E-Business Benchmarking Based on Hierarchical Customer Behavior Characterization

Christian Kurz, Günter Haring

University of Vienna

Department for Distributed Systems
Institute for Computer Science and Business Informatics
Vienna, Austria
{christian.kurz, guenter.haring}@univie.ac.at

Abstract

In this paper a methodology for the analysis of web server logfiles is presented. These logfiles were obtained from servers of a large European B2C e-business website. Resulting customer profiles characterize user behavior by using a hierarchical modeling approach. Consumers visiting a web shop are therefore described using a layered representation. At each layer the customer is described at different levels of detail. The resulting model is used to adapt the TPC-W benchmark suite to better characterize customer sessions. The paper presents a new approach to determine user sessions from the logfile data.

Keywords Benchmarking, Hierarchical Customer Behavior Characterization, Logfile Analysis, TPC-W

1 INTRODUCTION

According to the 8-second-rule, described by Zona Research in 1999 [23], after an idle time of about eight seconds most visitors leave a website and therefore potential customers turn away, that implies financial losses for the business. To be able to provide reasonable response times for an e-business site the performance of these sites has to be optimized by running appropriate benchmark suites imitating the behavior of prospective customers.

Therefore the definition and development of methodologies for the design and the implementation of benchmark experiments for e-business activities is crucial. For this purpose we develop a hierarchical workload characterization for e-business activities, adapt the TPC-W accordingly to cover real-world applications and environments, implement the extended TPC-W benchmark suite and perform measurements at selected sites. First results of this effort are presented in this paper.

Traditional benchmarks like the TPC-W don't describe customer behavior precisely enough. Thus we adapt the TPC-W benchmark with a more detailed model of customer behavior. For this purpose a hierarchical model describing customer behavior at different levels of detail is developed using layered characterization architecture. At the highest level the user performs general tasks like browsing the website or searching for a certain item. These tasks are broken down until http-requests are issued to the servers which are to be benchmarked.

We were able to acquire log files from servers running B2C e-business websites. These web logs are analyzed to gather the data necessary for the modeling of customer behavior within our hierarchical model. For this goal a methodology for the analysis, based on our hierarchical model of customer behavior, and some first measurement data is presented in this paper.

This paper mainly describes a methodology to benchmark e-business websites based on hierarchical customer behavior characterization. It extends the TPC-W benchmarking suite which is introduced in chapter 3. A description of the system where the data was obtained from is given in section 4. An insight in hierarchical customer behavior characterization is given in chapter 5. The methodology for data analysis is detailed in section 6.

2 RELATED WORK

There are some references related to the TPC-W benchmarking suite. In [22] Smith gives an overview about TPC-W. In [20] the TPC-W-workgroup shows a detailed specification about TPC-W implementation. Poess and Floyd [13] an introduction to TPC benchmarks, including TPC-W is shown. Cain, Rajavar, Marden and Lipasti [7] and Bezenek et al. [17] show an implementation of TPC-W in Java. Lou and Naughton [15] use TPC-W to evaluate form-based proxy caching for database-backed web sites. Brown, Kar and Keller demonstrate [1] an active approach to characterizing dynamic dependencies for problem determination in a distributed environment is experimentally using the TPC-W benchmarking suite.

Concerning the analysis of web server logfiles a few references could be found. Paraz and Yu [14] analyze apache web server logfiles to measure the performance of IP-networks using content metrics. The lack of scientific works in this area shows the need to develop a methodology for the analysis of logfiles.

Hierarchical modeling methodology can be found in various documents. Calzarossa et al. [12] develops a hierarchical approach to workload characterization in parallel systems. Hlavacs and Kotsis [9] construct a layered user behavior model. Furthermore Hlavacs, Hotop and Kotsis [8] apply this approach by generating workload for the OPNET network simulation tool. Kurz, Hlavacs and Kotsis [5] apply the layered user behavior methodology for modeling workload in an ISP subnet.

Everitt [3], Jain and Dubes [2] and Bacher [10] give an introduction to cluster analysis basics. Raatkainen examines [11] the validity of workload classification using cluster analysis. Fraley and Raftery [4] discuss which clustering method best to use and how to choose the appropriate number of clusters.

3 THE TPC-W

TPC-W is a transactional web e-Commerce benchmarking suite. It is meant to benchmark whole IT-infrastructures which service electronic commerce websites, for example online shops. Browsing customers are substituted by a series of clients which behave as if there are hundreds or thousands of customers which issue requests to the system under test. These clients are called remote browser emulators (see Figure 1). As a result it is measured how many transactions per second the tested system can deliver at most. Documented results for systems of large computer vendors can be found at the TPC homepage [21]. There are no regulations on which hardware and software to use, therefore a wide variety of e-commerce solutions can be compared using TPC-W.

The workload is performed in a controlled internet commerce environment that simulates the activities of a business oriented transactional web server. It models the following activities in an Online Bookstore: searching, browsing, shopping chart, secure purchasing, browsing best sellers and new products, customer registration and administrative updates. The TPC-W models dynamic web pages with static images, a durable shopping cart, it allows 30 seconds for page updates and incorporates various caching optimizations and scaling. A typical configuration is shown in Figure 1.

The measured performance metric is the number of web interactions processed per second. Store sizes vary from 1.000 to 10.000.000 items. Therefore the performance metric is expressed in WIPS@scale (e.g. 123@10.000). There are three different types of users distinguished by their behavior: primarily shopping (WIPS) – the main metric, browsing (WIPSb) and web-based ordering (WIPSo). The fastest system in September 2002 is a Dell PowerEdge 6650 1.6GHz with PowerEdge 1650 1.4GHz scoring 10499 WIPS@10000 (see [18] for the top ten TPC-W results by performance).

Additionally the number of web interactions per second is also measured in regard to system cost. The resulting metric is $/WIPS@scale which divides cost of the IT-infrastructure (see [20] for a detail specification) through the web interactions measured per second. In September the system with the lowest $/WIPS is a Dell Dell PowerEdge 6400/900 with PowerApp Web 120 scoring 24.50 US$/WIPS@10000 (see [19] for the top ten TPC-W results by price/performance).

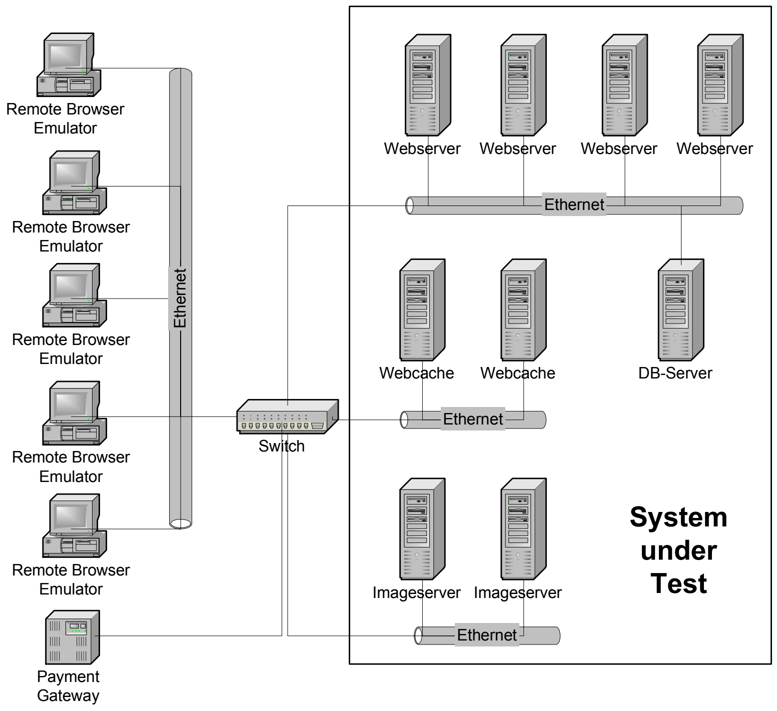

Figure 1: Typical TPC-W System Configuration [22]

A system configuration for measurement with TPC-W typically includes several web servers, load balancers, web caches, image servers, a database server and a number of remote browser emulators which issue http-requests to benchmark the servers.

4 SYSTEM UNDER STUDY

Or Analysis is based on log files obtained from a European B2C online shop. Data was collected for a period of one month in the beginning of 2002. The logs of the webservers contain more than 5 million lines. The area of operation of the online shop is the retail market, it mainly sells groceries.

The IT-infrastructure is based on two web servers (running Apache) connected to a load balancer, two application servers, a database server (Oracle) connected to a disk array and two firewalls. Figure 2 illustrates this structure:

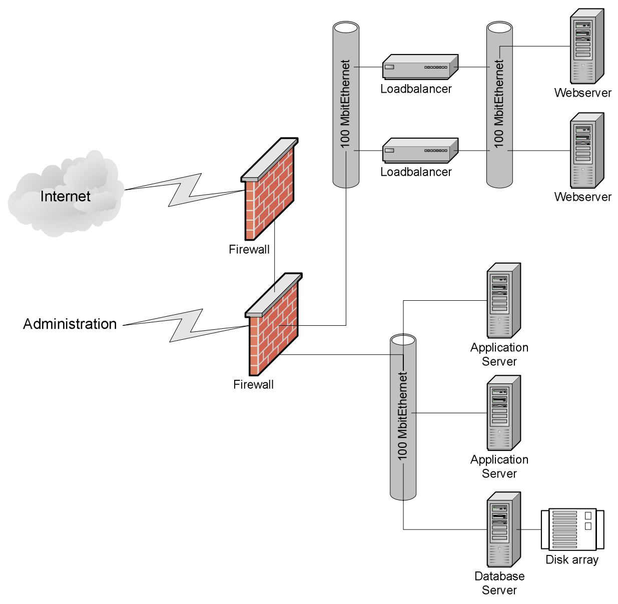

Figure 2: System Configuration

Servers and load-balancers are connected via a 100 Mbit Ethernet. The internal network is connected to external server administration and to the internet via two firewalls. The log files collected from the webservers contain the client IP-address, date and time, the request line (partially including session ID) the server status code and the object size for each http-request line. Additionally the CPU-usage broken down in user, system and I/O is recorded in snapshots every five minutes for the webservers, the application servers, the database server and the firewalls for the same period of time the web logs are collected.

5 HIERARCHICAL CUSTOMER BEHAVIOR CHARACTERIZATION

Today consumers can buy goods online easily by shopping at a wide variety of possible merchants. It is easy for the customer to shop at any “location” in the internet as the only thing he has to do is to type in a new web address into his browser. Thus it is necessary for an online shop to offer unique service to the customer which makes him choose the shop for his shopping. Main features where shops can differ from their competition are pricing and service. An online shop which doesn’t want to distinguish by price alone has to focus to satisfy customers with good service. To provide a high level of service the shopping experience should be tailor made for each individual customer. Therefore in the future more personalization will probably be needed to distinguish a business against its competitors.

To be able to personalize a website for its customers a business primarily has to know about their behavior and to describe it in a machine readable manner. This procedure and its output, a database representing user behavior, is called customer behavior characterization. When profiles are composed according to a hierarchical modeling approach this process is called Hierarchical Customer Behavior Characterization.

Hierarchical user profiles consist of levels of different granularity (see Figure 3). For describing user behavior in an online shopping environment we have chosen to divide the profiles into four different levels, the session layer, the task layer, the activity layer and the and request layer

At highest, the session layer, a user session is defined. In a user session the customer can perform a variety of tasks: Search, Browse, Order, Register and Administrative tasks. These tasks are again broken down to the activity layer:

- *Search:* type in a search, browse the results, go to the next results page
- *Browse:* browse a Top-10-list, browse a category, browse a product, expand picture, read text, comment product
- *Order:* Add item to chart, check out, enter shipping data, decide packaging, confirm order
- *Register:* input necessary data, confirm input, login, logout
- *Administrative* tasks: change registration, change order, change shipping, cancel, add data

At the lowest layer, the request layer, http-requests are generated according to the activities above. In this approach workload is transformed through different levels of abstraction. Besides customer behavior the source for network traffic, http-requests, are modeled. The gap between modeling user behavior and modeling network traffic can therefore be closed by using the proposed hierarchical customer behavior characterization. This means that customer behavior can be closely connected to the generation of network traffic, thus providing a means for network capacity planning. When an online merchant has obtained profiles describing its customers it is able to ensure service quality of its online shop by benchmarking its servers and projecting future service scenarios.

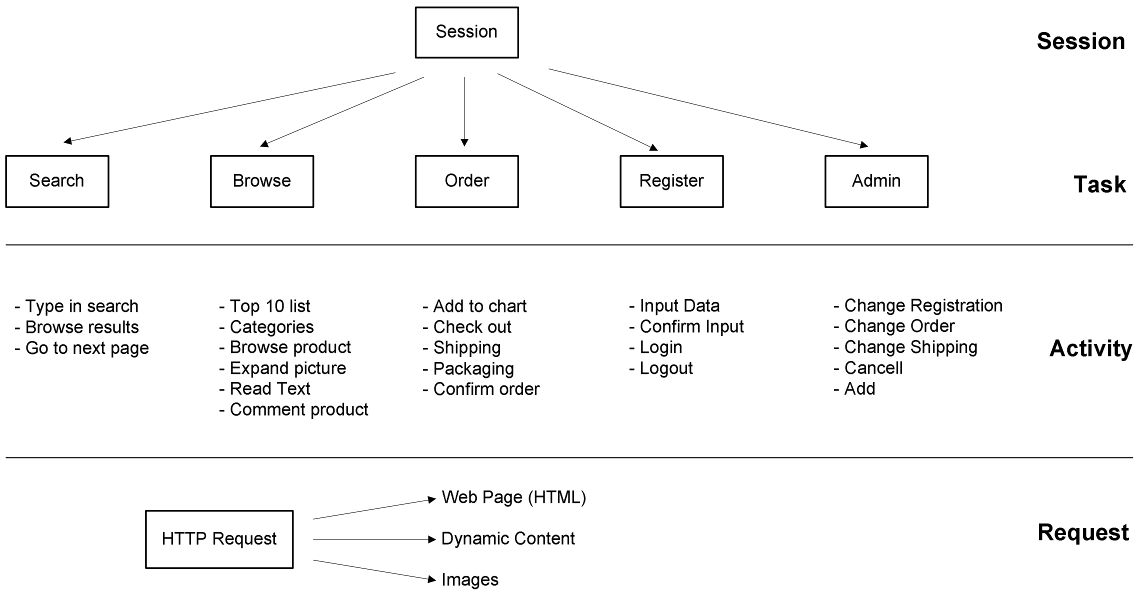

Figure 3: Hierarchical User Behavior Model

5.1 User Session Representation

Identified users sessions can be described using two measurements, the mode probability vector (comparable to customer visit model [6]) and the mode probability matrix (comparable to customer behavior graph [6]). The mode probability vector represents the probability of a user being in a certain state. The next statement shows an example for a mode probability vector:

$$\begin{aligned}
\mathrm{p_k} &= p_k(1),\ldots, p_k(5) \\
\mathrm{p_k} &= (\mathrm{Search, Browse, Order, Register, Admin}) \\
\mathrm{p_k} &= (0.15, 0.55, 0.15, 0.05, 0.05)
\end{aligned}$$

This means the user is with a probability of 0.15 in search state, with 0.55 in browse state, with 0.15 in order state, with 0.05 in register state and with a probability of 0.05 in administrative state.

The mode transition matrix displays the probability of a users transition from a state at time i to a certain state at time i+1. An example for a mode transition matrix is show in Figure 4:

$$p_k = p_k(1,1), \dots p_k(6,6)$$

	$Search_{i+1}$	$Browse_{i+1}$	$Order_{i+1}$	$Register_{i+1}$	$Admin_{i+1}$	Exit
Start	0.2	0.6	0.0	0.1	0.05	0.05
$Search_i$	0.2	0.65	0.05	0.02	0.03	0.05
$Browse_i$	0.1	0.6	0.2	0.05	0.0	0.05
$Order_i$	0.1	0.7	0.1	0.0	0.05	0.05
$Register_i$	0.2	0.78	0.0	0.0	0.0	0.02
$Admin_i$	0.2	0.7	0.0	0.0	0.07	0.03

Figure 4: Example for Mode Transition Matrix

Additionally to the five states derived from the task layer (search, browse, order, register, admin), two states are inserted into the mode transition matrix. Start is the first state a user is in when the session starts. This state can only be reached once, the state Exit means the end of an user session; this state cannot be left.

5.2 Characterizing Parameters

A methodology to describe user behavior at one level is detailed in section 5.1. It deals mostly with representing probabilities to start or schedule a task. This chapter is concerned about the interaction of the four levels in the hierarchical user behavior model where user behavior is broken down all the way to generate http-requests. To be able to characterize user behavior as illustrated in beginning of chapter 5 various data is necessary at each layer.

At the session layer the following information has to be stored:

- Interarrival time of sessions (time between the start of the session and the start of the next session)
- Duration of the session or probability to end the session
- Probability to start a certain task (maybe including the end of the session instead of the last mentioned duration of the session), can depend on the last task

Necessary information at the task layer:

- Probability to start a certain activity, can depend on the last activity
- When to stop that activity (afterwards start the next activity)
- Time the next activity is scheduled for

Information required at the activity layer:

- When to send a request
- Which request to send (html, images, dynamic content)
- Number of requests to send

If there is no data available for the activity layer, the task layer is connected to the request layer via a dummy activity layer. Then for each task it is determined which types of http-requests have to be sent out to the ecommerce servers. At the request layer there is no additional information needed, as requests specified by the activity layer are sent out.

6 DATA ANALYSIS

Chapter 6.1 illustrates the methodology for logfile analysis. It illustrates step by step how hierarchical customer profiles can be obtained from web server logfiles. The second section, chapter 6.2, deals with the important issue of choosing the right timeout when distinguishing between user sessions.

6.1 Methodology

This chapter describes how logged http-requests can be processed to obtain hierarchically structured profiles for user behavior. The following list shows the steps necessary in logfile analysis.

1. Data gathering
2. Parsing and storage
3. Identify user sessions
4. Calculate clustering criteria
5. Clustering
6. Compute hierarchical customer profiles

First the data has to be colleted. Webservers like Apache or Microsoft Internet Information Server provide means to automatically create files containing a line of information for each http-request. If an IT-infrastructure consists of webservers downstream of one or more load balancers the web logs of all servers must be joined. In the next step the information of the logged lines is parsed for separate fields like IP-address or date. For each line in the logfile a record containing the information segmented into the separate fields is created. This record can either be saved in a kind of spreadsheet or in a database. In our case study there are more than 5 million records, thus Microsoft Excel can not be used as a spreadsheet application because of its limitation to 65536 records, but statistical software like SPSS can be employed. For database applications it is no problem storing that many lines. It is also thinkable to work directly with the obtained text files using scripting languages like Perl for further data processing.

When data is available in a structured way, user sessions have to be identified. Therefore data is ordered in a timely manner and user sessions can be identified using either, the User-ID, the client computer name, parts of the client request string – for example there may be an identifier for the user included in the addresses of the web pages he loads – or the client IP-address. It is necessary to considering a reasonable timeout for each distinct session (see chapter 6.2). To all records stored a unique identifier denoting this record is added. If only modeling customers all computed sessions have to be checked whether they might belong to a customer or not. All sessions with source-IP addresses inside the domain of the company can be eliminated. Within our data some of this sessions just check in a periodical way whether the servers are still alive. Other sessions, mostly lasting for a whole working day are established for administrative purposes or represent call center agents entering their orders. Additionally other sessions may be found generated by proxy servers which might be eliminated from the data.

As a next step parameter describing users are created as a basis to divide customers into subgroups. Clustering is based on similarities between user sessions. According to the desired segmentation these sessions can be similar in regard to either:

- session length,
- the amount of data transferred,
- the interarrival time between requests,
- time of the day the session occurred (e.g. all users which surf the net between 8:00 am and 9:00 am),
- which sections of the website are viewed,
- their shopping behavior: Shoppers and non-shopper,
- the country they come from
- and many more.

Based on these criteria different data can be needed. For example one can chose session duration as clustering criteria. Then this parameter is calculated for all sessions. Or customer sessions are clustered according to the amount of data transferred. When clustering is based on customer behavior navigating in the online shop it is necessary to compute the mode probability vector and the mode probability matrix as illustrated in chapter 5.1.

Then the cluster analysis is performed. First hierarchical clustering can be used to get an idea on the appropriate number of clusters. Once the number of clusters is decided, analysis using cluster centers is used to actually cluster the data. Also two-step clustering (SPSS) can be applied. After processing it has to be ensured that clusters differ significantly among each other.

Finally the hierarchical customer profiles can be calculated for each of the distinct groups of customers. First data for the task layer is computed. For this reason the http-requests belonging to a session have to be distinguished in requests concerning browsing, searching, ordering, registration and administration. This differentiation can be done analyzing the recorded http-requests. Depending on the implementation of a website for example the name of the html-files or the directory accessed can give a hint to which task to assign the http-request. When this assignment is done the mode probability vector and the mode transition matrix can be generated. For the activity layer the tasks are broken further down wherever possible. Eventually to each activity a number of http-requests is assigned which represent the request layer. Other parameters necessary to be computed can be found in chapter 5.2.

When benchmarking a server weights for the customer mix or for the activity mix inside one or more of the user groups can be altered to perform benchmark runs with different customer behavior.

6.2 How to define a session?

This chapter deals with the issue of separating user sessions from the stream of http-requests in a logfile. Criteria to identify these sessions have to be established. Generally sessions cannot be distinguished by separating them on the basis of different IP-address requesting a site. Customers with a fixed address can perform several distinct sessions during the measurement period. Or users might connect to the internet using dynamic IP-addresses. Therefore the same IP-address can be used by different users, resulting in distinct user sessions. Thus it has to be defined that after a specified amount of idle-time (timeout) a new user session starts. It is the task of this section of the paper to define a methodology to transparently decide the appropriate timeout using quantitative measures obtained from the logfiles which are to be analyzed using this timeout.

First of all the number of sessions were calculated for various timeouts. It was computed for timeouts from 5 to 60 minutes with 5 minutes interval, for 120 minutes, from 180 to 1440 minutes (one day) with 180 minutes interval. For each timeout classes of session length could be obtained like shown in Figure 5.

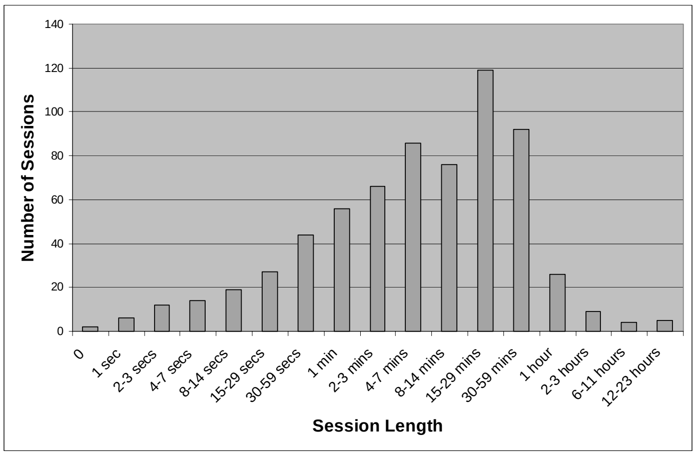

Figure 5: Number of Sessions vs. Session Length for a timeout of 30 minutes

Therefore it was possible to compute the average session length (1):

$$AverageSessionLength = \frac{\sum_{i=1}^{n} SessionCount_i * SessionLength_i}{TotalNumberOfSessions}$$
$$SessionLegth_i = \frac{UpperBound - LowerBound}{2} \qquad (1)$$

For each distinct timeout three parameters are available now, the chosen timeout, the average session length or session duration and the number of sessions. Additionally the mean value and the standard deviation of each parameter were calculated. Afterwards the timeout (2) and the number of sessions (3) are normalized using the following equations:

$$normalized\ Timeout = \frac{Timeout - MeanTimeout}{StandarddeviationTimeout}$$
$$MeanTimeout = \frac{\sum_{i=1}^{n} Timeout_i}{n} \qquad StandarddeviationTimeout = \sqrt{\frac{n\sum_{i=1}^{n} Timeout_i^{\,2}\left(\sum_{i=1}^{n} Timeout_i\right)^2}{n(n-1)}} \qquad (2)$$

$$normalized\ NoSessions = \frac{NoSessions - MeanNoSessions}{StandarddeviationNoSessions}$$
$$MeanNoSessions = \frac{\sum_{i=1}^{n} NoSessions_i}{n} \qquad StandarddeviationNoSessions = \sqrt{\frac{n\sum_{i=1}^{n} NoSessions_i^{\,2}\left(\sum_{i=1}^{n} NoSessions_i\right)^2}{n(n-1)}} \qquad (3)$$

With normalized timeout and the normalized number of sessions the Euclidean distance (4) to the origin of a graph (see Figure 6) is calculated. Finally the timeout with the lowest Euclidean distance is chosen for further analysis.

$$Euclidean\ Distance = \sqrt{(normalizedTimeout)^2 + (normalizedNoSessions)^2} \qquad (4)$$

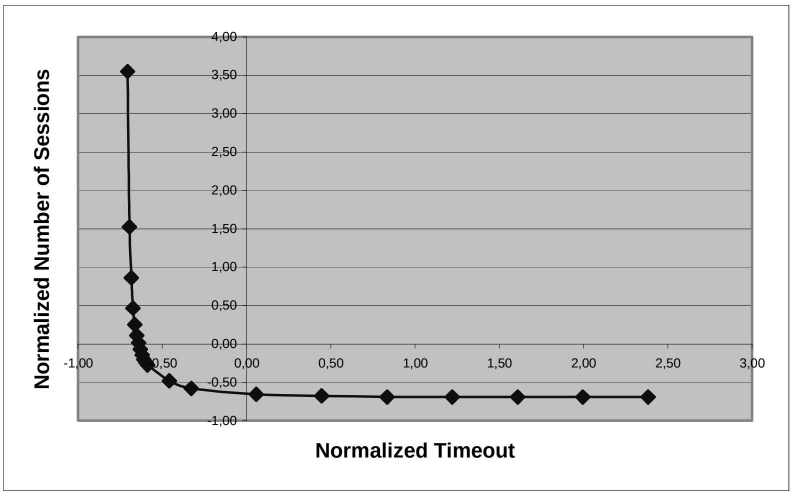

Figure 6: Normalized Number of Session vs. Normalized Timeout for all cases probed

The lowest value was obtained for a timeout of 40 minutes; therefore the timeout for further logfile analysis is set to this value.

7 CONCLUSIONS AND FUTURE WORK

In this paper a methodology dealing with the analysis of web server logfiles is presented (chapter 6.1). These logfiles are processed to obtain hierarchically structured profiles describing customer behavior. The main steps of analysis based on logged http-requests are data gathering, parsing and storage, identification of user sessions, the calculation of clustering criteria, the actual cluster analysis and finally the computation of hierarchical customer profiles. These profiles can then be used to drive the generation of http-requests for the TPC-W benchmark suite. Besides a format for user session description is illustrated (chapter 5.1). Finally the important issue of determining the right timeout when analyzing session inside logfiles is considered (chapter6.2).

The next step will be the detailed analysis of the logged http-requests to obtain mode probability vectors and mode transition matrices for each user session. The user sessions will then be clustered which should result in categories of similar user sessions. Then analysis is done to fill all layers (session-, task-, activity- and request-layer) with data. Eventually the database of clustered user session can be used to measure e-commerce installations depending on the actual mix of user types.

References

[1] A Brown, Gautam Kar, Alexander Keller, “An Active Approach to Characterizing Dynamic Dependencies for Problem Determination in a Distributed Environment”, Seventh IFIP/IEEE International Symposium on Integrated Network Management, Seattle, USA, May 2001

[2] A.K. Jain, R.C. Dubes, “Algorithms for Clustering Data”, Prentice Hall, 1988

[3] Brian S. Everitt, “Cluster Analysis”, Halsted Press, 3rd Edition, 1993

[4] C. Fraley, A. E. Raftery, “How Many Clusters? Which Cluster Method? Answers Via Model-Based Cluster-Analysis”, The Computer Journal, Vol. 41, No. 4, p. 578 - 588, 1998

[5] Christian Kurz, Helmut Hlavacs, Gabriele Kotsis, "Workload Generation by Modelling User Behaviour in an ISP Subnet", Proceedings of the International Symposium on Telecommunications (IST 2001), p. 365 - 368, Teheran, Iran, 2001

[6] Daniel A. Menasce, Virgilio A. F. Almeida, „Scaling for E-Business: Technologies, Models, Performance and Capacity Planning“, Prentice Hall, 2000

[7] Harold W. Cain, Ravi Rajwar, Morris Marden, Mikko H. Lipasti, “An architectural Evaluation of Java TPC-W”, Seventh International Symposium on High-Performance Computer Architecture, January, 2001

[8] Helmut Hlavacs, Ewald Hotop, Gabriele Kotsis, “Workload Generation by Modeling User Behavior”, OPNETWORK 2000, http://www.opnet.com/opnetwork2000/home.html

[9] Helmut Hlavacs, Gabriele Kotsis, “Modeling User Behavior: A Layered Approach”, MASCOTS'99, IEEE Computer Society, Los Alamitos, California, 1999, pp. 218-225.

[10] Johann Bacher, “Clusteranalyse: anwendungorientierte Einführung”, 2. Auflage, Oldenburg, 1996

[11] Kimmo E. E. Raatkainen, „Cluster Analysis and Workload Classification“, Performance Evaluation Review, Vol. 20 # 4, p. 24 – 30, May 1993

[12] M. Calzarossa, G. Haring, G. Kotsis, A. Merlo, D. Tessera, “A hierarchical approach to workload characterization for parallel systems”, B. Hertzberger and G. Serazzi, editors, High Performance Computing and Networking, LNCS vol. 919, pages 102–109. Springer, 1995.

[13] Meikel Poess, Chris Floyd, “New TPC Benchmarks for Decision Support and Web Commerce”, ACM Sigmod Record, Volume 29, No 4, December 2000

[14] Miguel A. Paraz and William Emmanuel S. Yu, “Measuring Performance of IP Networks Using Content Metrics”, Philippine Journal of ICT and Microelectronics, Volume 2, Aug 2002

[15] Qiong Luo, Jeffrey F. Naughton, “Form-Based Proxy Caching for Database-Backed Web Sites”, The VLDB Journal, p. 191 – 200, 2001

[16] S. V. Raghavan, R Kalyanakrishnan, “On the classification of interactive users based on user behaviour indices”, Performance Evaluation Review, Vol.13, No.2, pp 40-48, 1985

[17] Todd Bezenek, Trey Cain, Ross Dickson, Tim Heil, Milo Martin, Collin McCurdy, Ravi Rajwar, Eric Weglarz, Craig Zilles, and Mikko Lipasti, “Characterizing a Java Implementation of TPC-W”, Third Workshop on Computer Architecture Evaluation using Commercial Workloads, January, 2000

[18] Transaction Processing Performance Council, “Top Ten TCP-W Results by Performance”, http://www.tpc.org/tpcw/results/tpcw_perf_results.asp

[19] Transaction Processing Performance Council, “Top Ten TCP-W Results by Price/Performance”, http://www.tpc.org/tpcw/results/tpcw_price_perf_results.asp

[20] Transaction Processing Performance Council, “TPC Benchmark W (Web Commerce) Specification”, Version 1.8, 19th February 2002, http://www.tpc.org/tpcw/spec/tpcw_V1.8.pdf

[21] Transaction Processing Performance Council, TPC, http://www.tpc.org/

[22] Wayne D. Smith, “Benchmarking an Ecommerce Solution”, Revision 1.2

[23] Zona Research, “The Economic Impacts of Unacceptable Web Site Download Speeds”, www.zonaresearch.com/deliverables/white_papers/wp17/index.htm