Modeling and Analysis of 2D Service Differentiation on e-Commerce Servers

Xiaobo Zhou
Department of Computer Science
University of Colorado at Colorado Springs, CO 80933
zbo@cs.uccs.edu

Jianbin Wei and Cheng-Zhong Xu
Department of Electrical & Computer Engg.
Wayne State University, Detroit, MI 48202
{jbwei, czxu}@ece.eng.wayne.edu

Abstract

A scalable e-Commerce server should be able to provide different levels of quality of service (QoS) to different types of requests according to clients' navigation patterns and the server capacity. In this paper, we propose a two-dimensional (2D) service differentiation (DiffServ) model for on-line transactions: inter-session *and* intra-session*. The inter-session model aims to provide different levels of QoS to sessions from different customer classes, and the intra-session model aims to provide different levels of QoS to requests in different states of a session.*

We introduce service slowdown as a QoS metric of e-Commerce servers. It is defined as the weighted sum of request slowdown in different sessions and in different session states. We formulate the problem of 2D DiffServ provisioning as an optimization of processing rate allocation with the objective of minimizing service slowdown. We derive the optimal allocations for an M/G/1 server under various server load conditions and prove that the optimal allocations guarantees requests' slowdown to be square-root proportional to their pre-specified differentiation weights in both dimensions. We evaluate the optimal allocation scheme via extensive simulations and compare it with a tailored proportional DiffServ scheme. Simulation results validate that both allocation schemes can achieve predictable, controllable, and fair 2D slowdown differentiation on e-Commerce servers. The optimal allocation scheme guarantees 2D DiffServ at a minimum cost of service slowdown.

1. Introduction

In the server side, service differentiation (DiffServ) is to treat client requests differently based on clients' needs and servers' resource limitations. Because clients are different in their visiting patterns, receiving devices, and service fees, a scalable e-Commerce server needs to provide different levels of QoS to different clients. DiffServ has been an active research topic in the arena of networking since its architecture was first formulated by IETF [4]. Its goal is to define configurable types of packet forwarding so as to provide per-hop differentiated services for large aggregate of network traffic. Network alone is not sufficient to support end-to-end DiffServ. There are recent efforts on server-side DiffServ for various Web and multimedia applications; see [1, 3, 5, 22, 21] for examples. However, few exists for DiffServ in session-based e-Commerce applications.

A session is a sequence of individual requests of different types made by a single customer during a single visit to an e-Commerce site. During a session, a customer can issue consecutive requests of various functions such as browse, search, select, add to shopping cart, register and pay. It has been observed that different customers exhibit different navigation patterns. Actually, only few customers are heavy buyers and all others are occasional buyers or visitors. Recent studies on customer behaviors of some e-Commerce sites showed that only 5% to 10% customers were interested in buying something during the current visit and about 50% of these customers were capable of completing their purchases [15, 16, 17]. Although it is important to accommodate the remaining 90% to 95% customers in order to turn them into loyal customers in future, the 5% to 10% premium customers should be preferential. This requires a scalable e-Commerce server to provide different levels of QoS to sessions from different customers. We refer to this as *inter-session differentiation*.

An e-Commerce session contains a sequence of requests for various functions in different states. Requests in different states have different opportunities to turn themselves to be profitable. E-Commerce servers should also provide different levels of QoS to requests in different states in each session so that profitable requests like order and checkout are guaranteed to be completed in a timely manner. We refer to this as *intra-session differentiation*.

In this paper, we investigate the problem of the two-dimensional (2D) DiffServ provisioning on e-Commerce servers, where the customers can be classified according to their profiles and shopping behaviors. User-perceived QoS of on-line transactions is often measured by a performance metric of response time. It refers to the duration of a request between its arrival and departure times, including waiting time in a backlogged queue and actual processing time. Note network transmission delay is beyond the scope of the paper. DiffServ provisioning with respect to response time can be achieved to some extent by conventional priority-based request scheduling. The principle of priority-based scheduling is widely used to support packet queueing-delay differentiation in networking. Most of the delay differentiation algorithms can be tailored for request response time differentiation on e-Commerce servers [6, 13].

Proceedings of the 24th International Conference on Distributed Computing Systems (ICDCS'04)
1063-6927/04 $20.00 © 2004 **IEEE**

IEEE COMPUTER SOCIETY

Response time reflects user-perceived absolute performance of a server. It is not suitable for comparing the quality of requests that have different resource demands. Customers are likely to anticipate short delays for "small" requests like browsing, and are willing to tolerate long delays for "large" requests like search. A more important performance metric is *slowdown*. A request's slowdown is defined as the ratio of its delay in a backlogged queue relative to its service time. Since slowdown translates more directly to user-perceived system load, it is more often used as a performance metric of responsiveness on Interne servers [9, 18, 22]. DiffServ with respect to slowdown is beyond the capabilities of priority based request scheduling schemes, which adjust the priority of backlogged requests according to their experienced queueing delays, without taking into account any information about their service time. A queueing discipline based on request service time violates a fundamental Little's Law on which the priority-based scheduling principle is built.

This paper proposes and formulates the 2D DiffServ model with respect to slowdown as an optimization problem of processing rate allocation for the objective of minimizing service slowdown of the e-commerce server. We assume that the e-Commerce server has a single processing resource bottleneck. Although processing a request often needs to consume resources of different types, resource management usually focuses on the allocation of the most critical resource. This single resource bottleneck assumption was made in [1, 6, 7, 9], as well. We derive an optimal processing rate allocation scheme and prove that the scheme guarantees square-root proportional DiffServ and hence it is fair. We then evaluate the allocation scheme via extensive simulations and verify the differentiation predictability, controllability, and fairness of the scheme.

In the following, Section 2 gives the 2D DiffServ model and the problem formulation. Section 3 presents the rate allocation schemes. Sections 4 and 5 present implementation issues and simulation results. Related work is reviewed in Section 6. Section 7 concludes the article.

2. 2D Service Differentiation

2.1. Modeling of 2D Service Differentiation

For DiffServ, incoming requests from different clients need to be classified into multiple classes according to their profile, device, payment, etc. Basically, there are two types of DiffServ schemes [4]. One is absolute DiffServ, in which each request class receives an absolute share of resource usages. The other is relative DiffServ, in which a class with a higher desired QoS level (referred to as a higher class) will receive better (at least no worse) service quality than a lower class. Although absolute DiffServ is desired to hard real-time applications like audio/video streaming services, relative DiffServ is sufficient for soft real-time applications like e-Commerce transactions.

In order for a relative DiffServ scheme to be effective, the scheme must satisfy two basic properties: *predictability* and *controllability*. Predictability requires that higher classes receive better or no worse service quality than lower classes, independent of the class load distributions. Controllability requires that the scheduler contain a number of controllable parameters that are adjustable for the control of quality spacings between classes. An additional requirement on e-Commerce servers is *fairness*. That is, requests from lower classes should not be over-compromised for requests from higher classes. It is important to provide preferential treatments to sessions from premium customers and to requests that are likely to end with a purchase. However, e-Commerce servers should also handle other non-buying sessions that account for about 90% to 95% of visits if one wants to turn them into loyal customers [15, 16, 17].

Because different customers have different navigation patterns, the 2D DiffServ model classifies the customers into m classes according to statistics of their shopping behaviors, such as buy to visit ratio. Customers in the same class have similar navigation patterns. The 2D DiffServ model assumes that each session of customer class i ($1 \leq i \leq m$) has n states, each corresponding to a request function. A customer's request is classified as being in different states according to the type of function requested.

We assume that session arrivals from each customer class meet a Poisson process. Note that requests in each state from sessions of different customers are independent because the session head requests are independent. However, a customer may visit a state many times in a session. For example, a customer can submit a search request at time t_1, select a commodity at time t_2 (after some think time), and submit another search request at time t_3. Evidently, these requests at the search state are dependent and their dependency degree is determined by the navigation pattern of that customer. We notice that an e-Commerce server can accommodate many concurrent sessions from independent customers and that the number of re-visits in a session is limited (on average, the maximum number of visits at a state in a session is 2.71 and 6.76 for a heavy buyer class and for an occasional buyer class, respectively according to [15, 16]). That is, all the requests at the same state are weakly dependent. Therefore, we assume that request arrivals in each state from sessions of a customer class still meet a Poisson process. We model the server as a M/G/1 queue. The requests are scheduled in a processor-sharing manner by storing them into $m \times n$ queues, each associated with a state.

Assume requests in Poisson process arrive at a rate λ. Denote μ the request processing rate. It follows that the traffic intensity $\rho = \lambda/\mu$. Let S be a request's slowdown. According to queueing theories [12], when $\rho < 1$ ($\lambda < \mu$), we have the expected slowdown as

$$S = \frac{\rho}{1-\rho}. \tag{1}$$

2.2. Formulation of Processing Rate Allocation

The basic idea of the processing rate allocation for provisioning 2D DiffServ on an e-Commerce server is to divide the scheduling process into a sequence of short intervals. In each interval, based on the measured resource uti-

IEEE COMPUTER SOCIETY

lization and the predicted workload, the available processing resource usages are allocated to requests in different states from different sessions.

Let C be the total amount of the processing resource available during the current allocation interval. The server's scheduler has to determine the amount of the resource usages allocated to requests in each queue so that 2D DiffServ is achieved and resource utilization is maximized.

A session in different states usually demands different processing resource usages [2, 15, 16, 19]. Let r_j be the average resource demand of a session in state j. Let $c_{i,j}$ be the amount of the resource allocated to requests from sessions of class i in state j in the current allocation interval. Thus, $c_{i,j}/r_j$ is the processing rate of requests in state j from sessions of class i. Let $v_{i,j}$ denote the average number of visits to state j in a session from class i. Let $d_{i,j}$ denote the resource demand of a session from class i in state j. That is, $d_{i,j} = v_{i,j}r_j$. According to (1), the slowdown of a request from a session of class i in state j, $s_{i,j}$, is calculated as:

$$s_{i,j} = \frac{\lambda_i v_{i,j}}{c_{i,j}/r_j - \lambda_i v_{i,j}} = \frac{\lambda_i d_{i,j}}{c_{i,j} - \lambda_i d_{i,j}}, \tag{2}$$

where λ_i is the session arrival rate of class i.

We consider the processing rate allocation for 2D DiffServ when the following constraint holds in each resource allocation interval:

$$\sum_{i=1}^{m}\sum_{j=1}^{n} \lambda_i d_{i,j} < C. \tag{3}$$

That is, the request processing rate of the server is higher than the request arrival rate. Otherwise, a request's slowdown can be infinite. DiffServ would be infeasible.

Let α_i be the normalized quality differentiation weight of sessions from class i. That is, $\alpha_i > 0$ and $\sum_{i=1}^{m} \alpha_i = 1$. Because sessions from class i should receive better or no worse service quality than sessions from class $i+1$ according to inter-session DiffServ, without loss of generality, we assume $\alpha_1 \geq \alpha_2 \geq \cdots \geq \alpha_m$. The values of α_i can be determined according to the shopping behaviors of class i, such as their buy to visit ratio [15, 16].

Let β_j be the normalized quality differentiation weight of state j in a session. That is, $\beta_j > 0$ and $\sum_{j=1}^{n} \beta_j = 1$. The values of β_j can be determined according to the characterization of transition probability from state j to state pay in sessions from all classes. Without loss of generality, we assume $\beta_1 \geq \beta_2 \geq \cdots \geq \beta_n$. Based on the concept of slowdown for individual requests, we define a metric of *session slowdown* as $\sum_{j=1}^{n} \beta_j s_{i,j}$ to reflect the weighted slowdown of requests in a session from class i. We further define a metric of *service slowdown* as $\sum_{i=1}^{m}\sum_{j=1}^{n} \alpha_i\beta_j s_{i,j}$ to reflect weighted session slowdown of sessions from all classes.

We formulate the processing rate allocation for the 2D DiffServ as the following optimization problem:

$$\text{Minimize} \quad \sum_{i=1}^{m}\sum_{j=1}^{n} \alpha_i\beta_j s_{i,j} \tag{4}$$

$$\text{Subject to} \sum_{i=1}^{m}\sum_{j=1}^{n} c_{i,j} \leq C \tag{5}$$

$$s_{i,j} = \frac{\lambda_i d_{i,j}}{c_{i,j} - \lambda_i d_{i,j}} > 0. \tag{6}$$

The objective function (4) is to minimize the service slowdown of the server. It implies that sessions from higher classes get lower slowdown (higher QoS) and hence inter-session differentiation is achieved. It also implies that sessions in high states get lower slowdown and hence intra-session differentiation is achieved. The rationale behind the objective function is its feasibility, differentiation predictability, controllability and fairness, as we discussed in Section 2.1. (5) gives a resource allocation constraint over variables $c_{i,j}$. (6) ensures the positivity of slowdown.

3. Processing Rate Allocation Schemes

3.1. An Optimal Allocation Scheme

The above optimization problem is essentially a continuous convex separable resource allocation problem. According to theories of general resource allocation problems [10], its optimal solution occurs when the first order derivatives of the objective function (4) over variables $c_{i,j}, 1 \leq i \leq m$ and $1 \leq j \leq n$ are equivalent. Specifically, the optimal solution to (4) occurs when

$$-\frac{\alpha_{i_1}\beta_{j_1}\lambda_{i_1}d_{i_1,j_1}}{(c_{i_1,j_1} - \lambda_{i_1}d_{i_1,j_1})^2} - \frac{\alpha_{i_2}\beta_{j_2}\lambda_{i_2}d_{i_2,j_2}}{(c_{i_2,j_2} - \lambda_{i_2}d_{i_2,j_2})^2} \tag{7}$$

for $1 \leq i_1, i_2 \leq m$ and $1 \leq j_1, j_2 \leq n$.

It follows that

$$\frac{c_{i,j} - \lambda_i d_{i,j}}{c_{1,1} - \lambda_1 d_{1,1}} = \sqrt{\frac{\alpha_i\beta_j\lambda_i d_{i,j}}{\alpha_1\beta_1\lambda_1 d_{1,1}}} \tag{8}$$

for $1 \leq i \leq m$ and $1 \leq j \leq n$.

Let $\tilde{\lambda}_{i,j} = \alpha_i\beta_j\lambda_i d_{i,j}$. Together with the constraint (5), the set of equations (8) leads to the optimal allocation as

$$c_{i,j} = \lambda_i d_{i,j} + \frac{\tilde{\lambda}_{i,j}^{1/2}}{\sum_{i=1}^{m}\sum_{j=1}^{n} \tilde{\lambda}_{i,j}^{1/2}}(C - \sum_{i=1}^{m}\sum_{j=1}^{n} \lambda_i d_{i,j}). \tag{9}$$

Accordingly, the slowdown of a request is

$$s_{i,j} = \frac{\lambda_i d_{i,j} \sum_{i=1}^{m}\sum_{j=1}^{n} \tilde{\lambda}_{i,j}^{1/2}}{\tilde{\lambda}_{i,j}^{1/2}(C - \sum_{i=1}^{m}\sum_{j=1}^{n} \lambda_i d_{i,j})}. \tag{10}$$

¿From (10), we have the following three basic properties regarding the differentiation predictability and controllability given by the optimal processing rate allocation scheme:

1. If the session weight or the state weight of a request class increases, slowdown of all other classes increases, while slowdown of that class decreases.

IEEE COMPUTER SOCIETY

2. Slowdown of a request class increases with the increase of session arrival rate and the number of visits to that state of each request class.
3. Increasing the workload (session arrival rate or the number of visits to a state in a session) of a higher class causes a larger increase in slowdown of a class than increasing the workload of a lower class.

Recall $d_{i,j} = v_{i,j} r_j$. From (10), we further have the following DiffServ ratios:

$$\frac{s_{i_2,j}}{s_{i_1,j}} = \sqrt{\frac{\lambda_{i_2} v_{i_2,j}}{\lambda_{i_1} v_{i_1,j}}} \sqrt{\frac{\alpha_{i_1}}{\alpha_{i_2}}} \quad \text{for } j = 1, 2, \cdots, n \quad (11)$$

$$\frac{s_{i,j_2}}{s_{i,j_1}} = \sqrt{\frac{d_{i,j_2}}{d_{i,j_1}}} \sqrt{\frac{\beta_{j_1}}{\beta_{j_2}}} \quad \text{for } i = 1, 2, \cdots, m \quad (12)$$

$$\frac{s_{i_2,j_2}}{s_{i_1,j_1}} = \sqrt{\frac{\lambda_{i_2} d_{i_2,j_2}}{\lambda^{i_1} d_{i_1,j_1}}} \sqrt{\frac{\alpha_{i_1} \beta_{j_1}}{\alpha_{i_2} \beta_{j_2}}}. \quad (13)$$

¿From (11), (12), and (13), we can see that the optimal processing rate allocation has the property of fairness, as well. That is,

Theorem 3.1 The optimal allocation (9) guarantees relative service differentiation between the requests in both inter-session and intra-session dimensions and their quality spacings with respect to slowdown are square-root proportional to their per-defined differentiation weights.

Remark 1. If session arrival rate λ_i and the resource requirement of a session from a customer class in a state ($d_{i,j}$) are fixed, a request class (i, j) with a higher session weight α_i or with a higher state weight β_j gets more portion of available processing rate of the e-Commerce server. However, we note that the predictability of inter-session DiffServ holds iff $\sqrt{\frac{\lambda_{i_1} v_{i_1,j}}{\lambda_{i_2} v_{i_2,j}}} \leq \sqrt{\frac{\alpha_{i_1}}{\alpha_{i_2}}}$ for all $j = 1, 2, \cdots, n$. Also, the predictability of intra-session service differentiation holds iff $\sqrt{\frac{d_{i,j_1}}{d_{i,j_2}}} \leq \sqrt{\frac{\beta_{j_1}}{\beta_{j_2}}}$ for all $i = 1, 2, \cdots, m$. Otherwise, the essential requirement of 2D DiffServ, predictability, will be violated. For differentiation predictability, one solution is temporary weight promotion, as suggested in [22]. When it is applied in this context, based on the current session arrival rates and the number of visits to a state in sessions, the scheduler temporarily increases session weights α_i and state weights β_j in the current rate allocation interval so that the predictability of 2D DiffServ holds. In this case, the allocation scheme is heuristic.

Remark 2. We consider the problem of processing rate allocation for 2D DiffServ when constraint (3) holds. Otherwise, a request's slowdown can be infinite and provisioning slowdown differentiation would be infeasible. Session-based admission control mechanisms can be applied to drop sessions from low classes so that constraint (3) holds.

3.2. A Proportional Slowdown Allocation Scheme

To provide more insights into the impact of various processing rate allocation schemes on 2D DiffServ on e-Commerce servers, we present a proportional share allocation scheme that is tailored from proportional delay differentiation in network packet routing [8, 14].

A proportional resource allocation scheme assigns quality factors to request classes in proportion to their quality differentiation weights. In the 2D DiffServ model on e-Commerce servers, the quality factors of request classes are represented by their slowdown $s_{i,j}$. Consequently, the proportional model imposes the following constraints for inter-session DiffServ and intra-session DiffServ, respectively:

$$\frac{s_{i_2,j}}{s_{i_1,j}} = \frac{\alpha_{i_1}}{\alpha_{i_2}}, \quad \text{for all } j = 1, 2, \cdots, n \quad (14)$$

$$\frac{s_{i,j_2}}{s_{i,j_1}} = \frac{\beta_{j_1}}{\beta_{j_2}}, \quad \text{for all } i = 1, 2, \cdots, m \quad (15)$$

where α_i and β_i are the normalized quality weighting factors as defined in 2.2.

Based on the above analysis for the optimization-based processing-rate allocation scheme, we consider the proportional rate allocation problem when constraint (3) holds. Recall the definition of slowdown in (2). According to (14) and (15), we derive the following equation system:

$$\frac{s_{i_2,j_2}}{s_{i_1,j_1}} \frac{\alpha_{i_1} \beta_{j_1}}{\alpha_{i_2} \beta_{j_2}} \quad (16)$$

for $1 \leq i_1, i_2 \leq m$ and $1 \leq j_1, j_2 \leq n$.

Together with the constraint (5), the set of equations (16) leads to a processing rate allocation as

$$c_{i,j} = \lambda_i d_{i,j} + \frac{\tilde{\lambda}_{i,j}}{\sum_{i=1}^{m} \sum_{j=1}^{n} \tilde{\lambda}_{i,j}} (C - \sum_{i=1}^{m} \sum_{j=1}^{n} \lambda_i d_{i,j}). \quad (17)$$

Accordingly, the slowdown of a request is

$$s_{i,j} = \frac{\sum_{i=1}^{m} \sum_{j=1}^{n} \tilde{\lambda}_{i,j}}{\alpha_i \beta_j (C - \sum_{i=1}^{m} \sum_{j=1}^{n} \lambda_i d_{i,j})}. \quad (18)$$

According to (14) and (15), the proportional allocation scheme generates consistent and predictable schedules for 2D DiffServ on e-Commerce servers.

4. Implementation Issues

To evaluate the proposed processing rate allocation schemes on 2D DiffServ provisioning, we built a simulation model for e-Commerce servers. We used a synthetic workload generator derived from the real traces [6, 15, 16]. It allowed us to perform sensitivity analysis in a flexible way. Figure 1 outlines the basic architecture of the simulation model. It consists of a customer generator, a session generator, a request generator, a session/request rate estimator, a listen queue, and an e-Commerce server.

Based on the customer classification (*e.g.*, heavy buyer or occasional buyer), the customer generator assigns session arrival rate for each customer class (λ_i). The session generator then produces head requests that initiate sessions for the

IEEE COMPUTER SOCIETY

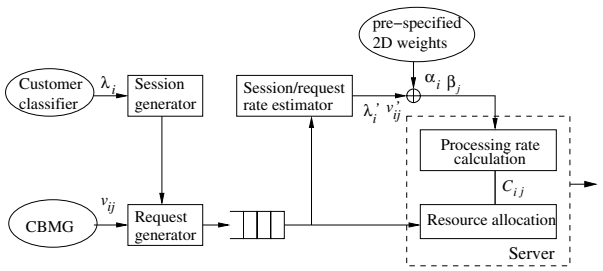


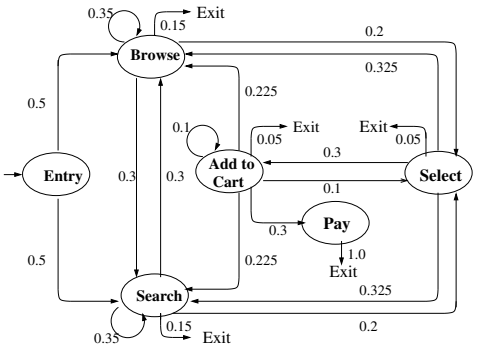
Figure 1: The arch. of the simulation model.

class. The session generation follows a Poisson process. The subsequent requests of a session are generated by the request generator according to its Customer Behavior Model Graph (CBMG). That is, based on the current state, transition probability and think time associated with each state, requests are generated for the session. Figure 2 shows two CBMGs for a heavy buyer class and an occasional buyer class, respectively. The CBMGs were derived from an on-line shopping store trace and given in [15, 16]. We implemented both profiles in our simulations.

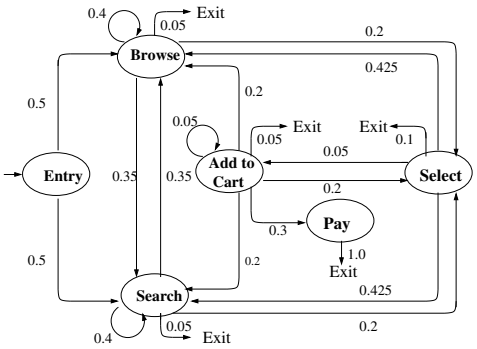
Each session request is sent to the e-Commerce server and stored in a listen queue. The listen queue is limited to 1,024 entries, which is a typical default value [7]. If the listen queue is full, a new request to the server is rejected and both the request and the whole session is aborted. In the simulations, we simulated a file mix as defined by TPC-W [19], a benchmark of e-Commerce workloads. The average resource demand of sessions in each state is assumed to be the same. It is exponentially distributed with a mean. The e-Commerce server's capacity is 1,000 requests per second for a TPC-W-like file mix and the service time for a request is proportional to the requested file size.

We divided the scheduling process into a sequence of short intervals of processing rate calculation and resource allocation. The calculation of processing rate for each class was based on the measured session arrival rate of each class, the number of visits to a state in a session from each class, the average resource demand of a request in a state in a single session, as well as the pre-specified 2D differentiation weights α_i and β_j. A fairly accurate estimation of these parameters is required so that the proposed rate allocation schemes can adapt to the dynamically changing workloads. We utilized history information to estimate these values in the session/request rate estimator. The estimate of session arrival rate of each customer class (λ_i') was obtained by counting the number of new sessions from each class occurring in a moving window of the past allocation intervals. As a typical way of calculating request arrival rate based on history information, the moving window is widely used in many similar experiments [7]. Smoothing techniques were applied to take weighted averages over past estimates. Similarly, the number of visits to a state in a session from each customer class ($v_{i,j}'$) was estimated.

The server maintained $m \times n$ listen queues. Given the processing rate for each class $c_{i,j}$ according to the results of (9) and (17), a generalized proportional-share scheduling algorithm (GPS) [11] was simulated to allocate CPU resource



(a) Heavy buyers.



(b) Occasional buyers.

Figure 2: CBMGs for different customer classes.

between $m \times n$ threads. Each thread processed requests from a class stored in the corresponding queue.

5. Performance Evaluation

We considered two customer classes: heavy buyer (class A) and occasional buyer (class B). Their CBMGs are shown in Figure 2 [15, 16]. Because the buy to visit ratio of two customer classes is 0.11 and 0.04, respectively, we assigned session differentiation weight as 11:4 to class A and B. We defined 6 states for each session: pay, add to shopping cart, select, search, browse and entry. They were sorted in a non-increasing order according to their number of state transitions needed to end in the pay state. We assigned state differentiation weight as 5:4:3:2:2:1 to the states correspondingly. The average number of visits to each state in a session is derived from the CBMGs. It is 0.11, 0.37, 1.12, 2.71, 2.71, and 1 for customer class A, and 0.04, 0.14, 2.73, 6.76, 6.76, and 1 for customer class B, respectively. The ratio of session arrival rate of class A and B was set to 1:9, according to [15, 16]. Each result is an average of 200 runs.

5.1. Impact of DiffServ on Service Slowdown

Figure 3 shows the results of service slowdown with the increase of server load. The results were obtained by the use of the optimal and the proportional allocation schemes. For comparison, the figure also includes the results without DiffServ. When the server load is below 10%, slowdown of all request classes is very small. When the server load is above

IEEE COMPUTER SOCIETY

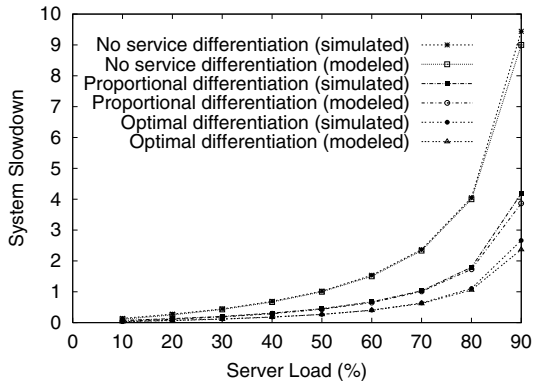


Figure 3: Service slowdown with the increase of server load.

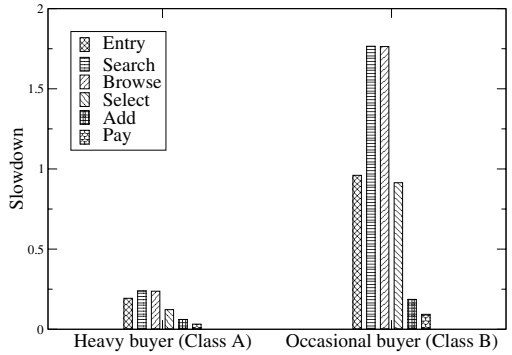
90%, slowdown of some request classes is very large. Actually, due to the limitation of listen queue size, some requests were rejected and their sessions were aborted. Session-based admission control mechanisms are required when the server is heavily loaded. Since our focus was on provisioning 2D DiffServ by the use of proposed processing rate allocation schemes, we varied the server load from 10% to 90%.

First, we found that simulation results agree with the expected results before the server is heavily loaded ($\leq 70\%$). The agreement verifies the assumption made in the modeling (Section 2.1) that request arrivals in each state from sessions of a class can be seen to be a Poisson process if session arrivals of that class meet a Poisson process. The gap between the simulated results and the expected ones increases as the server load increases. This is due to the variance of arrival distributions. Second, as we expected, the optimal allocation scheme minimizes service slowdown. The proportional allocation scheme achieves higher service slowdown. Obviously, an e-Commerce server without service differentiation provisioning receives much higher service slowdown. In the following, we give more sensitivity analysis of the optimal allocation scheme. Readers are referred to [20] for detailed analysis of the proportional allocation scheme.

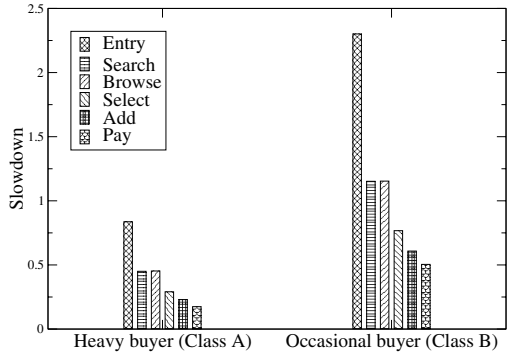
5.2. Impact of DiffServ on Request Slowdown

Figure 4 shows slowdown of individual requests in different states due to inter-session and intra-session DiffServ when the server has 50% (medium) load. It can be seen that the requests in the browse and search states almost have the same slowdown. This is because the customers have similar behaviors in these two states in terms of their state transition probabilities and resource demands. They were assigned the same state weight. In the following, we will use the search state to represent both and use the results of the search state to address inter-session DiffServ.

Both Figures 4(a) and 4(b) show that the objective of inter-session DiffServ is achieved. In each state, sessions from class A always have lower slowdown than those of class B. From Figure 4(a), it can be seen that the requirement of intra-session DiffServ predictability between the entry state and the search state is violated in both class A and B categories. Sessions in the entry state should have higher



(a) Optimal allocation scheme.



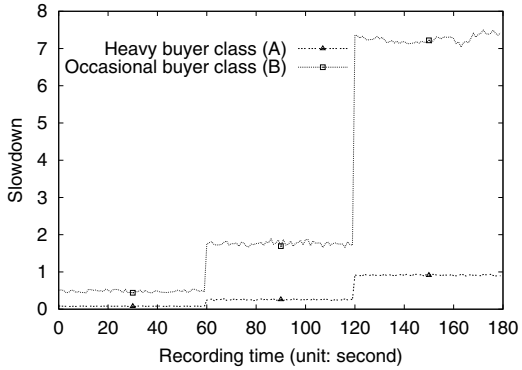
(b) Proportional allocation scheme.

Figure 4: Inter-session and intra-session DiffServ when the server has 50% load.

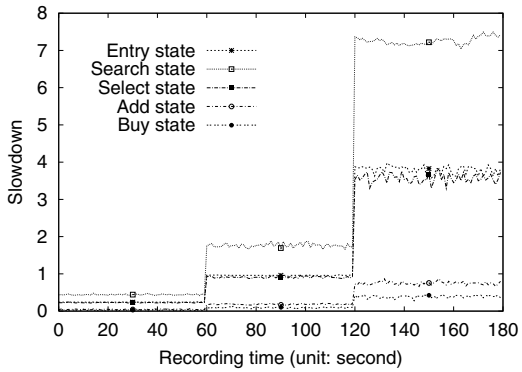
slowdown than sessions in the search state. Although the optimal allocation scheme minimizes service slowdown, the requirement of $\sqrt{\frac{d_{i,j_1}}{d_{i,j_2}}} \leq \sqrt{\frac{\beta_{j_1}}{\beta_{j_2}}}$ can be violated between the corresponding states, as we discussed in Section 3.1. This violation scenario provides an intuition into the fact that the DiffServ predictability of the optimal allocation scheme depends on class load distributions. It demonstrates that to provide predictable DiffServ, a scheduler must be able to control the settings of some parameters (*e.g.*, promoting differentiation weights). In contrast, there are no such violations in Figure 4(b). This is because the proportional allocation scheme can guarantee DiffServ predictability. However, this is achieved at the cost of much higher service slowdown, as shown in Figure 3.

Figure 5 shows a microscopic view of slowdown of individual requests due to the optimal allocation scheme when the server has 20% (light), 50% (medium) and 80% (heavy) load. At each load, the results were recorded for 60 seconds and shown in the figures from left to right. Each point represents slowdown of a request class in consecutive recording time units (seconds). Figure 5(a) illustrates inter-session DiffServ. All sessions are at search state. The plots from other states have the similar shapes. They show that the objective of inter-session DiffServ is achieved consistently in the short run. Figure 5(b) illustrates intra-session DiffServ over sessions of class B. The results of class A have simi-

IEEE COMPUTER SOCIETY

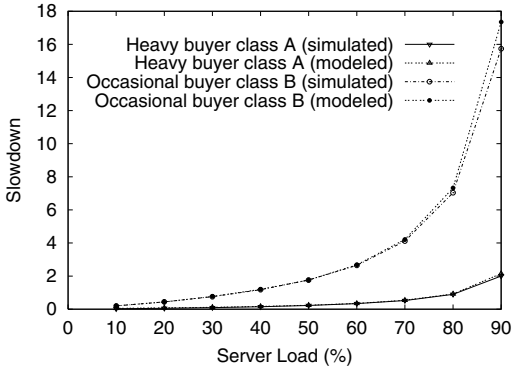


(a) Inter-session DiffServ (search state).

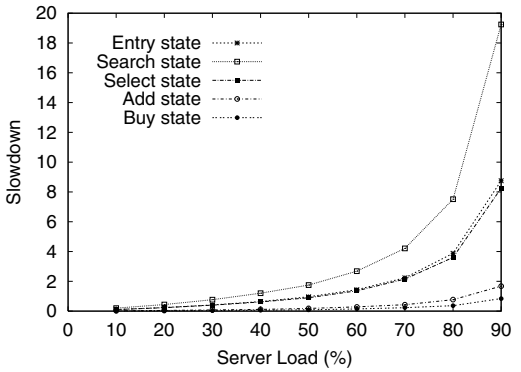


(b) Intra-session DiffServ (class B).

Figure 5: A microscopic view of slowdown of individual requests due to the optimal allocation scheme.



(a) Inter-session DiffServ (search state).



(b) Intra-session DiffServ (class B).

Figure 6: A long-term view of slowdown of request classes due to the optimal allocation scheme.

lar patterns. In the following, we use results of class B to address intra-session DiffServ.

Figure 6 shows average slowdown of request classes with the increase of server load from 10% to 90%. The simulated results meet the expectations according to (10) before the server is heavily loaded ($\leq$ 70%) in inter-session DiffServ. The gap between the simulated results and the expected ones increases as the load increases because the slowdown variance increases. The gap in intra-session DiffServ scenarios has similar shapes. Due to the space limitation, we omit the details in Figure 6(b). ¿From the figures, we can also see that the optimal allocation scheme can consistently achieve 2D DiffServ at various workloads in the long run.

6. Related Work

E-Commerce applications are session-based. Current research on scalable e-Commerce servers is mainly on workload characterization [2] and session-based admission control [6, 7]. In [7], the authors proposed a group of session-based admission control strategies to prevent a commercial Web server from becoming overloaded. The proposed mechanisms treat sessions from different clients equally and provide a fair guarantee of completion for any accepted session, independent of session length. The performance analysis focused on the throughput gains in terms of completed sessions instead of completed requests when servers were overloaded. In [6], the authors proposed a dynamic weighted fair scheduling algorithm to control overload in e-Commerce servers. It avoids processing of requests that belong to sessions that are likely to be aborted in the near future. In contrast, the focus of our work is to provision 2D DiffServ when resource demands of workloads are within resource capacity of the server. This is complementary to the previous work on session-based workload characterization and admission control for overload protection.

The DiffServ provisioning problem was firstly addressed at the network side. Most previous efforts focused on queueing-delay differentiation in packet level; see [8, 14] for examples. At the server side, a primary focus has been on admission control and priority-based request scheduling for responsive time differentiation [1, 3]. For example, in [1], the authors addressed strict priority scheduling strategies for controlling CPU utilization in Web content hosting servers. QoS was introduced by assigning strict priorities to requests for different contents. The results showed that DiffServ can be achieved but the quality spacings among different classes cannot be guaranteed by this kind of strict priority scheduling. Time-dependent priority scheduling has been used in achieving proportional queueing-delay DiffServ in network routers. It adjusts

IEEE COMPUTER SOCIETY

the priority of a backlogged class according to experienced delays of backlogged packets. Two representative algorithms are WTP [8] and adaptive WTP [14]. This kind of algorithms can be tailored in achieving queueing-delay differentiation at the server side [6, 13].

In comparison with response time, slowdown is a more accurate performance metric because it is desirable that a request's delay be proportional to its processing requirement. In [9], Harchol-Balter evaluated on-line job assignment strategies in a distributed server system, where the workload was heavy-tailed and job size was unknown to the scheduler. The primary objective was to minimize mean slowdown of the independent jobs in the distributed system. In contrast, the objective of this paper is to minimize service slowdown, a weighted sum of slowdown of the requests in different sessions and different session states. In [22], the Zhu, *et al.* used stretch factor, a variant of slowdown, as the performance metric for DiffServ in a cluster of Internet servers. Their node partitioning approach was targeted at stateless Web workload, instead of session-based transactions. The processing rate allocation strategy presented in this paper not only provides 2D DiffServ, but also lends itself to be realizable in various server environments.

The knowledge about customers' navigation patterns is important to DiffServ provisioning on e-Commerce servers. In [15, 16], the authors proposed CBMG to describe customers' navigation patterns in an e-Commerce site. Based on CBMGs, they presented a family of priority-based resource management policies for e-Commerce servers [16]. Priorities of sessions changed dynamically as a function of state a customer was in and as a function of the amount of money the shopping cart had accumulated. Resource management strategies were geared toward maximize a global system utility function. They cannot control quality spacings among different requests.

7. Conclusions

In this paper, we proposed a 2D DiffServ model with respect to slowdown for session-based e-Commerce applications. We formulated the model as an optimization problem of the processing rate allocation with the objective of minimizing service slowdown. We derived the optimal rate allocation scheme and showed that it can guarantee square-root proportional slowdown differentiation between the requests in both inter-session and intra-session dimensions. Simulation results shown that the scheme can achieve the objective consistently in both short and long timescales.

Acknowledgement This research was supported in part by NSF grants CCR-9988266 and ACI-0203592.

References

[1] J. Almeida, M. Dabu, A. Manikutty, and P. Cao. Providing differentiated levels of services in Web content hosting. In *Proc. ACM SIGMETRICS Workshop on Internet Server Performance*, pages 91–102, 1998.

[2] M. Arlitt, D. Krishnamurthy, and J. Rolia. Characterizing the scalability of a large Web-based shopping system. *ACM Trans. on Internet Technology*, 1(1):44–69, 2001.

[3] N. Bhatti and R. Friedrich. Web server support for tiered services. *IEEE Network*, 13(5):64–71, 1999.

[4] S. Blake, D. Black, M. Carlson, E. Davies, Wang Z., and W. Weiss. An architecture for differentiated services. *IETF RFC 2475*, 1998.

[5] S. Chandra, C. S. Ellis, and A. Vahdat. Application-level differentiated multimedia Web services using quality aware transcoding. *IEEE J. on Selected Areas in Communications*, 18(12):2544–2265, 2000.

[6] H. Chen and P. Mohapatra. Session-based overload control in QoS-ware Web servers. In *Proc. IEEE INFOCOM*, 2002.

[7] L. Cherkasova and P. Phaal. Session-based admission control: A mechanism for peak load management of commercial web sites. *IEEE Trans. on Computers*, 51(6):669–685, 2002.

[8] C. Dovrolis, D. Stiliadis, and P. Ramanathan. Proportional differentiated services: Delay differentiation and packet scheduling. *IEEE/ACM Trans. on Networking*, 10(1):12–26, 2002.

[9] M. Harchol-Balter. Task assignment with unknown duration. *Journal of ACM*, 29(2):260–288, 2002.

[10] T. Ibarkai and N. Katoh. *Resource allocation problem - Algorithmic approaches*. The MIT Press, 1988.

[11] J. Kay and P. Lauder. A fair share scheduler. *Communication of ACM*, 31(1):44–55, 1988.

[12] L. Kleinrock. *Queueing Systems, Volume II*. John Wiley and Sons, 1976.

[13] S. Lee, J. Lui, and D. Yau. Admission control and dynamic adaptation for a proportional-delay DiffServ-enabled Web server. In *Proc. ACM SIGMETRICS*, 2002.

[14] M. Leung, J. Lui, and D. Yau. Adaptive proportional delay differentiated services: Characterization and performance evaluation. *IEEE/ACM Trans. on Networking*, 9(6):801–817, 2001.

[15] D. A. Menascé, V. A. F. Almeida, R. Fonseca, and M. A. Mendes. A methodology for workload characterization of E-commerce sites. In *Proc. 1st ACM Conf. on Electronic Commerce*, 1999.

[16] D. A. Menascé, V. A. F. Almeida, R. Fonseca, and M. A. Mendes. Resource management policies for E-commerce servers. In *Proc. ACM SIGMETRICS Workshop on Internet Server Performance*, 1999.

[17] J. Nielsen. Why people shop on the Web. http://www.useit.com/alertbox/990207.html (Date of access: July 25, 2003).

[18] A. Riska, E. Smirni, and G. Ciardo. ADAPTLOAD: effective balancing in clustered Web servers under transient load conditions. In *Proc. IEEE Int'l Conf. on Distributed Computing Systems (ICDCS)*, 2002.

[19] W. D. Smith. TPC-W: Benchmarking an Ecommerce solution. http://www.tpc.org/tpcw (Date of access: Nov 28, 2002).

[20] X. Zhou, J. Wei, and C. Xu. Modeling and analysis of 2D service differentiaiton on e-Commerce servers. *Technical Report, CIC-03-06, Department of Electrical and Computer Engineering, Wayne State University*, 2003.

[21] X. Zhou, J. Wei, and C. Xu. Processing rate allocation for proportional slowdown differentiation on Internet servers. In *Proc. IEEE IPDPS*, 2004.

[22] H. Zhu, H. Tang, and T. Yang. Demand-driven service differentiation for cluster-based network servers. In *Proc. IEEE INFOCOM*, pages 679–688, 2001.

IEEE COMPUTER SOCIETY