



ELSEVIER

Performance Evaluation 42 (2000) 223–239

**PERFORMANCE
EVALUATION**
An International
Journal

www.elsevier.com/locate/peva

Business-oriented resource management policies for e-commerce servers[☆]

Daniel A. Menascé^{a,*}, Virgilio A.F. Almeida^b, Rodrigo Fonseca^b, Marco A. Mendes^b

^a *Department of Computer Science, MS4A5, George Mason University, Fairfax, VA 22030, USA*

^b *Department of Computer Science, University of Federal de Minas Gerais, Belo Horizonte, MG 30161, Brazil*

Abstract

Quality of service of e-commerce sites has been usually managed by the allocation of resources such as processors, disks, and network bandwidth, and by tracking conventional performance metrics such as response time, throughput, and availability. However, the metrics that are of utmost importance to the management and shareholders of a Web store are revenue and profit. Thus, resource management schemes for e-commerce servers should be geared towards optimizing business metrics as opposed to conventional performance metrics. This paper uses a state transition graph called customer behavior model graph (CBMG) to describe a customer session. It then presents a family of priority based resource management policies for e-commerce servers. Priorities change dynamically as a function of the state a customer is in and as a function of the amount of money the customer has accumulated in his/her shopping cart. A detailed simulation model was developed to assess the gain of these dynamic policies with respect to policies that are oblivious to economic considerations. Simulation results show that the multilevel dynamic priority scheme suggested here can significantly improve the values of business-oriented metrics, such as revenue per second, during peak periods. E-commerce sites that use this approach will be able to improve revenue at peak times with the same server capacity. © 2000 Elsevier Science B.V. All rights reserved.

Keywords: E-commerce; Resource management; Customer behavior model graph; Workload characterization; Business-oriented metrics

1. Introduction

It has been recognized by many that congestion and poor performance can be the major impediments for the success of e-commerce. Many e-commerce sites, especially those in the financial trading business, have been facing serious problems and financial losses when customers are not allowed to trade in a timely manner. Some disgruntled customers sue on-line trading services if they feel that they have been short changed and others just move their business elsewhere.

IT managers of Web stores have been managing allocation of resources such as processors, disks, and networks, by tracking conventional performance metrics such as response time, throughput, and

[☆] The subject matter of this paper is covered by a pending patent application at the United States Patent and Trademark Office.

* Corresponding author.

E-mail addresses: menasce@cs.gmu.edu (D.A. Menascé), virgilio@dcc.ufmg.br (V.A.F. Almeida), rfonseca@dcc.ufmg.br (R. Fonseca), corelio@dcc.ufmg.br (M.A. Mendes).

availability. However, the metrics that are of utmost importance to the management of a Web store are revenue and profit. Thus, resource management schemes for e-commerce servers should be aimed at optimizing business-oriented metrics such as revenue per second instead of focusing on conventional performance metrics.

Resource management policies for e-commerce sites should be based on the behavior of customers and on how they change state as they navigate through the site, going from browsing to searching, selecting items, adding them to their shopping carts and paying. We present in this paper a family of priority based resource management policies for e-commerce servers. Priorities change dynamically as a function of the state a customer is in, as a function of the user profile, and as a function of the amount of money the customer has accumulated in his/her shopping cart. The policies can also be tuned to provide good performance to customers who are just entering the Web store even before they add any items to their shopping carts. We believe that resource management policies such as the ones presented in this paper should be integrated into future commercial e-commerce products. This would allow e-commerce servers to handle peak loads with existing resources in a way that minimizes revenue loss due to poor quality of service.

A detailed simulation model was developed to assess the gain of our policies with respect to policies that are oblivious to monetary considerations. The simulation uses the same probability distributions employed by SURGE [2], a workload generator for Web sites, augmented by a generator of e-commerce requests that mimic typical customer behavior. Requests generated by a customer within a session are generated from a customer behavior model graph (CBMG) that captures how users navigate through the site. The CBMG representation is used in this paper as a means of characterizing workloads for e-commerce sites. As an example, two types of customer profiles, heavy and occasional, were considered. The results of our simulations show that the dynamic priority scheme introduced in this paper can increase business-oriented metrics such as revenue per second by almost 30% over the no priority case.

The rest of this paper is organized as follows. Section 2 discusses new metrics for e-commerce sites. Section 3 describes e-commerce workloads as composed of session requests and CBMGs. Section 4 describes new resource management policies for e-commerce servers. Section 5 describes the simulator and the simulation environment used to analyze the new policies proposed. Section 6 describes the numerical results obtained. Section 7 compares our work with that of others. Finally, Section 8 presents some concluding remarks.

2. Novel metrics for e-commerce

The quality of the service provided by online information systems, such as Web servers, has been traditionally assessed by metrics such as response time, throughput, reliability, and availability. Response time can be measured at the server side, in which case it does not include any client and external network time, or it can be measured from the user's perspective, in which case it includes components such as browser time, network access time at the client side, ISP time (at both ends), Internet time, network access time at the server side, and server response time.

Throughput is usually measured in requests per second or transactions per second and determines the rate at which the system can deliver work. While throughput is important from the perspective of the site administrator, it is irrelevant to end-users who are concerned about the response time they get. When the response time is too high, users complain if they have no choice but use the system. In e-commerce, customers usually have a choice: they leave the site and move to another Web store. This translates into lost revenue for the Web store and decreased throughput.

E-commerce brings the need of novel metrics that reflect at the same time the needs of the Web store and those of its users. One such metric defined here is *revenue throughput*, denoted by X^+ , measured in dollars per second generated by completed transactions. One of the goals of a Web store should be to maximize the revenue throughput. Implicit in this metric is a measure of customer satisfaction with response time, since if customers were really unhappy they would not shop at the Web store and the revenue throughput would decrease. A frustrated customer is a quickly exiting customer. Another metric is *potential lost revenue/sec*. This metric, denoted by X^- , represents the rate at which dollars accumulated in shopping carts are not converted into sales because customers leave the Web store due to inadequate response time. The resource management policies discussed in this paper attempt to maximize a Web store's revenue throughput without the need to increase server capacity.

3. The nature of e-commerce workloads

E-commerce workloads are composed of sessions. A *session* is defined as a sequence of requests made by a customer during a single visit to a site. Examples of requests made by an online shopper are: Browse, Search, Select, Add to the Shopping Cart, Register, and Pay. An online trader would have different operations, such as: Enter a Stock Order, Research a Fund, Obtain Real-time Quotes, and Retrieve Company Profiles. The allowed sequences of requests can be described by a state transition graph called CBMG [8,9]. This graph has one node for each possible state (e.g., Home Page (h), Browse (b), Search (s), Select (t), Add (a), and Pay (p)) and transitions between these states.

CBMGs can be obtained by analyzing HTTP logs as indicated in [7,8]. Each such CBMG represents a class of visits with similar navigational patterns. The CBMGs obtained this way are called *operational CBMGs*. Operational and stochastic approaches are characterized in detail in [4]. In the process of obtaining operational CBMGs, one can obtain the frequency of transitions between the states of the operational CBMG. If one makes the Markovian assumption that the next page to be visited depends strongly on the contents of the current page, we can view a CBMG as a model of classes of visits with similar navigational patterns. In this case, the CBMG is called a *model CBMG* and the transition frequencies are replaced by transition probabilities. A model CBMG can be used as a predictive tool of user behavior and can also be used to drive simulations of e-commerce sites, as done in Section 5. The qualifiers "operational" and "model" will be dropped when referring to CBMGs in case the meaning is clear from the context.

Different types of visits may be characterized by different CBMGs in terms of the transition probabilities. Thus, workload characterization for e-commerce entails in determining the set of CBMGs that best characterize customer behavior. As e-commerce sites become more sophisticated, they can process their logs to identify user profiles based on the navigation and buying patterns. These profiles could be specified as CBMGs. Thus, as a customer starts to navigate through a site, the Web store could attempt to match the customer to one of the existing profiles and assign priorities based on the user profile. The marketing and economic value of customized navigation experience made possible by the vast amount of information collected by Web servers has been pointed out in [11]. Some Web stores request that users login before they start to navigate through the site. In these cases, it may be even easier to match a customer with a profile.

As an example, consider two customer profiles: occasional and heavy buyers. The first category is composed of customers who use the Web store to find out about existing products, such as new books or best fares and itineraries for travel, but end up not buying, most of the time, at the Web store. The

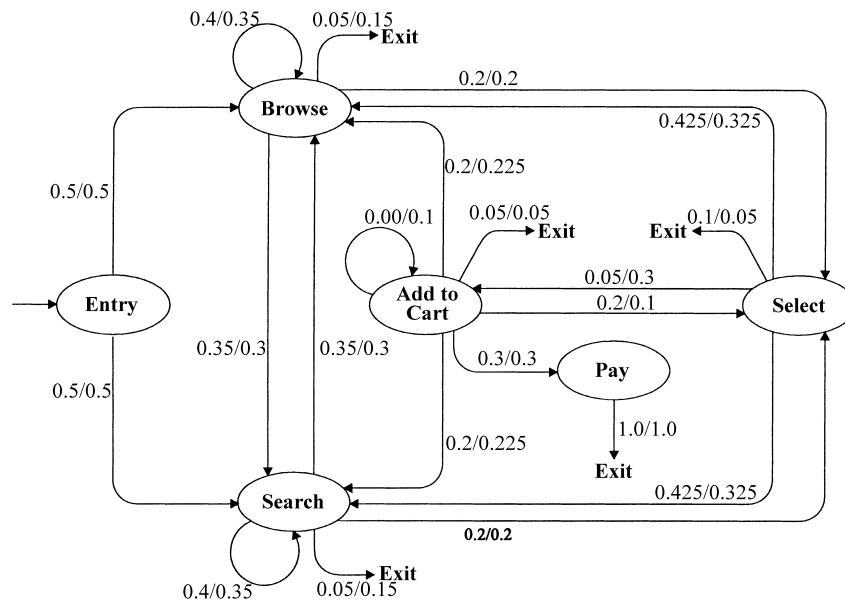


Fig. 1. A CBMG.

second category is composed of customers who have a higher probability of buying if they see a product that interest them at a suitable price. Fig. 1 shows a combined representation of the CBMGs for these types of customer profiles. The transition probabilities are indicated as a pair x/y , where x represents the transition probability for occasional buyers and y for heavy buyers.

Note that the CBMG of Fig. 1 is just an example. CBMGs can have many other states, depending on the nature of the electronic business. The exit state is not explicitly represented in Fig. 1 to improve its readability. Transitions to the exit state are represented as arrows leaving the states of the CBMG. The transitions that indicate exit from the Web store, from states other than Pay, are indicative of spontaneous exits, i.e., exits due to reasons such as unfriendly navigation scheme, poor contents, and unattractive prices. Spontaneous exits are not due to poor performance. The resource management policies described in this paper attempt to reduce the number of customers who leave the Web store due to poor performance.

A CBMG can be more formally characterized by a pair (P, Z) , where $P = [p_{i,j}]$ is an $n \times n$ matrix of transition probabilities between the n states of the CBMG and $Z = [z_{i,j}]$ is an $n \times n$ matrix that represents the average server-perceived think times between states of the CBMG, defined as the average time elapsed since the server completed a request for a customer until it receives the next request from the same customer within the same session. We adopt the convention that state 1 is always the entry (y) state and state n is always the exit (e) state. Note that the elements of the first column and last rows of matrix P are all zeroes since, by definition, there are no transitions back to the entry state from any state nor any transitions out of the exit state.

The CBMG provides useful information regarding the average number of visits V_j to each state of the CBMG for each visit to the site. The values of V_j can be obtained by solving the system of linear equations:

$$V_1 = 1, \quad V_j = \sum_{k=1}^n V_k \times p_{k,j} \quad \text{for } j = 2, \dots, n. \quad (1)$$

Note that since $p_{n,k} = 0 \quad \forall k = 1, \dots, n$, $V_n = 1$. For each CBMG, we have a different system of linear equations because the transition probabilities are different. The solutions to the system of linear equations in (1) for the CBMGs of occasional and heavy buyers are ($V_y = 1$, $V_b = 6.76$, $V_s = 6.76$, $V_a = 0.14$, $V_t = 2.73$, $V_p = 0.04$, $V_e = 1$) and ($V_y = 1$, $V_b = 2.71$, $V_s = 2.71$, $V_a = 0.37$, $V_t = 1.12$, $V_p = 0.11$, $V_e = 1$), respectively.

Useful metrics can be obtained from the average number of visits to the states of the CBMG. The first, called *average session length* and denoted by \bar{S} , is the average number of states visited by a customer per visit to the Web store. Thus, $\bar{S} = \sum_{k=2}^{n-1} V_k$. The average session length for occasional (o) and heavy (h) buyers, for the CBMGs of Fig. 1 is $\bar{S}^o = 17.45$ and $\bar{S}^h = 8.03$, respectively. Another metric of interest is the *buy to visit ratio*, denoted by BV. This is the ratio between the average number of customers who buy from the Web store and the total number of visits to the Web store. For each type of customer, BV is given by V_p , the average number of visits to the Pay state per visit to the site. Suppose that in the case of Fig. 1, 90% of customers who initiate sessions are occasional buyers. Then, $BV = 0.9 \times 0.04 + 0.1 \times 0.11 = 0.047$.

4. New resource management policies

One of the goals of an e-commerce server should be to maximize the conversion of traffic into sales transactions in order to increase the revenue generated by the site. To that end, one must understand the behavior of Web servers and how performance may impact user behavior, in particular during peak periods when the site's resources are stressed by bursts of customers. Therefore, resource management policies should be devised to support business-oriented goals.

4.1. Limitations of current policies

This is a typical scenario in e-commerce sites. Sometimes, the traffic to the site becomes too high (e.g., three or four times greater than the average traffic) and the server capacity is not able to handle the customers who are trying to enter the site. Usually, the quality of service is degraded to all customers who attempt to visit the site during these peak periods. Long waiting times to interact with the Web server and refused connections are typical indicators of poor quality of service in e-commerce sites. Current resource management policies do not make any distinction between customers and do not provide differentiated quality of service. For example, an e-store may want to favor customers who really intend to purchase something. However, in current systems, resources are equally shared between customers who are making a purchase and customers who are just browsing.

General-purpose operating systems and Web servers do not provide adequate support for resource management of e-commerce sites. They allocate resources to multiple processes considering only throughput and ignore the relative importance of processes and their meaning to business goals. Operating systems usually manage the allocation of CPU time, memory, and I/O, without taking into consideration any special characteristic of the workload, such as who generated the requests and what amount of money is associated with these requests.

4.2. E-commerce oriented resource management policies

E-commerce oriented resource management policies should be adaptive and evolve according to customer usage and profiles. Resources should be allocated on an economic basis. The resource allocation

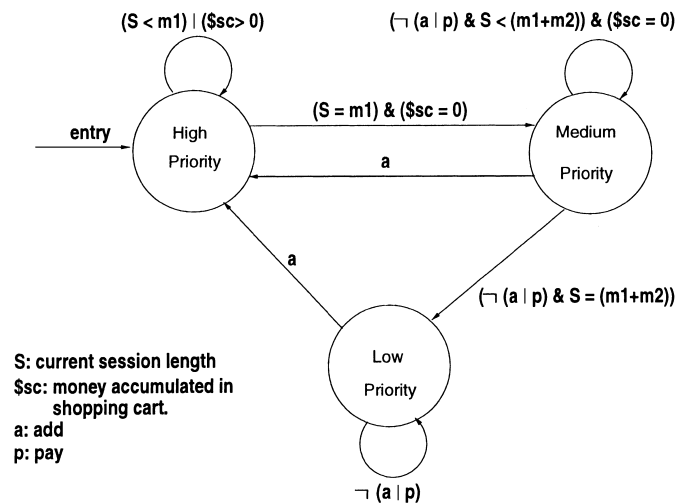


Fig. 2. Priority scheme for e-commerce sites.

policies examined in this paper are geared towards CPU and disk scheduling at the various servers of a Web store. The optimization criteria for the policies is to maximize the value of the revenue throughput.

To achieve this goal, the policies analyzed here use priorities based on user profiles, on current session length (S), on the amount of money accumulated in the customer's shopping cart ($\$sc$), and on the states visited in the CBMG. Three priority classes are considered: high, medium, and low. Customers transition between these priority classes is as shown in the priority transition diagram of Fig. 2.

The transitions between priority classes are labeled by conditions that may depend on the state visited, the session length S , and the accumulated money in the shopping cart $\$sc$. As shown in Fig. 2, all users entering the Web store start at high priority and remain there if they add at least one item to their shopping cart, in which case $\$sc > 0$, or their session length is less than a threshold m_1 . The rationale is to give high priority to users who are in the early stages of exploring the store so that they do not give up before they are given an opportunity to find out what the store has to offer. After the session length exceeds the threshold m_1 and the customer has not added anything to its shopping cart, its priority is lowered to Medium. The customer stays at that level of priority while no items are added to the shopping cart and the session length has not reached the second threshold $m_1 + m_2$. When this happens, the customer's priority is lowered to Low. The priority is changed to High from Medium or from Low if an item is added to the shopping cart. Note that by making m_2 large enough, the priority scheme can be reduced to a two-class priority scheme. By making m_1 large enough, the priority framework of Fig. 2 reduces to a single-class priority scheme. Thus, by varying the values of m_1 and m_2 one can generate a large number of policies.

Subclasses may be defined within each priority class according to user profiles. For example, if the profiles heavy buyer and occasional buyer are defined, higher priority would be given to all customers with a heavy buyer profile over customers with an occasional buyer profile. Within each priority subclass, we use the following disciplines at processors and disks:

- *CPU*. Processor sharing (PS) for classes Medium and Low and ordered by $\$sc$ for the High priority class. In other words, if $\$sc$ for customer 1 is higher than $\$sc$ for customer 2, then customer 1 has higher non-preemptive priority at the CPU than customer 2.
- *Disks*. FIFO for classes Medium and Low and ordered by $\$sc$ for the High priority class.

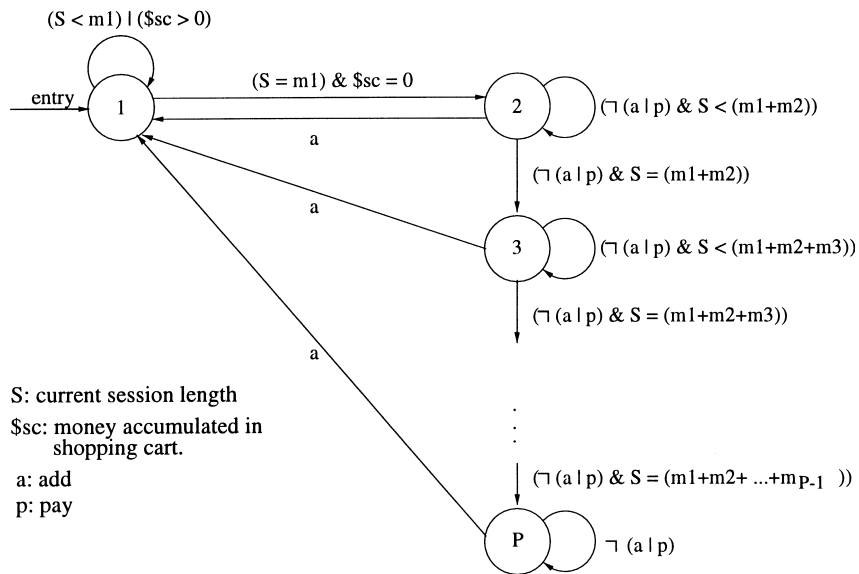


Fig. 3. Multilevel priority scheme for e-commerce sites.

The family of priorities discussed above can be generalized to a multilevel priority scheme in which there are P priority levels with 1 being the highest priority and P the lowest as shown in Fig. 3. In this case, there are $P - 1$ thresholds m_1, \dots, m_{P-1} . Transitions to the highest priority can occur from any of the other priority levels as soon as customers add items to their shopping carts.

5. A framework to compare the policies

Because no current system implements all the ideas and concepts proposed in this paper, we need to build an experimental environment to simulate adaptive systems, customer behavior, and workloads that conform to these systems. We built an e-commerce site simulator which is a hybrid between a trace-driven simulator and a discrete event simulator. We use a realistic Web workload generation tool to generate a stream of requests to start customer sessions and use the CBMGs and the simulator to generate the individual requests that make up a session.

The workload of an e-commerce site is highly dependent on the interaction between customers and the store site. For example, after deciding on which item to buy after a successful search in the store’s database, a customer may leave the store instead of completing the purchase, if it takes too long to receive a response from a request. In order to mimic this process and be able to evaluate changes in the user behavior caused by the proposed adaptive schemes of the site, we use a hierarchical simulation model for workload generation.

5.1. Hierarchical workload generation

The hierarchical simulation model is constructed as follows. First, the model creates service requests that initiate customer sessions. Once a session starts, the HTTP requests of the session are generated according

to the CBMG of the workload. For each HTTP request, the simulator generates specific requests for the site facilities, such as HTTP servers, CGI servers, DB servers, and LANs. Each facility simulates the execution of a request, demanding service from its main resources, such as CPU, disks, and network bandwidth.

At the higher-level, we use the Pareto distribution used by the SURGE [2] workload generator to represent request interarrival times. As in SURGE, this probability distribution is used to guarantee burstiness and realistic interarrival distribution for the service requests. At session start time though, we do not know yet the sequence of HTTP requests that will make up the session because the system is adaptive and the customer behavior depends on the system performance. Thus, the next request to be generated by the user is a function of the CBMG and the system responsiveness.

When a new session is initiated, the customer is assigned to a profile (i.e., heavy buyer or occasional buyer) represented by a CBMG. During a session, the customer goes through the states of the CBMG, and each of these states generates a single HTTP request to the store site, that processes it and sends the response. Then, based on the current state, the transition probabilities, the think times associated with each transition in the CBMG, and the response time of the last request, a new state is computed and a new request is generated for the site. The request generating process is repeated until the customer leaves the site.

There are two ways to leave the site. In the first one, a customer decides to leave the site spontaneously. This case is represented by the “exit arrow” in each state of a CBMG. A probability is assigned to the exit transition in each state. This exit probability represents all the reasons a customer quits a site (e.g., unfriendly navigation scheme, poor contents, unattractive prices) except slow response time. The second reason to leave is poor performance. If the site is slow to answer a request, the customer becomes impatient and quits the site. Industry usually assumes the “eight-second rule,” a de-facto standard for maximum download time of a Web page [7]. It is clear that customers may be willing to wait longer for some pages, such as the confirmation of a payment, than for other pages, such as the result of a search request. We model this phenomenon in our simulation by introducing a “customer impatience factor” defined by a timeout and a number of retries. The timeout is defined as $\text{timeout} = c_2(\text{state}) + c_1 \times \text{session length}$, where $c_2(\text{state})$ is a constant that depends on the state of the CBMG and c_1 is fixed at 0.1. The values of c_2 used in our simulations are $c_2(\text{b}) = 9$, $c_2(\text{s}) = 9$, $c_2(\text{t}) = 8$, $c_2(\text{a}) = 8$, and $c_2(\text{p}) = 30$. Given that the observed average session length in the actual logs we examined is around 9, we chose the values of c_1 and c_2 to reflect the case that customers at states other than the Pay state are willing to wait for periods of time ranging from 8 to 9 s. Customers in the Pay state may be willing to wait for a period time in the vicinity of 30 s. A retry means that a user presses the STOP button and the LOAD button of the browser in an attempt to get the response. Occasional buyers are assumed to retry once and heavy buyers three times. So, the CBMG combined with the impatience factor provides the model of user behavior used in the simulation.

5.2. Simulation model

To assess the impact of the resource management policies described here, we simulated an electronic bookstore. Customers are assumed to search for books by either keyword or by author. Two types of books were considered: technical and non-technical. Table 1 specifies the percent of technical books selected by customers of our online bookstore, the average price and price range for each type of book, the percentage of searches by author and keyword, and the average size of a page returned by searches by keyword and

Table 1
Workload parameters^a

Percent of technical books selected	20%
Average book prices	
Technical books	\$45.00
Non-technical books	\$18.00
Price range	
Technical books	[\$5.00, \$100.00]
Non-technical books	[\$5.00, \$60.00]
Searches	
By author	60%
By keyword	40%
Average size (in KB) of pages returned by search requests	
Technical books	
By keyword	20
By author	11
Non-technical books	
By keyword	25
By author	16
Average size (in KB) of pages returned by browse requests	20

^aThink times: 15 s for all transitions, except for: Select to Add: 45 s; Add to Pay: 25 s; Search to Select: 30 s.

author on technical and non-technical books. The distribution of book prices is a truncated Gaussian distribution. The values in Table 1 were obtained by analyzing the results of a large number of searches performed in various existing online bookstores. There are many possible values for the parameters used in the simulation studies. Due to space limitations, we are only presenting here results for a limited set of values.

The architecture of the e-commerce site for our simulated bookstore is composed of one authentication server, three HTTP servers, three application servers, three DB servers, and two 100-Mbps LANs. The authentication server is connected to LAN 1 which is connected to the Internet. A reverse proxy connects LANs 1 and 2. The HTTP, application, and DB servers are all connected to LAN 2. Each DB server has three disks with 9 ms average seek time, 4.7 ms average latency, and 30 MB/s transfer rate.

Table 2 contains additional parameters used by the simulator. These parameters along with the LAN bandwidth and disk performance characteristics were used to obtain the service demands for each type

Table 2
Request/reply size parameters

Parameter description	Value
Size of an HTTP request (includes protocol ovhd)	0.358 KB
Average round trip time (RTT)	0.125 s
Average size of a request to an application or DB server (includes ovhd)	0.258 KB
Average size of a file containing a CGI script	2 KB
Average KB read/written at the DB server per Add request	2 KB
Average size of a static page	2 KB
Average KB read/written at the DB server per Select request	3 KB
Average KB read/written at the DB server per Pay request	2 KB

Table 3
Components of the site configuration involved in the request execution

CBMG state	Site's components
Home	HTTP
Browse	HTTP, CGI, DB
Search	HTTP, CGI, DB
Select	HTTP, CGI, DB
Add to Cart	HTTP, CGI, DB
Pay	HTTP, CGI, DB, CGI, AS

of request at each state. The service demand D_{CPU} , in seconds, at the CPU of each server was computed as $D_{\text{CPU}}(\text{BD}, \text{BT}) = 0.00258(\text{BD} + \text{BT}) + 0.0027$, where BD is the number of KB read/written from disk and BT is the number of KB transmitted.

Associated with each state defined in the CBMG is a sequence of operations to be performed at the store site, i.e., the sequence of the site's components visited by a request. Ultimately, this sequence, together with the load in each of the store's components, determines the response time seen by the customer. Table 3 shows the site's components involved in the processing of the requests associated with each CBMG state. The simulation model was developed in C and used the discrete event simulation environment SMPL [6]. For all measured values, the 95% confidence intervals were within 5% of the mean values. To conclude this section, it is worth mentioning that a real log (consisting of HTTP requests) would be useless for our purpose of testing this new adaptive mechanism. The reason is that the user behavior, in terms of request generation, is a function of the CBMGs and system performance. A log of a system that does not implement the type of resource management scheme proposed here would not have the sequences that would result from the interaction of customers with a site that implements such policies.

6. Performance of new resource management policies

Many analyses were made with the use of the simulator. Due to space limitations, we only present here some of them. The metrics plotted in the figures that follow are: revenue throughput (X^+), in dollars per second, for occasional and heavy buyers, percent angry customers %A, defined as the percent of customers who leave the site due to poor performance, potential lost revenue per second (X^-) for the no priority case and for the priority scheme suggested here, and average response time for the priority and no priority cases. All graphs in this section depict the variation of a metric as a function of the session arrival rate λ_s . The value of m_2 used throughout the simulations reported here is 4. The value of m_1 varied as shown in the figures.

Figs. 4 and 5 show the variation of the revenue throughput X^+ as a function of λ_s for various values of m_1 for heavy and occasional buyers, respectively. In both figures, the no priority case is also shown. It can be observed from Fig. 4 that for heavy buyers, the revenue per second always increases as the system load increases and in virtually all cases, the revenue per second is better with the priority scheme than with no priorities, especially for higher load values. For example, for $\lambda_s = 30$ sessions/s, the value of X^+ for $m_1 = 2$ is 29% higher than for the no priority case. We also see from the figure that X^+ is very sensitive to m_1 as the load varies.

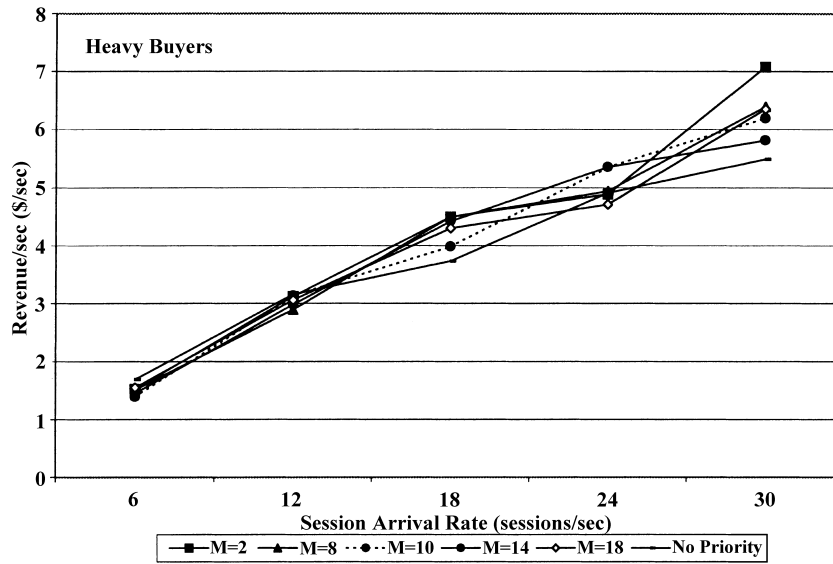


Fig. 4. Revenue per second vs. session arrival rate for heavy buyers.

For occasional buyers, as seen in Fig 5, the value of X^+ increases with λ_s up to a certain value of λ_s and then starts to decrease. The reason is that for heavy loads, occasional buyers experience much higher response times than heavy buyers and are more likely to exit the site by timeouts. As with the heavy buyers case, the value of X^+ is almost always better with priorities than without. The maximum value of X^+ for occasional buyers was observed to be \$10.59/s and it occurred for $\lambda_s = 24$ sessions/s

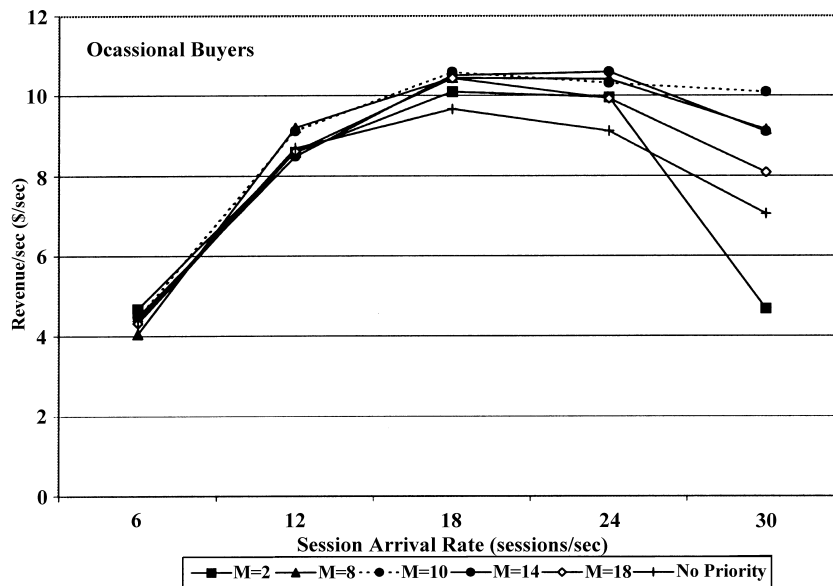


Fig. 5. Revenue per second vs. session arrival rate for occasional buyers.

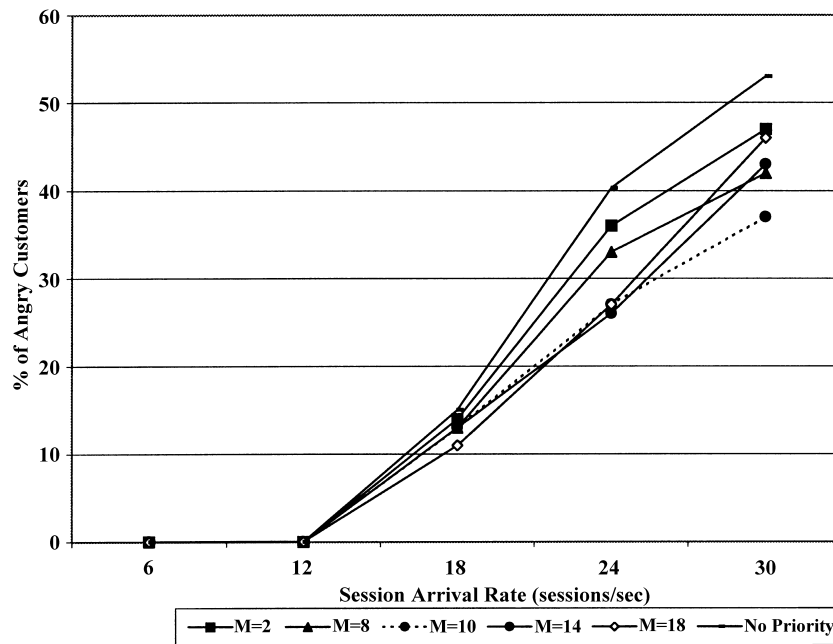


Fig. 6. Percent of angry customers vs. session arrival rate.

for $m_1 = 14$. For heavier loads, e.g., $\lambda_s = 30$ sessions/s, the maximum value of X^+ is 43% higher than that for the no priority case. This happens for $m_1 = 10$. Note that a very small value of m_1 ($m_1 = 2$) for occasional buyers, hurt them more than in the case of no priority. This is because, occasional buyers will very quickly move into medium or low priority and will very likely timeout due to high response times. To help put the values of revenue per second reported here in proper perspective, a Web store with the sustained revenue per second shown in Figs. 4 and 5 would have a yearly revenue of \$338 million assuming 16-h days at that revenue per second rate and 365 days/year, for the priority case and $m_1 = 10$ and for $\lambda_s = 30$ sessions/s. For the same load level and values of m_1 , the yearly revenue in the no priority case would be \$263 million. Thus, the priority scheme can improve the site revenue by almost 30%.

The goal of the proposed resource management policies is to improve the quality of service for some classes of customers during peak periods, i.e., when the site's resources are stressed by bursts of requests. This goal is achieved by giving priority to customers, according to the session length and amount of money accumulated in the customer's shopping cart. The simulation results provided in Figs. 4 and 5 clearly indicate the superiority of the priority policies when the system load is high. However, in certain cases, when the system is lightly loaded (i.e., for small values of λ), the no priority policy performs slightly better than the proposed policies. This behavior can be explained by the fact that some customers may quickly move into medium or low priority (specially for small m_1) and will very likely timeout, leaving their shopping cart with some selected items.

The behavior of Fig. 4 can be better understood by analyzing Fig. 6 that shows the variation of %A with the load. As it can be seen, for load intensity values up to 12 sessions/s, there are virtually no angry customers. However, as the load intensity increases, more and more customers become angry. The no-priority case displays the highest percent of angry customers and the $m_1 = 10$ case is the one with

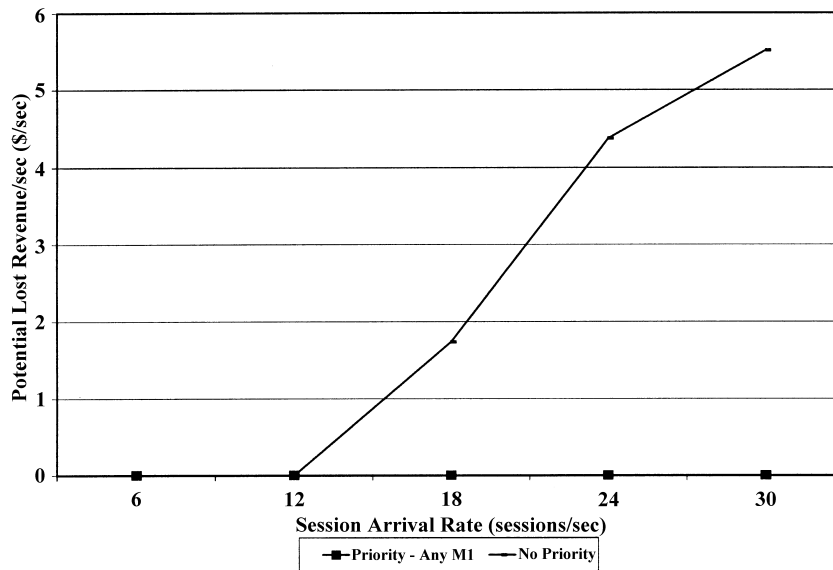


Fig. 7. Potential lost revenue per second vs. session arrival rate.

the lowest value of %A for very high loads. For $\lambda = 30$ sessions/s, 53% of the customers leave the site due to poor performance in the no-priority case while for the priority case with $m_1 = 10$ only 37% leave the site for this reason.

Fig. 7 shows the potential lost revenue per second, X^- , for all customers. This metric represents the maximum amount of dollars per second that the store would be losing due to customers that leave the site because of poor response time. As the figure indicates, with the priority scheme and all values of m_1 used, there is no potential lost revenue. However, for the no priority case, as the load increases, the potential loss in revenue increases. For $\lambda_s = 30$ sessions/s, the potential lost revenue per second is almost 50% of the combined revenue throughput brought to the Web store by heavy and occasional customers. An analysis of Figs. 6 and 7 shows that angry customers lost by the priority scheme do not have anything in their shopping carts and therefore do not contribute to X^- .

Finally, Fig. 8 shows the average response time for all types of customers as a function of λ_s for the no priority case and for the priority case and different values of m_1 . It is interesting to note that as λ_s increases, the average response time for the collection of all customers is higher for the priority case and values of $m_1 = 8, 10, 14$ and 18. Note that the improvements of the new metrics sometimes occur at the expense of the traditional performance metrics. Only the $m_1 = 2$ case presents a lower response time. The combined revenue of the site is maximized for $m_1 = 10$ for $\lambda_s = 30$ sessions/s. However, for this value of λ_s and m_1 the average response time is virtually the worst among all cases. This illustrates the fact that optimizing conventional performance metrics may be detrimental to business-oriented goals and vice-versa.

7. Related work

A few mechanisms have been developed to support quality of service in Web servers. All of them [1,3,10] aim at achieving good response time or high throughput. They are not adaptive to customer's

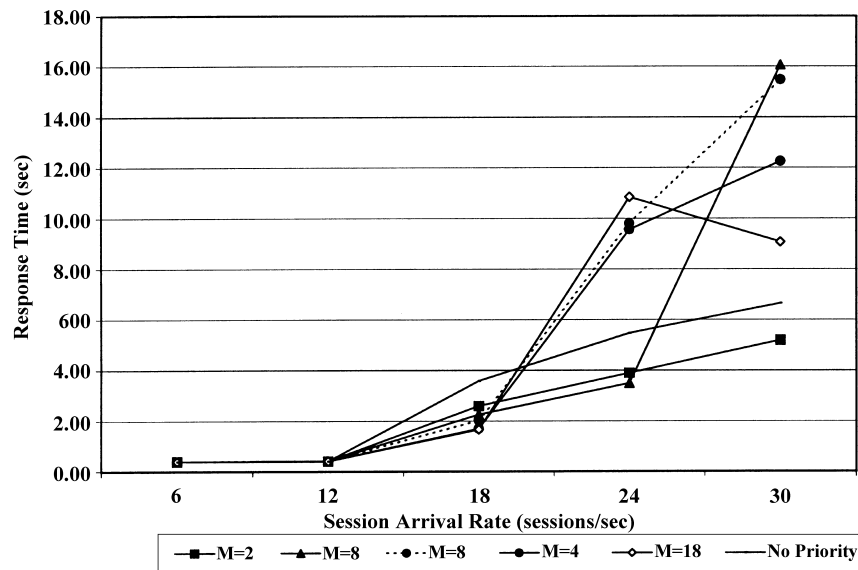


Fig. 8. Response times vs. session arrival rate.

behavior and they do not consider the new business-oriented performance metrics proposed in this paper. Ref. [1] introduces the notion of quality of service by associating priorities with requests, based on which document they are requesting, not where they come from. The metric for quality of service is latency in handling the HTTP requests. The priority mechanism is static and the study is restricted to priority schemes for CPU scheduling only. In [10], the authors develop a quality of service model that implements algorithms for scheduling CPU, memory, and networking resources. In their model, a site can determine how requests for various pages should be served. Therefore, the notion of quality of service is implemented by setting priorities among page requests and by associating constraints on resource usage.

An admission control mechanism to improve performance of overloaded Web servers is proposed and analyzed in [3]. The authors introduce the notion of session, consisting of many individual HTTP requests. The main goal of the session-based admission control mechanism is to prevent the overload of a Web server. The control scheme is based on the server CPU utilization. The analysis focuses only on the throughput gains obtained by an admission control mechanism that aims at guaranteeing the completion of any accepted session. Ref. [5] presents an overall description of a Hewlett-Packard product (WebQoS) that implements the session-based admission control scheme proposed in [3].

Our work differs from the one proposed in [3,5] in many ways. A fundamental difference is that our resource control mechanism is driven by business-oriented metrics, such as revenue throughput and potential lost revenue per second, and does not try to optimize conventional metrics such as response time, throughput, or server utilization. Our paper proposes resource management policies that aim at optimizing business-oriented metrics, and sometimes this can occur at the expense of traditional metrics, as shown in Fig. 8. In fact, the metrics proposed in this paper are an aggregation of many QoS properties, such as timeliness, availability, and security. Our metrics allow reasoning about and planning of e-commerce sites at a high level of abstraction. Another difference is the way we

characterize e-commerce workloads, as a combination of typical requests such as browse, search, select, add, and pay. A CBMG is used to represent different classes of customers and the dependencies existing among the several services of an e-commerce site. Refs. [3,5] treat an e-commerce workload as composed of sessions, that consists of many individual HTTP requests with no interdependence.

Finally, some experimental projects of operating systems have made use of economic and financial notions in the past. For example, the Spawn system [12] explores issues of fairness in resource distribution, currency as a form of priority, price equilibria, the dynamics of transients, and scaling in large distributed systems.

8. Concluding remarks

In this paper, we have introduced novel concepts for managing and evaluating e-commerce sites. We defined a set of new performance metrics to assess the performance of e-business sites in terms of business goals. Traditional quality of service models for the WWW are associated with two viewpoints: client-side performance, usually measured by response time, and server-side performance, usually represented by throughput. The metrics proposed here combine the two views. For example, *revenue throughput*, measured in dollars per second, implicitly represents customer and site behavior. If a customer is happy with the site service, he/she will shop at the Web store and the revenue throughput will increase.

In order to maximize the revenue generated by a site and support its business goals, we proposed a dynamic multilevel resource management policy. Priorities change dynamically as a function of the state a customer is in and as a function of the amount of money the customer has accumulated in his/her shopping cart. A detailed simulation model was developed to assess the gain of adaptive policies with respect to policies that are oblivious to economic considerations. Simulation results show that for an overloaded site, an increase of almost 30% in revenue could be obtained by the priority schemes of the adaptive policy, when compared to the no priority policy. We also show that the number of angry customers that leave a site due to poor performance could be decreased by 16% by the use of priority policies. The potential lost revenue could be reduced in 50% by the priority schemes. Overall, we observed a gain in the business metrics of an e-commerce site when adaptive policies for managing resources are used. The importance of the results discussed in this paper lies in the fact that e-commerce sites that use our dynamic priority scheme will be able to improve revenue at peak times with existing server capacity.

The priority techniques presented here can be adapted to handle cases in which customers can be identified by the site either through a login process or through cookies. In these situations, the site may use previously stored buying patterns to determine customer priorities. The use of a combination of current buying behavior with past buying patterns would avoid frustrating customers who have been very good buyers in the past but are mainly window shopping in the current session. It should also be noted that the policies presented here are designed to handle short term increases in traffic. In these cases, sites cannot add more resources but have to decide how best to serve a sudden load increase. One option is to refuse connections to guarantee a good quality of service to the accepted connections [3]. One problem with this approach is that we do not know ahead of time how valuable the rejected customers might be. We opted for policies that allow everybody into the site but dynamically allocate the site's resources in a manner that increases the overall revenue.

The priority scheme suggested here requires modification to Web server, operating system, and DBMS software. While this could be seen as a drawback, many e-commerce sites already rely on open source software such as Apache and Linux. This makes it easier to implement the techniques discussed here. Developers of software for e-commerce may be interested in implementing the techniques presented here. The purpose of this paper was to present the resource allocation mechanisms for e-commerce servers. In particular, we focused on priority based processor and disk scheduling techniques and showed their effectiveness. A natural next step is to implement them in a prototype of an e-commerce site. This would allow us, among other things, to evaluate the overhead of implementing these priority schemes.

References

- [1] J. Almeida, M. Dabu, A. Manikutty, P. Cao, Providing differentiated levels of service in Web content hosting, in: Proceedings of the First Workshop on Internet Server Performance, ACM SIGMETRICS'98, June 1998.
- [2] P. Barford, M. Crovella, Generating representative Web workloads for network and server performance evaluation, in: Proceedings of the 1998 ACM SIGMETRICS International Conference on Measurement and Modeling of Computing Systems, Madison, WI, June 1998.
- [3] L. Cherkasova, P. Phaal, Session based admission control: a mechanism for improving the performance of an overloaded Web server, HPL-98-119, HP Labs Technical Reports, 1998.
- [4] P.J. Denning, J. Buzen, The operational analysis of queuing network models, *Comput. Surveys* 10 (1978) 3.
- [5] HP Enterprise Computing, <http://www.useit.com/alertbox/990207.html>.
- [6] M. MacDougall, *Simulating Computer Systems: Techniques and Tools*, MIT Press, Cambridge, MA, 1987.
- [7] D.A. Menascé, V.A.F. Almeida, *Scaling for E-Business: Technologies, Models, Performance, and Capacity Planning*, Prentice-Hall, Upper Saddle River, NJ, 2000.
- [8] D.A. Menascé, V.A.F. Almeida, R. Fonseca, M.A. Mendes, A methodology for workload characterization of e-commerce sites, in: Proceedings of the 1999 ACM Conference on Electronic Commerce, Denver, CO, November 1999.
- [9] D.A. Menascé, V.A.F. Almeida, R. Fonseca, M.A. Mendes, Resource allocation policies for e-commerce servers, in: Proceedings of the Second Workshop on Internet Server Performance, ACM SIGMETRICS'99, Atlanta, GA, May 1999.
- [10] R. Pandey, J. Barnes, R. Olsson, Supporting quality of service in HTTP servers, in: Proceedings of the 17th Annual SIGACT-SIGOPS Symposium on Principles of Distributed Computing, 1998.
- [11] C. Shapiro, H. Varian, *Information Rules: A Strategic Guide to the Network Economy*, Harvard Business School Press, Cambridge, MA, 1999.
- [12] C. Waldspurger, T. Hogg, B. Huberman, J. Kephart, W. Stornetta, Spawn: a distributed computational economy, *IEEE Trans. Software Eng.* 18 (1992) 2.



Daniel A. Menascé is a Professor of Computer Science at George Mason University. He received a Ph.D. in Computer Science from the University of California at Los Angeles (UCLA) in 1978. He is a Fellow of the ACM and an elected member of IFIPs WG 7.3. He has published over 110 papers and he is the chief author of five books. He received various teaching and best paper awards. For more information visit www.cs.gmu.edu/faculty/menasce.html.



Virgilio A.F. Almeida is a Professor of Computer Science at the Federal University of Minas Gerais (UFMG), Brazil. He received a Ph.D. degree in Computer Science from Vanderbilt University in 1987. His research interests are large distributed systems and the World Wide Web. He is the co-author of three books on performance modeling and evaluation. He held Visiting Faculty and research positions at Boston University and XEROX PARC. For more information visit www.dcc.ufmg.br/~virgilio/.

Rodrigo Fonseca is a graduate student at the Computer Science Department at the Federal University of Minas Gerais, Brazil.

Marco Aurelio Mendes is a graduate student at the Computer Science Department at the Federal University of Minas Gerais, Brazil.