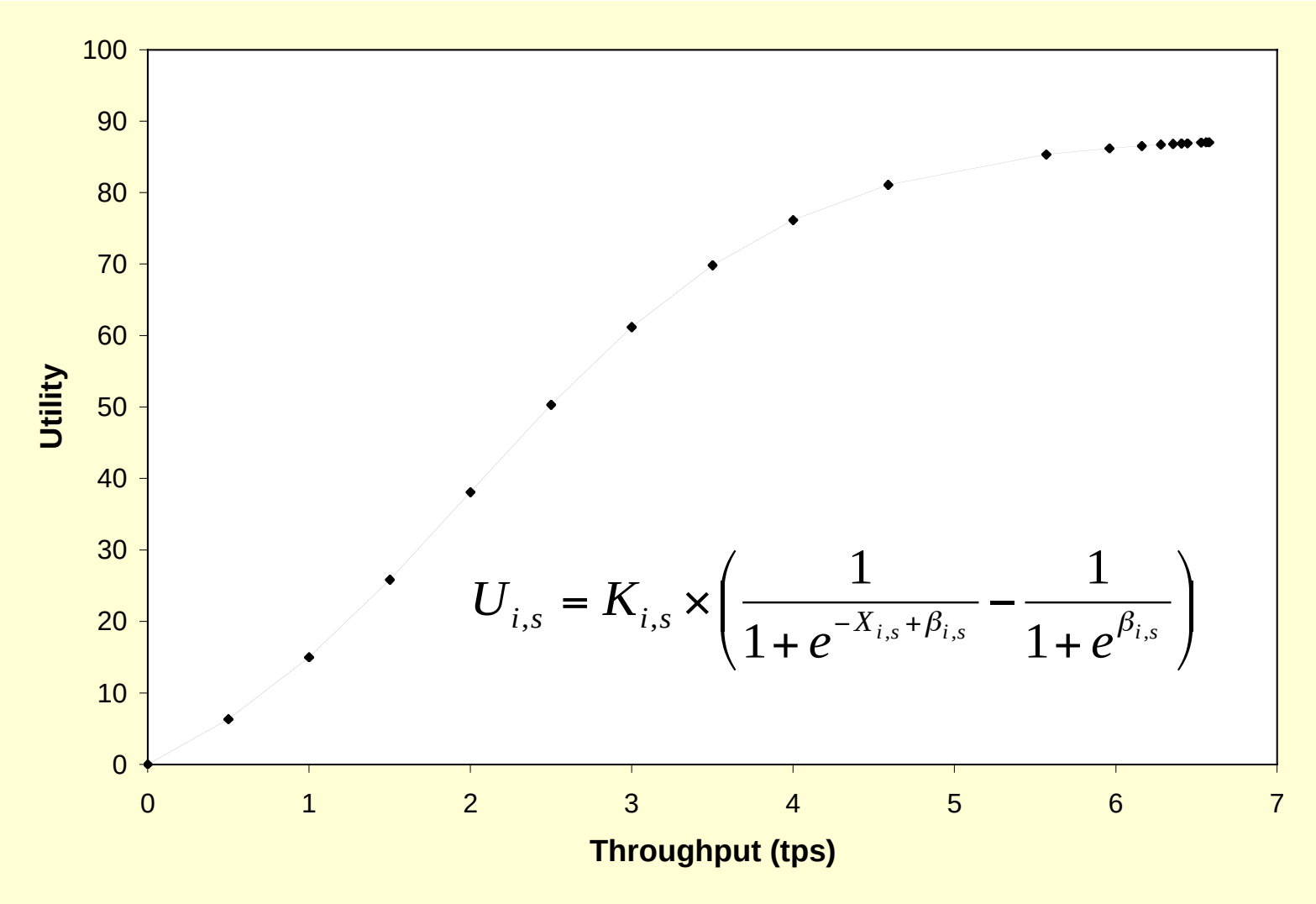
$$\sum_{s=1}^{S_i} a_{i,s} = 1$$

Utility Function as a Function of the Response Time

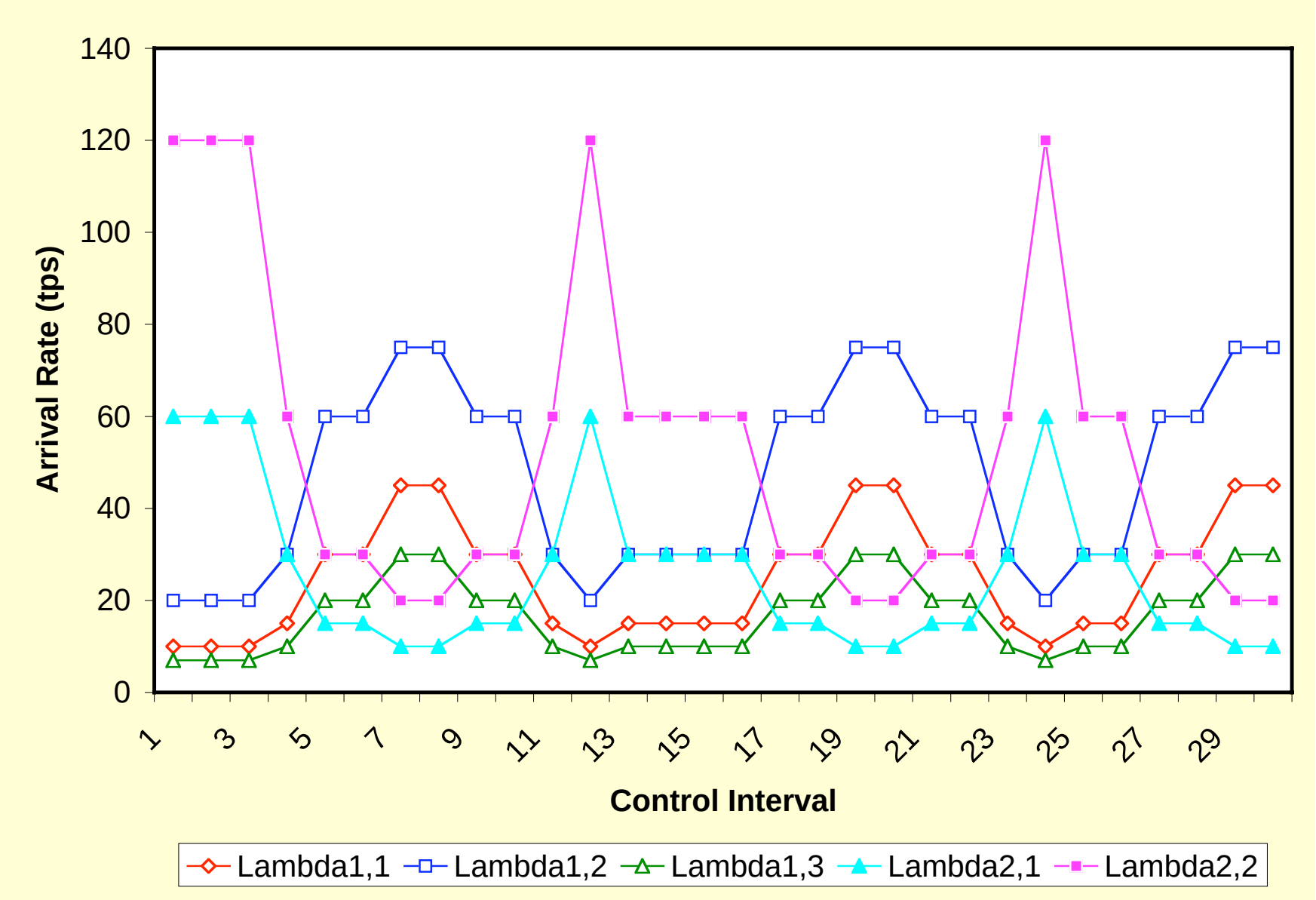


$$U_{i,s} = \frac{K_{i,s} \times e^{-R_{i,s} + \beta_{i,s}}}{1 + e^{-R_{i,s} + \beta_{i,s}}}$$

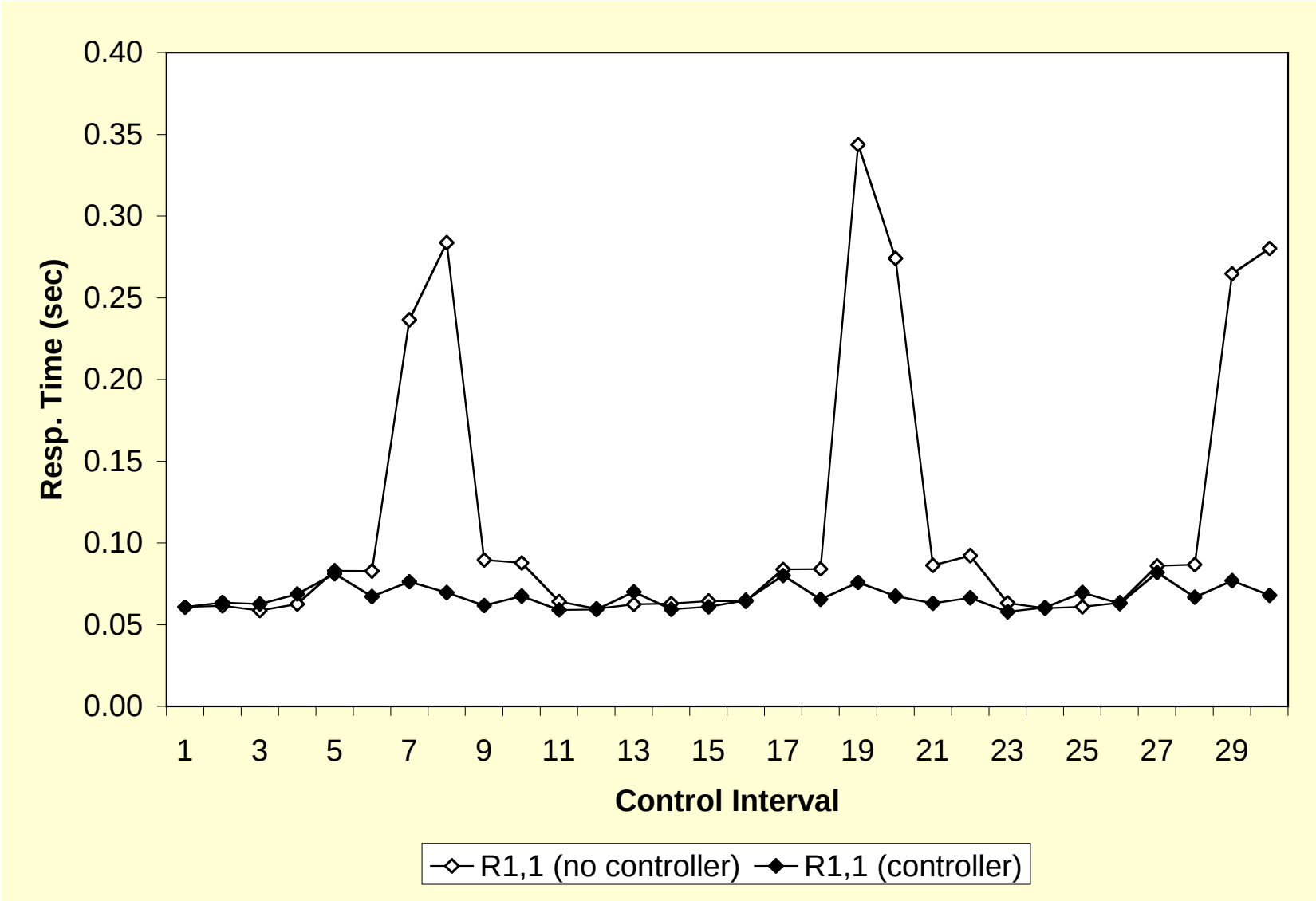
Utility Function as a Function of the Throughput



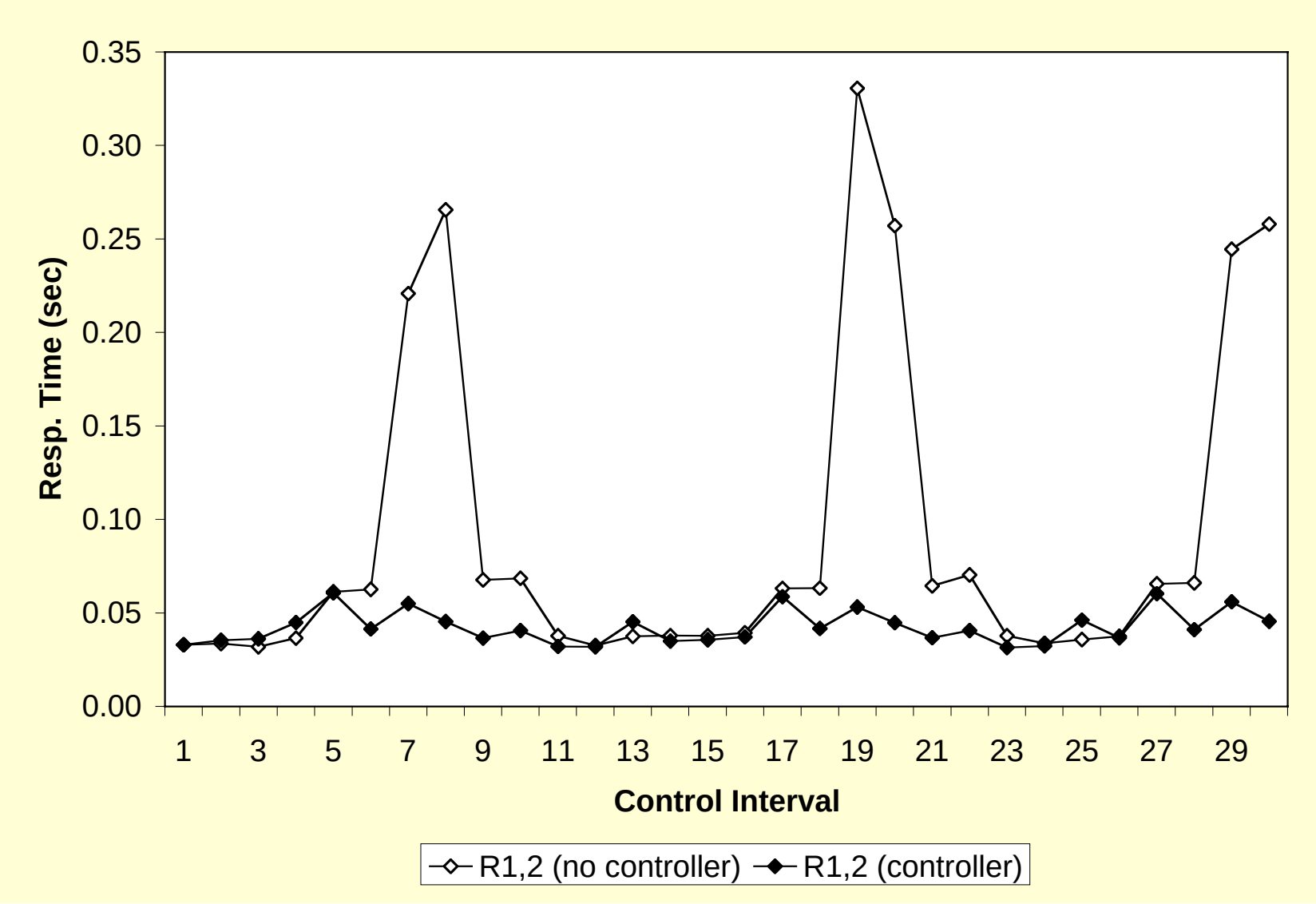
Workload Variation for Online AEs



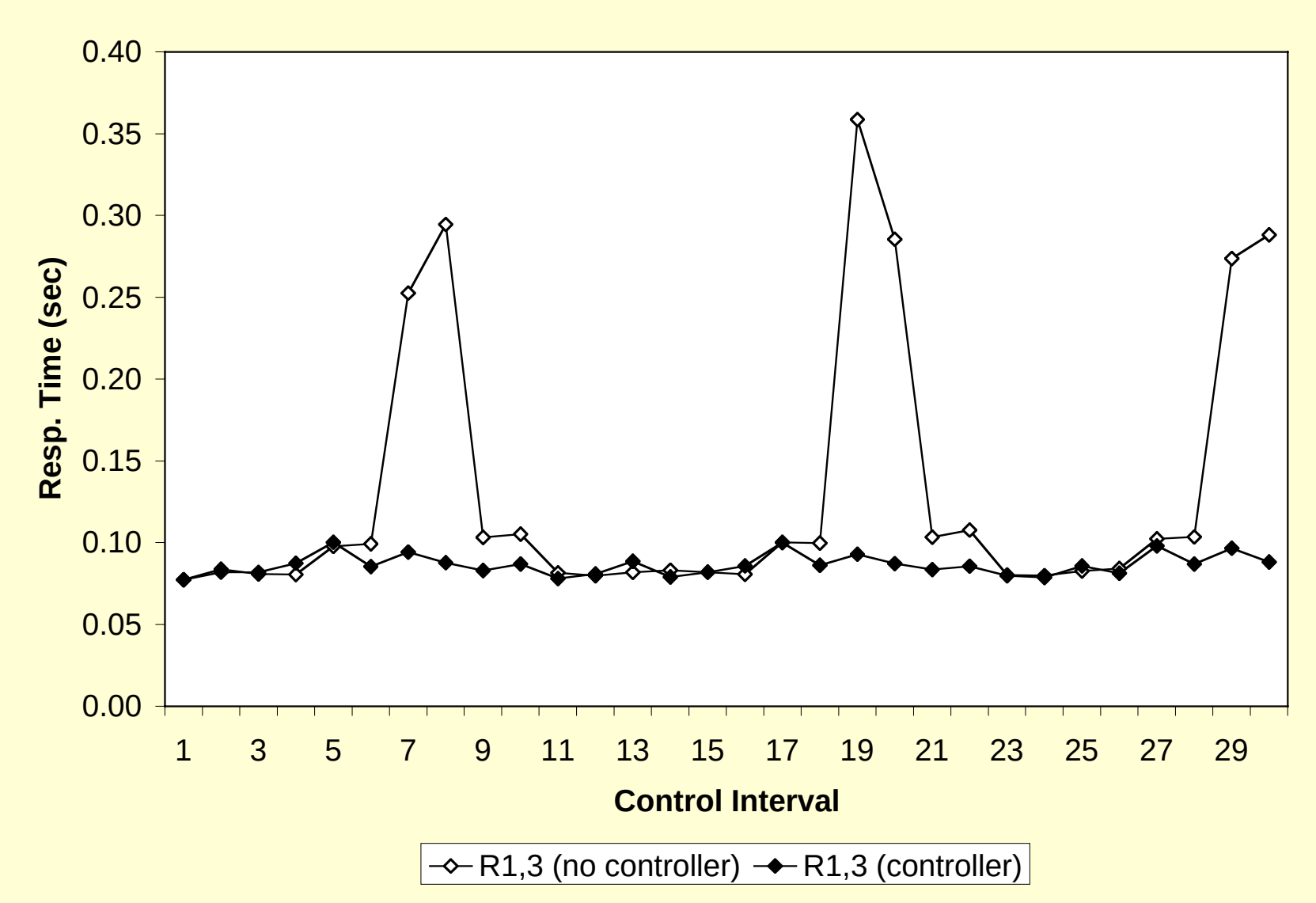
Response Times for Class 1 of AE 1



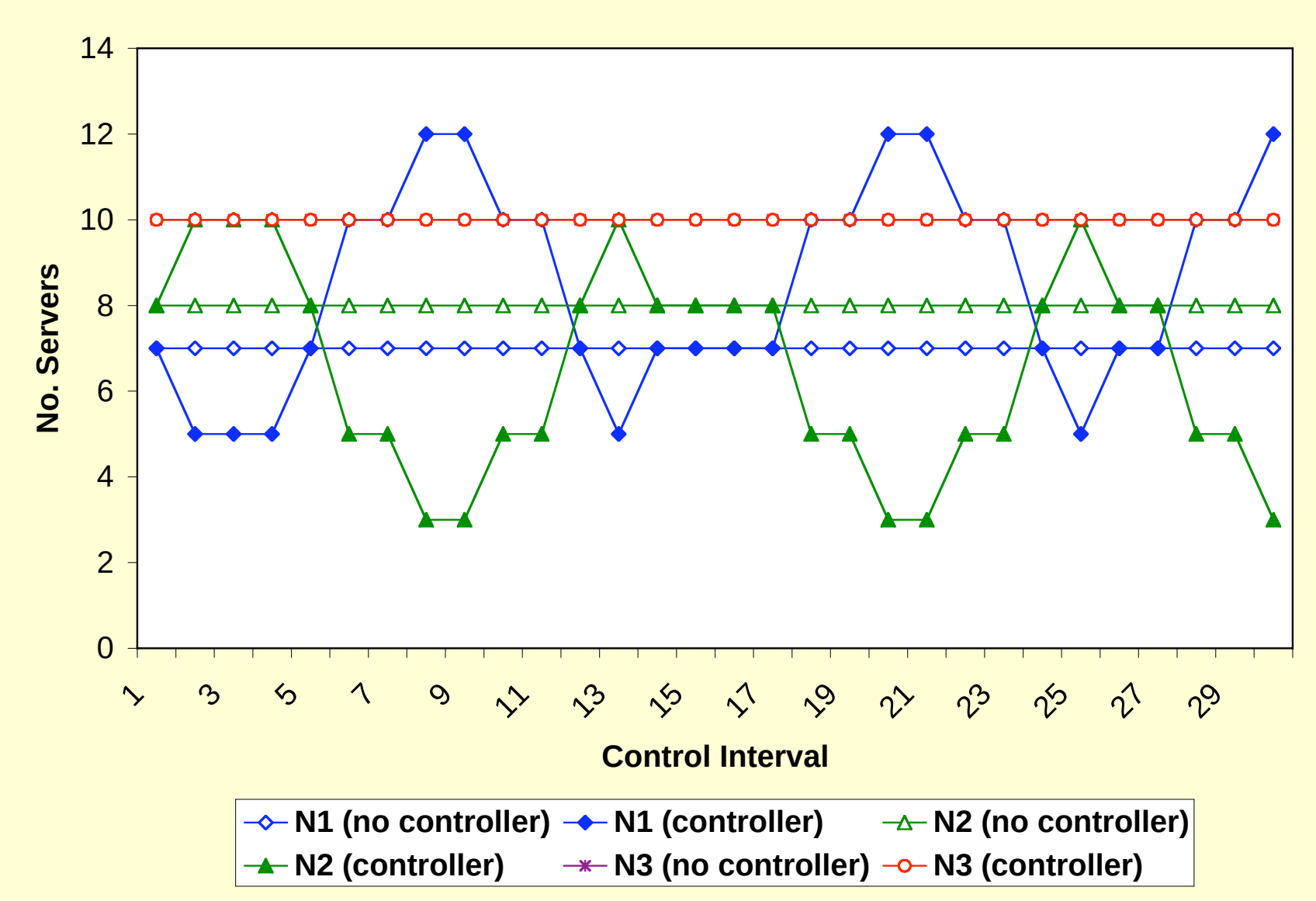
Response Times for Class 2 of AE 1



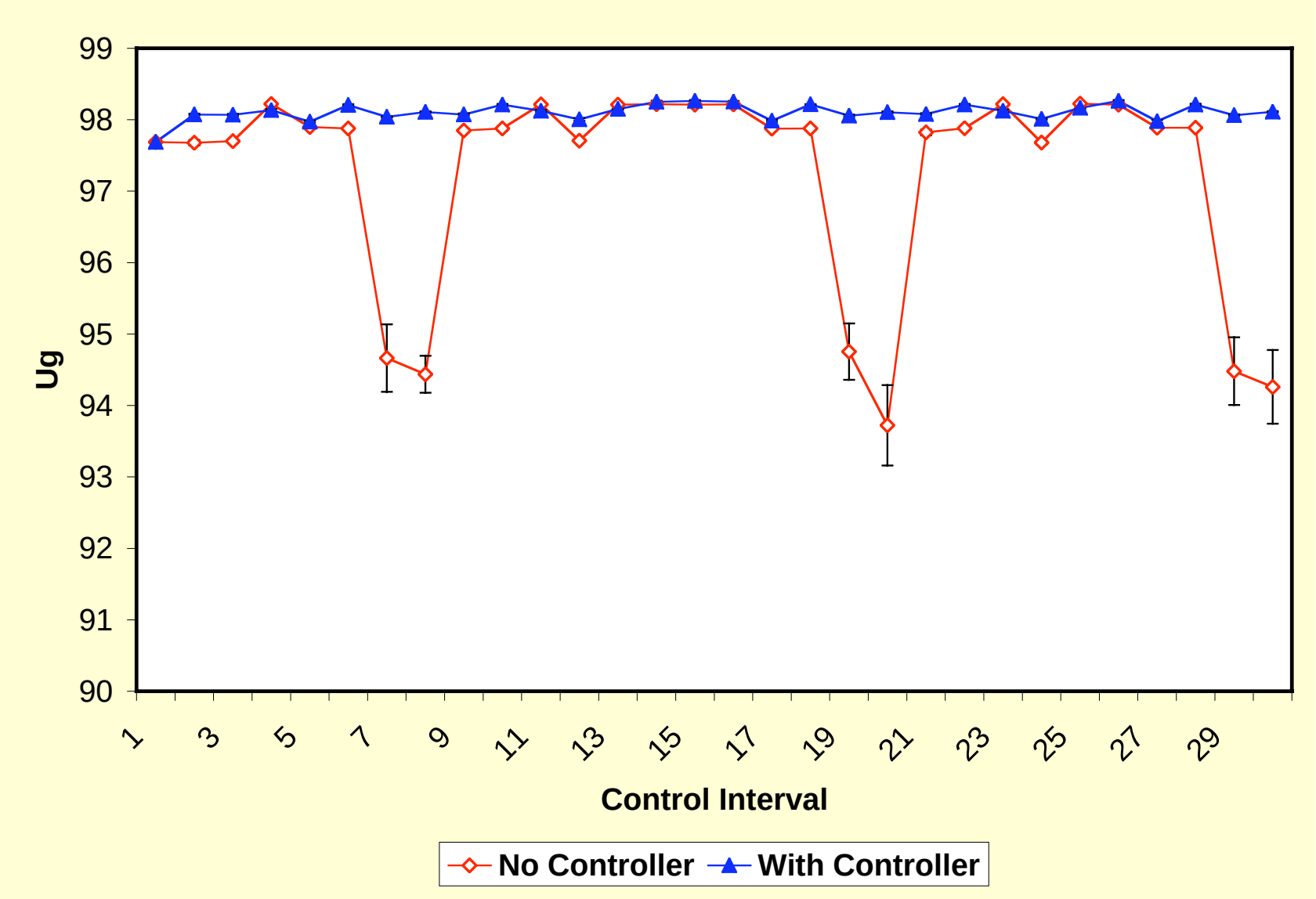
Response Times for Class 3 of AE 1



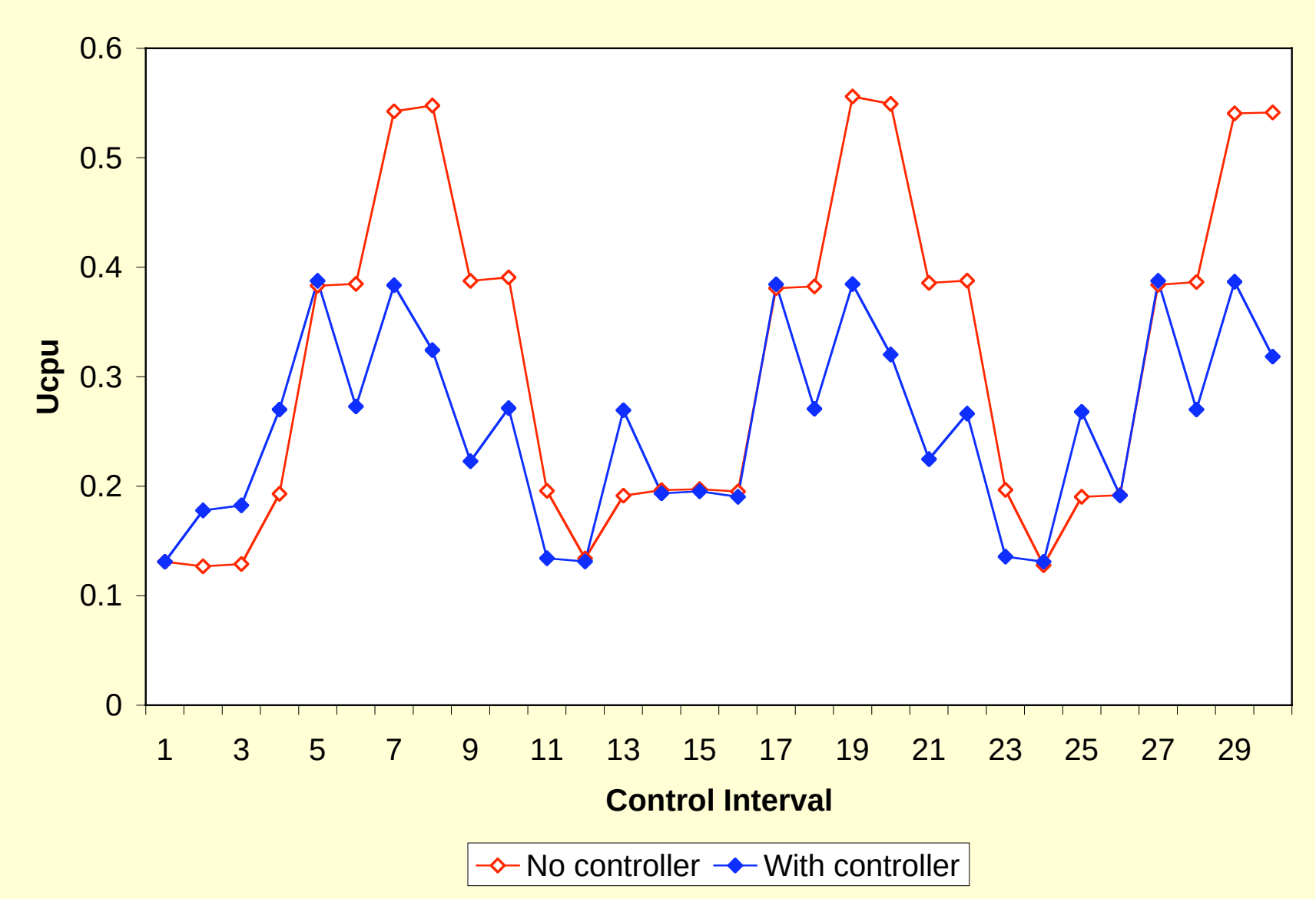
Variation of the Number of Servers



Variation of Global Utility



CPU Utilization for Servers in AE 1

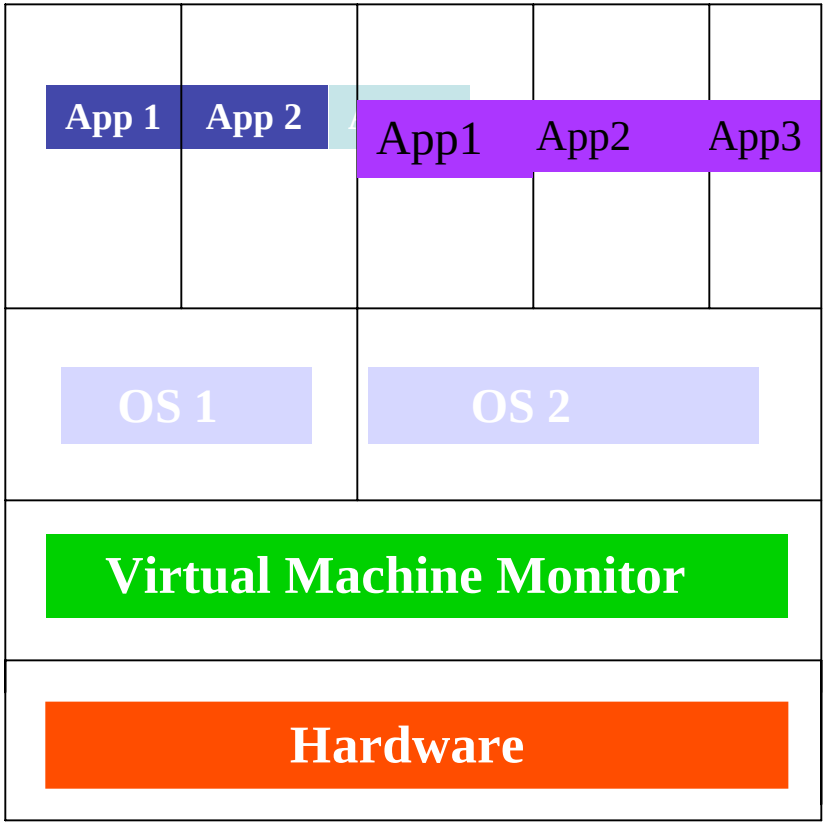


Examples of Autonomic Computing Systems

- E-commerce system
- Internet Data Center
- Virtualized Environment

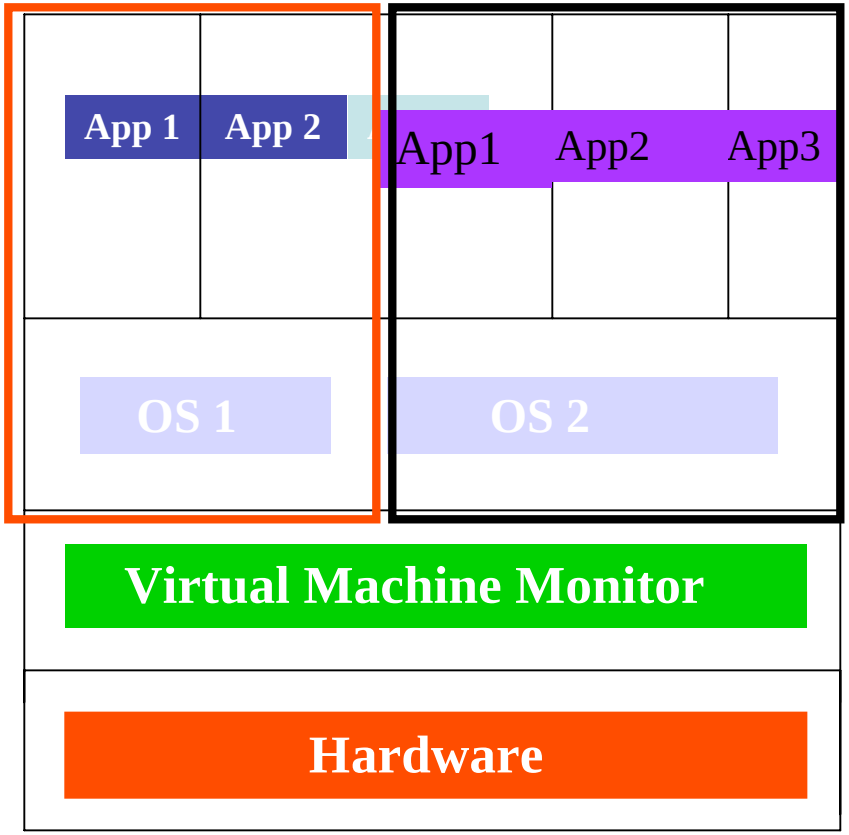
Autonomic Virtualized Environments

- OSs (guest OSs) run in user mode and the VMM runs in supervisor mode
- Advantages of virtualization:
  - Increased security and reliability
  - Reduced cost
  - Better maintenance
  - Easy support for legacy systems
  - Load balancing
- Virtual machines are allocated the CPU based on their priority or assigned CPU shares
- Examples: VMWare, Sun Zones, Xen, IBM LPARs.



Autonomic Virtualized Environments

- OSs (guest OSs) run in user mode and the VMM runs in supervisor mode
- Advantages of virtualization:
  - Increased security and reliability
  - Reduced cost
  - Better maintenance
  - Easy support for legacy systems
  - Load balancing
- Virtual machines are allocated the CPU based on their priority or assigned CPU shares
- Examples: VMWare, Sun Zones, Xen, IBM LPARs.



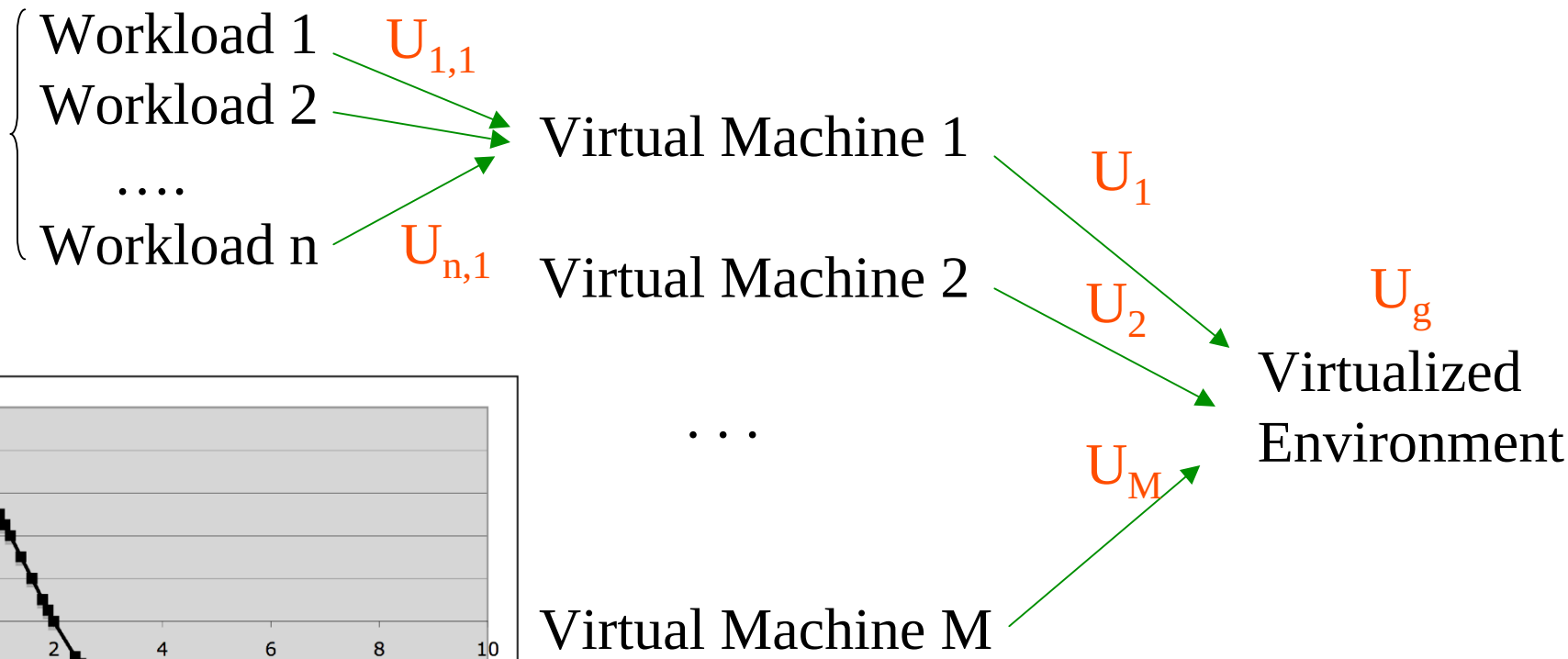
CPU Allocation Problem for Autonomic Virtualized Environments

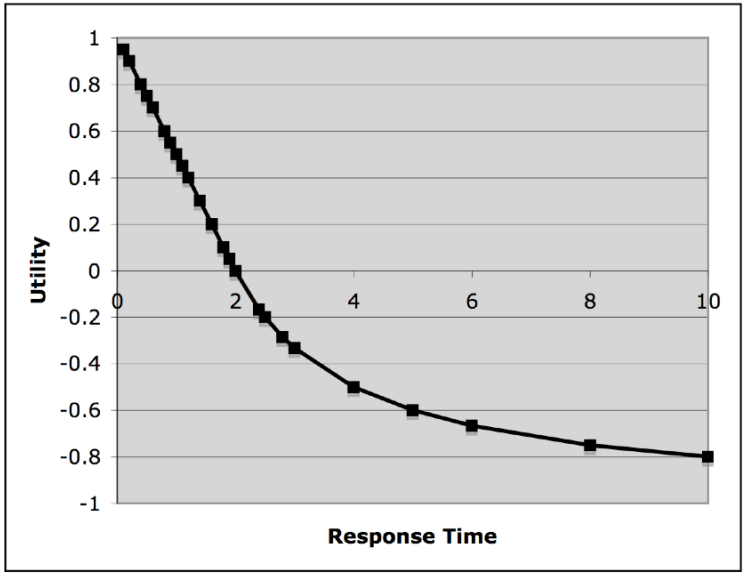
- Existing systems allow for manual allocation of CPU resources to VMs using *CPU priorities* or *CPU shares*.
- Need automated mechanism for the adjustment of CPU shares or the priorities of the virtual machines in order to maximize the global utility of the entire virtualized environment

CPU Allocation Problem for Autonomic Virtualized Environments (Cont'd)

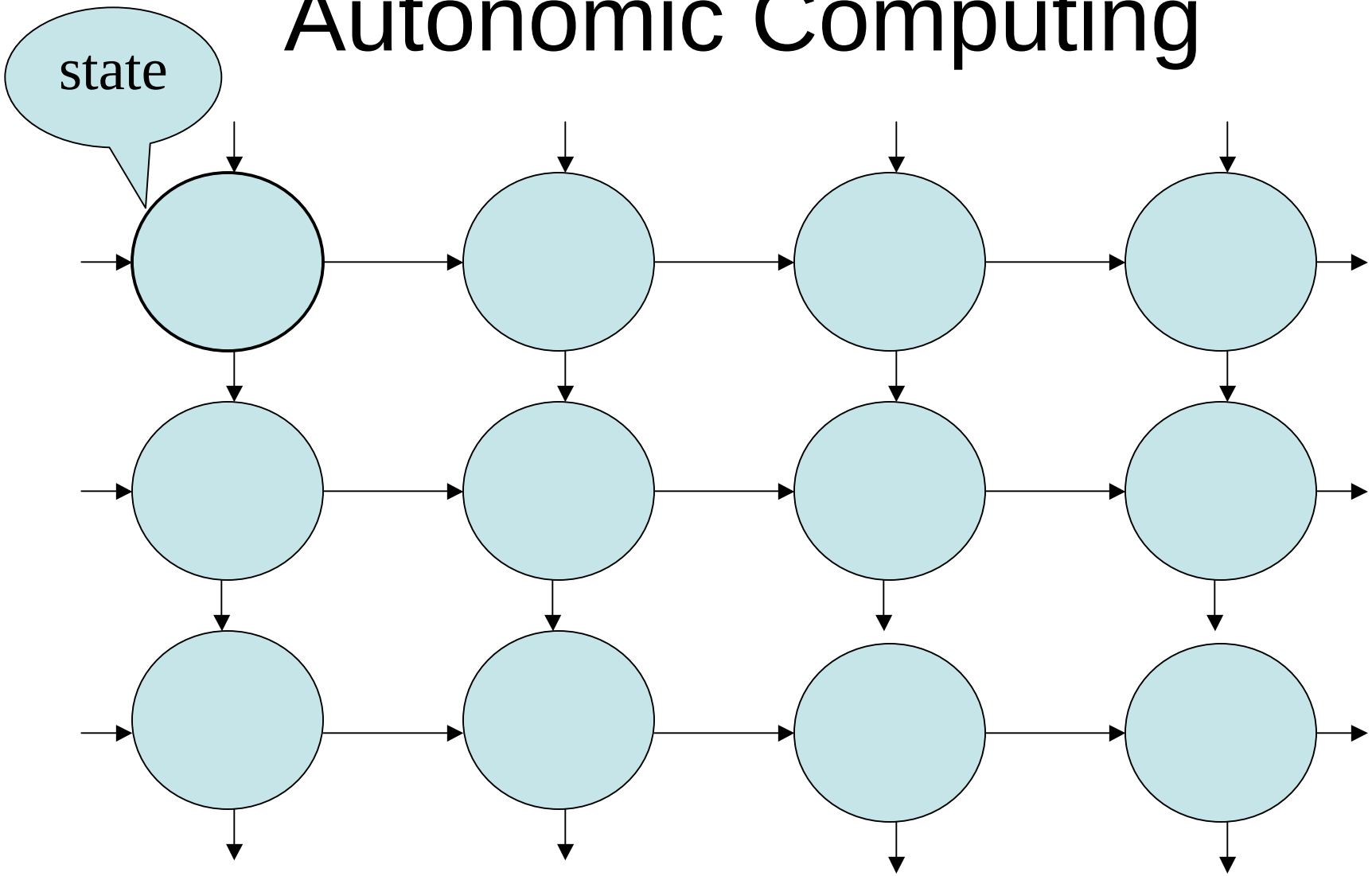
- Use utility functions: maps performance metrics and SLAs to a scalar.
- Each service class of any given VM has a utility function that depends on the type of the class, the value of the performance metric for the class, and on the SLA for the class.
- The total utility function of a VM is a function of the utility functions of all its service classes.
- The global utility function is a function of the utility functions of all VMs.

CPU Allocation Problem for Autonomic Virtualized Environments (Cont'd)

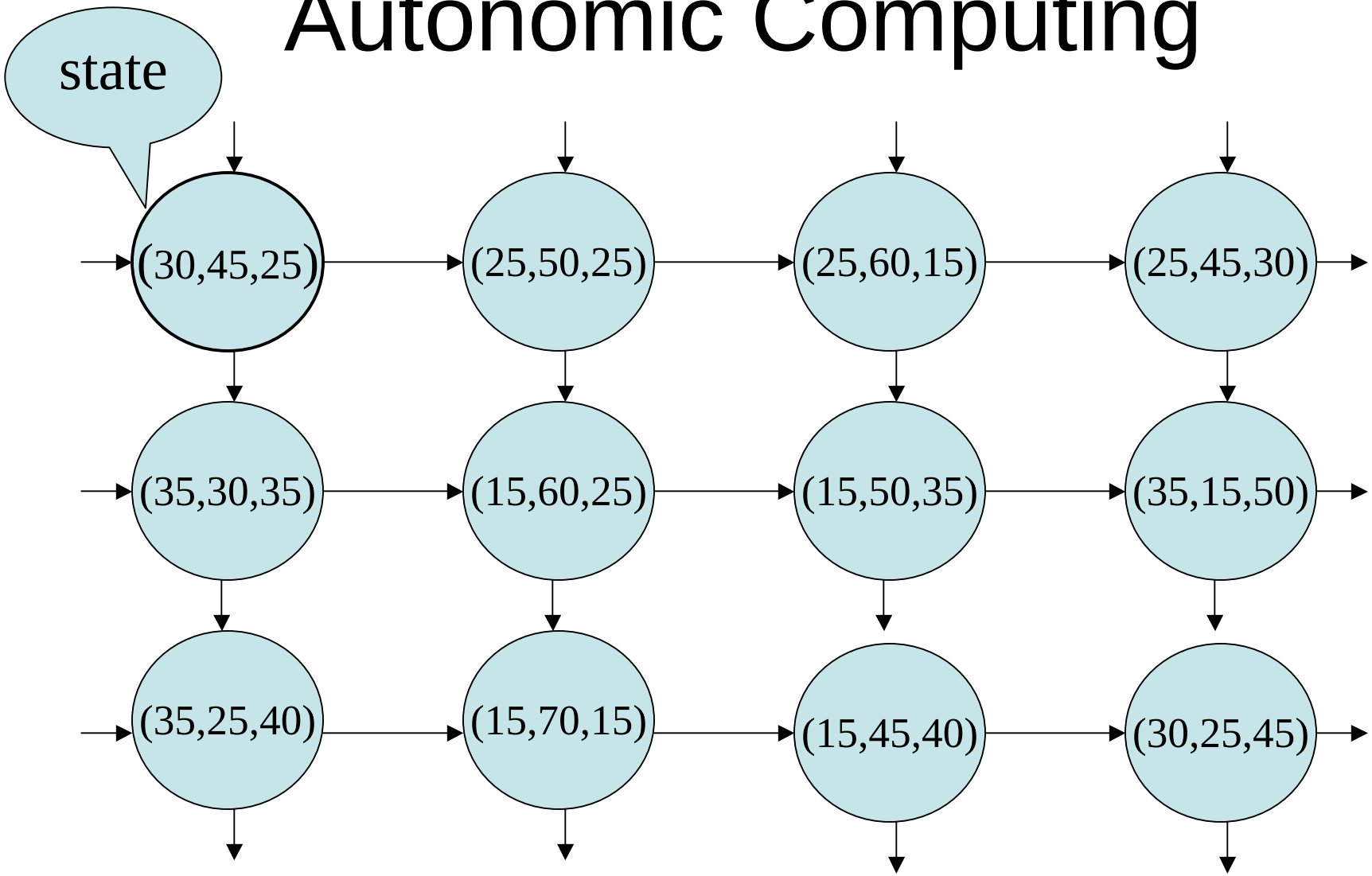




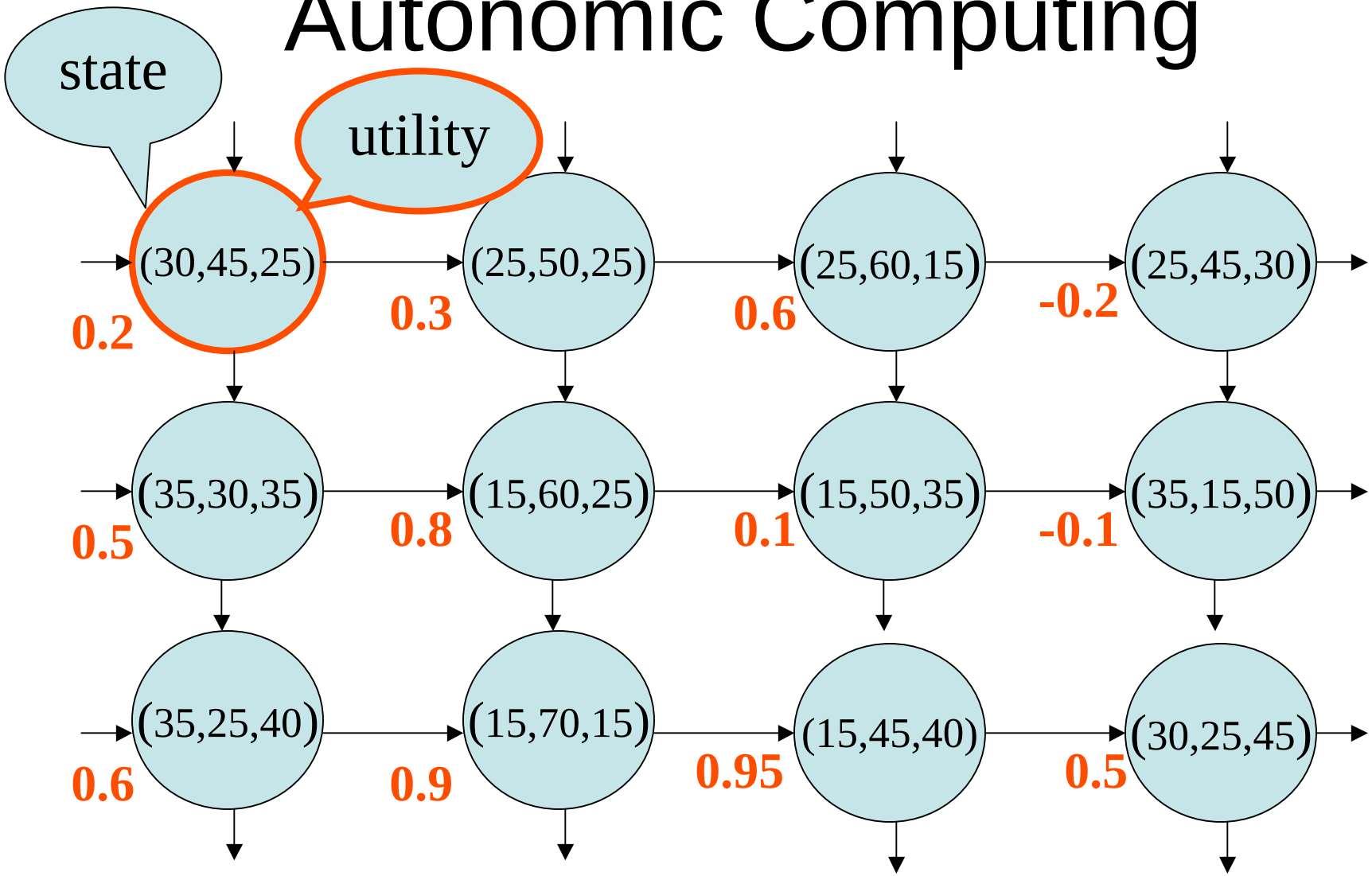
Performance Model-Based Autonomic Computing



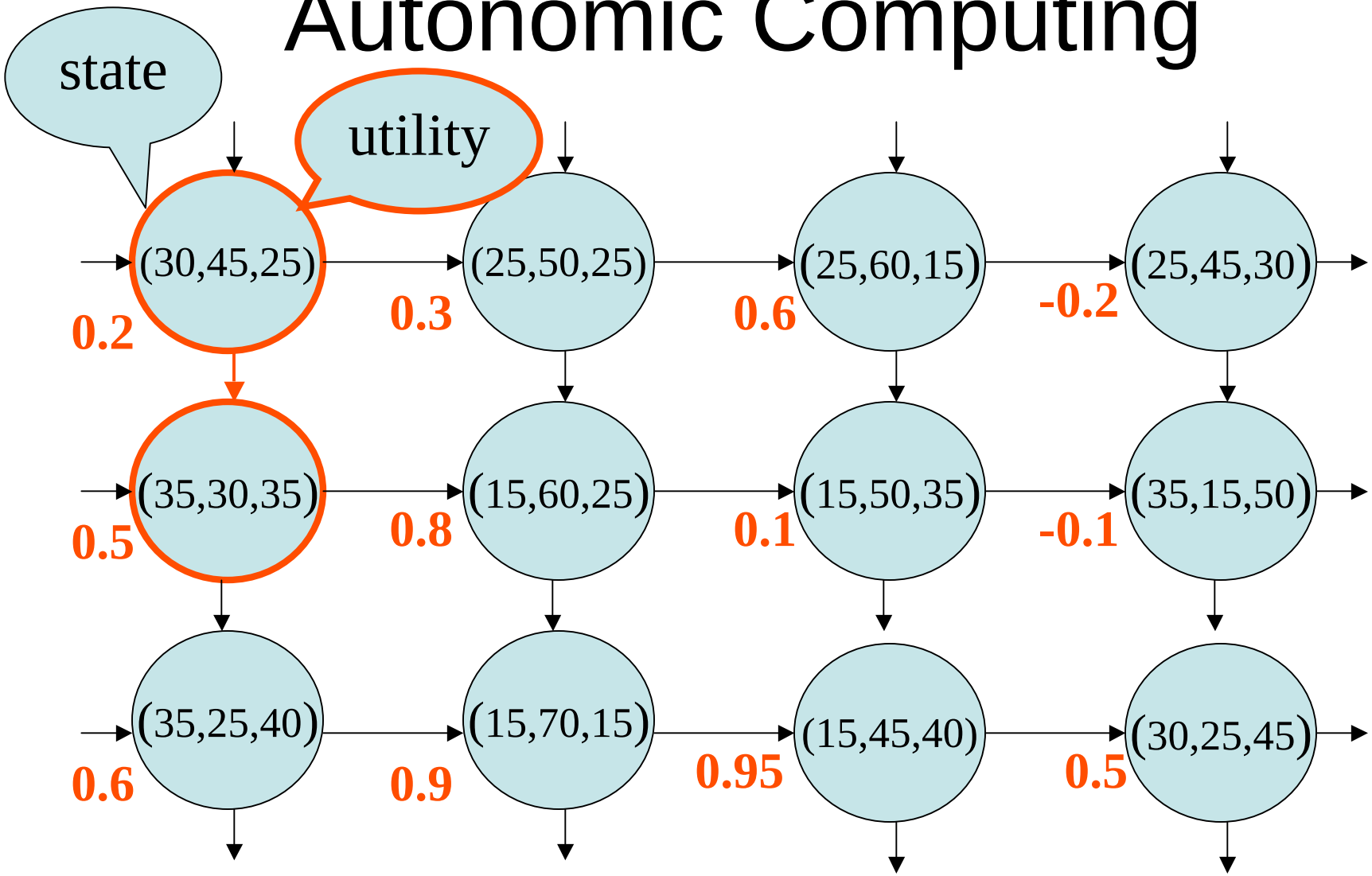
Performance Model-Based Autonomic Computing



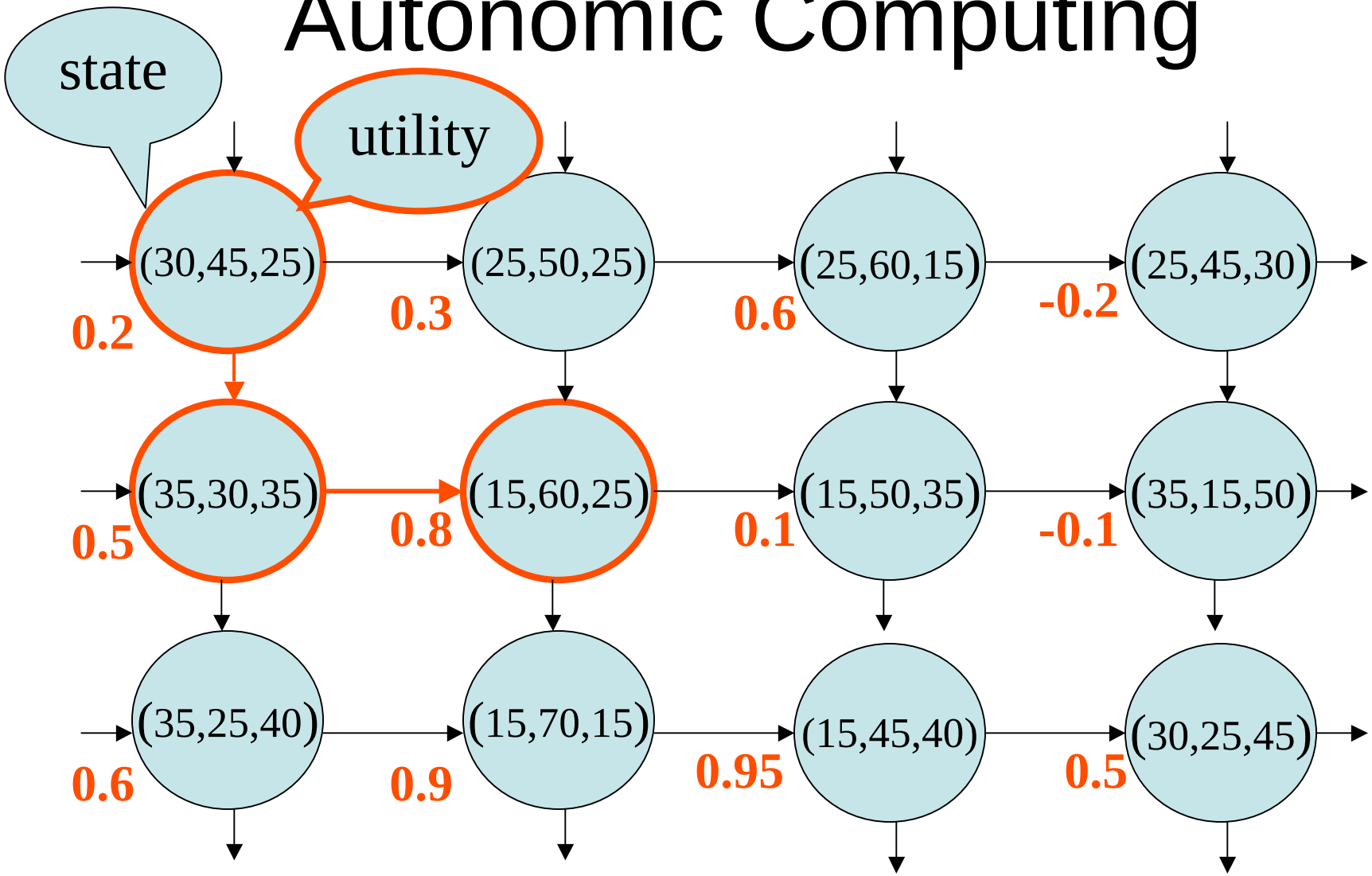
Performance Model-Based Autonomic Computing



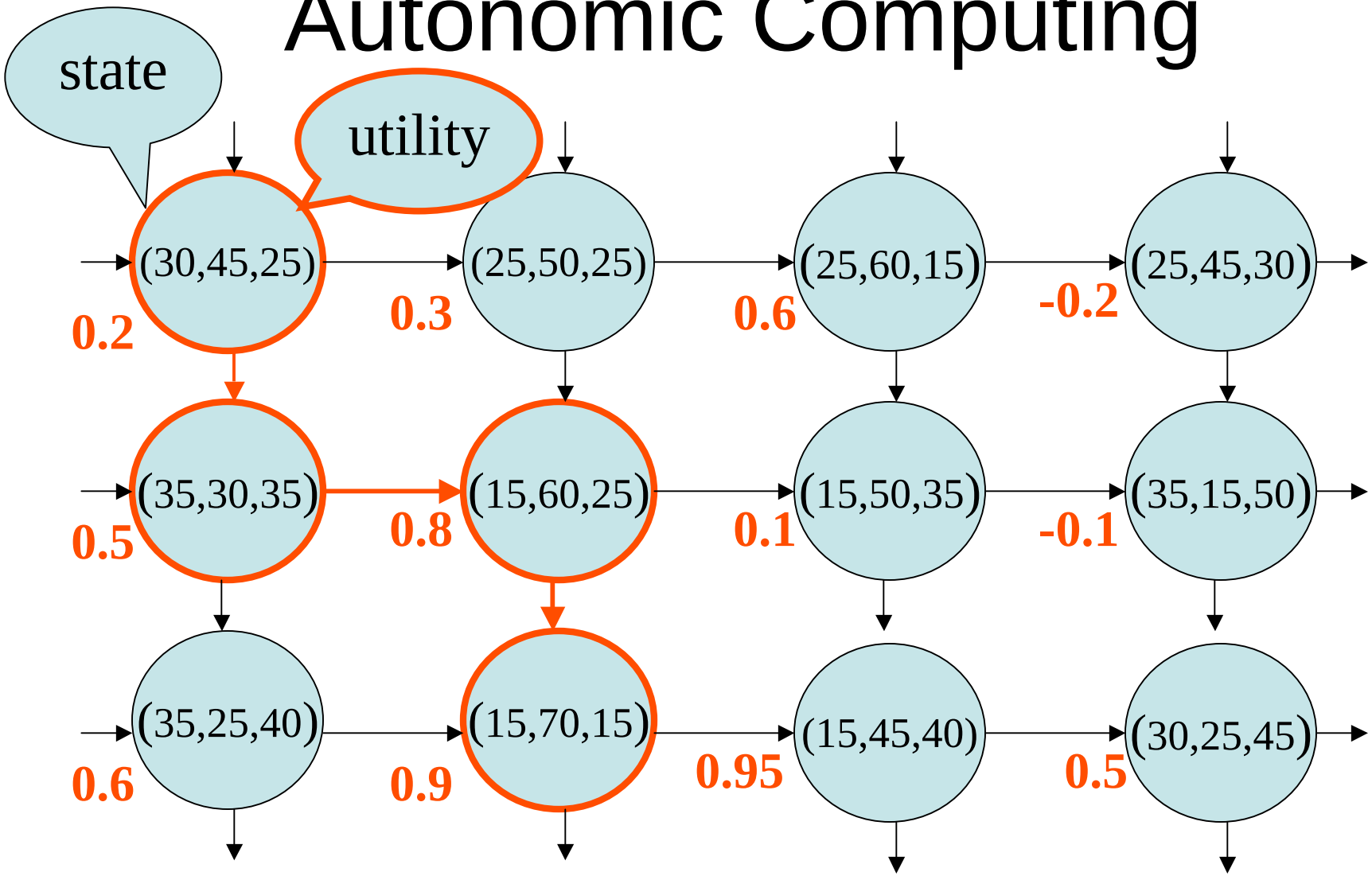
Performance Model-Based Autonomic Computing



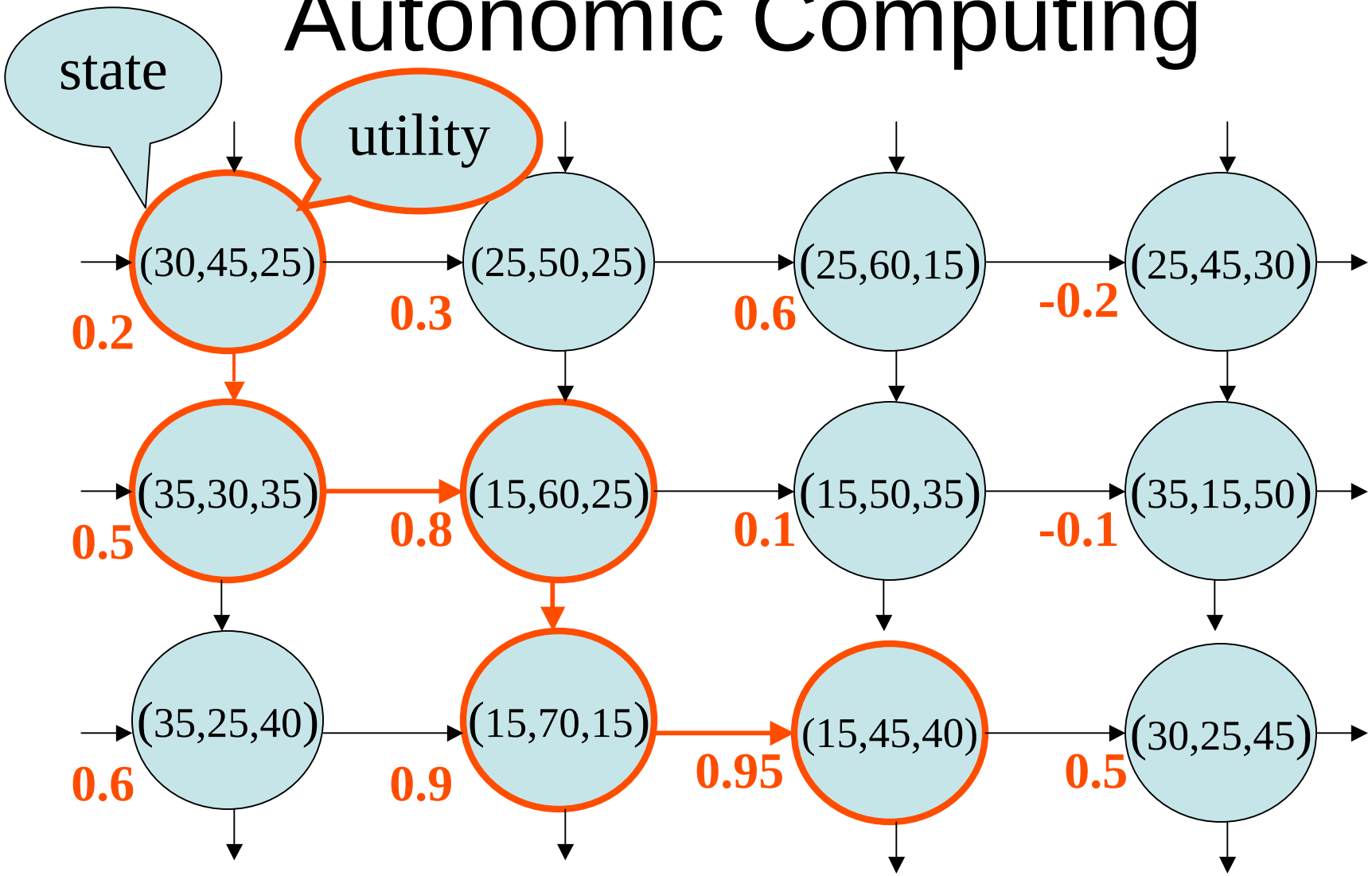
Performance Model-Based Autonomic Computing



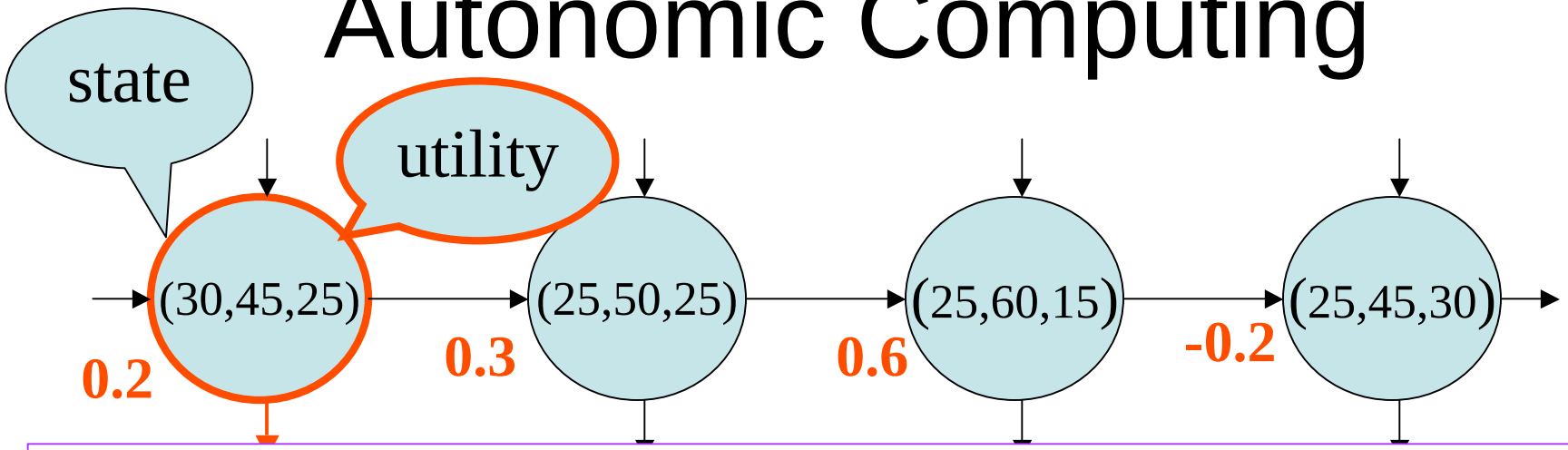
Performance Model-Based Autonomic Computing



Performance Model-Based Autonomic Computing



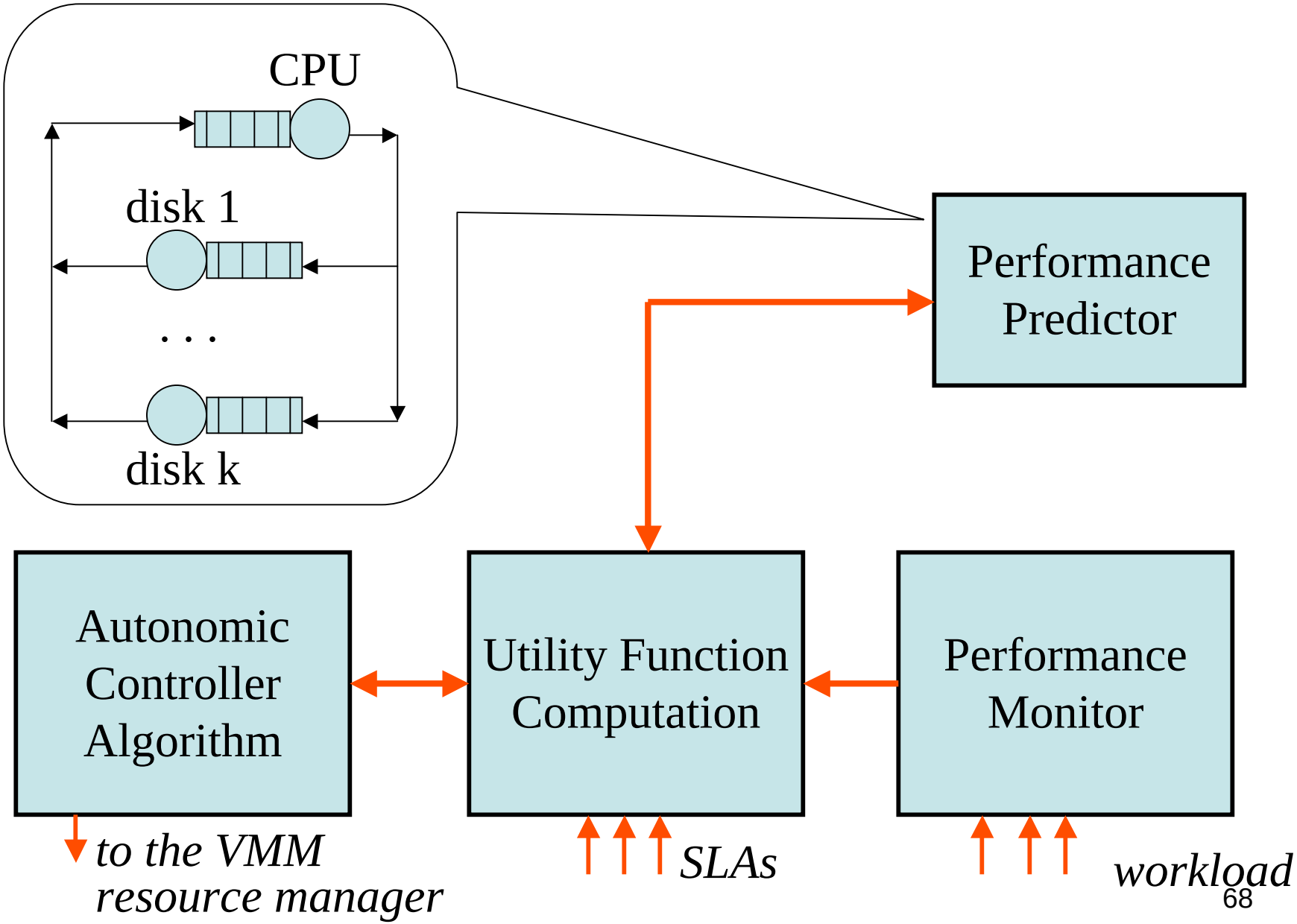
Performance Model-Based Autonomic Computing



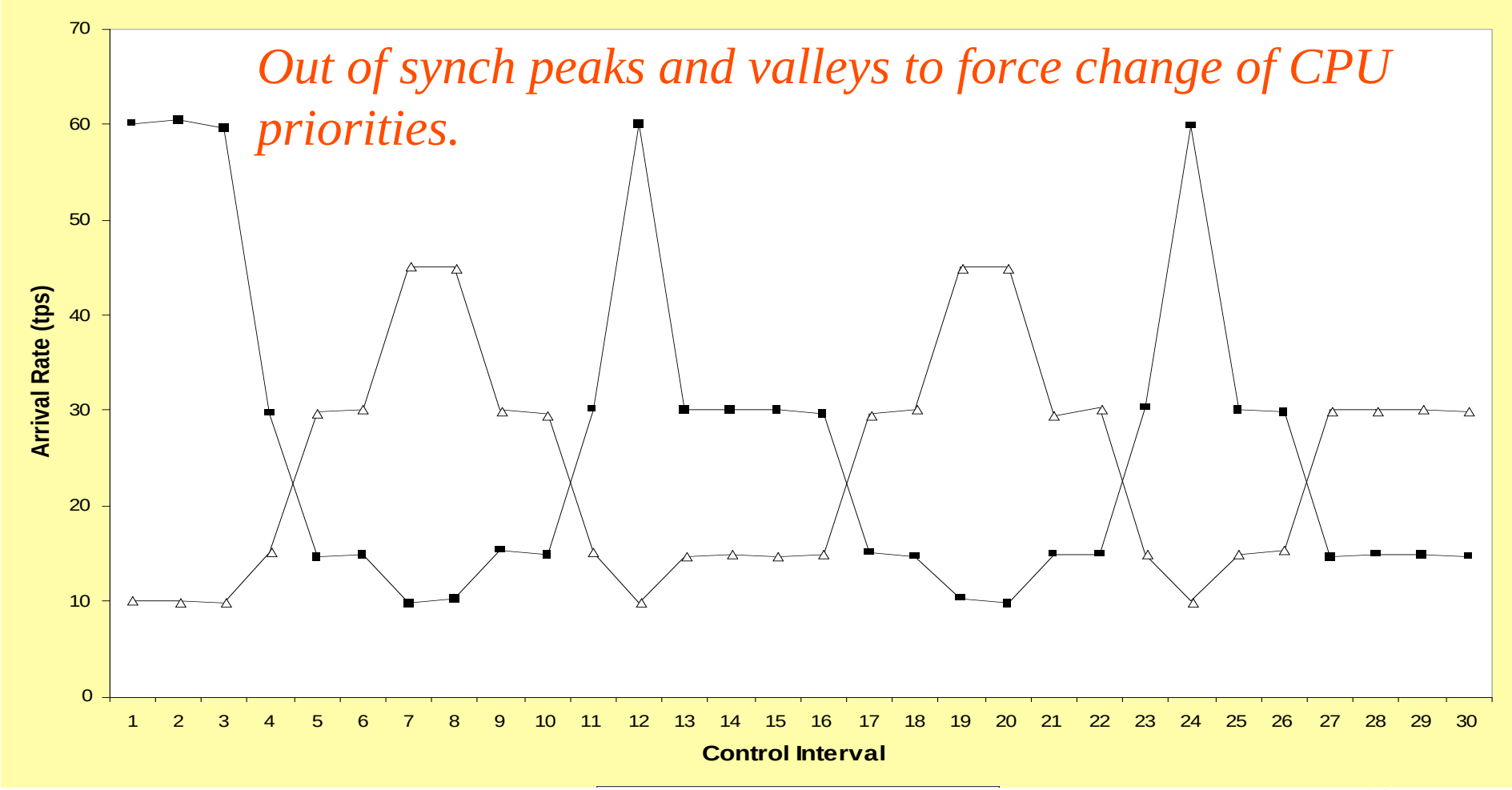
State: CPU share or priority allocated to each VM: (a1, ..., aM)

The utility function at each state is computed with the help of a **predictive queuing network model**. The best state to move to is determined using **combinatorial search** techniques.

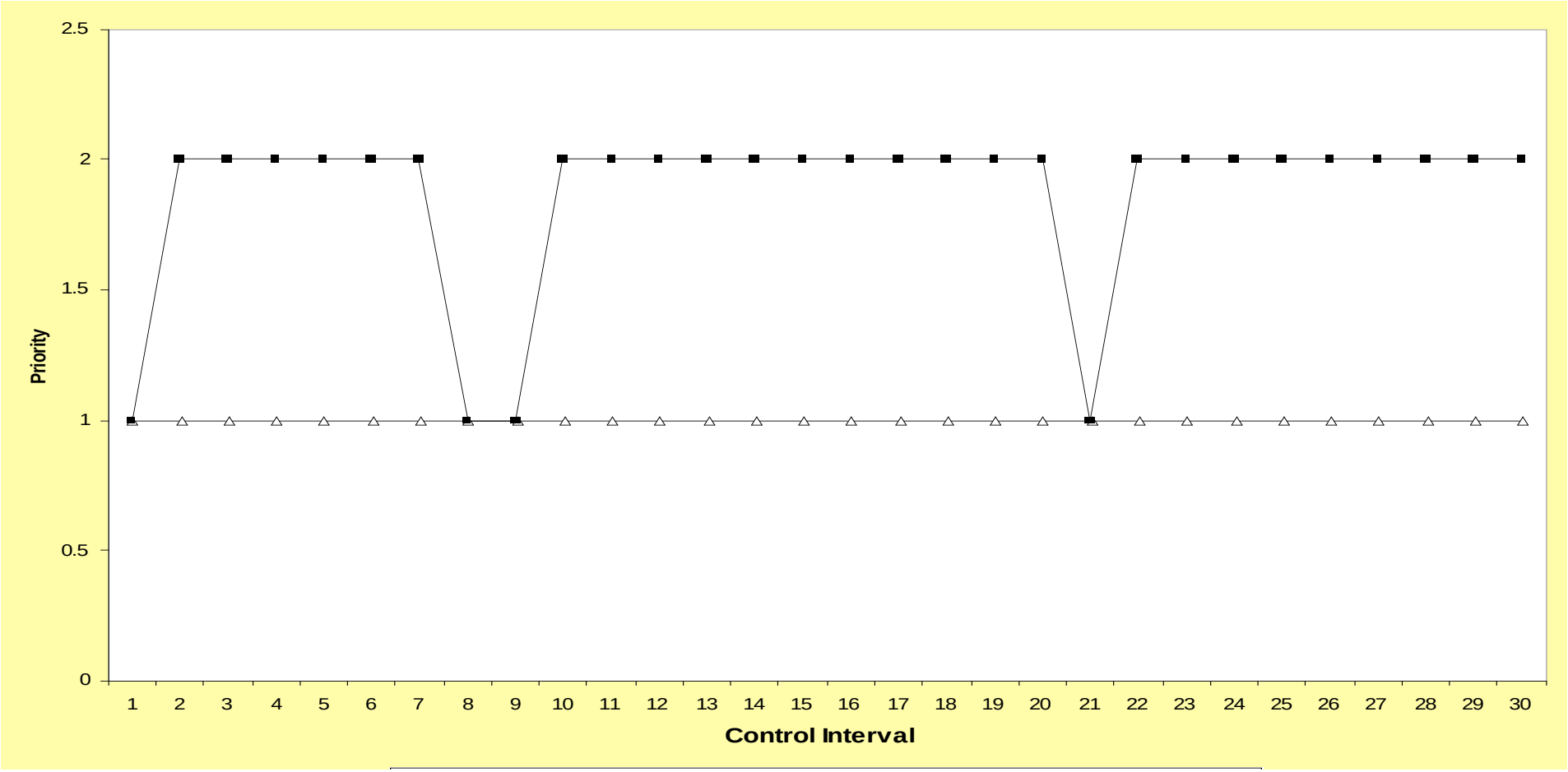
Virtualization Controller Architecture



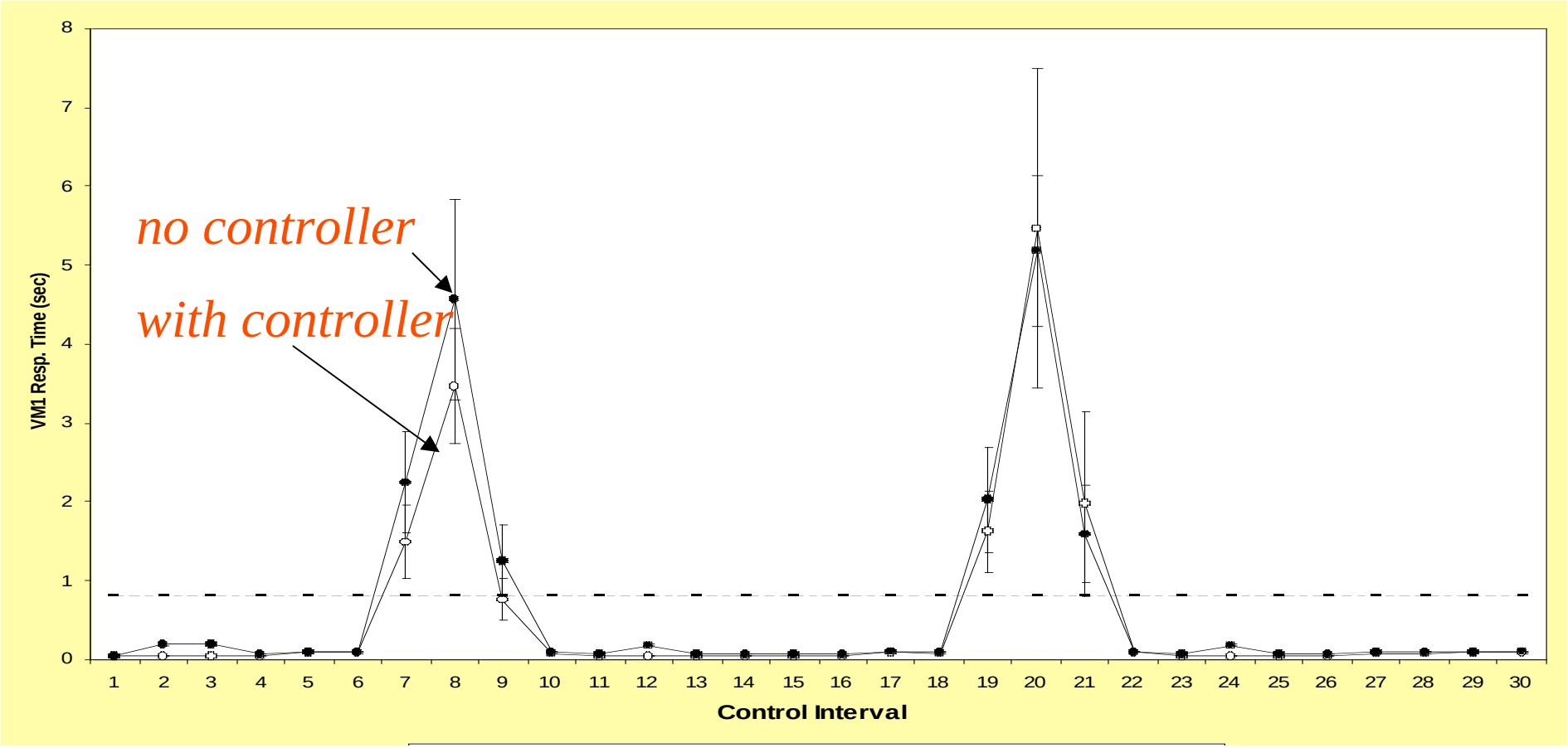
Priority Based CPU Allocation: Workload Variation



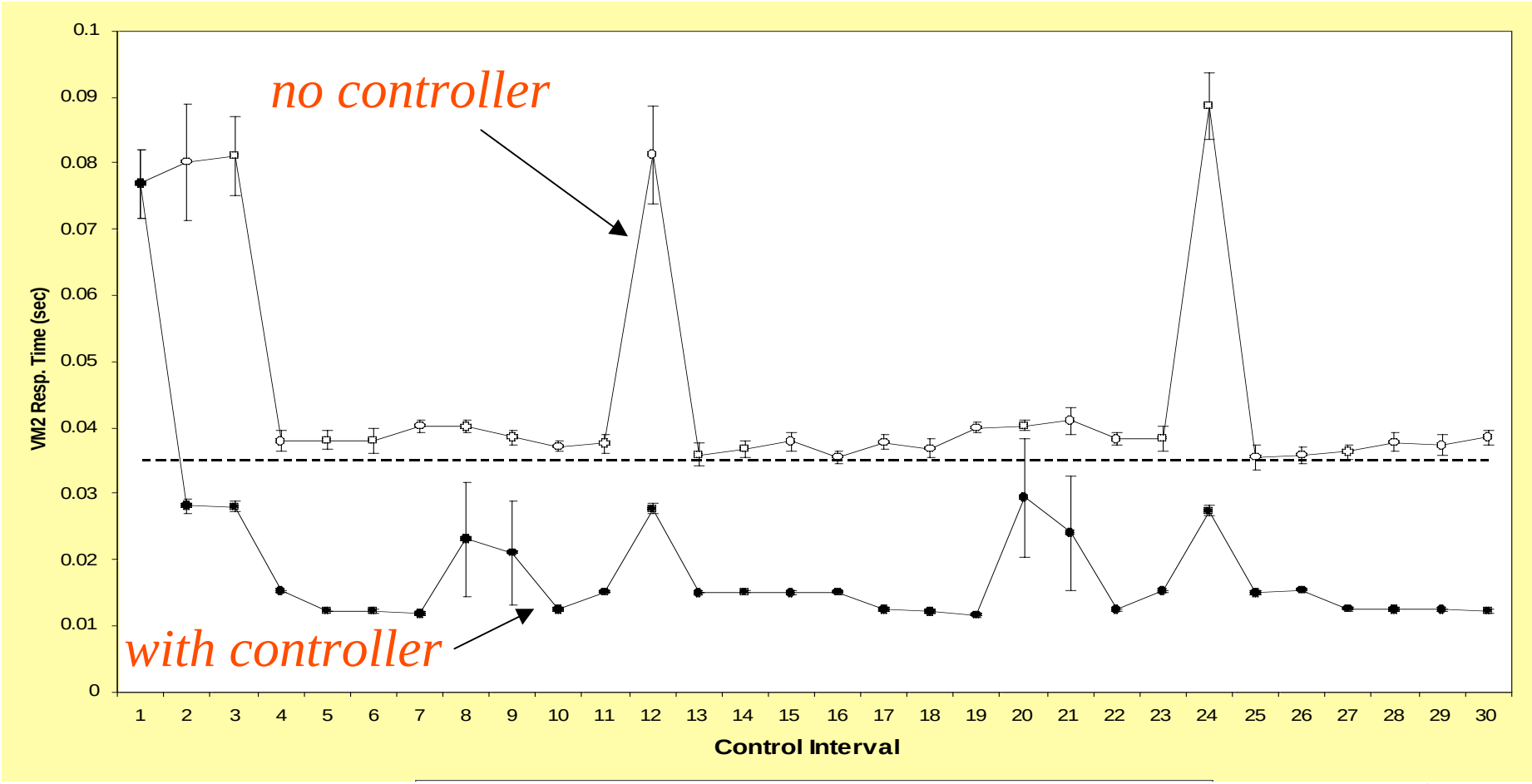
Priority Based CPU Allocation: Variation of VM Priority



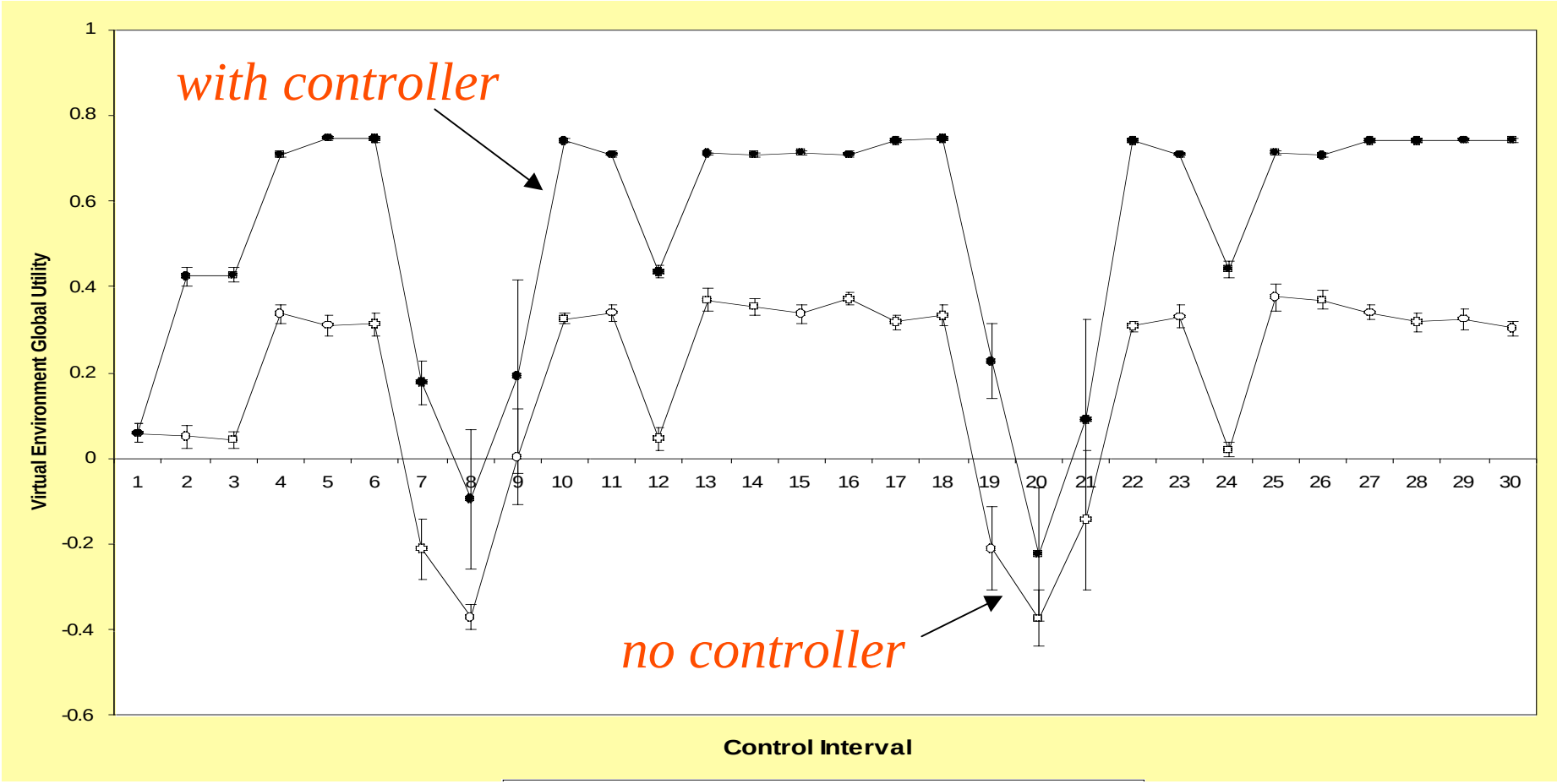
Priority Based CPU Allocation: Response Time for VM1



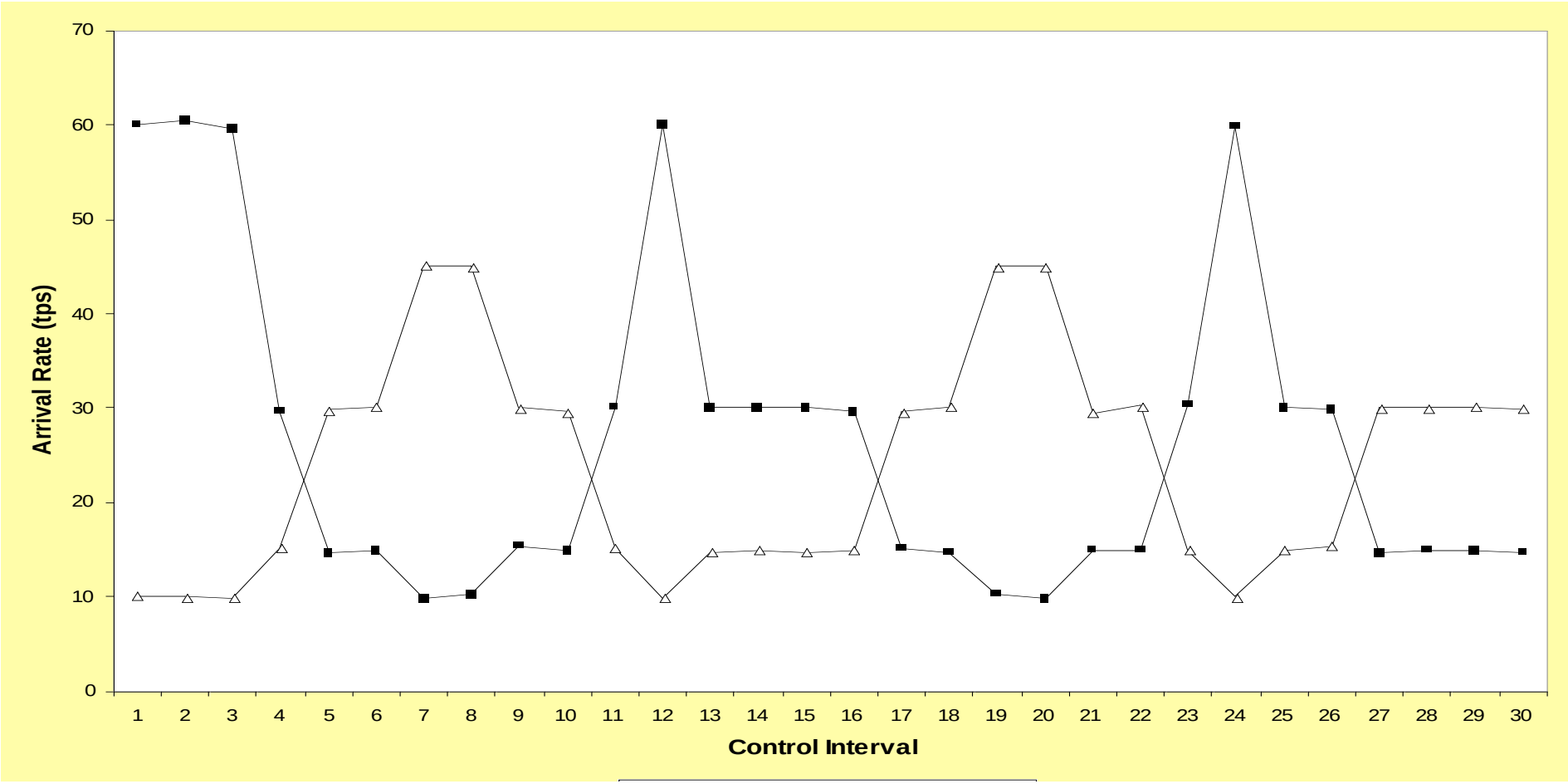
Priority Based CPU Allocation: Response Time for VM2



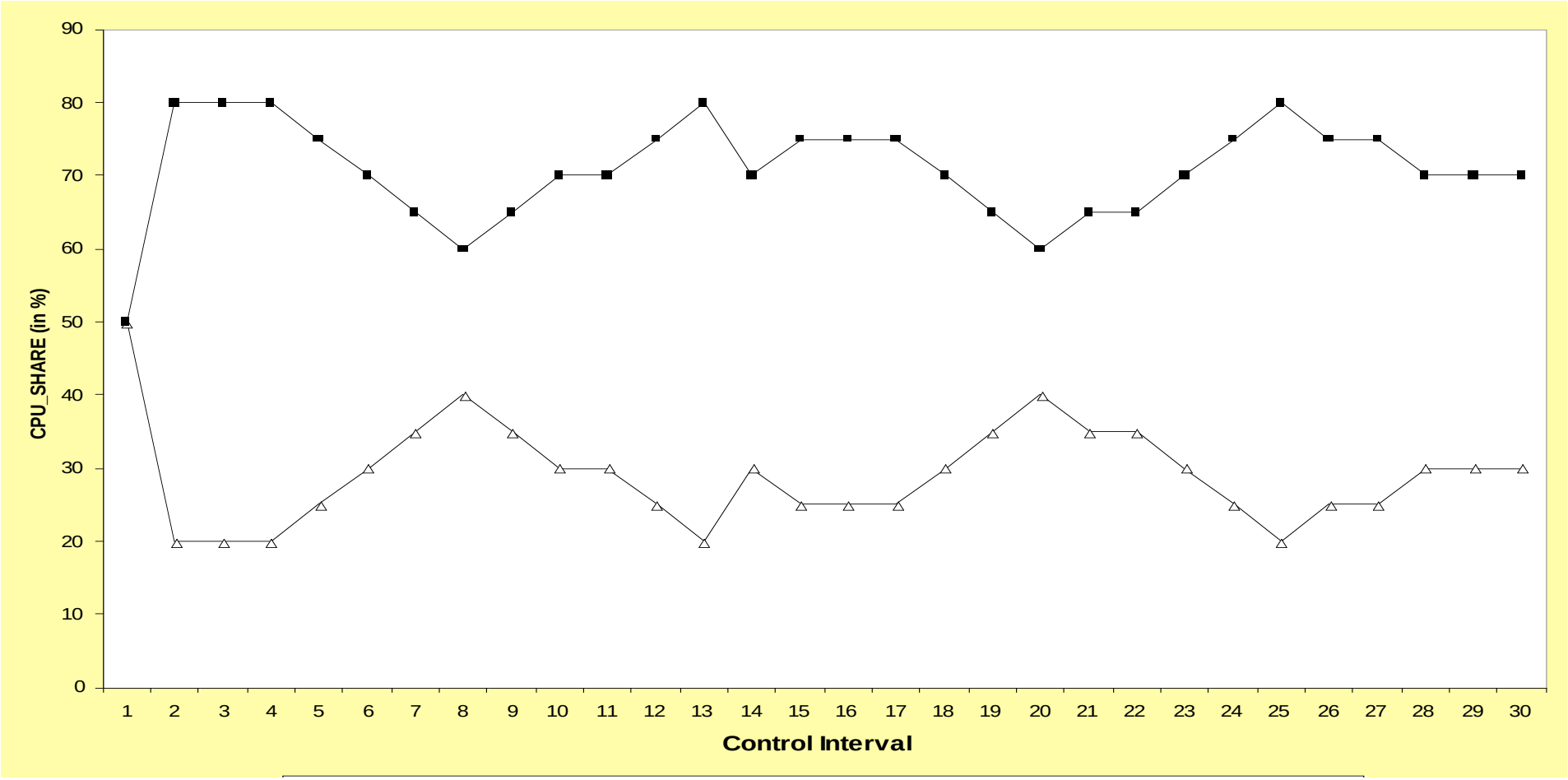
Priority Based CPU Allocation: Global Utility



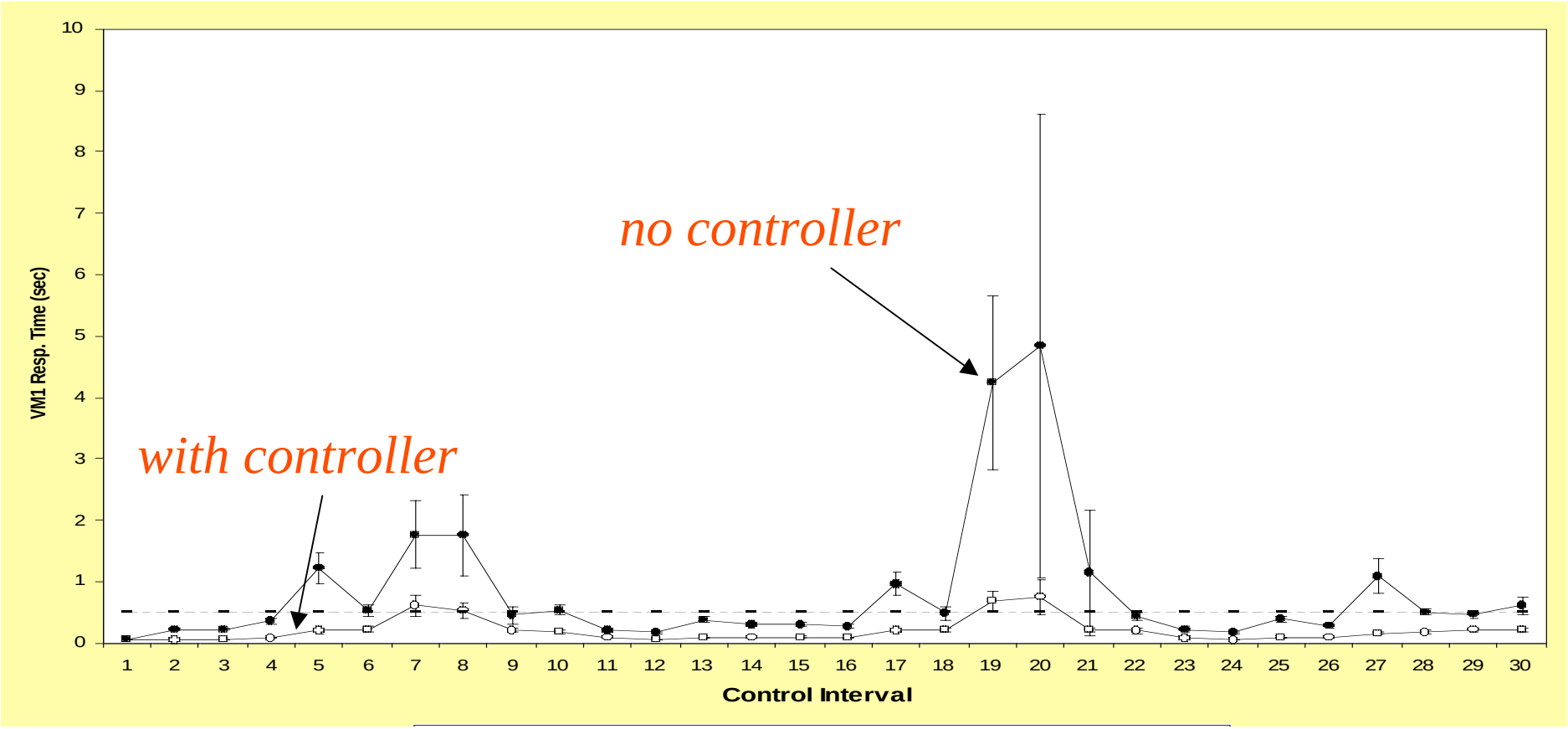
CPU Shares Based Allocation: Workload Variation



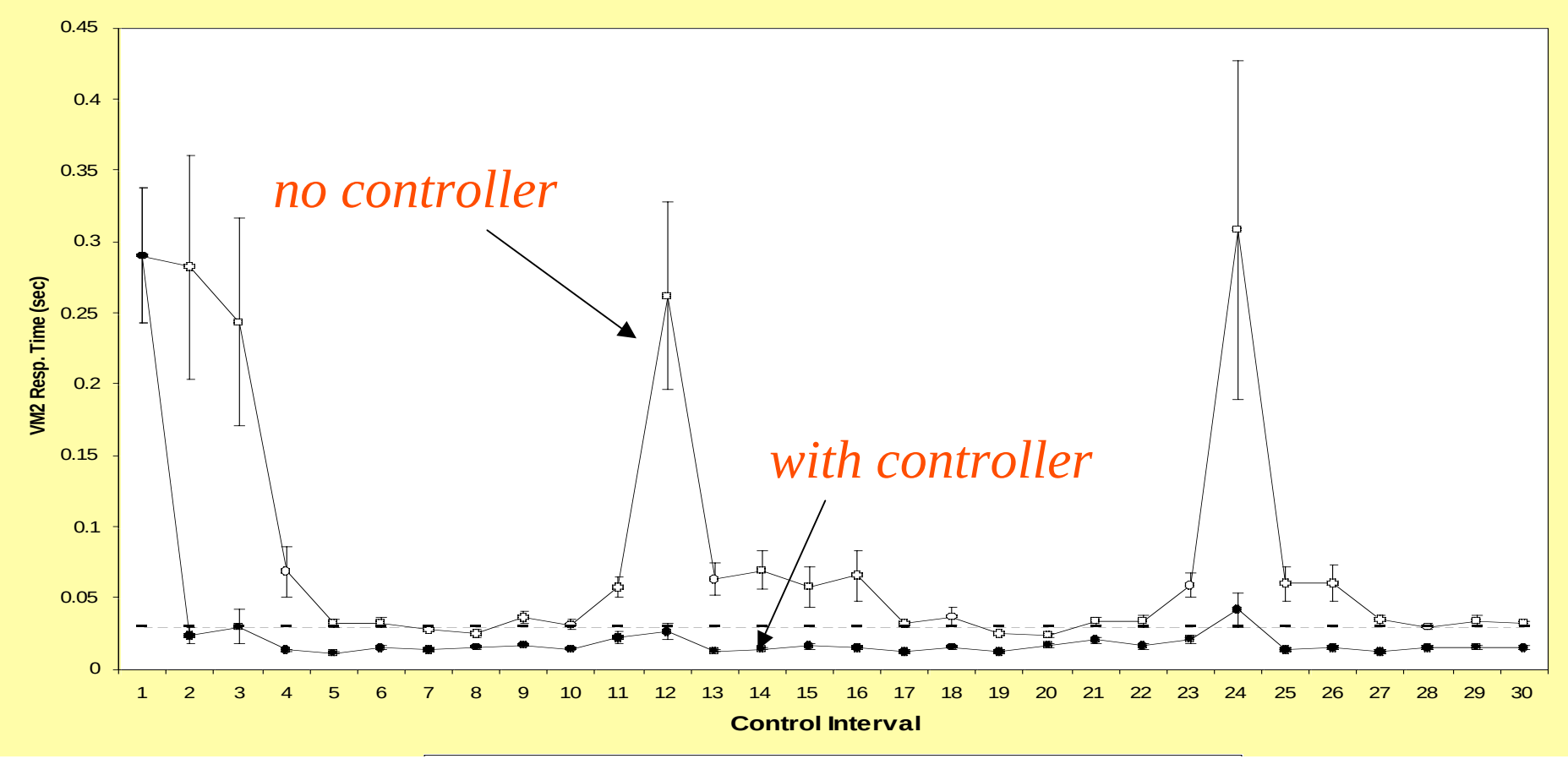
CPU Shares Based Allocation: CPU Shares Variation



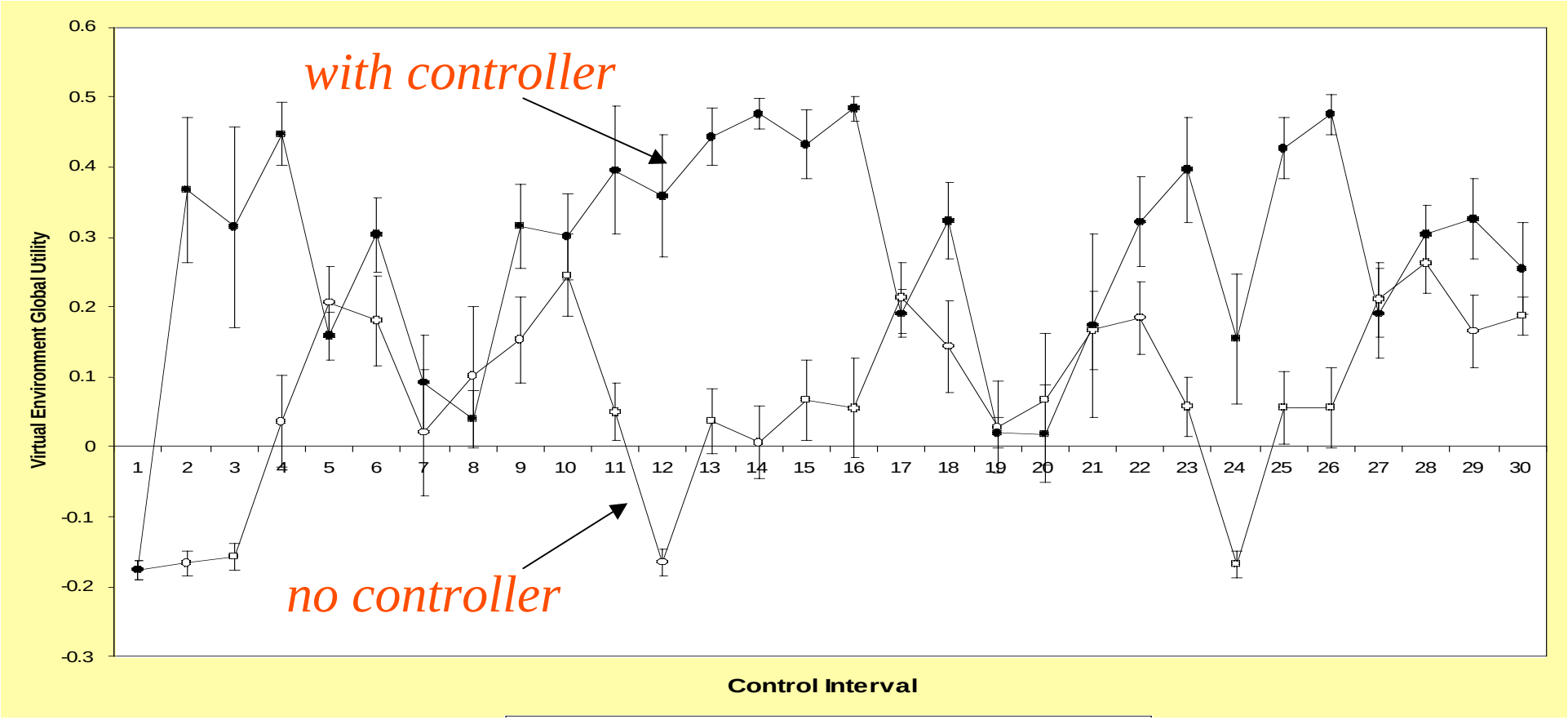
CPU Shares Based Allocation: Response Time for VM1



CPU Shares Based Allocation: Response Time for VM2



CPU Shares Based Allocation: Global Utility



Concluding remarks

- Autonomic computing is an area of growing importance due to its potential to deal with complexity.
- Other techniques used:
  - Machine learning
  - Control theory

Bibliography

- “Autonomic Computing: Guest-editor introduction,” D.A. Menasce and J.O. Kephart, IEEE Internet Computing, Vol. 11, No. 1, January/Febrary 2007.
- “Dynamic Server Allocation for Autonomic Service Centers in the Presence of Failures," Menascé and Bennanni, in the book *Autonomic Computing: Concepts, Infrastructure, and Applications*, eds. S. Hariri and M. Parashar, CRC Press, 2006.
- "Autonomic Virtualized Environments," D.A. Menasce and M.N. Bennani, IEEE International Conference on Autonomic and Autonomous Systems, July 19-21, 2006, Silicon Valley, CA, USA.
- "Resource Allocation for Autonomic Data Centers using Analytic Performance Models," Bennani and Menascé, *Proc. 2005 IEEE International Conf. Autonomic Computing (ICAC-05)*, Seattle, WA, June 13-16, 2005.
- "On the Use of Online Analytic Performance Models in Self-Managing and Self-Organizing Computer Systems," Menascé, Bennanni and Ruan, in the book *Self-Star Properties in Complex Information Systems*, O. Babaoglu, M. Jelasity, A. Montresor, C. Fetzer , S. Leonardi, A. van Moorsel, and M. van Steen, eds., Lecture Notes in Computer Science, Vol. 3460, Springer Verlag, 2005.
- "Assessing the Robustness of Self-Managing Computer Systems under Highly Variable Workloads," Bennani and Menascé, *Proc. 2004 IEEE International Conf. Autonomic Computing (ICAC-04)*, New York, NY, May 17-18, 2004.
- "On the Use of Performance Models to Design Self-Managing Computer Systems," Menascé and Bennani, *Proc. 2003 Computer Measurement Group Conference*, Dallas, TX, Dec. 7-12, 2003.
- Preserving QoS of E-commerce Sites Through Self-Tuning: A Performance Model Approach," D.A. Menasce, R. Dodge and D. Barbara, Proc. 2001 ACM Conference on E-commerce, Tampa, FL, October 14-17, 2001

Thanks! Questions?

www.cs.gmu.edu/faculty/menasce.html