A Methodology for Workload Characterization of E-commerce Sites

Daniel A. Menascé
Dept. of Computer Science
George Mason University
Fairfax, VA 22030
USA
menasce@cs.gmu.edu

Virgilio A. F. Almeida
Dept. of Computer Science
Univ. Federal de Minas Gerais
Belo Horizonte, MG 30161
Brazil
virgilio@dcc.ufmg.br

Rodrigo Fonseca
Dept. of Computer Science
Univ. Federal de Minas Gerais
Belo Horizonte, MG 30161
Brazil
rfonseca@dcc.ufmg.br

Marco A. Mendes
Dept. of Computer Science
Univ. Federal de Minas Gerais
Belo Horizonte, MG 30161
Brazil
corelio@dcc.ufmg.br

Abstract

Performance analysis and capacity planning for e-commerce sites poses an interesting problem: how to best characterize the workload of these sites. Tradition al workload characterization methods, based on hits/sec, page views/sec, or visits/sec, are not appropriate for e-commerce sites. In these environments, customers interact with the site through a series of consecutive and related requests, called sessions. Different navigational patterns can be observed for different groups of customers. In this paper, we propose a methodology for characterizing and generating e-commerce workload models. First, we introduce a state transition graph called Customer Behavior Model Graph (CBMG), that is used to describe the behavior of groups of customers who exhibit similar navigational patterns. A set of useful metrics, analytically derived from the analysis of the CBMG, is presented. Next, we define a workload model and show the steps required to obtain its parameters. We then propose a clustering algorithm to characterize workloads of e-commerce sites in terms of CBMGs. Finally, we present and discuss experimental results of the use of proposed methodology.

1 Introduction

It has been recognized by many that congestion and poor performance can be the major impediments for the growth of e-commerce. The survey in [8] showed that around 19% of the people surveyed attribute to bad performance the bad experiences they had with e-commerce sites. 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 online trading services if they feel they have been short changed and others just move their business elsewhere.

Capacity planning procedures have been used to assure that customers receive adequate quality of service as they navigate through the site. A key step of any performance evaluation and capacity planning study is workload characterization [4, 12, 13]. Traditional workload characterization metrics, based on hits/sec, page views/sec, or visits/sec, are not appropriate for e-commerce sites. In these environments, customers interact with the site through a series of consecutive and related requests, called sessions. It has also been observed that different customers exhibit different navigational patterns. The goal of this paper is to present a workload characterization methodology that captures these novel characteristics of the interaction between customers and site in e-commerce environments. First, we introduce a state transition graph called Customer Behavior Model Graph (CBMG), that is used to describe the behavior of groups of customers who exhibit similar navigational patterns. Useful metrics, such as average session length, average number of items bought per customer visit, and buy to visit ratio can be directly derived from the analysis of the CBMG. We then describe the steps needed to construct a resource usage-oriented workload model for e-commerce sites. The construction of a workload model relies on two algorithms proposed in the paper. The first one takes as input conventional HTTP logs and generates a session log. The second algorithm takes as input the session logs and performs a clustering analysis which results in a set of CBMGs that can be used as a compact representation of the e-commerce workload. To verify our approach, we generated artificial e-commerce logs based on different CBMG patterns drawn from data collected from the operation of actual online bookstores [3, 11]. We then applied clustering algorithms to the logs and determined groups of customers that exhibit similar navigational behav-

E-COMMERCE 99, Denver, Colorado

ior. We also applied the algorithms to logs of an actual e-commerce site.

It is fundamental in evaluating the performance of Web servers to have a solid understanding of WWW workloads. Web workload is unique in its characteristics. However, a few references have addressed the problem of workload characterization for e-commerce sites. Most of the existing work concentrates on characterizing Web workloads composed of sequence of page requests [2, 5]. They identified some WWW workload properties and invariants, such as the heavy-tailed distributions (e.g., Pareto distribution) of file sizes in the Web. It has been also observed that WWW traffic is bursty in several time scales [5].

In [6], the authors introduce the notion of session, consisting of many individual HTTP requests. However, they do not characterize the workload of e-commerce sites, that is composed of typical requests such as browse, search, select, add, and pay. The analysis focuses only on the throughput gains obtained by an admission control mechanism that aims at guaranteeing the completion of any accepted session. The work in [9] proposes a workload characterization for e-commerce servers, where customers follow typical sequences of URLs as they move towards the completion of transactions. The authors though do not propose any clustering algorithms or mechanisms to generate workload models. In [18], the authors describe WebLogMiner, a web log analysis tool that uses online analytic processing (OLAP) and data mining techniques to extract information on user behavior from HTTP logs. In [14], the authors present techniques to model and predict user surfing paths. These techniques include a web-mining method that extracts significant surfing patterns.

The rest of the paper is organized as follows. Section two discusses the nature of e-commerce workloads. Section three shows new workload metrics calculated from the analysis of the CBMG. Section four describes the workload characterization methodology. The next section presents the session generation algorithm and the clustering algorithm. Section six discusses the numerical results obtained with synthetic logs and section seven presents preliminary results obtained with real logs. Finally, section eight presents concluding remarks.

2 The Nature of E-commerce Workloads

E-commerce workloads are composed of sessions. A *session* is defined as a sequence of requests of different types made by a single customer during a single visit to a site. Examples of requests for an online shopper are: browse, search, select, add to the shopping cart, user registration, and pay. An online trader would have different operations, such as: enter a stock order, research a fund, obtain real-time quotes, retrieve company profiles, and compute earning estimates. The allowed sequences of requests can be described by a state transition graph called *Customer Behavior Model Graph (CBMG)*. This graph has one node for each possible state (e.g., home page, browse (b), search (s), select (t), add (a), and pay (p)) and transitions between these states. A probability is assigned to each transition. Different types of users 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 [17]. 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 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. Figs. 1 and 2 show the CBMGs for occasional and heavy buyers, respectively.

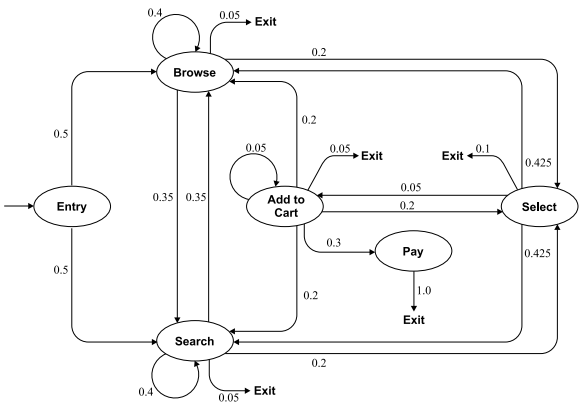

Figure 1: Customer Behavior Model Graph for an occasional buyer.

Note that the CBMGs of Figs. 1 and 2 are just examples. CBMGs can have many other states, depending on the nature of the electronic business. These figures do not explicitly represent the exit state to improve their readability. Transitions to the exit state are represented though. The transitions that indicate exit from the Web store, from states other than "pay", are indicative of spontaneous exits.

It is important to note that the CBMG is a characterization of the navigational pattern as viewed from the server side. That means that a transition from state i to state j is said to occur when the request to go to state j arrives at the server. Therefore, user requests that are resolved at the browser cache or at a proxy server cache

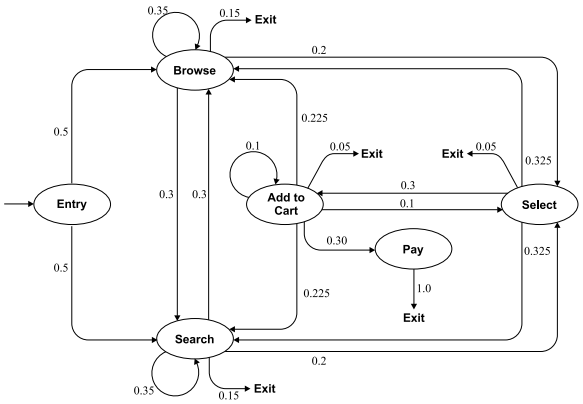

Figure 2: Customer Behavior Model Graph for a heavy buyer.

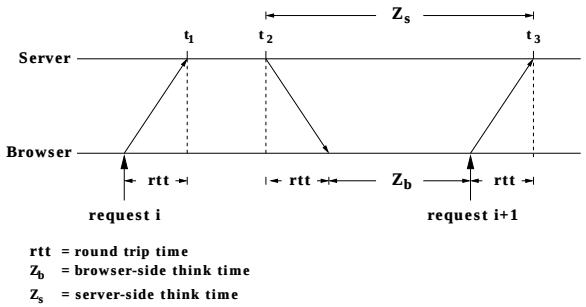

Figure 3: Browser-side and Server-side Think Times.

are not seen by the e-commerce server and therefore are not reflected in the CBMG. However, these requests do not use resources of the e-commerce site and therefore do not have to be considered when workload characterization is done for server sizing and capacity planning purposes [12, 13]. In these cases, it is important to capture the load imposed on the server resources by the various requests submitted by a customer. It should also be noted that, in the case of e-commerce, many pages that are intrinsically static, are generated dynamically, and therefore not cached, because they contain advertisement. The problem of relying solely on server side information is discussed in [18] in the context of using data mining and online analytic processing techniques (OLAP) on HTTP logs. The authors conclude that despite the fact that server side information is not 100% complete, much useful information can be discovered from them.

Another aspect of the nature of the e-commerce workload is the workload intensity. This aspect of the workload is characterized by two main parameters:

- the arrival rate of session initiation requests, measured in sessions/sec, for each type of session. Different session types are characterized by different CBMGs. These arrival rates reflect the mix of customers who used the site. It has been observed in many e-commerce sites that occasional buyers constitute a very large percentage of all customers. In fact, the percentage of customers who end up buying, i.e., visiting the "pay" state has been found to be around 5% [15].
- the average server-side think time (Z_s) between state transitions of the CBMG. This is 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. Figure 3 illustrates this definition. As it can be seen, the server-side think time is given by $t_3 - t_2$ and is equal to $2 \times \text{rtt} + Z_b$ where rtt is a network round-trip time and Z_b is the browser-side think time. From now on, the server-side think time will be simply referred to as think time. A think time can be associated with each transition in the CBMG. Think times are not shown in Figs. 1 and 2. No think times are associated with transitions to the exit state.

So, 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 think times between states of the CBMG. A matrix similar to the transition probability matrix P is described in [18] where Online Analytical Processing (OLAP) and Data Mining techniques are used to analyze Web logs.

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.

3 Metrics Derived from the CBMG

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

$$\begin{aligned} V_1 &= 1 \\ V_j &= \sum_{k=1}^{n} V_k \times p_{k,j} \qquad \text{for all } j = 2, \cdots, n \end{aligned} \tag{1}$$

The system of linear equations of (1) can be written in vector form as $\vec{V} - \vec{1} = \vec{V} \times P$ where $\vec{1} = (1, 0, \cdots, 0)$. As a consequence of the fact that $p_{n,k} = 0 \ \forall \ k = 1, \cdots, n$, $V_n = 1$.

For the CBMGs of Figs. 1 and 2, this system of linear equations becomes

$$\begin{aligned} 1 &= V_y \\ p_{y,b}\ V_y + p_{b,b}\ V_b + p_{s,b}\ V_s + p_{a,b}\ V_a + p_{t,b}\ V_t &= V_b \\ p_{y,s}\ V_y + p_{b,s}\ V_b + p_{s,s}\ V_s + p_{a,s}\ V_a + p_{t,s}\ V_t &= V_s \\ p_{a,a}\ V_a + p_{t,a}\ V_t &= V_a \end{aligned}$$

$$
\begin{aligned}
p_{b,t}\ V_b + p_{s,t}\ V_s + p_{a,t}\ V_a &= V_t \\
p_{a,p}\ V_a &= V_p \\
p_{b,e}\ V_b + p_{s,e}\ V_s + p_{a,e}\ V_a + p_{t,e}\ V_t + p_{p,e}\ V_p &= V_e
\end{aligned} \tag{2}
$$

where $V_y, V_b, V_s, V_a, V_t, V_p$, and V_e are the average number of visits to states Entry, Browse, Search, Add to Cart, Select, Pay, and Exit, respectively.

Note that for each CBMG, we have a different system of linear equations because the transition probabilities are different. The solution to the system of linear equations $\vec{V} - \vec{1} = \vec{V} \times P$ for the graphs of Figs. 1 and 2 is $(V_y = 1, V_b = 6.76, V_s = 6.76, V_a = 0.14, V_t = 2.73, V_p = 0.04, V_e = 1)$ 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. A closed form solution to this system of linear equations for the case of Figs. 1 and 2 is given in the Appendix.

From the vector $\vec{V}$ one can obtain some useful metrics. The first, called *average session length* and denoted by $\bar{S}$, is the average number of states visited by a customer for each visit to the Web store. Thus, $\bar{S} = \sum_{k=1}^{n-1} V_k$. The average session length for occasional and heavy buyers, for the CBMGs of Figs. 1 and 2 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 equal to 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 the buy to visit ratio is given by V_p, where V_p represents the pay state. Suppose that in the case of Figs. 1 and 2, 90% of customers who initiate sessions are occasional buyers. Then, $BV = 0.9 \times 0.04 + 0.1 \times 0.11 = 0.047$.

4 Workload Characterization Methodology

With the growing importance of performance for e-commerce sites, comes an increasing need to characterize and model their workloads. A workload model can be: i) a trace of an actual e-commerce site, ii) an artificially generated stream of requests that mimic a population of customers or iii) a set of parameters that represent the resource usage of an actual workload. Resource usage oriented workload models are useful for performance modeling and capacity planning purposes.

As pointed out earlier, a session is the basic component of the workload model for e-commerce sites. The characterization methodology describes the steps required to construct a resource-oriented model of an e-commerce workload. Starting from the customer navigational patterns, the following methodology enables one to obtain the appropriate physical characterization of the workload.

1. Identify the different types of sessions that compose the workload. Each type of session is associated with a CBMG. Using the terminology of performance modeling, each CBMG represents a class of customers, that are similar to each other concerning request pattern and resource usage. The algorithm to identify and characterize different types of CBMGs is presented in the next section. Once the CBMG has been defined as the basic component, the next step is to define the set of parameters that characterize the workload. The parameters are separated into two groups, one that concerns the workload intensity and the other one referring to the service demands at the resources that compose the site architecture (e.g., network, servers and their components, such as processors, and disks).

2. Calculate the workload intensity parameters. For each customer class r, determine the session arrival rate (λ_r^s) and the average think times between requests of the CBMG, denoted by the matrix Z_r. It is worth mentioning that arrival rates for each type of request (j) can be calculated as $\lambda_r^j = \lambda_r^s \times V_j^r$, where V_j^r is the average number of visits to state j of the CBMG associated with class r.

3. Determine the resource usage parameters. Let us denote by i a resource of the e-commerce site. It could be a specific server, such as a home page server or a database server or it could be a component of a server, such as a processor, disk or network interface. The sum of all service times during one execution of request j of class r at resource i is called service demand and is denoted by $D_{i,r,j}$. For example, the *Add to Cart (a)* operation that is executed in the application server could be characterized by the following parameters: $D_{\mathrm{CPU},r,a} = 30$ msec and $D_{\mathrm{disk},r,a} = 95$ msec. The methods used to compute service demands are well-known in the literature [12, 13]. The resource usage by a session can also be calculated in terms of the parameters derived from the CBMG. $D_{i,r}$ denotes the average service demand of a customer of class r at resource i and is given by:

$$
D_{i,r} = \sum_{j=1}^{n-1} D_{i,r,j} \times V_j^r. \tag{3}
$$

where $D_{i,r,j}$ the service demand at resource i for class r due to one visit to state j and V_j^r is the average number of visits to state j for the CBMG of class r. The workload parameters described here can be used as input to analytical performance models of e-commerce sites. Given that the computation of service demands is a well-known issue, the rest of the paper concentrates on algorithms to characterize and generate high-level e-commerce workload models, based on customer behavior graphs.

5 A CBMG-based Workload Characterization Algorithm

Consider a request log $\mathcal{L}$, derived from an HTTP log. If a site has more than one HTTP server, more than one HTTP log will be generated. We assume that before the log $\mathcal{L}$ is obtained, all HTTP logs are merged into a single log using the timestamp. Clock synchronization

services such as the ones available in Linux and NT can be used to facilitate merging of distributed logs. There is one line in $\mathcal{L}$ for each line in the HTTP log. Each line in $\mathcal{L}$ is assumed to have the following information (uid, request_type, request_time, exec_time), according to the definitions given below.

- uid (u) is an identification of the customer submitting the request. Cookies, dynamic URLs, or even authentication mechanisms can be used to uniquely identify requests as coming from the same browser during a session [16].
- request_type (r) indicates the type of request. Examples include a GET on the home page, a browse request (i.e., a GET on another page), a request to execute a search, a selection of one of the results of a search, a request to add an item to the shopping cart, or a request to pay.
- request_time (t) is the time at which the request arrived at the site.
- exec_time (x) is the execution time of the request. Even though this value is not normally recorded in the HTTP log, servers can be modified to record this information.

From now on, we will represent a line of the request log $\mathcal{L}$ by the tuple (u, r, t, x). Figure 4 illustrates the steps of the workload characterization methodology. The first step consists of merging and filtering HTTP logs to discard irrelevant entries such as errors and others. The result of this step is the request log $\mathcal{L}$. The next step takes as input the request log and generates a session log $\mathcal{S}$ to be described below. The next step takes as input the session log $\mathcal{S}$ and performs a clustering analysis which results in a set of CBMGs that can be used as a compact representation of the sessions in the log $\mathcal{S}$. We now describe the steps GetSessions and GetCBMGs in turn.

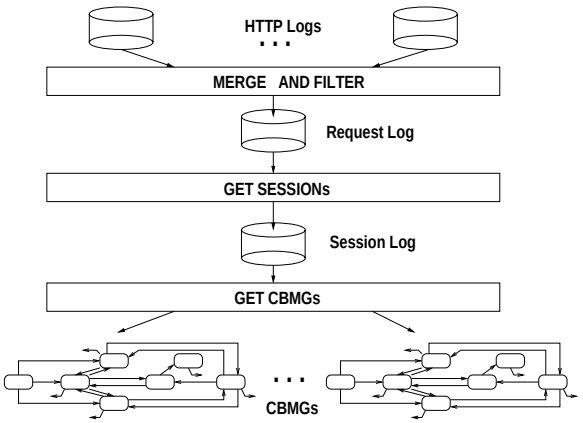

Figure 4: Workload characterization methodology.

5.1 Algorithm GetSessions

Before we describe the step GetSessions, we need to describe the log $\mathcal{S}$. The k-th entry in the log $\mathcal{S}$ is composed of the tuple (C_k, W_k) where $C_k = [c_{i,j}]$ is an $n \times n$ matrix of transition counts between states i and j of the CBMG for one session, and $W_k = [w_{i,j}]$ is an $n \times n$ matrix of accumulated think times between states i and j of the CBMG for one session. To illustrate the notation, consider that for a given session, there were 3 transitions between states s and t, and that the think times for each of the transitions were 20 sec, 45 sec, and 38 sec, respectively. Then, $c_{s,t} = 3$ and $w_{s,t} = 20 + 45 + 38 = 103$ sec.

Before we describe the algorithm GetSessions some definitions are in order:

- $\mathcal{O}$: set of open sessions. An *open session* is a session for which the last request has not yet appeared in the request log $\mathcal{L}$ or has not yet been identified. An open session is uniquely identified by the uid of the customer. Associated with each open session o we must store the transition count and accumulated think time matrices for the session. These matrices for session o are denoted as C_o and W_o, respectively. We must also keep the following variables for each session:
  - LastState_o: last state visited during open session o,
  - TimeLastState_o: time at which state LastState_o was visited, and
  - LastExecTime_o : execution time of the last request executed for that session.
- τ: threshold after which the next request from the same customer is considered to belong to a new session.

Figures 5 and 6 show the algorithm to generate the session log $\mathcal{S}$. Basically, the request log $\mathcal{L}$ is scanned sequentially. A determination is made for each request in this log as to whether it constitutes a request that starts a new session or a request from an existing open session. A request is considered to start a new session if there is no open session for that customer or if the last request from this customer occurred more than τ seconds ago. If it is a request for a new session, one must close an existing session for the same customer, if any is open. If the request is for an existing open session, the relevant entries for the transition count and accumulated think time matrices are updated. An end of session is detected when a new session for the same customer is opened or at the end of processing the request log. Every time a session is closed (see Fig. 6) a new entry is written to the session log. At intervals that are at least τ sec in duration, an attempt is made to close all existing open sessions that have their last entry in the request log more than τ sec ago. This is made to prevent the number of open sessions in $\mathcal{O}$ to grow excessively and use up a lot of main memory.

If the data structure $\mathcal{O}$ is stored as a heap, then the worst-case running time of the GetSessions algorithm is $O(S \log S)$ where S is the number of sessions in the request log.

```
Input Parameters: L, τ
LastClosingTime ← 0; O ← ∅;
For each (u, r, t, x) ∈ L do
  begin
    NewSession ← true;
    if (u ∈ O) and ((t − TimeLastState_u) < τ))
    then NewSession ← false;
    if NewSession
    then begin
           /* close previous session, if any, */
           /* for the same customer */
           if u ∈ O then CloseSession (u);
           /* start new session */
           O ← O ⋃ {u};
           C_u[i, j] ← 0  ∀ i, j;
           W_u[i, j] ← 0  ∀ i, j
         end
    else begin /* not a new session */
           C_u[LastState_u, r] ←
                    C_u[LastState_u, r] + 1;
           W_u[LastState_u, r] ←
                W_u[LastState_u, r]+
                    (t − TimeLastState_u
                     −LastExecTime_u)
         end
    LastState_u ← r;
    TimeLastState_u ← t;
    LastExecTime_u ← x;
    if (t − LastClosingTime) > τ
    then begin /* close old sessions */
           For each o ∈ O s.t.
               (t − TimeLastState_o) > τ do
                          CloseSession (o);
           LastClosingTime_o ← t
         end;
  end;
/* close all open sessions */
For each o ∈ O do CloseSession (o)
```

Figure 5: Algorithm GetSessions

```
For all o ∈ O do
  begin
    /* update number of exit transitions */
    C_o[LastState_o, exit] ← 1
    /* write session log entry */
    Write (C_o, W_o) to the session log S
    /* delete open session o */
    O ← O − {o}
  end
```

Figure 6: Procedure CloseSession

5.2 Procedure GetCBMGs

Once the session log $\mathcal{S}$ is generated, we need to perform a clustering analysis on it to generate a synthetic workload composed of a relatively small number of CBMGs. The centroid of a cluster determines the characteristics of the CBMG. Any number of clustering algorithms could be used. We will use the k-means clustering algorithm [1, 7, 12, 13]. This algorithm begins by finding k points in the space of points, which act as an initial estimate of the centroids of the k clusters. The remaining points are then allocated to the cluster with the nearest centroid. The allocation procedure iterates several times over the input points until no point switches cluster assignment or a maximum number of iterations is performed. Clustering algorithms require a definition of a distance metric to be used in the computation of the distance between a point and a centroid. Assume that the session log is composed of M points $X_m = (C_m, W_m), m = 1, \cdots, M$ where C_m and W_m are the transition count and accumulated think time matrices defined above. Our definition of distance is based on the transition count matrix only since this is a factor that more clearly defines the interaction between a customer and an e-commerce site. We define the distance $d_{a,b}$ between two points X_a and X_b in the session log as the Euclidean distance

$$d_{a,b} = \sqrt{\sum_{i=1}^{n}\sum_{j=1}^{n}(C_a[i,j] - C_b[i,j])^2}. \qquad (4)$$

At any point during the execution of the k-means clustering algorithm we have k centroids. The clustering algorithm needs to keep track of the number of points, $s(k)$, represented by centroid k. We now show how the coordinates of a new centroid, i.e, the new values of the matrices C and W are obtained when a new point is added to a cluster. Suppose that point $X_m = (C_m, W_m)$ is to be added to centroid k represented by point (C, W). The new centroid will be represented by the point (C', W') where the elements of the matrices C' and W' are computed as

$$c'[i,j] = \frac{s(k) \times c[i,j] + c_m[i,j]}{s(k)+1} \qquad (5)$$

$$w'[i,j] = \frac{s(k) \times w[i,j] + w_m[i,j]}{s(k)+1}. \qquad (6)$$

Once all the clusters have been obtained, we can derive the matrices P and Z that characterize the CBMG associated with each cluster, as follows

$$p[i,j] = c[i,j] / \sum_{k=1}^{n} c[i,k] \qquad (7)$$

$$z[i,j] = w[i,j]/c[i,j]. \qquad (8)$$

The arrival rate, λ_k^s, of sessions represented by the CBMG of cluster k is given by $\lambda_k^s = s(k)/T$ where T is the time interval during which the request log $\mathcal{L}$ was obtained.

6 Experimental Setup and Results

In this section, we report on simulation experiments designed to test the effectiveness of the proposed clustering algorithms for e-commerce workloads. Actual e-commerce workloads are considered a sensitive issue for all e-commerce companies. Therefore, we decided to construct an experimental environment to simulate an online bookstore, based on information described in [3, 11]. 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 CBMGs 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 generate requests to start sessions at the e-commerce site. The request generation process follows a bursty pattern typical of distributions for service requests [5]. 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 sessionis 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 and 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. Let us describe how request timestamps are generated. Assume that a customer is at time t_1 at a given state and decides to generate a transition to another state. Then, the timestamp for the new request will be $t_1 + Z_b + 2 \times RTT$, where RTT is the network roundtrip time and Z_b is the browser-side think time. The request generating process is repeated until the customer leaves the site spontaneously, without any specific reason. This case is represented by the "exit arrow" in each state of a CBMG.

6.1 Results of the Clustering Algorithms

To assess the efficiency of the clustering algorithm we focused on two random variables:

- $\tilde{d}_k$: average intracluster distance for cluster k, defined as the average distance of all points of cluster k to its centroid. So,

$$\tilde{d}_k = \frac{1}{s(k)} \sum_{x \in \mathcal{C}_k} d(x, C_k) \tag{9}$$

  where $\mathcal{C}_k$ is the set of points in cluster k, and C_k is the centroid of cluster k.

- $\tilde{D}_{i,j}$: intercluster distance between clusters i and j for $i \neq j$. This is defined as $d(C_i, Cj)$.

We then compute the sample mean ($\bar{d}$), sample variance (σ^2_{intra}), and sample coefficient of variation (C_{intra}) for the intracluster distance as

$$\bar{d} = \frac{1}{k} \sum_{j=1}^{k} \tilde{d}_k \tag{10}$$

$$\sigma^2_{\text{intra}} = \frac{1}{k-1} \sum_{j=1}^{k} (\tilde{d}_k - \bar{d})^2 \qquad k > 1 \tag{11}$$

$$C_{\text{intra}} = \sigma_{\text{intra}} / \bar{d} \tag{12}$$

The sample mean ($\bar{D}$), sample variance (σ^2_{inter}),and sample coefficient of variation (C_{inter})of the intercluster distance is computed as

$$\bar{D} = \frac{1}{k\ (k-1)/2} \sum_{i=1}^{k} \sum_{j=i+1}^{k} \tilde{D}_{i,j} \qquad k > 1 \tag{13}$$

$$\sigma^2_{\text{inter}} = \frac{1}{k\ (k-1)/2 - 1} \sum_{i=1}^{k} \sum_{j=i+1}^{k} (\tilde{D}_{i,j} - \bar{D})^2 \qquad k > 2 \tag{14}$$

$$C_{\text{inter}} = \sigma_{\text{inter}} / \bar{D} \tag{15}$$

In general, the purpose of clustering is to minimize intracluster variance while maximizing intercluster variance. It is clear that if the number of clusters is made equal to the number of points, we will have achieved this goal. On the other hand, we want a compact representation of the workload. So, we need to select a relatively small number of clusters such that the intracluster variance is small and the intercluster variance is large. The ratio between the intra and intercluster variance, denoted β_{var}, and the ratio between the intra and intercluster coefficient of variation, denoted β_{cv}, are important in determining the quality of the clustering process. The smaller the values of β_{cv} and β_{var} the better.

Figure 7 plots the inter and intracluster coefficient of variation as well as β_{cv} versus k. As it can be seen in the figure, C_{intra} does not vary much with the number of clusters. On the other hand, C_{inter} increases with k. The important observation is that β_{cv} drops significantly from $k = 3$ to $k = 6$ and then exhibits a much slower rate of decrease. This is an indication that we should select $k = 6$ as the number of clusters in our workload. This can be also verified by the fact that for $k = 6$, β_{var} exhibits a local minimum, as shown in Fig. 8. It should be noted that even though we generated our synthetic workload from two CBMGs, the resulting six-workload characterization is more meaningful since it provides a further refinement of the two original CBMGs.

6.2 Cluster Analysis

The goal of clustering algorithms is to identify natural groups of customers, based on similar navigational

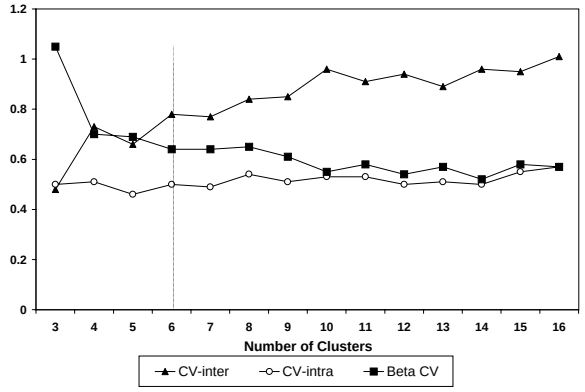

Figure 7: Inter and intra cluster coefficients of variation and β_{cv} vs. k.

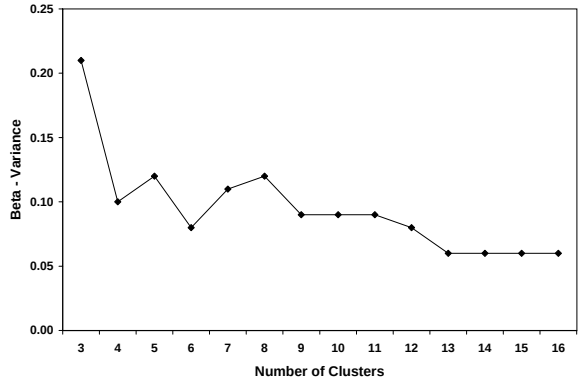

Figure 8: β_{var} vs. k.

patterns. Once we have partitioned the workload into clusters or classes, the next step is to interpret the structures that exist in the sets considered. This section analyzes the six clusters obtained by the proposed algorithms and shows how the cluster's parameters can be used to interpret characteristics of the customer behavior. The GetSessions procedure was applied to log $\mathcal{L}$ with 340,000 lines, representing HTTP operations. The number of sessions identified by GetSessions in log $\mathcal{L}$ is 20,000.

Table 1 reports the values of some parameters of the centroids of the six clusters obtained from the analysis of the 20,000 sessions. The first line shows the percentage of sessions that fall into each cluster. For instance, cluster 1 represents almost half of all the sessions. Line 2 shows the *Buy to Visit Ratio* (BV), that represents the percentage of customers who buy from the Web store. Session length indicates the average number of shopper operations requested by a customer for each visit to the electronic store. Line 5 exhibits the *Add to Shopping Cart Visit Ratio* (V_a), that represents the fraction of times a customer adds an item to his/her shopping cart. However, this operation does not necessarily imply a buy operation, as can be noticed from the comparison between its values and the BV's values. The last line of the Table indicates the number of browsing and searching operations associated with customers of each cluster. The natural question that arises now is: What kind of conclusions can we draw from the above characterization of the e-commerce workload?

Cluster	1	2	3	4	5	6
% of the Sessions	44.28	28	10.6	9.29	6.20	1.5
BV Ratio (%)	5.7	4.5	3.7	4	3.5	2
Session Length	5.6	15	27	28	50	81
AV Ratio (%)	11	15	21	20	32	50
$V_b + V_s$	3.6	11.4	20	23	39	70

Table 1: Resulting Cluster Attributes

In the sample we analyzed, we can note two very different behavior patterns. Cluster 1, that represents the majority of the sessions (44.28%) has a very short average session length (5.6) and the highest percentage of customers that buy from the store. On the other extreme, we notice that cluster six represents a small portion of the customers, that exhibit the longest session length and the smallest buying ratio. In order to try to correlate these parameters, we plotted in Fig. 9 the percentage of customers who buy as a function of the average session length. We can observe that for this sample, an interesting pattern was found: the longer the session, the less likely it is for a customer to buy an item from the Web store. This observation adds validity to our workload characterization technique. As seen in Sec. 3, for the two CBMGs used to generate the synthetic workload, the average session length for occasional buyers is more than double that of heavy buyers.

Another important aspect of the workload characterization is that we can map the visiting patterns identified for each cluster of customers onto software and hardware resources demanded from the system. For instance, looking at Table 1, we note that browsing and searching functions represent almost two thirds of the operations requested in all groups of customers. Using the service demand parameters of these two operations, one could build a multiple class performance model to evaluate, for example, the impact of a new database software on the performance of the e-commerce site.

7 Experiments with Real Logs

Obtaining HTTP logs from actual e-commerce sites can be a challenge since these logs may contain information that is quite revealing about the nature and degree of success of the business. We were able to obtain an HTTP log file from an e-commerce company that was kind enough to provide us with a sanitized version of their logs. Due to a non-disclosure agreement, we will not be able to name the company nor provide information on sale-related matters. Notwithstanding, we provide in this section some preliminary results of the analysis we carried out on these logs. More work is in progress.

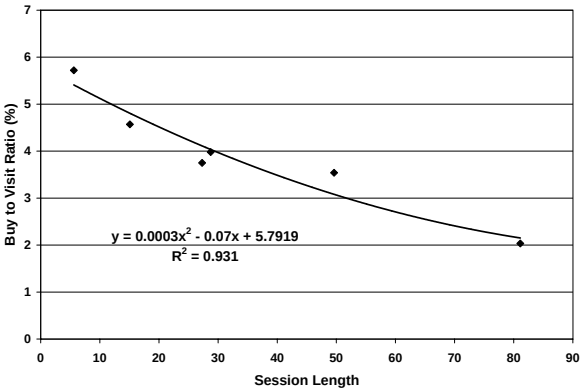

Figure 9: Buy to Visit Ratio vs. Session Length

Cluster	Percent of Points (%)	Avg. Session Length
1	6.50	12.00
2	42.6	6.90
3	20.4	7.15
4	12.71	8.96
5	2.67	14.77
6	7.97	11.97
7	7.17	11.21

Table 2: Cluster Analysis of a Real Log

We had six log files with a total of 17,727,573 HTTP requests. After eliminating requests for images, we were left with 6,004,813 HTTP requests. These are requests for HTML files and execution of cgi scripts. After running the GetSessions algorithm on 628,573 requests, we identified a significant number of very small sessions and a few very large sessions, due to accesses by robot agents. In order to analyze meaningful sessions, we decided to work with sessions whose length was greater than 3 and smaller than 20 requests. As a consequence, we were able to identify 34,811 sessions, described by CBMGs, with an average size of 8.46 requests each. We report here results with one of the logs that had almost 295,000 filtered requests.

We then applied the clustering algorithm and observed that the intercluster and intracluster variances achieve a local minimum for seven clusters. We show in Table 2 the results obtained in the workload characterization. The table shows the percent of points as well as the average session length for each cluster.

The values of the inter and intra cluster variances, coefficients of variation, and β are: $\sigma^2_{\text{inter}} = 18.38$, $\sigma^2_{\text{intra}} = 00.10$, $C_{\text{inter}} = 0.77$, $C_{\text{intra}} = 0.11$, $\beta_{var} = 0.0054$, and $\beta_{cv} = 0.1428$.

We noticed from the clustering analysis of the real logs that the buy probability tends to decrease as sessions get longer. It is interesting to observe that the same result was found in the analysis of the synthetic logs (see Fig. 9). Additional work to characterize the behavior of actual e-commerce workload is in progress.

8 Concluding Remarks

Workload characterization is a key step of any capacity planning and performance evaluation study. This paper presented a novel approach to workload characterization in the context of e-commerce servers. To address the e-commerce workload characterization issue, we introduced a state transition graph called Customer Behavior Model Graph (CBMG), that describes a customer session, i.e., a sequence of requests of different types. A CBMG is formally defined by a matrix of transition probabilities between the states of the graph and by a matrix that represents the average server-side think times between states of the CBMG. We showed how to analytically derive from the CBMG useful workload information, such as the average number of visits to each state per visit to the electronic store, the average session length, and the buy to visit ratio.

In e-commerce, customers interact with the site through a series of consecutive and related requests, called sessions. The workload characterization methodology presented here starts with a determination of the sessions present in the HTTP request log. A CBMG is computed for each session. Then, using clustering algorithms, we show how individual CBMGs can be grouped into groups of users with similar behavioral characteristics. The methods presented here were applied to a simulated electronic bookstore. Starting with a request log with 340,000 records, we detected approximately 20,000 sessions, which were then clustered into six groups of customers with similar navigational patterns. A method to find out the best number of clusters was also presented. Clusters' parameters can then be used for the input definition of multiple class performance models of e-commerce services.

Appendix: Solution to $\vec{V} - \vec{1} = \vec{V} \times P$

After some algebraic manipulation of the equations in (2), we obtain the following relationships that provide the values of $V_y, V_b, V_s, V_a, V_t, V_p$, and V_e.

$$
\begin{aligned}
a_1 &= \frac{1 - p_{a,a} - p_{a,t}\ p_{t,a}}{1 - p_{a,a}} \\
a_2 &= \frac{p_{a,b}\ p_{t,a} + p_{t,b} - p_{a,a}\ p_{t,b}}{1 - p_{a,a}} \\
a_3 &= \frac{a_1\ (1 - p_{b,b})}{p_{b,t}} - a_2 \\
a_4 &= \frac{p_{s,t}\ (1 - p_{b,b})}{p_{b,t}} + p_{s,b} \\
a_5 &= 1 - p_{s,s} - \frac{p_{b,s}\ p_{s,b}}{1 - p_{b,b}} \\
a_6 &= p_{h,s} + \frac{p_{b,s}\ p_{h,b}}{1 - p_{b,b}} \\
a_7 &= \frac{p_{b,s}\ a_2}{1 - p_{b,b}} + \frac{p_{a,s}\ p_{t,a}}{1 - p_{a,a}} + p_{t,s} \\
V_t &= \frac{a_5\ p_{h,b} + a_4\ a_6}{a_3\ a_5 - a_4\ a_7}
\end{aligned}
$$

$$
\begin{aligned}
V_s &= \frac{a_6 + a_7\ V_t}{a_5} \\
V_a &= \frac{p_{t,a}\ V_t}{1 - p_{a,a}} \\
V_p &= \frac{p_{a,p}\ p_{t,a}\ V_t}{1 - p_{a,a}} \\
V_b &= \frac{a_1\ V_t - p_{s,t}\ V_s}{p_{b,t}} \\
V_y &= 1 \\
V_e &= 1.
\end{aligned}
$$

References

[1] B. Everitt, *Cluster Analysis*, Halsted Press, New York, 1980

[2] M. Arlitt and C. Williamson, Web server Workload Characterization: the search for invariants, in *Proc. 1996 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, Philadelphia, May 1996.

[3] V. Almeida, N. Ziviani, V. Ribeiro, and W. Meira, Efficiency Analysis of Brokers in the Electronic Marketplace, *Proc. of the 8^{th} International World Wide Web Conference*, Toronto, May 1999.

[4] M. Calzarossa and G. Serazzi, Workload Characterization: A Survey, Proceedings of the IEEE, Vol. 81, No. 8, August 1993.

[5] P. Barford and M. Crovella, Generating Representative Web Workloads for Network and Server Performance Evaluation, in *Proc. 1998 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, Madison, July 1998.

[6] L. Cherkasova and 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.

[7] D. Ferrari, G. Serazzi, and A. Zeigner, *Measurement and Tuning of Computer Systems*, Upper Saddle River, Prentice Real, 1983.

[8] GVU's WWW User Surveys, http://www.gvu.gatech.edu/user_surveys/

[9] D. Krishnamurthy and J. Rolia, Predicting the Performance of an E-Commerce Server: Those Mean Percentiles, in *Proc. First Workshop on Internet Server Performance*, ACM SIGMETRICS 98, June 1998.

[10] M. MacDougall, *Simulating Computer Systems: Techniques and Tools*, The MIT Press, 1987.

[11] Menascé, D. A., V. A. F. Almeida, R. Fonseca, M. A. Mendes, Resource Allocation Policies for E-commerce Servers, in *Proc. Second Workshop on Internet Server Performance*, held in conjunction with ACM Sigmetrics'99, Atlanta, GA, May 1, 1999.

[12] Menascé, D. A., and V. A. F. Almeida, *Capacity Planning for Web Performance: metrics, models, and methods*, Prentice Hall, Upper Saddle River, NJ, 1998.

[13] Menascé, D. A., V. A. F. Almeida, L. W. Dowdy, *Capacity Planning and Performance Modeling: from mainframes to client-server systems*, Prentice Hall, Upper Saddle River, 1994.

[14] Pitkow, J. and P. Pirolli, Mining Longest Repeating Subsequences to Predict World Wide Web Surfing, *Proc. 2nd. Usenix Symposium on Internet Technologies and Systems*, Boulder, CO, October 1999.

[15] Nielsen, J., http://www.useit.com/alertbox/990207.html

[16] Treese, G. W. and L. C. Stewart, *Designing Systems for Internet Commerce*, Addison Wesley, Reading, MA, 1998.

[17] C. Shapiro and H. Varian, *Information Rules: a strategic guide to the network economy*, Harvard Business School Press, 1999.

[18] Zaiane, O. R., M. Xin, and J. Han, Discovering Web Access Patterns and Trends by Applying OLAP and Data Mining Technology on Web Logs, *Proc. Advances in Digital Libraries Conf. (ADL'98)*, Santa Barbara, CA, April 1998, pp. 19–29.