

Overhead and Resource-Contention Aware Analytical Model for Overloaded Web Servers

Vipul Mathur, Varsha Apte

Department of Computer Science and Engineering
Indian Institute of Technology - Bombay
Mumbai

November 6, 2006

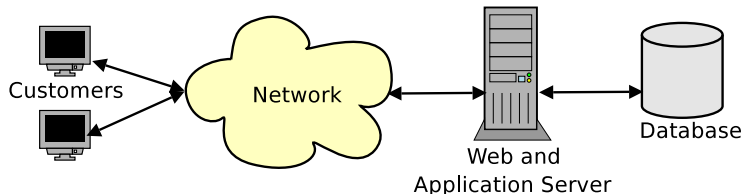


Paper accepted at WOSP'07

Overhead and Resource-Contention Aware Analytical Model for Overloaded Web Servers



Overload in Web Servers

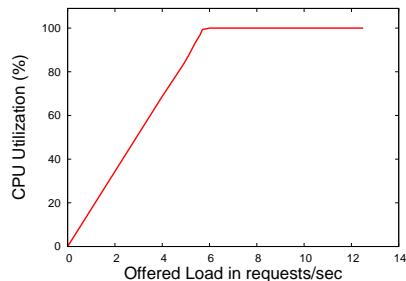
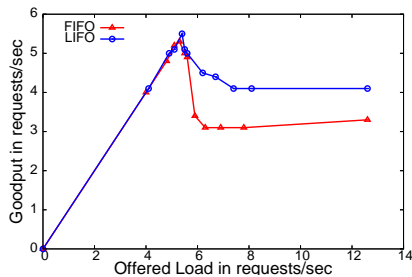


Overload

- Offered load exceeds capacity
- Increased delays in service
- Customers are impatient and may abandon requests



Overloaded Web Server: Experimental Observations



- Abandonments lead to wasted processing time
- Usable throughput (*goodput*) degrades
- Resource utilization remains at 100%
- LIFO performs better during overload
- Experimental observations from [Singhmar et al., 2004]



Analytical Model

Why?

Existing models do not predict goodput drop at onset of overload and subsequent leveling out with 100% utilization.

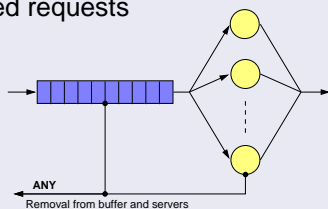
Objectives

- Reproduce goodput degradation and leveling
- Explain better performance of LIFO
- Understand factors that cause this behaviour



System Characteristics of Interest

- Abandonments
 - non-exponential timeout distribution
 - removal behaviour of abandoned requests
- Contention for shared resource
- Overhead processing
- Multi-threaded server
- Limited buffers

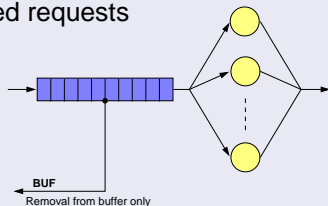


ANY: Removal from Buffer and Servers



System Characteristics of Interest

- Abandonments
 - non-exponential timeout distribution
 - removal behaviour of abandoned requests
- Contention for shared resource
- Overhead processing
- Multi-threaded server
- Limited buffers

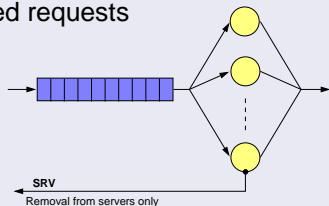


BUF: Removal from Buffer Only



System Characteristics of Interest

- Abandonments
 - non-exponential timeout distribution
 - removal behaviour of abandoned requests
- Contention for shared resource
- Overhead processing
- Multi-threaded server
- Limited buffers

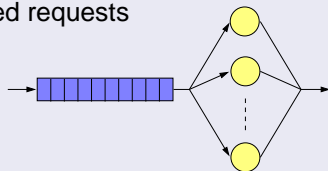


SRV: Removal from Server Only



System Characteristics of Interest

- Abandonments
 - non-exponential timeout distribution
 - removal behaviour of abandoned requests
- Contention for shared resource
- Overhead processing
- Multi-threaded server
- Limited buffers



NONE

No removal of abandoned requests

NONE: No Removal of Abandoned Requests



System Characteristics of Interest

- Abandonments
 - non-exponential timeout distribution
 - removal behaviour of abandoned requests
- Contention for shared resource
- Overhead processing
- Multi-threaded server
- Limited buffers

Our approach

Use queueing systems to model such software servers



Extended Kendall Notation

$$A/S/C/K/P/D + T/R$$

A	arrival process	P	population (default ∞)
S	service time distribution	D	scheduling discipline timeout distribution
C	number of servers	T	{ <i>EXP, ERL, G, PH, D</i> }
K	maximum number of requests in the system	R	request removal policy { <i>NONE, BUF, SRV, ANY</i> }



Notation

Extended Kendall Notation

$$A/S/C/K/P/D + T/R$$

A	arrival process	P	population (default ∞)
S	service time distribution	D	scheduling discipline timeout distribution
C	number of servers	T	{ <i>EXP, ERL, G, PH, D</i> }
K	maximum number of requests in the system	R	request removal policy { <i>NONE, BUF, SRV, ANY</i> }

Example

$$M/M/3/10/\bullet + ERL/BUF$$

a \bullet indicates that one of many choices is used



Basic Model

$M/M/C/K/$ with LIFO and Abandonments

[Movaghar, 1996] [Movaghar, 2000]

- Solves the $M/M/C/K/FIFO + G/BUF$ and $M/M/C/K/FIFO + G/ANY$ models
- LIFO not considered
- In BUF, users stop being impatient upon entering service



Basic Model

$M/M/C/K/$ with LIFO and Abandonments

[Movaghar, 1996] [Movaghar, 2000]

- Solves the $M/M/C/K/FIFO + G/BUF$ and $M/M/C/K/FIFO + G/ANY$ models
- LIFO not considered
- In BUF, users stop being impatient upon entering service

[Pla et al., 2004]

- Solves $M/M/C/K/FIFO + PH/BUF$ and $M/M/C/K/LIFO + PH/BUF$
- Users stop being impatient on entering service



Basic Model

$M/M/C/K$ with LIFO and Abandonments

We adapt/extend these for our use

- Users may abandon requests in service
- Such requests are completed but count towards 'unsuccessful work'
- EXP and ERL timeout distributions are considered



Basic Model

$M/M/C/K$ with LIFO and Abandonments

We adapt/extend these for our use

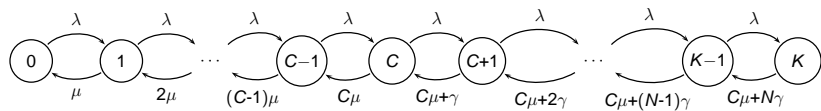
- Users may abandon requests in service
- Such requests are completed but count towards 'unsuccessful work'
- EXP and ERL timeout distributions are considered

Comparison of LIFO vs. FIFO in this model

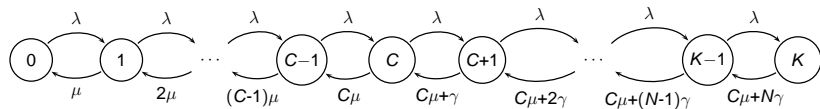
- Goodput
- Response time



$M/M/C/K/\bullet + EXP/BUF$ model



$M/M/C/K/\bullet + EXP/BUF$ model

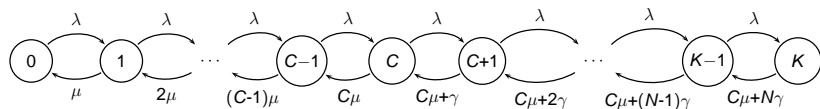


Observation

With $EXP(\gamma)$ timeout distribution, the model is the same for both LIFO and FIFO



$M/M/C/K/\bullet + EXP/BUF$ model



Observation

With $EXP(\gamma)$ timeout distribution, the model is the same for both LIFO and FIFO

Conclusion

Goodput of LIFO and FIFO is same for exponentially distributed user timeouts



Choice of Timeout Distribution

Observation

- Most users wait for some non-zero amount of time before becoming impatient
- Mode of $X \sim EXP(\gamma)$ is at $X = 0$



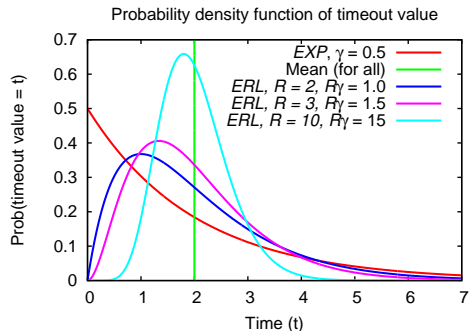
Choice of Timeout Distribution

Observation

- Most users wait for some non-zero amount of time before becoming impatient
- Mode of $X \sim EXP(\gamma)$ is at $X = 0$

Choice

- We choose $Erlang(R, R\gamma)$ as a representative non-exponential distribution



Erlang rate parameter is set at $R\lambda$ to keep overall mean timeout at $1/\lambda$



Tracking Impatience State

State of each position (buffer/server)

$$i_k = \begin{cases} 0, & \text{position } k \text{ is empty,} \\ r, & \text{impatience stage of user, } 1 \leq r \leq R, \\ -1, & \text{user timed out, but request still in system.} \end{cases}$$



Figure: $ERL(R, R\lambda)$ customer impatience model



Solving the $M/M/C/K/\bullet + ERL/\bullet$ Model

- Construct Markov chain whose state describes the state of each position.

- $s = (\overbrace{i_{K-1}, \dots, i_{C+1}, i_C}^{(K-C) \text{ buffers}}, \overbrace{i_{C-1}, \dots, i_0}^{C \text{ servers}})$ $s \in \mathcal{S}$

- Solve numerically¹ to obtain steady-state probabilities π_s

¹Using tools such as PRISM and SHARPE .



Solving the $M/M/C/K/\bullet + ERL/\bullet$ Model

- Construct Markov chain whose state describes the state of each position.

- $s = (\overbrace{i_{K-1}, \dots, i_{C+1}, i_C}^{(K-C) \text{ buffers}}, \overbrace{i_{C-1}, \dots, i_0}^{C \text{ servers}})$ $s \in S$

- Solve numerically¹ to obtain steady-state probabilities π_s
- Hence obtain
 - Throughput Λ_C , Goodput Λ_{GOOD}
 - Blocking probability P_B
 - Successful completion P_{GOOD}
 - Completion of abandoned requests P_{BAD}
 - Timeout and removal from buffer P_T

¹Using tools such as PRISM and SHARPE .



Solving the $M/M/C/K/\bullet + ERL/\bullet$ Model

- Construct Markov chain whose state describes the state of each position.

$$\text{■ } s = \left(\overbrace{i_{K-1}, \dots, i_{C+1}, i_C}^{(K-C) \text{ buffers}}, \overbrace{i_{C-1}, \dots, i_0}^{C \text{ servers}} \right) \quad s \in \mathcal{S}$$

- Solve numerically¹ to obtain steady-state probabilities π_s

- Hence obtain

- Throughput Λ_C , Goodput Λ_{GOOD}
- Blocking probability P_B
- Successful completion P_{GOOD}
- Completion of abandoned requests P_{BAD}
- Timeout and removal from buffer P_T

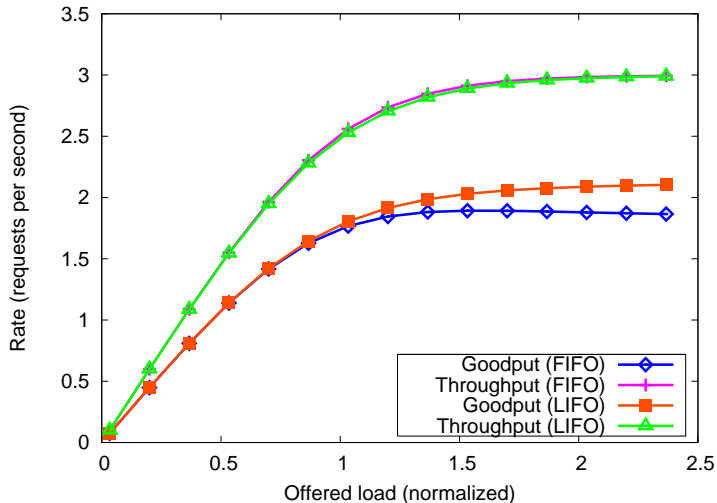
- We solve the $M/M/C/K/\bullet + ERL/BUF$ and $M/M/C/K/\bullet + ERL/NONE$ models for FIFO and LIFO

¹Using tools such as PRISM and SHARPE .



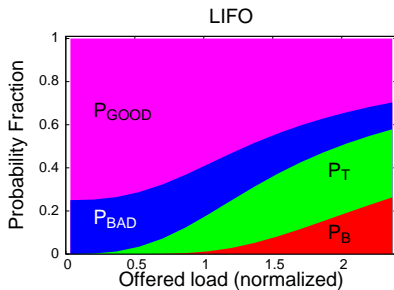
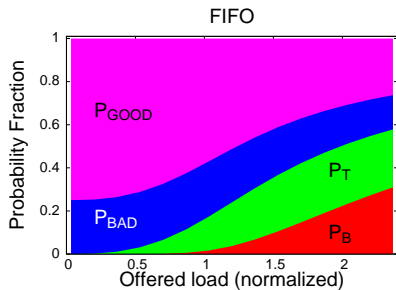
Results for the $M/M/3/10/\bullet + ERL/BUF$ Model

$1/\gamma = 2s, \mu = 1$ request/s



Results for the $M/M/3/10/\bullet + ERL/BUF$ Model

$1/\gamma = 2s, \mu = 1$ request/s

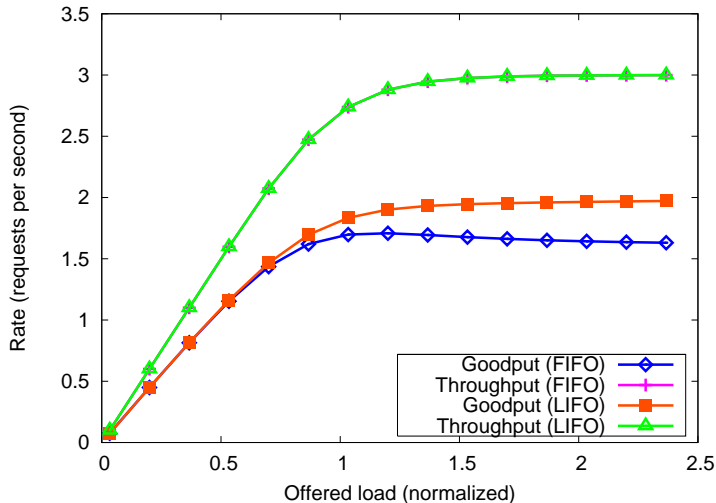


Probability Fractions



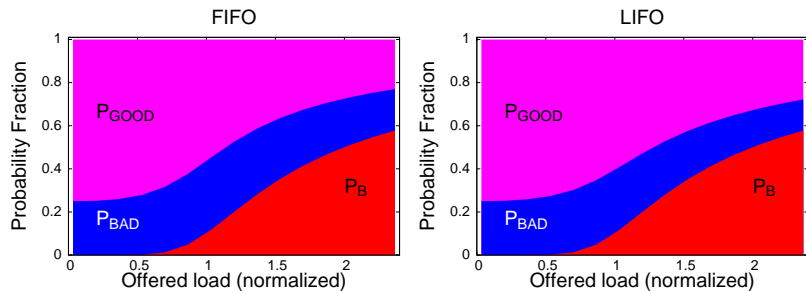
Results for the $M/M/3/10/\bullet + ERL/NONE$ Model

$1/\gamma = 2s, \mu = 1$ request/s



Results for the $M/M/3/10/\bullet + ERL/NONE$ Model

$1/\gamma = 2s, \mu = 1$ request/s



Probability Fractions



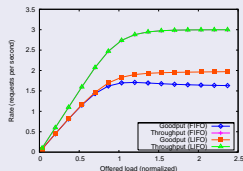
Basic $M/M/C/K$ Model Takeaway

LIFO vs. FIFO in $M/M/C/K$ model with abandonments

- Non-exponential timeout distribution necessary to explain goodput difference
- Goodput when overloaded: LIFO > FIFO (Erlang timeouts)
- Response time: LIFO better even for exponential timeouts

Basic model of overloaded Web server

- Shows goodput being a fraction of throughput



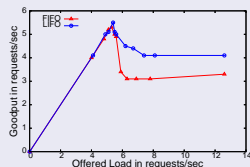
Basic $M/M/C/K$ Model Takeaway

LIFO vs. FIFO in $M/M/C/K$ model with abandonments

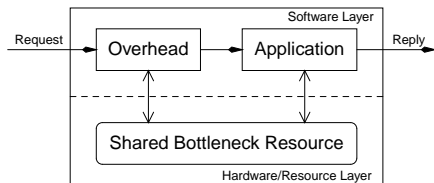
- Non-exponential timeout distribution necessary to explain goodput difference
- Goodput when overloaded: LIFO > FIFO (Erlang timeouts)
- Response time: LIFO better even for exponential timeouts

Basic model of overloaded Web server

- Shows goodput being a fraction of throughput
- Unable to explain observed drastic drop in goodput



Layered Model with Overhead and Resource Sharing



- Capture behaviour of software servers: overhead processing and resource sharing

Figure: High-level model of request processing. Overhead (OHD) and application (APP) processing stages contend for shared bottleneck resource.



Layered Model with Overhead and Resource Sharing

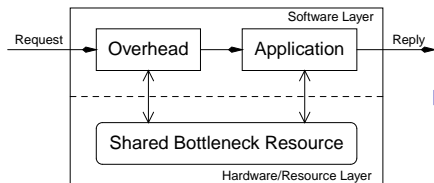


Figure: High-level model of request processing. Overhead (OHD) and application (APP) processing stages contend for shared bottleneck resource.

- Capture behaviour of software servers: overhead processing and resource sharing
- *OHD* stage
 - TCP connection setup
 - Root *listener* thread
 - Receive HTTP request, parse, assign *worker* thread



Layered Model with Overhead and Resource Sharing

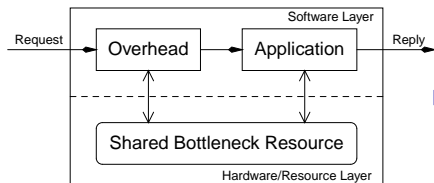


Figure: High-level model of request processing. Overhead (OHD) and application (APP) processing stages contend for shared bottleneck resource.

- Capture behaviour of software servers: overhead processing and resource sharing
- *OHD* stage
 - TCP connection setup
 - Root *listener* thread
 - Receive HTTP request, parse, assign *worker* thread
- *APP* stage
 - Request processing by *worker*
 - Scripting engine
 - Back-end DB queries, processing results. . .



Choice of Models

OHD stage model

- Client timeout
⇒ remove request from queue/service (*ANY*)
- $M/M/1/K/FIFO + EXP/ANY$ [Movaghar, 2000]



Choice of Models

OHD stage model

- Client timeout
⇒ remove request from queue/service (*ANY*)
- $M/M/1/K/FIFO + EXP/ANY$ [Movaghar, 2000]

APP stage model

- Remove request from buffer only (*BUF*)
or no removal (*NONE*)
- $M/M/C/K/\bullet + ERL/BUF$ or $M/M/C/K/\bullet + ERL/NONE$



OHD and APP Stages

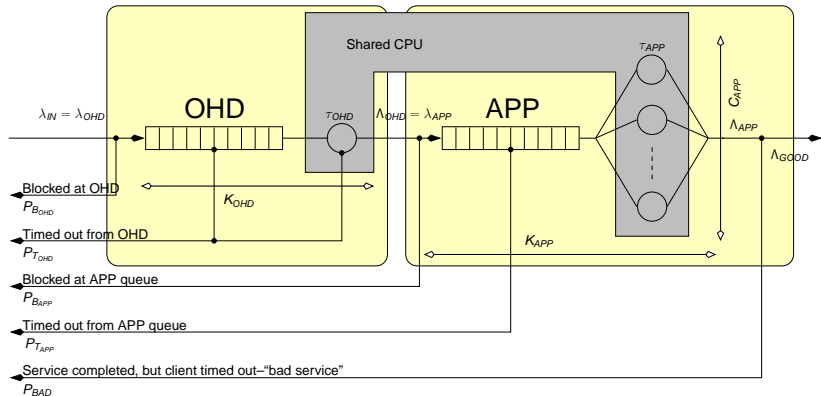


Figure: Queuing Model Representation



Resource Sharing Model

Shared CPU Model: Time Sharing

Service time requirements (τ_{OHD} , τ_{APP}) are scaled up by average number of active threads (n_{CPU}) in the system



Resource Sharing Model

Shared CPU Model: Time Sharing

Service time requirements (τ_{OHD} , τ_{APP}) are scaled up by average number of active threads (n_{CPU}) in the system

$$\tau'_{OHD} = n_{CPU} \tau_{OHD}$$

$$\tau'_{APP} = n_{CPU} \tau_{APP}$$



Resource Sharing Model

Shared CPU Model: Time Sharing

Service time requirements (τ_{OHD} , τ_{APP}) are scaled up by average number of active threads (n_{CPU}) in the system

$$\tau'_{OHD} = n_{CPU} \tau_{OHD}$$

$$\tau'_{APP} = n_{CPU} \tau_{APP}$$

$$n_{CPU} = \max(1, n_{OHD} + n_{APP})$$



Resource Sharing Model

Shared CPU Model: Time Sharing

Service time requirements (τ_{OHD} , τ_{APP}) are scaled up by average number of active threads (n_{CPU}) in the system

$$\tau'_{OHD} = n_{CPU} \tau_{OHD}$$

$$\tau'_{APP} = n_{CPU} \tau_{APP}$$

$$n_{CPU} = \max(1, n_{OHD} + n_{APP})$$

$$n_{OHD} = \lambda_{OHD} (1 - P_{B_{OHD}} - P_{T_{OHD}}) \tau'_{OHD}$$

$$n_{APP} = \lambda_{APP} (1 - P_{B_{APP}} - P_{T_{APP}}) \tau'_{APP}$$



Resource Sharing Model

Shared CPU Model: Time Sharing

Service time requirements (τ_{OHD} , τ_{APP}) are scaled up by average number of active threads (n_{CPU}) in the system

$$\tau'_{OHD} = n_{CPU} \tau_{OHD}$$

$$\tau'_{APP} = n_{CPU} \tau_{APP}$$

$$n_{CPU} = \max(1, n_{OHD} + n_{APP})$$

$$n_{OHD} = \lambda_{OHD} (1 - P_{B_{OHD}} - P_{T_{OHD}}) \tau'_{OHD}$$

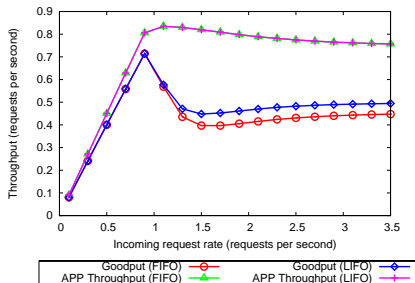
$$n_{APP} = \lambda_{APP} (1 - P_{B_{APP}} - P_{T_{APP}}) \tau'_{APP}$$

Solved iteratively

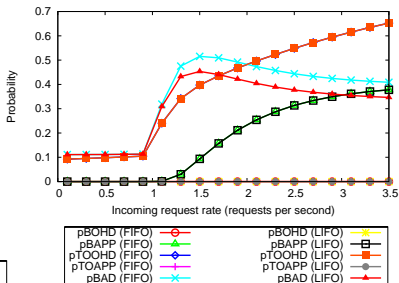


Results from Layered Model I

- APP model is $M/M/3/13/\bullet + ERL/NONE$
- OHD model is $M/M/1/101/FIFO + EXP/ANY$
- $T_0 = 2s$, $\tau_{OHD} = 0.2s$ and $\tau_{APP} = 1.0s$.



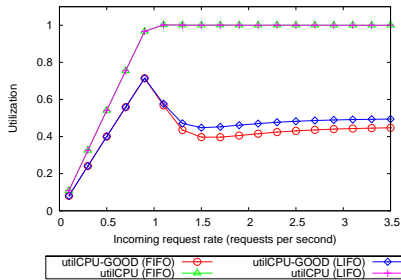
(a) Throughput vs. offered load



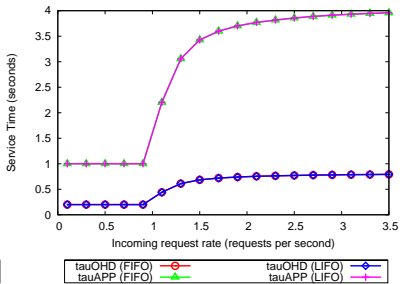
(b) Probabilities vs. offered load



Results from Layered Model II



(c) Utilization vs. offered load



(d) Service Time vs. offered load

'Good CPU Utilization' is the percentage of time CPU spends processing requests that count towards goodput.



Summary of Analytical Model

- Existing models do not explain observed goodput behaviour
- LIFO response time better during overload
- For non-exponential timeouts, LIFO goodput higher when overloaded
- Our resource contention and overhead aware model reproduces goodput degradation and flattening behaviour
- The model gives insights into the dynamics of an overloaded software server



Thank You! Questions?



<http://www.cse.iitb.ac.in/perfnet>



References I



C.Hirel, Sahner, R., Zang, X., and Trivedi, K. (2000).
Reliability and performability modeling using SHARPE 2000.
In 11th International Conference, TOOLS 2000, Schaumburg, Illinois USA.



Diao, Y., Hellerstein, J. L., Parekh, S., and Bigus, J. P. (2003).
Managing Web server performance with AutoTune agents.
IBM Systems Journal, 42(1):136–149.



Kamra, A., Misra, V., and Nahum, E. M. (2004).
Yaksha: a self-tuning controller for managing the performance of 3-tiered web sites.
In Proceedings of the Twelfth IEEE International Workshop on Quality of Service (IWQOS'04), pages 47–56.



References II



Kwiatkowska, M., Norman, G., and Parker, D. (2002).

PRISM: Probabilistic symbolic model checker.

In Proceedings of the 12th International Conference on Modelling Techniques and Tools for Computer Performance Evaluation (TOOLS'02), pages 200–204. Springer.



Mathur, V. and Apte, V. (2004).

A computational complexity-aware model for performance analysis of software servers.

In Proceedings of the 12th IEEE Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS'04), Volendam, Netherlands. IEEE Computer Society.



References III



Menascé, D. A., Bennani, M. N., and Ruan, H. (2005).

On the Use of Online Analytic Performance Models in Self-Managing and Self-Organizing Computer Systems, volume 3460 of *Lecture Notes in Computer Science*, chapter Self-Star Properties in Complex Information Systems.

Springer Verlag.



Movaghar, A. (1996).

On queueing with customer impatience until the beginning of service. In *Proceedings of the 2nd International Computer Performance and Dependability Symposium*, page 150, Washington, DC, USA. IEEE Computer Society.



Movaghar, A. (2000).

On queuing with customer impatience until the end of service.

In *Proceedings of the 4th International Computer Performance and Dependability Symposium*, page 167, Washington, DC, USA. IEEE Computer Society.



References IV



Pla, V., Casares-Giner, V., and Martínez, J. (2004).

On a multiserver finite buffer queue with impatient customers.

In Proceedings of the 16th ITC Specialist Seminar on Performance Evaluation of Wireless and Mobile Systems, Antwerp, Belgium.



Singhmar, N., Mathur, V., Apte, V., and Manjunath, D. (2004).

A combined LIFO-priority scheme for overload control of E-commerce Web servers.

In the International Infrastructure Survivability Workshop (affiliated with the 25th IEEE International Real-Time Systems Symposium), Lisbon, Portugal.



Welsh, M., Culler, D., and Brewer, E. (2001).

SEDA: an architecture for well-conditioned, scalable internet services.

SIGOPS Operating Systems Review, 35(5):230–243.

