Journal of Web Engineering, Vol. 1, No. 1 (2002) 074–090

CHARACTERIZING E-BUSINESS WORKLOADS USING FRACTAL METHODS

Daniel Menascé*, Bruno Abrahão†, Daniel Barbará§,
Virgílio Almeida† and Flávia Ribeiro†

(*) Dept. of Computer Science, George Mason University,
Fairfax VA, USA
E-mail: menasce@cs.gmu.edu

(†) Dept. of Computer Science, Universidade Federal de Minas Gerais,
Belo Horizonte, MG, Brazil
E-mail: {brut,virgilio,flavia}@dcc.ufmg.br

(§) Dept. of Information and Software Engineering, George Mason University,
Fairfax VA, USA
E-mail: dbarbara@gmu.edu

Received October 30, 2002

Understanding the workload of Web and e-business sites is a fundamental step in sizing the IT infrastructure that supports these sites and in planning for their evolution so that Quality of Service (QoS) goals are met within cost constraints. This paper presents two approaches for characterizing e-business sessions: distance-based and fractal (session similarity). We apply both approaches to an actual e-business workload to understand what customers do, what navigational patterns they follow, and to identify groups of users that have similar behavior. We also present the benefits and drawbacks of both approaches. The main contribution of this work is the presentation of techniques that improve the process of workload characterization.

Keywords: e-business, e-business workload characterization, k-means clustering, fractal dimension, fractal feature selection, fractal clustering.
Communicated by: D Schwabe

1. Introduction

Understanding the workload of Web and e-business sites is a fundamental step in sizing the IT infrastructure that supports these sites and in planning their evolution so that Quality of Service is met within cost constraints. Most important, one of the main potential benefits of properly characterizing the workload is to improve the quality of experience of a customer at a web site.

The first attempts to characterize [3, 6, 10] the workload of web sites focused on information providing sites only and considered only the stream of HTTP requests coming to the sites. They analyzed statistics related to the traffic, file sizes, and relationship between popularity and frequency of access. More recently, the authors of [16] proposed a characterization approach for e-business workloads using a hierarchical model as shown in Figure 1. Workload characterization can be accomplished at many levels: user level (sessions), application level (functions requested), protocol level (HTTP), and resource level.

This approach analyzes each layer individually in order to obtain a characterization of the arrival process and usage statistics. At the session layer, the analysis included information

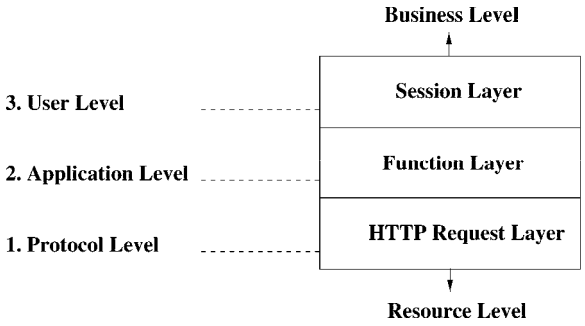

Fig. 1. A hierarchical workload model

such as session inter-arrival times, session length distribution and number of active sessions and initiated sessions. But the user interaction with the web site was not deeply studied.

The popularity of an e-business site depends directly on the quality of the experience of a user. This is why it is so important to understand and model how users behave while they interact with the e-business site. So, based on the approach of [16], we focused on the session layer (user level) for a more careful characterization.

To characterize the user's behavior we use the information obtained in a session [5], defined as a sequence of requests of different types made by a single customer during a single visit to a site. There are several ways of representing a web session such as the CVM (customer visit model), the CBMG (customer behavior model graph) and as sequences. The best representation depends on the information you want to extract from the web log.

In this paper, we use the CVM, which represents sessions by collections of vectors (one per session) that count the number of e-business functions of each type requested within a session. One advantage of this representation over the CBMG model is its simplicity.

Our goal is to group similar sessions in order to characterize the user interaction pattern with the site. To achieve this, we first used a distance-based clustering approach. As this method presented limitations, we tried to circumvent the limitations using a fractal methodology. These methods allow us to reduce the dimension of the data sets by removing attributes that are not relevant and do not contribute to the characterization of the workload. A preliminary discussion of the techniques presented here appeared in a previous paper by the authors [21].

This paper is organized as follows. Section 2 briefly summarizes the results of previous work relevant to this paper. Section 3 presents some of the characteristics of the e-business data we use and discusses how we represent its sessions. The next section presents our first approach to the problem using distance-based clustering algorithm. Section 5 presents the fractal methodology developed to characterize e-business workloads. Finally, Section 6 presents some concluding remarks.

2. Related Work

There are many references on workload characterization of Web workloads for information

provider Web sites [5, 6, 10], but very few for e-commerce sites [4, 14, 16].

In [9], the notion of a session which consists of several individual HTTP requests is introduced. Reference [17] presents several models (e.g., the Customer Behavior Model Graph and the Customer Visit Model) and shows how user sessions can be represented by these models and how to obtain them from HTTP logs. In [16], the authors use a hierarchical model to characterize e-commerce workloads from a real Web store log. This method takes into account, not only requests for files, but also user sessions consisting of requests for e-commerce functions (i.e., add to cart and pay). In addition, they detect and characterize the presence of Shopbots and Crawlers in the workload. That work uses the same log used in this paper.

In the database field, the fractal dimension has been proven to be a suitable tool to estimate spatial joins and range queries [8], indexing and feature selection [20] amongst others. A good coverage of fractal dimension, including the correlation fractal dimension D_2, is given by Belussi and Faloutsos [8], which show an efficient algorithm to compute these dimensions.

In [12], the Pair Count Exponent power law is presented along with its application to estimating the selectivity of spatial joins queries. The correlation fractal dimension D_2 can be obtained from the Pair Count plot.

described in [20]. A recent paper by the same authors [23] presents an algorithm to search for correlated attributes and thus perform dimensionality reduction (however, the complexity of the algorithm is higher than that of the method we chose to use). In addition to presenting the algorithm, that work also shows results using real and synthetic databases. We use here the Fractal Clustering algorithm (FC) proposed by Barbará et. al. [7] to cluster datasets of n-dimensional points and group them in clusters not restricted by its shape.

A methodology to discover web robots sessions in a stream of requests to an e-commerce service can be found in [22]. The authors use classification models that distinguish robot from non-robot sessions.

3. E-Business Workload

E-business workloads should be characterized at higher levels of abstraction, i.e., sessions as opposed to HTTP requests. For that purpose, we analyzed large HTTP logs of an online bookstore. The logs correspond to 15 days in which 955,818 HTTP requests were processed. Entries corresponding to images (representing 71% of the web log) and errors were deleted and the URLs of the remaining entries in the log were mapped to one of twelve e-business functions defined in Table 1. Requests for e-business functions amounted to 26.3% of the requests received by the bookstore.

Then, sessions within the log were identified using a combination of session ids generated by the online bookstore and a 30-minute inactivity period threshold [17].

We considered the two approaches proposed by Menascé and Almeida [15] for representing sessions: Customer Behavior Model Graphs (CBMGs) and Customer Visit Models (CVM). The CBMG is a state transition graph, in which the nodes correspond to states in the session (e.g., browsing, searching, selecting, checking out, and paying) and the arcs correspond to transitions between states. Probabilities are associated with transitions as in a Markov Chain. In [18], a clustering-based method was presented to process HTTP logs and obtain clusters of CBMGs with "similar" patterns (e.g., heavy buyers, occasional buyers).

A Customer Visit Model (CVM) represents sessions of a web site log as a collection of

Table 1. List of E-business Functions

Function Name	Description
Acc	Account Login and Creation
Add	Add items to the cart
Browse	Navigate through product categories
Help	Obtain help
Home	Request the site's home page
Info	Obtain product information
Pay	Pay for an item
Post	Request a summary of items paid for
Pre	Submit payment info
Robot	Request the file `robot.txt`
Search	Search for products based on keywords
Undef	Undefined function

session vectors, one per session. A session vector $V_j = (v_1, \cdots, v_m)$ for the j-th session indicates the number of times, v_i $(i = 1, \cdots, m)$, that each of the m different functions (e.g., search, browse, add to cart, etc) were invoked during the session.

In this paper, we applied clustering techniques using the CVM model rather than the CBMG, since the CVM is a more compact representation of the workload than a CBMG. In the CBMG representation, each session would be represented by a 12 $\times$ 12 matrix of transitions probabilities while in a CVM each session is represented by a vector with twelve dimensions. This reduces the size of the representation but at the same time gives us enough information to find session patterns.

A session vector was generated for each session and the resulting dataset containing all sessions is called hereafter the *complete* dataset. We define the session length, S, as the total number of requests to execute e-business functions during the session. So, $S = \sum_{i=1}^{12} v_i$.

4. Distance-based Characterization of E-Business Workloads

One of the ways of improving the session layer characterization is to cluster similar sessions into groups and then characterize these groups according to their the navigational pattern. In order to do this, we apply a clustering algorithm to the e-business dataset described in the previous section and identify different classes of users given the clusters generated by the algorithm.

Clustering techniques based on the definition of distance (e.g., Euclidean or Manhattan) are commonly used in many applications. Some examples of these algorithms include k-means, minimum spanning tree, and others [11]. These clustering techniques assume that the clusters are shaped as hyperspheres and the centroid is the center of the hypersphere. In this paper, we have chosen the k-means algorithm using the Euclidean distance.

The k-means algorithm selects k points as the initial k clusters and adds each remaining point to the closest cluster using a predefined distance metric. Every time a point is added to a cluster, the coordinates of the centroid of the cluster have to be recomputed [11]. Point allocation to clusters may have to be repeated until no point changes its cluster allocation or

a maximum number of steps is performed.

k-means requires, as its input, the number of clusters into which we intend to partition the dataset. A common question is how many clusters accurately represent the workload. For characterization purposes, it is desirable to keep this number small. A more precise way of answering this question involves the following two metrics: the average distance between points of a cluster and its centroid—the intra-cluster distance—and the average distance between centroids—the inter-cluster distance. The purpose of clustering is to minimize the intra-cluster distance while maximizing the inter-cluster distance. Excluding the case where every point represents a cluster, we determine the number of clusters by determining the smaller value of the ratio between the intra-cluster and inter-cluster distances, denoted by β_{CV}. In a previous study based on the same online bookstore log [18], the authors determined, using β_{CV}, that the number of clusters that best characterizes the dataset is six.

4.1. *First Attempt - Complete Dataset*

Table 2 presents the results of applying k-means clustering to our e-business workload represented by session vectors and using 5, 6, and 7 clusters.

For each number of clusters, the table shows the coordinates of the centroid for each cluster. These coordinates represent the average number of executions of each of the twelve e-business functions by sessions represented by that cluster. Column 15 indicates the percentage of sessions in each cluster. The previous column, S, indicates the sum of the number of executions of each of the twelve e-business functions for that cluster. The last column of the table presents a possible interpretation for the type of sessions represented by each cluster.

Table 2. Results of k-means clustering for the complete log.

Cl	undef	acc	add	aux	browse	home	info	pay	post	pre	robo	search	S	%	Interpr.
5 clusters															
0	0	0.1	0.3	0.1	1.0	0.8	1.1	0	0	0	0	1.5	4	92	hit&run
1	1.8	0.3	0.4	1.6	2.2	1.2	3.6	0	0	0.1	0	26.2	37	5	searchers
2	0.3	1.5	3.2	0.9	2.8	1.4	3.2	0.2	0.1	2.7	0	3.0	19	2	buyers
3	1.3	5.5	5.2	20.9	74.3	11.8	81.4	0	0	0	1.0	6.0	207	0	bots
4	1.0	3.1	0.9	1.1	0.9	1.5	1.1	0.1	3.3	0.3	0	1.1	14	0	buyers
6 clusters															
0	0	0.1	0.3	0.1	1.0	0.8	1.1	0	0	0	0	1.5	4	92	hit&run
1	0.1	0.8	3.4	0.9	2.8	1.4	3.3	0	0	2.7	0	3.1	18	2	chm
2	1.3	4.0	2.8	0.9	2.5	1.5	2.8	1.0	0.6	2.9	0	2.8	23	1	buyers
3	1.3	5.5	5.2	20.9	74.3	11.8	81.4	0	0	0	1.0	6.0	207	0	bots
4	0.9	2.8	0.6	1.0	0.8	1.4	0.9	0	3.0	0.1	0	0.9	12	0	hit&run
5	1.7	0.3	0.4	1.5	2.2	1.2	3.6	0	0	0.1	0	25.8	36	5	searchers
7 clusters															
0	0	0.1	0.3	0.1	1.0	0.8	1.1	0	0	0	0	1.8	5	92	hit&run
1	1.7	0.3	0.4	1.6	2.2	1.2	2.7	0	0	0.1	0	0.7	10	5	chm
2	0.5	2.5	2.3	16.3	55.6	5.6	38.0	0	0	0	1.0	3.0	124	0	browsers
3	4.0	0	0	0	0	0	0	0	0	0	0	23225	23229	0	shopbots
4	1.0	3.1	0.9	1.1	0.9	1.5	1.1	0.1	3.3	0.3	0	1.1	14	0	buyer
5	142.6	224.2	214.8	360	1466	473.2	3277.6	0	0	0	0.6	224.4	6383	0	crawlers
6	0.3	1.5	3.2	0.9	2.8	1.4	3.2	0.2	0.1	2.7	0	3.0	19	2	buyer

We used the following interpretation in every clustering table to indicate the following categories of sessions:

- *shopbots*: sessions generated by shopbots that send requests to various sites for price comparison purposes. As indicated in [2], shopbots are characterized by relatively large

sessions and a very large percentage of search requests as compared to other requests.

- *crawlers*: a typical crawler requests a site home page, parses it, and then follows the links present in that page. It then repeats the process for each new link found in all pages retrieved. As indicated in [2], sessions generated by crawlers tend to be very long—thousands of requests—and tend to cover most e-business functions, except pay and postpay.

- *buyers*: sessions with buying activity (i.e, $v_{\text{pay}} \neq 0$).

- *hit&run*: these sessions are very small, do not show any significant interest from the customer in the site, as indicated by very little product selection (e.g., browse, info, search) activity and very little product ordering (e.g., acc, add, pay) activity.

- *chm*: these sessions characterize customers who seem to have changed their minds with respect to buying as indicated by add to cart activity not followed by a checkout (i.e., pay) activity.

- *bots*: these sessions include a mix of shopbots and crawlers.

- *info*: sessions dominated by info requests with negligible paying activity.

- *browsers*: sessions dominated by browse requests with negligible paying activity.

- *searchers*: sessions dominated by search requests with negligible paying activity.

The characterization of sessions could be useful for planning purposes. It helps to answer questions such as What would be the response time if the percentage of bots increase by 100%? How could one reduce the percentage of *chm* customers? What kind of customers usually buy? Is there a common reason why customers change their mind?

In two of the three cases, ($k = 5$ and 7) there are two clusters named buyers with an average session length of 14 and 19. In the 6-cluster case there is only one cluster named buyer with an average session length of 23. This means that users who buy do not have small sessions.

In the first two cases (5 and 6 clusters) we could not distinguish between the two types of bots, but we could identify the bots' cluster. In the last case (7 clusters), we could identify the two kinds of bots: crawlers and shopbots.

We can also point out that most of the user sessions are in one cluster that contains 92% of the total number of sessions and most of these sessions are human ones.

The problem concerning this big cluster is that the Euclidean distance among the points (or sessions) is small. Sessions with completely different patterns end up in the same cluster. For instance, a session that has only one search access and a session that has only one home.

Here we identify the following problem. We obtained a big cluster with 92% of the sessions in our dataset and we identified that increasing the number of clusters does not change it. So we decided to clean the dataset in order to emphasize the users characteristics we want to extract. In the next subsection, we removed from the dataset sessions with $length = 1$ because these sessions are not very meaningful. Although these sessions do not represent much in terms of number of requests, they are significant in number of sessions, since they

account for 46% of the sessions in the dataset. Besides that, we also tried to remove the robot sessions by deleting sessions bigger than 50 functions requests. We feel that 50 is a safe number to use for that purpose since as seen in the table, the average session length of the buyers cluster does not exceed 23. The robot sessions (or session bigger than 50) represent 0.5% of the total sessions. This new dataset is called *Upto50.*

4.2. *Second Attempt - Small sessions*

Table 3 shows the results of applying the $k-$means algorithm to the *Upto50* dataset.

Table 3. Results of k-means clustering for the upto50 minus 1length sessions log.

Cl	undef	acc	add	help	browse	home	info	pay	post	pre	robo	search	S	%	Interpr.
5 clusters															
0	0.1	0.2	0.2	0.3	0.7	1.0	0.8	0.0	0.0	0.0	0.0	1.2	4	83	hit&run
1	0.1	0.2	0.9	0.2	3.8	1.2	11.3	0.0	0.0	0.1	0.0	2.6	20	4	info
2	0.1	0.7	3.9	0.5	1.5	1.5	2.9	0.0	0.0	0.4	0.0	6.9	18	6	chm
3	0.8	3.1	2.9	0.9	1.8	1.5	2.4	0.6	0.3	3.7	0.0	2.7	20	1	buyers
4	0.1	0.2	0.6	0.2	9.3	1.4	3.2	0.0	0.0	0.1	0.0	1.8	16	6	browsers
6 clusters															
0	0.1	0.2	0.3	0.3	0.2	1.0	0.7	0.0	0.0	0.0	0.0	1.3	4	66	hit&run
1	0.1	0.2	0.8	0.2	3.2	1.2	11.1	0.0	0.0	0.1	0.0	2.6	19	3	info
2	0.1	0.7	4.2	0.6	1.5	1.5	3.1	0.0	0.0	0.4	0.0	7.1	19	5	chm
3	0.8	3.2	2.9	0.9	1.8	1.5	2.4	0.6	0.3	3.7	0.0	2.7	20	1	buyers
4	0.1	0.2	1.0	0.3	12.9	1.5	4.9	0.0	0.0	0.1	0.0	2.4	23	3	browsers
5	0.1	0.1	0.2	0.1	3.4	1.0	1.4	0.0	0.0	0.0	0.0	1.0	7	21	hit&run
7 clusters															
0	0.1	0.2	0.2	0.3	0.4	1.0	0.7	0.0	0.0	0.0	0.0	1.1	4	71	hit&run
1	0.1	0.2	0.7	0.2	3.2	1.1	11.4	0.0	0.0	0.1	0.0	2.3	19	3	info
2	0.0	0.1	0.5	0.1	1.1	1.3	1.7	0.0	0.0	0.0	0.0	8.7	13	5	searchers
3	0.9	3.4	2.7	0.9	1.8	1.5	2.3	0.7	0.4	3.6	0.0	2.6	20	1	buyers
4	0.1	0.1	0.3	0.2	4.4	1.1	1.9	0.0	0.0	0.0	0.0	1.1	9	14	browsers
5	0.1	0.9	5.5	0.7	1.7	1.5	3.7	0.0	0.0	0.7	0.0	4.3	19	4	chm
6	0.1	0.2	1.1	0.3	14.0	1.5	5.6	0.0	0.0	0.1	0.0	2.5	25	2	browsers

Again, we were faced with the same problem found when we used the complete web log. There is still a big cluster, although it is smaller than the one in Table 2 and it contains more than 60% of the number of sessions. Again, this big cluster is of the *hit&run* type.

The main classes that appear in these clusters are *hit&run, info, chm, buyers* and *browsers.* When we increase the number of clusters, a new class shows up: the *searchers.*

An interesting observation this time is that the centroid of the buyers cluster is almost the same regardless of the number of clusters selected.

Since we experienced the same problem as before, we decided to try another dataset. This time, we selected only the users who buy or at least had the intention of buying. This is indicated by $v_{\text{Add}} \geq 1$, i.e., at least one item was placed in the shopping cart. The other dataset, called *Pay*, includes sessions in which $v_{\text{Pay}} \neq 0$ and $v_{\text{Prepay}} \neq 0$ and $v_{\text{Postpay}} \neq 0$, i.e., sessions in which a purchase occurred. This represents 17% of the total number of sessions. We call this dataset *add-pay.* The next subsection shows the results of the clustering algorithm applied to this dataset.

4.3. *Third Attempt - Buying Intention*

Table 4 shows the results of the clustering algorithm applied to *add-pay* dataset.

One interesting observation of Table 4 is that the centroid of the buyers cluster is exactly the same for all clusters selected: $(v_{undef} = 1.3, v_{acc} = 4.0, v_{add} = 2.8, v_{aux} = 0.9, v_{browse} =$

Table 4. Results of k-means clustering for the add-pay log.

Cl	undef	acc	add	help	browse	home	info	pay	post	pre	robo	search	S	%	Interpr.
5 clusters															
0	0.0	0.6	2.1	0.4	2.0	1.1	2.3	0.0	0.0	0.0	0.0	1.9	10	84	hit&run
1	0.1	0.8	3.4	0.9	2.8	1.4	3.3	0.0	0.0	2.7	0.0	3.1	18	11	chm
2	0.9	2.8	0.6	1.0	0.8	1.4	0.9	0.0	3.0	0.1	0.0	0.9	12	1	hit&run
3	3.1	23.7	22.4	35.3	257.2	47.4	321.1	0.0	0.0	0.0	1.1	24.0	735	0	bots
4	1.3	4.0	2.8	0.9	2.5	1.5	2.8	1.0	0.6	2.9	0.0	2.8	23	3	buyers
6 clusters															
0	0.0	0.5	2.1	0.4	2.0	1.1	2.3	0.0	0.0	0.0	0.0	1.9	10	82	hit&run
1	3.1	23.7	22.4	35.3	257.2	47.4	321.1	0.0	0.0	0.0	1.1	24.0	735	0	bots
2	0.9	2.8	0.6	1.0	0.8	1.4	0.9	0.0	3.0	0.1	0.0	0.9	12	1	hit&run
3	0.2	1.2	3.2	0.8	2.4	1.5	3.0	0.0	0.0	1.0	0.0	2.7	16	2	chm
4	1.3	4.0	2.8	0.9	2.5	1.5	2.8	1.0	0.6	2.9	0.0	2.8	23	3	buyers
5	0.1	0.8	3.4	0.9	2.8	1.4	3.3	0.0	0.0	2.7	0.0	3.1	18	11	chm
7 clusters															
0	0.0	0.5	2.1	0.4	2.0	1.1	2.3	0.0	0.0	0.0	0.0	1.9	10	82	hit&run
1	2.0	10.9	10.1	15.1	179.3	23.0	136.8	0.0	0.0	0.0	1.1	11.3	389	0	bots
2	0.9	2.8	0.6	1.0	0.8	1.4	0.9	0.0	3.0	0.1	0.0	0.9	12	1	hit&run
3	0.2	1.2	3.2	0.8	2.4	1.5	3.0	0.0	0.0	1.0	0.0	2.7	16	2	chm
4	1.3	4.0	2.8	0.9	2.5	1.5	2.8	1.0	0.6	2.9	0.0	2.8	23	3	buyers
5	0.1	0.8	3.4	0.9	2.8	1.4	3.3	0.0	0.0	2.7	0.0	3.1	18	11	chm
6	99	1121	1074	1766	6955	2147	16167	0.0	0.0	0.0	1	1122	30452	0	crawlers

$2.5, v_{home} = 1.5, v_{info} = 2.8, v_{pay} = 1.0, v_{prepay} = 2.9, v_{postpay} = 0.6, v_{search} = 2.8$) and it is quite similar to the ones found in the previous section. The average session length is 23. So, we can say that we can distinguish, this time, pretty well the buyers cluster.

There are some robots sessions is this web log. This is because of the crawlers. They can access the function add due to its crawling function. We could distinguish the crawlers in the case of seven clusters.

Again we have a big cluster with 82% of the sessions and a larger session length than the big clusters shown in sections 4.1 and 4.2.

Since we could not solve the problem of breaking up the big cluster, containing more than 60% of the sessions using distance based clustering, we tried another approach, described in the next section, which consists of using a fractal methodology. In this methodology a fractal clustering technique takes into account the similarity of the sessions instead of the Euclidean distance among them.

5. Fractal Methodology

In order to address the drawbacks pointed out in the previous section, we searched for other methods of characterizing user sessions so that those undesirable effects could be minimized. We were motivated to investigate the use of fractal tools due to the following facts:

- Distance-based clustering failed to distinguish points due to the little variation of sessions length. Fractal clustering uses the similarity between points and not their proximity as a clustering criterion. This could lead to a more refined way of grouping similar sessions, providing better differentiation of human generated sessions.

- Distance based clustering forms clusters of regular geometric shapes imposing an artificial way of grouping e-business sessions. Fractal clustering algorithms are able to form clusters of any arbitrary shape, thereby improving the "quality" of the clusters so that one can assign meaningful interpretations to each of them.

- The fractal approach can be used to reduce the fractal dimension, which in addition to reducing the complexity and the size of the dataset, it also reveals the most relevant attributes that should be used to characterize the workload. Moreover, fractal clustering provides a better understanding of the dataset by uncovering hidden relationships among attributes of the dataset.

In this section, we develop and illustrate the fractal methodology to characterize the session layer of the hierarchical workload model. First, we show how the fractal dimension of an e-business workload can be computed. Second, we present the fractal dimension reduction which yields an optimization for clustering points based on the fractal dimension. Last, we show how clusters can be formed, exploiting the similarity of the sessions.

5.1. *Fractal Dimension of e-Business Workloads*

Many phenomena observed in nature seem to have chaotic behavior. However, as we vary the observation scale, they exhibit patterns that repeat themselves. When this happen, we say that these phenomena have *fractal* behavior and this repetition of patterns is called *self-similarity* [7, 13]. Self-similarity has been observed in many phenomena related to computer systems and to Web and e-business workloads. For example, the number of bytes retrieved from a Web server in time slots of duration Δ is a self-similar process over a wide range of values of Δ [10]. Another example is the number of HTTP requests arriving at an e-business site over a time slot of duration Δ [16].

The fractal behavior of a phenomenon is characterized by a fractal dimension [19]. There is more than one kind of fractal dimension for the same phenomenon [8]. We focus on the *correlation fractal dimension (D_2)* [12, 20] since the methods used in this paper make use of this metric. We can calculate D_2 for any dataset using two methods.

An extremely slow method consists in computing all pairwise distances between every pair of points. This is quadratic in nature, and therefore not normally recommended, except on unusually small sets, where an approximation method may not yield particularly good results. Using this method we obtain the pair-count plot or PC-plot of the dataset. The pair count, $PC\ (r)$, of a dataset for a given radius r is defined as the number of pairs in the dataset in which points are within a distance r of each other. The study in [12] showed that $PC\ (r)$ can be approximated by $PC\ (r) = K\ .\ r^{\alpha}$ for many real datasets, if they are analyzed in a usable range of distances, where K is a constant and the exponent α is called the pair-count exponent. The PC-plot is then a plot of log $PC\ (r)$ vs. log r. In other words, the PC-plot is a straight line for a given range of distances. The pair-count exponent is the slope of this straight line in the PC-plot and it is also the "correlation fractal dimension" D_2 of the dataset.

Figure 2 shows the PC-plot of our dataset. The distance metric used here is the Euclidean distance and the natural logarithm was used for both axes. As expected [12], similar results can be obtained using the Manhattan distance [11] and with a sample of the dataset. As it can be seen, for a range of distances from 1 ($= e^0$) to 5.5 ($= e^{1.7}$) the PC-plot is pretty much linear with a slope (correlation fractal dimension) of approximately 3.86.

The second, and commonly used algorithm is box-counting [8]. Box-counting estimates the pairwise counts. The idea is to partition the space into a grid of n-dimensional cells of side equal to r. We then count the number of points in the i-th cell of side r and compute

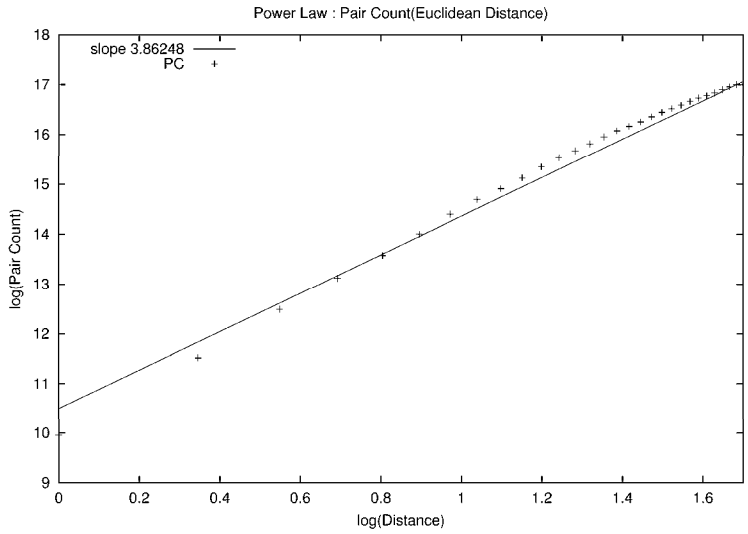

Fig. 2. PC-plot for of the dataset using Euclidean distance.

the frequency, C_i^r, in which points fall within cell i, i.e., the count of points in the cell divided by the total number of points in the dataset. The "correlation fractal dimension" D_2 is then defined as

$$D_2 = \frac{\partial \ log \ \sum_i (C_i^r)^2}{\partial \ log \ r}, \quad r \in [r_1, r_2] \tag{1}$$

for the range $[r_1, r_2]$ in which the dataset presents some self-similarity features [12, 20].

The series of the sums of the second moments of the occupancies of the cells mimics the series of pairs yielded by the full pairwise method, and the slopes (in logarithmic scale) should be very close. This method is generally much faster than the pairwise method. Figure 3 plots the log of the sum of the squares of the cell occupancy frequencies vs. the log of r in selected ranges. Thus, the slope of this graph ($\approx$ 3.6) is a good approximation of the correlation fractal dimension(D_2) of the dataset.

5.2. *Fractal Dimension Reduction*

Our original dataset has an embedded dimension equal to 12 since there are twelve e-business functions in the session vector. As pointed out by [20], we can find some attributes that may not add information to the correlation fractal dimension of the dataset, that is, they do not change the value of D_2 if they are removed from the dataset. Since the fractal-based clustering algorithm uses the correlation fractal dimension as the similarity metric, we use this dimension as the criterium for determining the minimum set of attributes that are representative in the dataset. This means that some of these attributes are not relevant information and they do not need to be taken into account to perform this clustering technique. We examine, in this section, how one can select the relevant attributes in a dataset, in terms of fractal dimension.

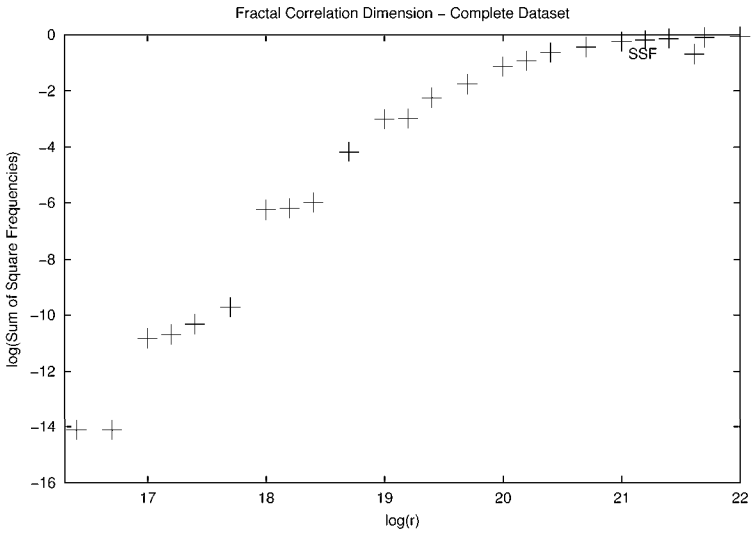

Fig. 3. Log of the sum of cell occupancy frequency vs. cell side of our dataset

The basic process works as follows [20]. We start by computing the fractal dimension D of the complete dataset. Then we examine one attribute (say attribute i) at a time and compute the partial fractal dimension D_i, defined as the fractal dimension of the dataset using all attributes except attribute i. Select the attribute j such that $(D - D_j) = min_i(D - D_i)$. Set D equal to D_j and remove attribute j from the dataset. The process continues until all attributes have been removed.

Figure 4 shows the results of applying this process in the dataset. The y-axis value for a given attribute i is the partial fractal dimension before the attribute is removed. The order in which attributes are removed, as a result of executing the procedure outlined above, is from right to left. So, as the figure shows, removing Postpay through Undef does not significantly change the fractal dimension of the dataset. The results of attribute selection lead us to some optimizations. First, the fractal clustering analysis of these datasets can be carried out with a number of attributes much smaller than the original 12. Second, if an attribute that distinguishes two or more tuples is removed, as these tuples become equal, these tuples can be aggregated in a single point[a], reducing the amount of data to be handled in the clustering process.

The removed attributes fall into one of the three following cases:

- They are null throughout the tuples.
- They are constant throughout the tuples.
- They are correlated with one or more attributes which were not removed.

[a]Fractal clustering forms clusters based on the similarity between points. Thus, removing duplicate points does not affect the resulting clusters.

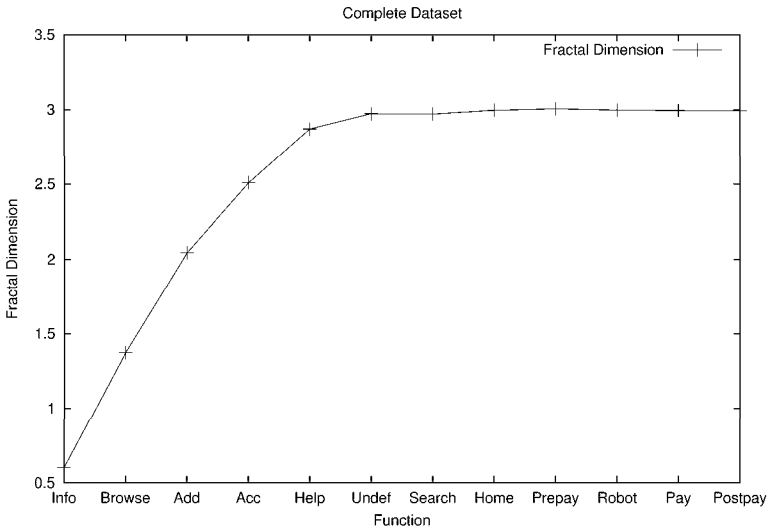

Fig. 4. Attribute selection of the "Complete" dataset

If there are two or more attributes that are correlated, the method drops attributes using this correlation until only the attributes that correspond to independent attributes remain.

In [20] it is shown that the order in which the reduction method removes attributes yields the minimum set of attributes that fully characterize the dataset.

It is important to notice that the method can reveal the existence of correlations and uncover hidden relationships among attributes. However, it does not determine what the correlations are. Finding correlations among attributes can bring valuable information in explaining user behavior.

The computational complexity of finding correlations between attributes of a dataset can be significantly reduced because we just need to search for correlations among each removed attribute and the remaining ones. Correlations among removed attributes, as well as among the remaining ones, do not have to be examined.

As an illustration of how one could find some correlations among attributes, suppose we want to determine some typical behavior of the users who usually buy something from the online bookstore. For this purpose, we perform the fractal dimension reduction method and show the results in Figure 5 for a dataset consisting only of sessions in which a purchase occurs.

Some simple correlations can be found through the use of *association rules* [1]. In this method, the data is assumed to be a basket of items, which, in its simplest form, is a vector of binary values: an item is either in the basket or it is not. This technique finds rules of the form $A \Rightarrow B$, where A and B are *itemsets* in the data. The rule comes with two measures: the *support*, which indicates the probability of the itemset A, B to be in the data set (number of times A and B occur together in the baskets divided by the number of baskets), and the *confidence*, or probability that B occurs in the baskets already containing A (number of times

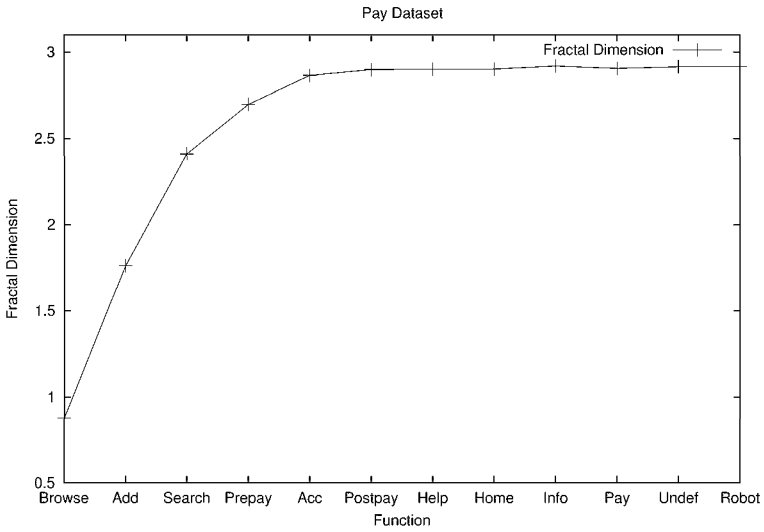

Fig. 5. Attribute selection of the "buyers" dataset

A and B occur together in the baskets divided by the number of baskets containing A). The rules by themselves do not indicate correlation.

However, a different measure, called the *lift* of the rule, defined as the ratio between the confidence and the probability of the consequent (in our example the itemset B) is useful in establishing correlations. A lift greater than 1 indicates a positive correlation and a lift less than 1 a negative correlation.

In our dataset, we took as the baskets, the sessions; as items, the presence or absence of the functions (attributes of the log). For instance, one can have the item $browse$ or $\overline{browse}$, indicating whether the session requested the browse function or not. We tried to find correlations connecting each attribute of the set of removed ones with the attributes in the set of relevant ones. The most interesting correlations found are shown in Table 5.

Rule	**Lift**	**Confidence**
$browse \Rightarrow info$	1.58	0.86
$\overline{info} \Rightarrow \overline{browse}$	1.36	0.89
$\overline{browse}$ and $info \Rightarrow search$	1.26	0.95

Table 5. Best rules for paying customers in the web log

The first two rules indicate a strong correlation between browsing and getting information pages (the customer either uses both or none). The third one indicates that when the rule is broken (the customer does not use the browsing function, but visits the information pages), then the search function is commonly used. This represents a predictable user action: first the user tries to find the product in the web shop and when he/she finds something he/she wants, looks for a more detailed information on it.

5.3. *Fractal Clustering*

The *Fractal Clustering(FC)* algorithm used was proposed in [7]. There, the authors presented a novel algorithm for clustering data points, based on exploiting self-similarity. Starting with a set of initial clusters, this algorithm incrementally processes each point by placing it in the cluster in which it causes the minimum *fractal impact*, that is, the cluster whose fractal dimension changes the least when the point is added to it. A threshold of fractal disturbance due to the addition of a point in a cluster is established in order to prevent the method from being influenced by noise. The points that exceed this threshold are considered outliers by the algorithm.

Despite the fact that the resulting clusters can be of any arbitrary shape, and, in order to provide a better understanding of the resulting clusters of FC, we tried to do a rough approximation of the geometrical centroid of each cluster by reconsidering the removed attributes and equal sessions, and, performing k-means with $k = 1$ for each one of them. The results of this procedure applied to our dataset are shown in Table 6 in the form of the centroids of the three clusters found. The last column indicates the percentage of points in each cluster. The three clusters correspond to human sessions, since the robot sessions were considered as outliers by FC (and form a fourth very small cluster). The clustering was done in a reduced-attribute set and we present in the table the set of attributes that better represent the characteristics of the session in terms of e-business: add, browse, home, info, and search. The centroids suggest the presence of three groups of human sessions, whose main difference is the intensity on requesting the functions.

Table 6. Centroids found by FC on the complete web logs

Cluster	add	browse	home	info	search	%
1	0.251	0.772	0.778	0.811	0.944	96.6
2	3.148	6.312	1.963	6.553	5.100	2.57
3	5.027	8.699	3.208	10.321	8.732	0.83

Looking at the results of Table 6, obtained when FC was performed, we were able to derive a relationship between the sum of the variables browse, info, and search and the add variable. This relationship would not have revealed itself without performing FC. We then performed Chi-Square tests over the variables v_{add} and the sum $v_{\text{browse}} + v_{\text{info}} + v_{\text{search}}$, along with a linear regression between these variables. The results are summarized in Table 7. The interesting fact is that the variables are correlated in all cases (each cluster), but while the correlation is positive in Cluster 1 and the complete set, it is negative in both of the other clusters (the slope of the regression curve is negative). These correlations are an indication that the clusters were well formed and there was a good separation of the characteristics of each cluster in terms of session similarity. Clusters 2 and 3 have significantly more Add activity than cluster 1 (see Table 6). In these two cases, considering the negative slope of each cluster, as customers increase their product selection activity, they tend to add less items to the shopping cart.

The results obtained by FC can be roughly compared with the results obtained by the k-means applied to the *Upto50* session log. In the former, the robots were removed by being considered outliers and in the latest, sessions with a large number of operations, which consist

mostly of robots activity, were also removed. If we inspect the percentage of points in each FC resulting cluster, we can see that 96.6% of the sessions were grouped in the same cluster. This shows the tendency of both methods to consider human sessions as one big cluster as seen in Section 4 . Opposing to what we expected, the FC method was not able to refine the clustering process when there are only subtle differences in the behavior of users, for instance, when we intend to analyze different profiles of human sessions.

Table 7. Correlations between v_{add} and $v_{\text{browse}} + v_{\text{info}} + v_{\text{search}}$ in the complete set and each of the clusters

Cluster	Slope	Constant	Correlation
Cluster1	3.885e-02	0.153	Correlated
Cluster2	-0.179	6.362	Correlated
Cluster3	-0.206	10.754	Correlated
Complete	3.885e-02	0.153	Correlated

6. Conclusion and Future Work

The workload characterization of an e-business site can be divided by layers in a hierarchical model. This model includes the HTTP request layer, the function layer, and the session layer. This paper focuses on the session layer since we want to determine the user access patterns when they interact with the site. It may be useful, from a management standpoint, to separate customers based on their behavior. The interactions of customers of e-business sites are grouped by sessions, which are sequences of requests of different types made by a single customer during a single visit to a site. Within a session, a customer requests the execution of various e-business functions such as browse, search, select, add to the shopping cart, register, and pay. To represent these sessions, we used the Customer Visit Model (CVM) [17], which provides a compact representation of the workload.

In order to group "similar" sessions and understand common patterns of user behavior, we used clustering techniques. First, we used the well known method based on the Euclidean distance to group points—k-means. This method was first used in the e-business context by Menascé and Almeida to cluster similar sessions represented by the CBMG [17]. The results of the application of this method in our workload showed that it is not able to extract meaningful clusters from the data when the sessions have subtle differences and have little variation in length. We realized that the problem persists even if we try to change the characteristics of the dataset in order to make it more obvious what characteristics we want to extract.

As a tentative to address the drawbacks of the distance-based method, we present an initial contribution, proposing a fractal methodology. In contrast to the distance-based method, it uses the similarity to cluster sessions with the same characteristics, based on the fact that distance-based methods will fail to group human generated sessions since the points are pretty close and the distance does not vary much. First, we showed how the fractal dimension of the dataset can be used to reduce the number of attributes, the dimension of the session vector. By analyzing variations in the fractal dimension, the method selects the minimum set of attributes that are relevant to the workload characterization. This process leads to a reduced complexity of the e-business workload characterization and provides better understanding of real workloads by uncovering hidden relationships among the data. We,

then derived association rules that help explain important user navigation patterns. This type of information can be used to redesign a Web site in order to improve user quality of experience.

As the last step of the methodology, a fractal-based clustering algorithm was used to identify groups of user sessions that share some common features. We conclude that, in contrast to what we expected, this method was not able to determine the different patterns of human sessions in our workload. Notwithstanding, the main contribution of this paper is to present the fractal methodology as a possible method to improve user session characterization. More experiments on other logs are required to generalize the advantages and disadvantages of the method. It should be realized that the nature of e-business workloads may vary significantly from site to site. Thus, the main contributions of this paper is on the methodology aspect rather than as an encompassing workload characterization study.

Last but not least, the study of an actual log from an online bookstore is one of the main contributions of this paper. Obtaining logs from actual e-business sites is not trivial due to the proprietary nature of these logs. We were fortunate to be given logs from a real site, that shall remain unnamed for obvious reasons.

Acknowledgements

The authors would like to thank Ping Chen and Julia Couto for their help in running some of the experiments.

References

1. R. Agrawal, T. Imielinski, and A. Swami, "Mining Association rules between sets of items in large databases," Proc. ACM SIGMOD Conference on Management of Data, Washington, DC, May, 1993.
2. V. Almeida, D. A. Menascé, R. Riedi, F. P. Ribeiro, R. Fonseca, and W. Meira, Jr., "Analyzing Web Robots and their Impact on Caching," Proc. Sixth Workshop on Web Caching and Content Distribution, June, 2001, pp. 299–310.
3. V. Almeida, M. Crovella, A. Bestavros, and A. Oliveira, "Characterizing Reference Locality in the WWW," Proc. IEEE/ACM International Conference on Parallel and Distributed Systems (PDIS), December 1996.
4. M. Arlitt, D. Krishnamurthy, and J. Rolia, "Characterizing the Scalability of a Large Web-based Shopping System," *ACM Trans actions on Internet Technology*, Vol. 1, No. 1, August 2001, pp. 44–69.
5. M. Arlitt, "Characterizing web user sessions," HP Laboratories Technical Report, Palo Alto, CA, March 2000.
6. M. Arlitt and C. Williamson, "Web Server Workload Characterization: The search of invariants," Pro. 1996 ACM Sigmetrics Conference on Measurement of Computer Systems, May 1996.
7. D. Barbará and P. Chen, "Using the fractal dimension to cluster datasets," Proc. Int'l Conf. on Knowledge Discovery and Data Mining, 2000, pp. 260-264.
8. Alberto Belussi and Christos Faloutsos, "Estimating the Selectivity of Spatial Queries using the 'Correlation' Fractal Dimension," Proc. 21st Int. Conf. Very Large Databases (VLDB), November, 1995, Morgan Kaufmann, Umeshwar Dayal, Peter M. D. Gray, Shojiro Nishio, eds, pp. 299–310.
9. L. Cherkasova and P. Phaal, "Session Based Admission Control: A Mechanism for Improving the Performance of an Overloaded Web Server," HP Labs Technical Report HPL-98-119, 1998.
10. M. Crovella and A. Bestavros, "Self-Similarity in the World Wide Web Traffic: Evidence and Possible Causes," Proc. IEEE/ACM Transactions on Networking, December 1997, Vol. 5, pp.

836–846.
11. B. Everitt, *Cluster Analysis*, 4th ed., Oxford University Press, Oxford, 2001.
12. C. Faloutsos, B. Seeger, A. Traina, and Caetano Traina, "Spatial join selectivity using power laws," Proc. 2000 ACM SIGMOD International Conference on Management of Data, 2000, pp. 177–188.
13. B. B. Mandelbrot, *The Fractal Geometry of Nature*, W. H. Freeman, New York, 1983.
14. D. A. Menascé, V. Almeida, R. Riedi, F. Ribeiro, R. Fonseca, and W. Meira Jr., "A Hierarchical and Multiscale Approach to Analyze E-Business Workloads," Performance Evaluation Journal, North-Holland, to appear in 2003.
15. D. Menascé and V. Almeida, *Capacity Planning for Web Services: metrics, models, and methods*, Prentice Hall, NJ, 2002.
16. D. A. Menascé, V. Almeida, R. Riedi, F. Ribeiro, R. Fonseca, and W. Meira Jr., "In Search of Invariants for E-Business Workloads," Proc. ACM Conference in Electronic Commerce 2000, Mineapolis, MN, Oct. 2000.
17. D. Menascé and V. Almeida, *Scaling for E-business: Technologies, Models and Performance and Capacity Planning*, Prentice Hall, NJ, May, 2000.
18. D. Menascé, V. Almeida, R. Fonseca, and M. Mendes, "A Methodology for Workload Characterization for E-commerce Servers," Proc. 1st ACM Conference in Eletronic Commerce, Denver, CO, Nov. 1999, pp. 119–128.
19. M. Schroeder, *Fractals, Chaos, Power Laws: Minutes from an Infinite Paradise*, W. H. Freeman and Company, NY, 1990.
20. C. Traina, A. Traina, L. Wu, and C. Faloutsos," "Fast feature selection using fractal dimension," Proc. XV Brazilian Database Symposium, October 2000.
21. D. Menascé, B. Abrahao, D. Barbara, V. Almeida and F. Ribeiro, "Fractal Characterization of Web Workloads," Web Engeneering track, Eleventh International World Wide Web Conference, Honolulu, Hawaii, USA, May 2002.
22. P. Tan and V. Kumar, "Discovery of Web Robot Sessions Based on their Navigational Patterns," Data Mining and Knowledge Discovery, 6:9-35, 2002.
23. E. Parros Machado de Sousa and C. Traina and A. Traina and C.Faloutsos, "How to use the Fractal Dimension to Find Correlations between Attributes," KDD 2002 Workshop on Fractals and Self-Similarity in Data Mining, August 2002.