A Framework for the Evaluation of Session Reconstruction Heuristics in Web Usage Analysis

Myra Spiliopoulou • Bamshad Mobasher • Bettina Berendt • Miki Nakagawa

Department of E-Business, Leipzig Graduate School of Management, Jahnallee 59, D-04109 Leipzig, Germany

School of Computer Science, Telecommunications & Information Systems, DePaul University, 243 South Wabash Avenue, Chicago, Illinois 60604, USA

Institute of Information Systems, Humboldt University Berlin, Spandauer Str. 1, D-10178 Berlin, Germany

School of Computer Science, Telecommunications & Information Systems, DePaul University, 243 South Wabash Avenue, Chicago, Illinois 60604, USA

myra@ebusiness.hhl.de • mobasher@cs.depaul.edu • berendt@wiwi.hu-berlin.de • miki@cs.depaul.edu

---

Web usage mining has become the subject of intensive research, as its potential for personalized services, adaptive Web sites and customer profiling is recognized. However, the reliability of Web usage mining results depends heavily on the proper preparation of the input datasets. In particular, errors in the reconstruction of sessions and incomplete tracing of users' activities in a site can easily result in invalid patterns and wrong conclusions. In this study, we evaluate the performance of heuristics employed to reconstruct sessions from the server log data. Such heuristics are called to partition activities first by user and then by visit of the user in the site, where user identification mechanisms, such as cookies, may or may not be available. We propose a set of performance measures that are sensitive to two types of reconstruction errors and appropriate for different applications in knowledge discovery (KDD) applications.

We have tested our framework on the Web server data of a frame-based Web site. The first experiment concerned a specific KDD application and has shown the sensitivity of the heuristics to particularities of the site's structure and traffic. The second experiment is not bound to a specific application but rather compares the performance of the heuristics for different measures and thus for different application types. Our results show that there is no single best heuristic, but our measures help the analyst in the selection of the heuristic best suited for the application at hand.

1. Introduction

The discovery of Web usage patterns is an essential prerequisite for the personalization of Web services, the tuning of Web services, the presentation of promotional contents, the launching of e-marketing campaigns, and in short, for all applications in which user interests, preferences, requirements, and behavioral conventions must be assessed and served.

Web usage analysis is a subdomain of knowledge discovery. It employs data mining techniques to extract patterns from data. These data are Web activity logs, often enriched with external information such as customer transactions from a pay-bill department, customer records from a customer care division or demographics from an external third party. The reliability of data mining results depends on the quality of these data, which may be noisy, erroneous or incomplete.

Web server logs are the primary source of data in which the activities of Web users are captured. However, due to constraints of the HTTP specification, these records are incomplete, in the sense that the requests they contain cannot be uniquely assigned to the individual that has performed them. This ambiguity has led to several "proactive" and "reactive" strategies. Proactive strategies aim at an unambiguous association of each request with an individual *before* or *during* the individual's interaction with the Web site. Reactive strategies attempt to associate requests to individuals *after* the interaction with the Web site, based on the existing, incomplete records.

Proactive strategies include user authentication, the activation of cookies that are installed at the user's work area and can thus attach a unique identifier to her requests, as well as the replacement of a site's static pages with dynamically generated pages that are uniquely associated with the browser that invokes them. The following characteristics are pertinent to proactive strategies:

- They raise privacy considerations among site users, and may thus have a negative impact on the contact efficiency of the Web site.

- They presume that Web site owners understand the importance of good quality data for their business, can quantify and compare the negative and the positive potential

impact of proactive strategies in their business domain, and are willing to perform the investment needed to implement these strategies.

Reactive strategies exploit background knowledge on user navigational behavior to assess whether requests registered by the Web server can belong to the same individual, and whether these requests were performed during the same or subsequent visits of the individual to the site. It is generally believed that their results are of lower accuracy than those of the proactive strategies. It is also believed that proactive strategies are error-proof, except for some statistically negligible number of cases. However, the validity of such beliefs, which is in essence the accuracy of proactive and reactive strategies, has not been quantified to date.

The absence of well-founded accuracy results is problematic from a business perspective: Should a business venue in a privacy-sensitive market segment (as are many customer segments in Europe) opt for a proactive strategy, or would a good reactive strategy incur only a minor quality loss? Are all proactive strategies equally reliable? If yes, why are some sites operating multiple proactive strategies, some of which require non-minor investments? If not, what is the best strategy or the best combination? And what is the accuracy loss when opting for the second-best one?

In this study, we introduce a framework for answering such questions. Our framework is designed for the comparison of proactive and reactive strategies that (re)construct the activities of individual users from the stream of requests accessing a site and assign them into sessions, i.e. into separate visits to the site. Our framework incorporates a set of measures for the comparison of session reconstruction results, a test environment where alternative strategies have been implemented, and an experimental setting for our comparisons.

In the next section, we discuss related research. In Section 3, we give a detailed description of proactive and reactive strategies for session reconstruction, and we elaborate on the particularities in their behavior. In Section 4, we measure the impact of reactive strategies on the reconstruction of basic session features, and we give an example of how reconstruction errors can affect mining results. Section 5 extends this by introducing a set of measures for the performance of the heuristics. In Section 6, we describe our first experimental results with the proposed set of measures, interpret the performance differences among the investigated reactive strategies, and use these results to identify appropriate application areas for each strategy. Section 7 concludes our study.

2. Related Work

Studies related to our work come from two areas: data preparation and evaluation methodologies for data mining.

Data preparation. In the process of knowledge discovery from data, the data preparation phase precedes data analysis. As pointed out by Pyle (1999), this phase is critical for the success of pattern discovery: errors committed in this phase may make the data useless for further analysis. In this book, Pyle describes the many tasks of data preparation, including both general-purpose tasks and specific activities imposed by the type of data analysis to follow (Pyle 1999). The most common generic tasks are the detection and repair of erroneous data and the treatment of missing values. For example, a negative age value should be corrected, and a missing birth date should be filled with a reasonable one, e.g. with the population average or median. While such modifications would be unacceptable for operative tasks, they are necessary for the statistical evaluation of the dataset. In the latter, the concrete values of individual records are not critical, as long as they do not distort the distribution of the whole population.

Data preparation tasks depend on both the nature of the data and the type of analysis. For example, a continuous variable like "income" can be used as classification variable by a neural network, but should be mapped into a set of discrete values if decision trees are used instead. Conversely, the values of a discrete variable such as postal code must be mapped into reasonable values in the $[0, 1]$ interval before a neural network can process them (Hand et al. 2001).

In Web usage analysis, data preparation includes similar tasks of repairing erroneous data and treating missing values (cf. Cooley et al. 1999, 2000, Spiliopoulou and Faulstich 1999). Requests that do not reflect human navigation behavior, i.e. requests from robots and other software agents, have to be removed (Tan and Kumar 2002). An additional data preparation task, which is peculiar to the nature of Web usage data, is the faithful reconstruction of the activities of each individual user during each visit to the Web site. This is no trivial task, because the data recorded by a Web server are not sufficient for distinguishing among different users and for distinguishing among multiple visits of the same person. For the achievement of this task, there are "proactive strategies" that guarantee a faithful reconstruction *during usage recording*, and "reactive strategies" that attempt a

faithful reconstruction *after usage recording*. Since the evaluation of such strategies is the objective of our study, we provide a thorough discussion of individual strategies of each type in the next section.

Research on evaluation methods. Although data preparation is essential for knowledge discovery, studies on the evaluation of data preparation methods are comparatively scarce. In Pyle (1999), the major importance of data preparation for the success of data mining is stressed and examples on the pitfalls of poor data preparation are discussed. However, a comparison of alternative data preparation methods is beyond the scope of Pyle's book.

Padmanabhan, Zheng, and Kimbrough have studied the impact of data preparation alternatives upon Web usage mining in Padmanabhan et al. (2001) and in Zheng et al. (2003). In Padmanabhan et al. (2001), they focus on the prediction of purchase for users visiting multiple sites. Best predictive accuracy is achieved when all records of all visited sites are available. The authors show that when the analysis is based on the activities inside one site only, the accuracy of the predictors drops significantly.

Zheng et al. (2003) compare a set of methods for purchase prediction, each of which exploits different components of the users' sessions on which to make predictions. They compute the prediction accuracy of these methods using several classifiers. They show that these methods perform very differently, to the extent that one of them is unable to predict purchases while another produces patterns of high quality.

In our study, we compare data preparation strategies whose goal is the reconstruction of the sessions, upon which preparation methods for prediction, as in Padmanabhan et al. (2001) and Zheng et al. (2003) are applied. Differently from these authors, we do not evaluate the performance of these strategies for a specific KDD application. Rather, we provide measures for performance scoring, we propose a framework for comparing the strategies, and we identify types of KDD applications for which each strategy and each of our proposed measures are most appropriate.

3. Heuristic Methods for Session Reconstruction

Heuristic methods for session reconstruction must fulfill two tasks: First, all activities performed by the same physical person should be grouped together. Second, all activities belonging to the same visit should be placed into the same group. Knowledge about a user's

identity is not necessary to fulfill these tasks. However, a mechanism for distinguishing among different users is indeed needed.

In accordance with W3C (W3C Web Usage Characterization Activity 1999), we term as *(server) session* or *visit* the group of activities performed by a user from the moment she enters the site to the moment she leaves it. Since a user may visit a site more than once, the Web server log records multiple sessions for each user. We use the name *user activity log* for the sequence of logged activities belonging to the same user. Thus, *sessionizing* is the process of segmenting the user activity log of each user into sessions. A *sessionization heuristic* is a method for performing such a segmentation on the basis of assumption about users' behavior or the site characteristics.

We will first describe sessions and their reconstructions formally, and then go on to describe and formalize strategies for session reconstruction.

3.1 Sessions in a Web Server Log

Let U be the set of possible page requests in the Web site. This set consists of all static URLs and all pages that can be generated by underlining scripts and database invocations with different input settings.

The Web server log L is a list of actual page requests from U. L can be regarded as a set. However, its records are ordered by timestamp of the invocation. We exploit this ordering and refer to the j^{th} entry in the log, denoted as $L[j]$. In conformance with the widespread Web server log format specifications described in detail in Section 3.1.2 below, we use $l.url$ to denote the URL requested in the entry $l \in L$, and use self-explanatory names for all data fields recorded in the log.

3.1.1 The Notion of "Real Session"

Each user visits the Web site one or more times. The sequence of page requests she performs during one visit constitutes a "real session." Thus, a real session r is a list of elements of L. We denote by $\mathcal{R}$ the set of real sessions constituting L. We impose an arbitrary order over the elements of $\mathcal{R}$, so that we can refer to the i^{th} real session $\mathcal{R}[i] \in \mathcal{R}$. The following properties hold:

(1) The elements of a real session, i.e. the requests for objects in the site, are ordered by

timestamp. This means that sessions preserve the order of requests:

$$\forall \mathcal{R}[i] \in \mathcal{R}, \forall k = 2, \ldots, length(\mathcal{R}[i]) : \mathcal{R}[i][k].timestamp > \mathcal{R}[i][k-1].timestamp \ .$$

where $length(\cdot)$ denotes the number of elements of a list.

(2) All requests in L and only these appear in the sessions of $\mathcal{R}$:

$$\cup_{\mathcal{R}[i] \in \mathcal{R}} \left(\cup_{k=1}^{k=length\mathcal{R}[i]} \mathcal{R}[i][k] \right) = L \ .$$

(3) Each request in L belongs to exactly one session of $\mathcal{R}$:

$$\forall \mathcal{R}[i] \in \mathcal{R}, \forall k = 1, \ldots, length(\mathcal{R}[i]) : \nexists i' \neq i, k' : \mathcal{R}[i][k] = \mathcal{R}[i'][k'] \ .$$

The above properties ensure that $\mathcal{R}$ partitions L in an order-preserving way.

3.1.2 Basic Entities for Session Reconstruction, and the Notion of "Constructed Session"

The base information for session reconstruction are the Web server records on the users' activities. The most widespread formats for HTTP servers are the W3C common and extended log file formats. In the common log file format (cf. http://www.w3.org/Daemon/-User/Config/Logging.html), the only recorded data related to a user as a person are the IP address or DNS hostname of the user's host or proxy. The extended log file format (see http://iishelp.web.cern.ch/IISHelp/iis/htm/core/iiintlg.htm) also allows the recording of the user's software agent (browser or batch client) that performs the requests on her behalf, as well as the "referrer" URL, i.e. the page from which a request was initiated. None of these data items is sufficient to distinguish among different persons, because several persons may access a site through the same host or proxy and may employ the same software agent to this purpose.

A sessionizing heuristic h attempts to assign all activities performed during a user visit to the site together. The result is a set of "constructed sessions" $\mathcal{C} \equiv \mathcal{C}_h$. We impose an order on $\mathcal{C}$, so that we can refer to the i^{th} constructed session as generated by the heuristic h.

The set of constructed sessions produced by heuristic h, $\mathcal{C}_h$, is "valid" if and only if it exhibits the same properties as the set of real sessions $\mathcal{R}$, namely: (1) the elements of each constructed session are ordered by timestamp, (2) the union of these elements is L,

and (3) their intersection is the empty set. This means that h partitions L into a set of constructed sessions, while respecting the timestamp ordering. In the following, we consider only heuristics that produce valid sets of constructed sessions.

3.2 Mapping Activities to Users

The extended log file format allows for a "cookie identifier" to be recorded. A cookie is a small piece of code associated with a Web site; it installs itself on the user's host and associates a cookie identifier with the user's browser. Whenever the user requests a page from the Web server, the cookie identifier is attached to the request and returned to the server. Thus, under this format, each request can be assigned reliably to its initiator.

In the sense of our work, the cookie-based mapping of activities to users is a "data preparation strategy." It is a "proactive" strategy because the reliability of the mapping is guaranteed *while* (or actually even before) the user accesses the site. In contrast, a "reactive strategy" would attempt to establish such a mapping from the server's Web log *after* the user has accessed the site. If the Web server does not employ cookie-based identification or a user authentication mechanism, then only reactive strategies can be used to reconstruct the activities of the users.

Cookie-based identification of users is not always a feasible option. First, cookie-based identification of users presupposes an appropriate configuration of the Web server. Data gathered prior to this configuration should still be evaluated. Secondly, cookies can be turned off by users due to privacy concerns. Some aspects of how cookies are commonly employed appear to be incompatible with the principles of European data protection legislation (Mayer-Schönberger 1997), so that after recent EU debates it has again been demanded that in order for cookie use to be compliant with existing laws, at least Internet users must be clearly made aware of cookies' presence in electronic systems (Reuters 2001). Restrictions on the use of cookies in the European Union will affect Web site owners outside the European Union as well, because transactions involving partners/users inside the European Union require compliance with the European data protection principles under the Safe Harbor agreement (http://europa.eu.int/eur-lex/en/consleg/pdf/2000/en_2000D0520_do_001.pdf).

From the viewpoint of the site owner, the absence of a proactive data preparation strategy implies the need for a reactive strategy alternative. Reactive strategies do not map activities to users: rather, they focus on identifying the requests performed during a single visit of a

user to a site. Recognizing the user when she comes again is beyond their scope. This will be discussed in Section 3.4 below.

3.3 Proactive Mapping of Activities to Sessions

A cookie identifier distinguishes among users that share the same environment or pool of dynamically assigned IP addresses. However, it cannot detect that the user has left the site: as far as a cookie is concerned, all visits performed by a user to a site are equivalent to a single long visit during the same period. For several data analysis applications, this simplification is not acceptable: frequent visitors become more experienced with a site and its content, and they should be treated differently from newcomers.

A "session identification mechanism" is a proactive strategy that maps user activities during one visit into a *session*. It is usually implemented as a Web server software option that associates a unique identifier to each client process accessing the server. This identifier is attached to each request made by the user's client to the server, thus allowing for the unique assignment of requests to users during one visit. The identifier expires when the user's client process is terminated, when the connection is broken, or when a timeout occurs. Its expiration determines the end of the session.

Session (re)construction does not only involve the correct assignment of activities performed during the same visit: a session should contain all these activities, and only these. Completeness may be compromised due to caching. In particular, when a user requests an object already accessed in the past, the browser may decide to load it from its local cache instead of accessing the remote server. This implies that repeated requests are not registered by the server. For proactive strategies, this is simply a loss of information. For reactive strategies that exploit knowledge about previous requests in order to assign requests to a user or session, caching may lead to erroneous assignments of requests. To avoid this problem, some servers implement cache-prevention methods, i.e. prevent caching with a proactive strategy. Cooley et al. (1999) proposed a reactive session reconstruction strategy that partially alleviates the caching problem by exploiting site topology.

3.4 Reactive Mapping of Activities to Sessions

A reactive strategy takes as input a Web server log and partitions it into a set of "constructed sessions," i.e. into groups of requests constituting different user visits. In the evaluation

model presented in Section 5, we will juxtapose these constructed sessions to the "real ones," as provided by a reference model.

In this study, we consider three reactive strategies. In particular, we describe two *time oriented heuristics* that rely on the temporal properties of the user activity log, and one *navigation oriented heuristic* that derives its sessionizing rules from assumptions on the way users navigate.

3.4.1 Time Oriented Heuristics

Time oriented heuristics use an upper bound on the time spent in the entire site during a visit or an upper bound on page-stay time.

To approximate the total time spent inside a site, Catledge and Pitkow (1995) measured mean inactivity time within a site, and came to a value of 9.3 minutes. By adding 1.5 standard deviations, they derived a 25.5 minute cutoff for the duration of a visit. This has been rounded to 30 minutes and is used in many applications as a rule of thumb for maximal session length (Cooley et al. 1999, Spiliopoulou and Faulstich 1999).

During a visit, the time spent by a user to read and process the contents of any single page varies within certain limits. If a long time elapses between one request and the next, it is likely that the latter request is the first of a new visit. This observation led to a second type of time oriented heuristics, which use a threshold on the total page-stay time. Unlike the total session duration cutoff of 30 minutes, there is no widespread threshold for page-stay time. This is a reasonable implication of the fact that page-stay time is affected by the information content of a page, by the time needed to load the components of the page and by the transfer speed of the communication line. Heuristics of this type are used in Cooley et al. (1999) and Spiliopoulou and Faulstich (1999).

3.4.2 Navigation Oriented Heuristics

Navigation oriented heuristics do not consider the time a user spends on a page or inside a site but rather exploit behavioral habits associated with Web navigation. Concretely, Web users reach pages by following hyperlinks rather than by typing URLs. Hence, a request for a page that is unreachable through the pages visited by the user thus far is likely to have been initiated by another user. Navigation oriented heuristics elaborate on this observation to partition a Web server log into sessions.

According to a navigation-based heuristic proposed by Cooley et al. (1999, 2000), a requested Web page P that is not reachable from previously visited pages, should be assigned to a different session. This heuristic also accounts for the fact that P need not be accessible from the page immediately accessed before it. Rather, the user may backtrack to a page visited earlier, from which P is reachable. These backward moves are not always registered by the Web server, because the pages to which they refer can be already available in the client's cache. In that case, the heuristic reconstructs the shortest path of backward moves leading to P and adds it to the user's session (Cooley et al. 1999, 2000).

The previous heuristic uses the topology of the Web site graph. Consultation of the site topology presupposes the availability of the site's graph in an appropriate format. A less demanding heuristic proposed in Cooley et al. (1999) is based on the referrer information recorded in the Web server log according to the extended log file format. The "referrer" of a URL request is the page from which the request was issued. This heuristic states that the referrer of a requested page P should be a page already in the session; otherwise P is assigned to a different session (Cooley et al. 1999). If the page has an empty referrer, then it is likely to be the first page of a new session.

Referrer-based heuristics are more restrictive than the topology-based heuristic above, because there are cases where a page request has an empty referrer although it has been reached via a hyperlink from some page already in the session. In general, an empty (also called undefined) referrer may appear inside a session for several reasons, namely (1) in the unlikely case that the user has typed the URL, (2) if the requests are performed by a robot, because robots do not follow hyperlinks, or (3) for some of the frames belonging to the same page. In Berendt et al. (2001) we propose a referrer-based heuristic that alleviates this problem by including to the same session all consecutive requests that have an empty referrer but are invoked within a small time interval Δ.

These heuristics are applicable across a wide range of site types. In particular, sites with dynamically generated pages lend themselves to sessionizing. All the heuristics mentioned above can be applied to sequences of such pages, because at least the URL stem is recorded in the logs like any other URL. In fact, sessionizing will often be easier for sites with dynamic pages, because these sites are less affected by caching.

In addition, application-specific versions of referrer heuristics become possible if the site's log file can record the parameters that control the generated pages in the URL query strings. For example, two consecutive URLs can be assigned to the same session if the second query

string contains all parameters of the first plus some more, thereby refining the first. Such a heuristic has been proposed in Berendt and Spiliopoulou (2000).

3.5 A Selection of Sessionization Heuristics for Evaluation

In this study, we evaluate the performance of two time oriented heuristics and of the extended referrer heuristic described in Berendt et al. (2001). The concrete specifications of these reactive strategies are as follows:

h1: Time oriented heuristic: The duration of a session must not exceed a threshold θ. Let t_0 be the timestamp of the first URL request in a constructed session. A URL request with timestamp t is assigned to this session iff $t - t_0 \leq \theta$. The first URL request with timestamp larger than $t_0 + \theta$ becomes the first of the next constructed session.

h2: Time oriented heuristic: The time spent on a page must not exceed a threshold δ. Let t' be the timestamp of the URL most recently assigned to a constructed session. The next URL request belongs to the same session iff for its timestamp t'' it holds that $t'' - t' \leq \delta$. Otherwise, this URL becomes the first of the next constructed session.

h-ref: Referrer-based heuristic: Let p and q be two consecutive page requests with p belonging to a session S. Let t_p and t_q denote the timestamps for p and q, respectively. Then q will be added to S if the referrer for q was previously invoked within S, or if the referrer is undefined and $(t_q - t_p) \leq \Delta$, for a specified time delay Δ. Otherwise, q is added to a new constructed session.

We have considered these three heuristics in two settings.

The first setting is a combination with a cookie-based mechanism for user identification ("**cookie**" in Table 1). As discussed in sections 3.2 and 3.3, a cookie provides a proactive strategy for mapping activities to users, but it does not provide a means for mapping activities to sessions. In the absence of session identification mechanisms, this latter task needs to be performed by a reactive strategy. However, the heuristics **h1, h2,** and **h-ref** that we use as reactive strategies need only identify the session borders within the activities of each user.

In the second setting, requests are only identified by the requesting host's **IP** address and the user **a**gent employed ("**ipa**" in Table 1); no cookie information was used to distinguish

between different users. In this setting, the reactive strategies must perform two tasks: They must identify session borders, and at the same time assign temporally interleaved activities to different users.

The three heuristics **h1, h2,** and **h-ref** were tested in the two settings **cookie** and **ipa**, with different values for the heuristics' parameters θ, δ, and Δ. In the remainder of the paper, we concentrate on results from experiments using the parameter settings described in Table 1.

Table 1: The Six Heuristics Used in the Experiments: Underlying Reactive Sessionization Strategy (**h1, h2, h-ref**), Experimental Setting (**cookie** or **ipa**), and Temporal Threshold Values

Acronym of heuristic	Description	Cutoff threshold value	CookieID available?
h1-30-cookie	total session duration	$\theta = 30$ min	YES
h1-30-ipa	total session duration	$\theta = 30$ min	NO
h2-10-cookie	page-stay duration	$\delta = 10$ min	YES
h2-10-ipa	page-stay duration	$\delta = 10$ min	NO
h-ref-cookie	referrer heuristic	$\Delta = 10$ sec for frameset loading	YES
h-ref-ipa	referrer heuristic	$\Delta = 10$ sec for frameset loading	NO

The 30-minute threshold for total session duration is a value derived from empirical findings, as described in Section 3.4.1 above. The 10-minute threshold for page-stay time is a very conservative maximum cutoff, intended to capture the time for loading and studying the contents of a page. The 10-second cutoff used by the referrer heuristic also allows for overhead associated with loading all pages within a frameset.

Since the purpose of our experiments is an evaluation of the comparative performance of the sessionization heuristics, the selection of the actual cutoff values for individual heuristics is of less importance. However, independent experiments (Berendt et al. 2001) clearly indicate that the two time-based heuristics are quite robust with respect to variations of the threshold parameters. One exception to this observation is that the **h2** heuristic shows a degradation in performance if the page-stay time is overestimated considerably, i.e. if it becomes more than 30 minutes.

4. Evaluating Session Reconstruction in Basic Usage Analysis

The heuristics of the reactive strategies discussed in the previous section are based on various assumptions about user behavior: the thresholds considered by the time-based heuristics are conservative approximations of the maximum time a user spends in a site or on one page of a site. On the other hand, the referrer heuristic exploits the basic principle of navigation, namely hyperlink traversal instead of URL typing.

Therefore, before establishing our evaluation framework, we study the basic statistics of the sessions built by each heuristic. Moreover, we investigate the performance of the heuristics for a simple application, namely the detection of frequent entry and exit pages. This application is often the first step of Web site analysis, since it gives insights to which Web pages are first sighted and at which pages the users abandon the site.

4.1 Session Splitting in a Test Web Site

For our experiments, we have applied the sessionizing heuristics to the server log of a university site. The site is frame-based, thus posing a challenge at least for the referrer-heuristic. Caching was disabled during the recording of the log data for our experiment. The log consists of 174663 requests, recorded between the 18^{th} and 29^{th} of November, 2000.

The sessions of the Web site are constructed by the combination of two proactive strategies: First, the site employs cookies to distinguish among users. Second, the Web server assigns a session identifier to each browser process of a user. This session identifier is valid for the lifetime of the browser process, subject to a conservative timeout. This is the best approximation of a session's end page that can be achieved without cooperation with other Web servers. Hence, in our experiments, we assume that the set of sessions thus constructed is identical to the set of real sessions $\mathcal{R}$, and term them as "real sessions" hereafter.

During the experimentation period, 13829 real sessions were recorded, giving the following statistics:

Table 2: Basic Statistics of Real Sessions

	Session size (in pages)	Session duration	Page-stay time
Average value	12.63	31 min 56 sec	2 min 12 sec
Median value	7.00	1 min 39 sec	0 min 12 sec

As can be seen from the difference between average and median values, there is a considerable skew caused by a small number of very long sessions.

4.2 Reconstructed Sessions in the Test Web Site

For our experiments with the reactive strategies, we have retained the cookie identifier and removed the session identifier from the Web server log. Thus, each sequence of activities was uniquely associated with a user, but the beginning and the end of each session inside the sequence were not known. Then, we used for the heuristics the parameter settings shown in Table 1.

In Table 3, we show the number of sessions built by each of the three heuristics and the base statistics derived from them. As can be seen, there are considerable differences among the heuristics, whereby the **h-ref-cookie** heuristic produced an exorbitantly large number of sessions.

Table 3: Statistics of Real and Reconstructed Sessions

	Real sessions	**h1-30-cookie**	**h2-10-cookie**	**h-ref-cookie**
Number of sessions	13829	14234	15971	29195
Average session duration	31:56	7:05	3:49	35:15
Median session duration – both (min:sec) –	1:39	2:59	1:35	0:36
Average page-stay time	2:12	0:44	0:18	4:55
Median page-stay time – both (min:sec) –	0:12	0:16	0:11	0:07
Average session size	12.63	12.18	10.85	5.53
Median session size – both (no. of pages) –	7.00	8.00	7.00	3.00

The first row in Table 3 shows the number of sessions produced by the proactive strategy (real sessions) and the corresponding numbers of reconstructed sessions per heuristic. All reactive strategies generated more sessions than were there, which leads to sessions that are, on average, shorter than the real sessions. We use the term *undersizing* for this phenomenon, and the term *oversizing* for its opposite.

The undersizing trend is best illustrated by the average session-size values. While the average session size of the two time-based heuristics was close to the real one, the average for the referrer heuristic was much lower. Indeed, the referrer heuristic produced more than twice as many sessions than really existed. It seems that the referrer heuristic split almost

every real session erroneously. However, this is not the case: as can be seen by the average session durations of the heuristics, **h-ref-cookie** has the values that are closest to the real average.

The behavior of the referrer heuristic can be interpreted if we consider the average and median values. Average values are much higher than medians in this dataset. Since the referrer heuristic has a very high average session duration, it has built some very long sessions. At the same time, it has also split many sessions erroneously: if one long session is partitioned into a large set of very short ones, the average session duration may be less affected than the total number of sessions.

The undersizing behavior of the referrer heuristic becomes more visible when studying the number of sessions constructed for small session sizes, as shown in Table 4. A comparison of the number of real sessions with one page and the number of one-page sessions built by the referrer heuristic shows that more than 5000 pages originally belonging to a larger session were treated as self-standing sessions. The values for sessions with two or three pages are similar.

Table 4: Real and Reconstructed Sessions With One to Three Pages

	Real sessions	**h1-30-cookie**	**h2-10-cookie**	**h-ref-cookie**
Sessions with 1 page	401	482	645	5789
Sessions with 2 pages	277	372	531	3076
Sessions with 3 pages	3192	2130	3743	7400

The values in Table 4 show that the time-based heuristics performed some undersizing of the real sessions but kept a close approximation of the original session structures, while the referrer heuristic's assignment of pages to sessions does not reflect the original distribution.

In the following, we further investigate simple measures of reconstruction quality, namely the identification of entry and exit pages.

4.3 Classification of Entry and Exit Pages

Our second study concerned the impact of the reactive strategies in the characterization of a session page as entry or exit page. This characterization is very important for customer-oriented services: the entry page is the first one a user sees when coming to the site; its quality determines whether the user will visit further pages. The exit page is one at which the user abandoned the site; if it is desirable that the user does not leave the site at this

page, then page redesign is necessary. Obviously, a misclassification of entry and exit pages is undesirable, because it leads to misinterpretation of user behavior and to a non-rewarding redesign effort.

To evaluate the performance of the three reactive strategies, we use the measures of *precision* and *recall* in a way similar to the evaluation of conventional classifiers. In particular, let E be the set of entry pages in the real sessions and let E_h be the set of pages characterized as entry pages by the heuristic h, where h is one of **h1-30-cookie**, **h2-10-cookie**, and **h-ref-cookie**. Then, the *precision* of h is the ratio of pages correctly classified as entry pages to all pages characterized by the heuristic as such:

$$precision(h) = \frac{|E \cap E_h|}{|E_h|} ,$$

while *recall* is the ratio of correctly classified entry pages to all real entry pages:

$$recall(h) = \frac{|E \cap E_h|}{|E|} .$$

For exit pages, precision and recall are defined in the same way. In Table 5, we show the precision and recall scores for the three heuristics.

Table 5: Identification of Entry and Exit Pages

	h1-30-cookie	**h2-10-cookie**	**h-ref-cookie**
Entry page precision	0.9168	0.8650	0.4734
Entry page recall	0.9437	0.9991	0.9996
Exit page precision	0.9363	0.8576	0.4708
Exit page recall	0.9638	0.9905	0.9940

According to the results in Table 5, the time-based heuristic on total session duration achieves a precision of more than 90%, while the heuristic on page-stay time has a slightly lower precision. The precision of the referrer heuristic is less than 50%. The reason for the poor performance is the large number of sessions constructed by this heuristic, which turned almost all pages in the site into entry pages. For the same reason, the recall of the referrer heuristic is high: since almost all pages are entry (respectively exit) pages of some session, chances are that the pages actually having this property are indeed among them.

Not all entry and exit pages are of equal importance, though. In our example log, a single page serves as an entry for 96.28% of the real sessions. Thus, it is important that the reactive strategies identify the pages that are most frequently accessed as entry (respectively exit) pages. To this purpose, we adjust the aforementioned notions of precision and recall to reflect access frequencies as follows:

For a page p, let $entry(p)$ be the number of real sessions having p as their entry page and let $exit(p)$ be the number of real sessions ending at p. Analogously, $entry_h(p)$ denotes the number of constructed sessions starting at p for h being one of the reactive strategies, and $exit_h(p)$ is defined accordingly.

The number of correct classifications of p by h is the minimum of $entry(p), entry_h(p)$. The precision of h with respect to p is the ratio $\frac{min\{entry(p),entry_h(p)\}}{entry_h(p)}$ and the recall is the ratio $\frac{min\{entry(p),entry_h(p)\}}{entry(p)}$.

Using this approach, we compute the precision of heuristic h with respect to a set of pages P as

$$precision(h,P) = \sum_{p \in P} \frac{min\{entry(p), entry_h(p)\}}{entry_h(p)} \ .$$

The recall of a heuristic h is computed accordingly as

$$recall(h,P) = \sum_{p \in P} \frac{min\{entry(p), entry_h(p)\}}{entry(p)} \ .$$

However, the results of Table 5 indicate that the recall measure is not indicative of the performance of the heuristics and is not used further.

To reflect the performance of the heuristics for the identification of the most frequent entry and exit pages, we have computed their precision values for the top 10 and for the top 20 entry and exit pages. These results are shown in Table 6. In this table, the first column reflects the share of the top 10 (respectively top 20) pages among all the sessions. For example, the value of 96.28% in the first row means that more than 95% of the sessions start with an entry page belonging to those top 10.

Table 6: Identification of the Most Frequent Entry and Exit Pages

	Share	**h1-30-cookie**	**h2-10-cookie**	**h-ref-cookie**
Top 10 entry pages	96.28%	0.8	0.7	0.6
Top 20 entry pages	97.23%	0.85	0.8	0.7
Top 10 exit pages	50.85%	1	1	0.7
Top 20 exit pages	56.81%	1	1	0.8

The findings of Table 6 draw a different picture of the performance of the heuristics. The precision quotes achieved by the time-based heuristics for the top 10 and top 20 pages are lower than the values reached when classifying all entry pages. On the other hand, the precision scores for the exit pages became 100%. This is very important, because (1) pages

causing the abandonment of the site are likely to need improvement and (2) there is much more diversity among the most frequent exit pages than among the most frequent entry pages, thus making the need for correct guesses more urgent.

The performance of the referrer heuristic has improved in comparison to the results of Table 5. Although its precision scores are lower than those of the other two heuristics, they are much better than the 50% borderline.

In summary, our first experiment showed that some of the features of real sessions, like skew in the distribution of session duration values, have significant impact on the performance of the reactive strategies. However, the second experiment signals that the quality of each heuristic depends on the knowledge discovery application that deploys it. Even in the simple application of identifying the site's exit pages, the precision scores differ significantly between the rather simplistic approach of finding *all* exit pages and the more reasonable one of detecting the most frequent exit pages only.

Differently from the studies of Padmanabhan et al. (2001), we do not concentrate on a specific group of KDD applications but rather attempt to quantify the appropriateness of the heuristics for different application types. Therefore, in the next section we propose a set of evaluation measures intended for KDD applications with different demands.

5. Measures for Session Reconstruction Quality

The experiments of the previous section demonstrated the impact of reconstruction heuristics on the reliable classification of entry and exit pages of a Web site. Most Web usage analysis tasks have more complicated objectives, though, including the prediction of the users' next activities and the correlation between navigation patterns and desirable or undesirable activities. For such tasks, the identification of the entry and exit page of each session is not sufficient; session contents should be reliably reconstructed. In this section, we propose a set of measures on the quality of session reconstruction.

Intuitively, the perfect heuristic would reconstruct all sessions by placing all activities of each user during each visit into the same session, omitting no activity and assigning no activity to the wrong session. As described in Section 3, user identification data, end-of-session notifying mechanisms, and cache-prevention policies are not always in place. The reactive heuristics described in Section 3 produce approximations of the "real sessions." In such an approximation, some activities of a session may be missing while others are

erroneously assigned. Thus, it is necessary to quantify the performance of each heuristic with respect to the quality of *all* sessions it builds.

5.1 Sources of Session Reconstruction Errors

The goal of a sessionizing heuristic is the establishment of a set of constructed sessions that is as close as possible to the set of real sessions.

The reader may recall from Section 3.1.1 that a real session is the sequence of activities performed by one user, and that all users' sessions constitute a set $\mathcal{R}$. As described in Section 3.1.2, a sessionizing heuristic h produces a set of constructed sessions $\mathcal{C}_h$. The perfect heuristic ph would produce the set of real sessions, i.e. $\mathcal{C}_{ph} = \mathcal{R}$. Reactive sessionization heuristics are prone to the following mis-assignments: (1) the heuristic may place requests of different users into the same session, and (2) the heuristic may place requests of one session into several different sessions.

Berendt et al. (2001) evaluated the performance of sessionizing heuristics when distinct users are safely recognized as such, e.g. through a cookie mechanism. In that case, requests of different users are never merged. However, the heuristic may still corrupt the real sessions by specifying wrong session boundaries.

If a site does not use cookies or another proactive mechanism to distinguish among its users, or if some users employ anonymizers or reject cookies, then both kinds of error may occur. The second error occurs when a heuristic specifies wrong boundaries between adjacent real sessions and when it splits a real session into multiple parallel constructed ones. In order to make an appropriate mapping between an error and its cause, we consider the following types of error:

Interleaving error: The heuristic considers multiple users to be one user or a single user to be multiple users. In the former case, the heuristic places requests of multiple users into one constructed session. In the latter case, requests of the same user are placed into different parallel constructed sessions.

Segmentation error: The heuristic distinguishes correctly among users but makes erroneous guesses about the end of the real sessions.

This error typification is based on the assumption that if a user performs parallel activities on a site, then these activities belong to the same conceptual course of action. Thus, they

should be assigned to the same real session. This implies that the interleaving error only concerns the mixing of sessions from different users; if the set of activities of a user is correctly identified, this set can only be partitioned into subsequent sessions, and is thus subject to the segmentation error only.

5.2 Modelling and Measuring the Effectiveness of Sessionizers

The measures we propose quantify the successful mappings of real sessions to constructed sessions, i.e. the "reconstructions of real sessions." In particular, a measure M evaluates a heuristic h based on the difference between $\mathcal{C}_h$ and $\mathcal{R}$. It assigns to h a value $M(h) \in [0, 1]$ so that the score for the perfect heuristic ph is $M(ph) = 1$. We distinguish between "categorical" measures that reflect the number of real sessions that are reconstructed by the heuristic *in their entirety*, and "gradual" measures that take account of the extent to which the real sessions are reconstructed.

5.2.1 Categorical Measures

Categorical measures count how many real sessions were recognized by the heuristics as distinct visits and were thus mapped *into* constructed sessions. We first introduce the notion of session containment:

Definition 1 *Let a, b be sessions in $\mathcal{R} \cup \mathcal{C}$ with n, respectively m, elements. a "is contained in" b, $a \preceq b$ iff $\exists k \in \{0, \ldots, m-1\} : \forall i = 1, \ldots, n : a[i] = b[k+i]$.*

In this definition, we require that the elements of the contained session a are in adjacent positions inside b, i.e. they are not interleaved with elements not belonging to a.

Definition 2 *A real session $r \in \mathcal{R}$ is "completely reconstructed" if all its elements are contained in the same constructed session, i.e. there is a $c \in \mathcal{C}$ such that $r \preceq c$.*

(In Berendt et al. (2001), the notion of "completely reconstructed session" is defined without requiring adjacency of the elements. However, the context of that study implies that the elements of a completely reconstructed session are adjacent.)

Definition 3 *The* base categorical measure *M_{cr} grades a heuristic h by the number of completely reconstructed real sessions contained in $\mathcal{C}_h$ divided by the total number of real sessions $|\mathcal{R}|$:*

$$M_{cr}(h) = \frac{|\{r \in \mathcal{R} | \exists c \in \mathcal{C}_h : r \preceq c\}|}{|\mathcal{R}|} \ .$$

From this, we establish the following *derivative categorical measures* by elaborating on the containment relationship between a real and a constructed session:

Definition 4

- *The measure $M_{cr,start}$ counts only completely reconstructed real sessions whose first element is also the first element of the constructed session:*

$$M_{cr,start}(h) = \frac{|\{r \in \mathcal{R} | \exists c \in \mathcal{C}_h : r \preceq c \wedge r[1] = c[1]\}|}{|\mathcal{R}|} \ .$$

- *The measure $M_{cr,end}$ counts only completely reconstructed real sessions whose last element is also the last element of a constructed session:*

$$M_{cr,end}(h) = \frac{|\{r \in \mathcal{R} | \exists c \in \mathcal{C}_h : r \preceq c \wedge r[length(r)] = c[length(c)]\}|}{|\mathcal{R}|} \ .$$

- *The measure $M_{cr,start-end}$ counts only real sessions that are identical to the constructed sessions:*

$$M_{cr,start-end}(h) = \frac{|\mathcal{R} \cap \mathcal{C}_h|}{|\mathcal{R}|} \ .$$

The last measure $M_{cr,start-end}$ is an extension of the previous ones. In particular, a real and a constructed session are identical if and only if the former is completely reconstructed by the latter and the two sessions have the same start element and the same end element.

Application domain of the categorical measures. Categorical measures consider only completely reconstructed real sessions. They are designed to grade the performance of heuristics against segmentation errors. Thus, they are appropriate for applications where both of the following conditions hold:

(1) A user identification mechanism is available, so that interleaving errors on the sessions of different users cannot occur.

(2) Faithful reconstruction of the real sessions in their completeness is necessary for the analysis.

The first condition can be guaranteed by a proactive mechanism like cookies or user authentication. The second condition holds for applications that analyze behavior patterns in their entirety: evaluation of user satisfaction with a site and analysis of the improvement potential of the site as a whole belong to this category.

For some application domains, these conditions are too restrictive. For example, consider the page prefetching problem, in which the next request must be predicted given the requests performed thus far. User identification is not essential here, because prefetching is based on the former requests inside the same session only. Furthermore, complete session reconstruction is desirable but not necessary; at least for approaches using hidden Markov chains, only the most recent fragment of former requests (of size 1 or k) is of interest. For such applications we introduce a set of *gradual measures*.

5.2.2 Gradual Measures

Categorical measures presuppose the existence of a user identification mechanism, whereupon they can score heuristics with respect to segmentation errors. In the absence of such a mechanism, interleaving errors may occur. Applications exploiting only fragments of sessions in their analysis require measures sensitive to this type of error. Before introducing a group of measures to this purpose, we define the notion of "common fragment" between sessions.

Definition 5 *Let a (respectively b) be sessions in $\mathcal{R} \cup \mathcal{C}$ with n (respectively m) elements. The "(k_1, k_2)-common fragment" between a, b is defined as*

$$a \sqcap_{k_1}^{k_2} b = \{a[i], i = k_1, \ldots, k_2 | \exists k : \forall j = k, \ldots, k + k_2 - k_1 : a[i] = b[j]\} \ .$$

Two sessions may have several common fragments. We introduce the notion of "largest continuous intersection" as the largest common fragment between two sessions, using the first such fragment in case of ties.

Definition 6 *The "largest continuous intersection" between a, b, denoted as $a \sqcap b$, is the common fragment $x = a \sqcap_{k_1}^{k_2} b$, for which the following holds:*

1. *x is maximal, i.e. there is no common fragment containing it:*

$$\nexists l_1, l_2 \in \{1, \ldots, n\} : \left(a \sqcap_{k_1}^{k_2} b \preceq a \sqcap_{l_1}^{l_2} b\right) \wedge \left(a \sqcap_{k_1}^{k_2} b \neq a \sqcap_{l_1}^{l_2} b\right) \ .$$

2. *There is no common fragment y with $length(y) > length(x)$.*

3. *For each other maximal common fragment $y = a \sqcap_{l_1}^{l_2} b$ with $length(y) = length(x)$, $k_1 < l_1$.*

We use the largest continuous intersection to define the degree of overlap between a real and a constructed session.

Definition 7 *The "degree of overlap" between a real session and a constructed session is the number of elements in their largest continuous intersection divided by the total number of elements of the real session. Formally, for each $r \in \mathcal{R}$ and $c \in \mathcal{C}_h$, we define*

$$deg_o(r, c) = \frac{|r \sqcap c|}{|r|} \ .$$

The degree of overlap in Definition 7 refers to a particular constructed session. However, the fragments of a real session may be placed in multiple constructed sessions. Thus, the "degree of overlap for a real session" is an aggregate of the individual overlap degrees for each constructed session built by the heuristic. In general, this aggregate degree is a function f that combines the individual overlap degrees, for example by computing their average, median, or maximum. In this study, we define f as the maximum, so that:

$$f(r, h, deg_o) \ = \ \max_{c \in \mathcal{C}_h} \{deg_o(r, c)\} \ . \tag{1}$$

Finally, to compute a score for a heuristic on the basis of the degree of overlap it achieves, we need a function g that returns an aggregate of the degrees of overlap computed over all real sessions. Different aggregates are appropriate to this purpose, such as the robust median, the maximum, or the average. In this study, we opt for the average, so that for a heuristic h and its degree of overlap function deg_o we have

$$g(h, deg_o) = avg_{r \in \mathcal{R}} \{f(r, h, deg_o)\} \ . \tag{2}$$

On the basis of these definitions, we introduce the first gradual measure as follows.

Definition 8 *Let $\mathcal{R}$ be the set of real sessions and $\mathcal{C}_h$ the set of constructed sessions produced by heuristic h. Further, let f be a function that aggregates degrees of overlap for a particular real session towards all constructed sessions in $\mathcal{C}_h$, and let g be a function that aggregates the degrees of overlap among all real sessions.*

Then, $M_o(g \odot f)(h)$ is a gradual measure

$$M_o(g)(h) = g(h, deg_o) \ ,$$

where g is an aggregate on $f(r, h, deg_o)$ over all $r \in \mathcal{R}$, as in Equation (2). We select

$$M_o(avg \odot max)(h) = avg_{r \in \mathcal{R}} \{\max_{c \in \mathcal{C}_h} (r, h, deg_o)\} \ .$$

If a real session r is completely reconstructed by a session $c \in \mathcal{C}_h$, then $deg_o(r, c) = 1$. However, the number of additional elements in the constructed session is not taken into account. To alleviate this problem, we introduce the notion of degree of similarity:

Definition 9 *The "degree of similarity" between a real session and a constructed session is the number of elements in their largest continuous intersection divided by the total number of elements in the two sessions. Formally, for each $r \in \mathcal{R}$ and $c \in \mathcal{C}_h$, we define*

$$deg_s(r, c) = \frac{|r \sqcap c|}{|r \cup c|} \ .$$

Using the degree of similarity, we define the measure $M_s(g \odot f)$ analogously to $M_o(g \odot f)$ for each pair of functions f, g. For our experiments, we use the gradual measure $M_s(avg \odot max)$:

$$M_s(agv \odot max)(h) = avg_{r \in \mathcal{R}}\{\max_{c \in \mathcal{C}_h}(r, h, deg_s)\} \ .$$

Application domain of the gradual measures. Gradual measures are intended for applications that do not require complete session reconstruction but are still sensitive to interleaving and segmentation errors. Page prefetching is a characteristic application of this type. Gradual measures are also appropriate for applications that do not need user identification, because the behavior of users during a session is not affected by their behavior during earlier sessions. Evaluation of the usability of information acquisition services (Berendt and Spiliopoulou 2000) is another example of this application type.

For sites lacking a proactive user identification mechanism, only knowledge discovery applications confined to the analysis of sessions are feasible. In Geyer-Schulz et al. (2001), it is shown that session analysis is sufficient to make recommendations to customers. For some applications, the distinction between newcomers and experienced users is sufficient: session analysis can then be performed for the two groups separately. All these applications are sensitive to interleaving errors.

All measures are summarized in Table 7.

In the following section, we use these categorical and gradual measures to evaluate the performance of the heuristics described in Section 3.

Table 7: The Six Measures Used in the Experiments.

Acronym	Description
M_cr	proportion of real sessions that are completely reconstructed
M_crs	proportion of real sessions that are completely reconstructed, with the correct first element
M_cre	proportion of real sessions that are completely reconstructed, with the correct last element
M_crse	proportion of real sessions that are identically reconstructed
M_o	average maximal degree of overlap of real sessions with constructed sessions
M_s	average maximal degree of similarity of real sessions with constructed sessions

6. Experimental Evaluation

In the previous section, we introduced two groups of measures. A categorical measure scores sessionization heuristics on the basis of the sessions they have reconstructed in their entirety. Such a measure is appropriate when a site has a user identification mechanism and the KDD applications require complete session reconstruction. A gradual measure, on the other hand, assigns scores according to how large the reconstructed session *fragments* are. Gradual measures are appropriate for applications that do not observe users but only sessions, either because no user identification mechanism is available, or because past user behavior is not relevant for the analysis.

We have used the categorical and gradual measures to evaluate the performance of the sessionizing heuristics described in Section 3.5, using the parameter settings described there. For our experiments, we used the log of the Web site server described in Section 4.1. The reader may recall that a cookie-based mechanism was used for user identification, and a proactive sessionizing strategy split the user activities into sessions by assigning a session identifier to the user's agent process. The combination of the two proactive strategies served as reference mechanism to produce the real sessions. This produced 4400 users with 13829 sessions.

To evaluate the performance of the reactive sessionization heuristics of Section 3.5, we have applied them on two copies of the dataset: In the first dataset copy, the session identifiers were removed but the cookie identifiers were retained, thus allowing segmentation errors only ("**cookie**" in Table 1). In the second dataset copy, both session identifiers and cookie

identifiers were removed, so that interleaving errors could occur ("**ipa**" in Table 1).

6.1 Experimental Results

In Figure 1, we compare the performance of the heuristics for each of the categorical measures. Since categorical measures presuppose user identification, we consider only the cookie-based version of the three reactive strategies. The lines connecting the scores of the same heuristic for different measures show the performance deterioration between the least restrictive measure M_{cr} and the most restrictive one M_{crse}. The gradual measures are shown for comparison.

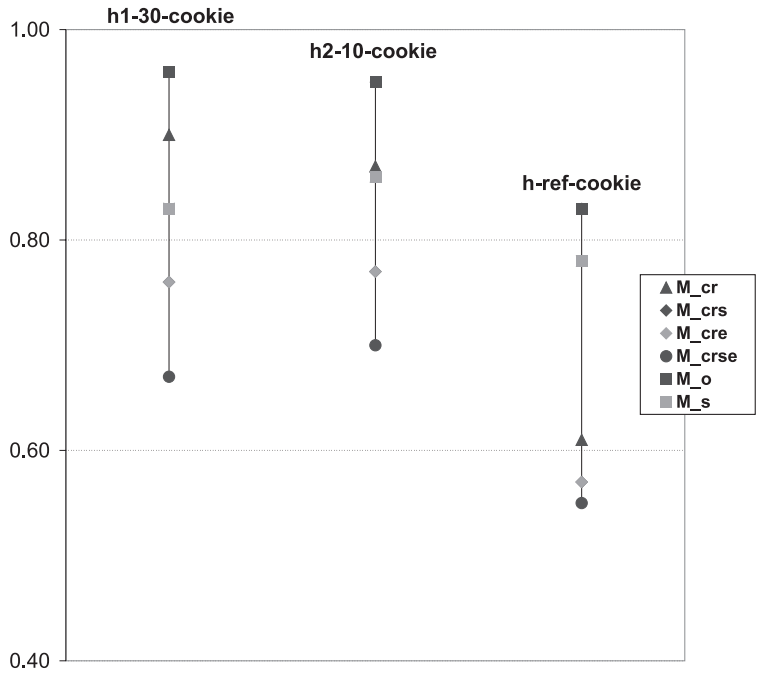

Figure 1: Categorical and Gradual Measures for the **cookie** Setting

Figure 2 depicts the performance of the heuristics for the gradual measures. Next to the cookie-based version of each heuristic, we can see the heuristic's performance when no user identifiers are available. The lines connecting the scores of the same heuristic show the degradation in performance for the gradual measures. In this figure, we have added the scores for the M_{crse} categorical measure as a reference.

Finally, Figure 3 depicts the performance drop between the **cookie** and the **ipa** version of the heuristics. In other words, it shows the performance drop incurred by the additional

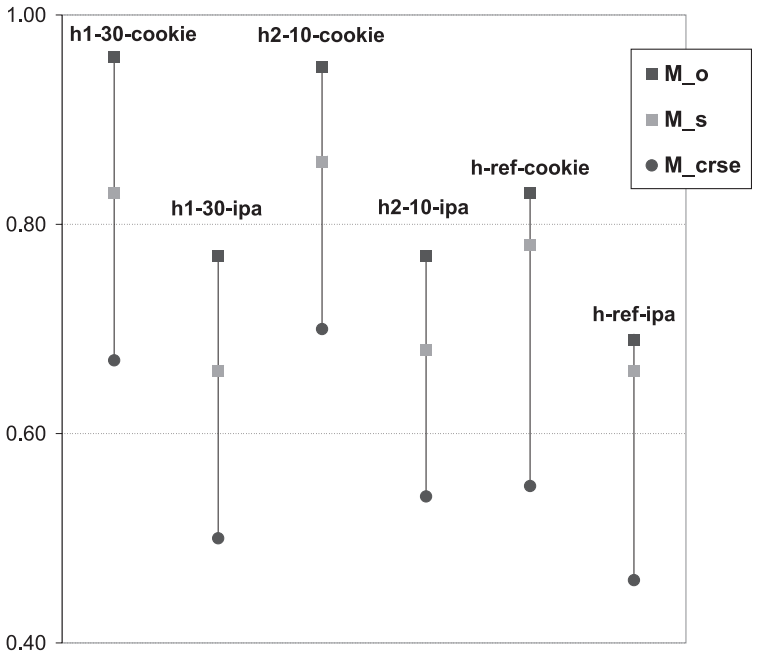

Figure 2: Gradual Measures and M_{crse} for the **cookie** and **ipa** Settings

reconstruction task in the **ipa** setting: while the heuristics' only task in the **cookie** setting is to identify session borders, in the **ipa** setting they must also assign temporally interleaved activities to different users. For all heuristics, performance loss varies between 10% and 20%. The referrer heuristic shows the least performance loss.

All figures indicate that there is no perfect reactive strategy. Even when cookie identifiers are known, segmentation errors still occur, to the effect that the best reactive strategy rebuilds the set of real sessions only to 65%. On the other hand, applications not interested in user identification should focus on the performance of the reactive strategies for the gradual measures: the time-based heuristics approximate the performance of the proactive strategies to more than 90%.

Berendt et al. (2001) provide a thorough discussion of the relative performance of the heuristics for the cookie-based case. We include part of this discussion in Section 6.2 for completeness.

6.2 Discussion of the Results

Before elaborating on the scores of the heuristics for each measure, we note a number of logical dependencies: the measures M_{crs} and M_{cre} add constraints to M_{cr}. Thus, the score of

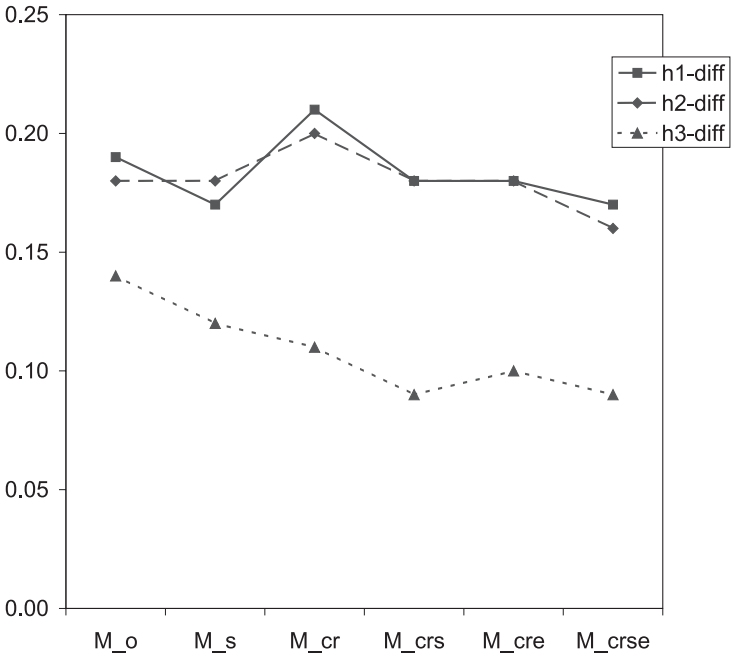

Figure 3: Degradation in Performance Between **cookie** and **ipa**

a heuristic h for them cannot be higher than $M_{cr}(h)$. The same holds for M_{crse} compared to M_{crs} or M_{cre}. Furthermore, the scores given by the gradual measures are higher than those of the categorical measures, since gradual measures also count partially reconstructed real sessions.

For any real and constructed session, the degree of similarity cannot be higher than the degree of overlap: both of them are ratios with the same nominator, while the denominator of the former is bounded below by the value of the denominator of the latter. Thus, the scores of gradual measures based on the degree of similarity cannot exceed the scores of their counterparts on the degree of overlap.

These dependencies reflect the relative scores of each heuristic across the whole palette of measures.

Relative performance of the heuristics. The two temporal heuristics **h1-30** and **h2-10** have similar performance with and without user identification. The performance of the referrer heuristic is lower for all but the gradual measure on the degree of similarity.

When we compare the scores of **h1-30** and **h2-10** in either setting, we see that **h1-30** has slightly lower scores for the derived categorical measures and for the gradual measure on the

degree of similarity. Notably, these measures give low scores when the constructed sessions are larger than the real sessions mapped into them, while the base categorical measure and the gradual measure on the degree of overlap do not. Thus, a possible explanation of the performance difference is that the dataset contains some short real sessions of the same user(s), separated by intervals of more than 10 minutes: these sessions are placed by **h1-30** into the same 30-minutes long constructed session, while **h2-10** builds a new constructed session whenever the 10-minute page-stay threshold expires.

Although the scores of the referrer heuristic are lower than those of the temporal ones for both settings (with one exception discussed below), the performance of this heuristic is more stable across the categorical measures in both settings. The reader may recall that the base categorical measure just counts the completely reconstructed real sessions, while the derived categorical measures refine this count by skipping real sessions embedded within constructed sessions together with foreign elements. Thus, the low but stable performance of the referrer heuristic for these measures implies that:

- The referrer heuristic outputs fewer real sessions that are completely reconstructed, but
- The constructed sessions containing these real sessions are more likely to have foreign elements.

Furthermore, the performance deterioration caused by session interleaving is much less for the referrer heuristic than for the temporal heuristics. We elaborate on this observation below, when we discuss the impact of the interleaving error.

Impact of session segmentation. The impact of session segmentation is reflected in the scores of the heuristics for the cookie-based setting. As already noted, the two temporal heuristics perform similarly, with a slight superiority of **h2-10** for the derived gradual measures.

The performance of the referrer heuristic is comparatively low, i.e. many real session segmentation errors are encountered. Possible explanations are:

1. There are many real sessions where the referrer of a request is not the request directly preceding it, e.g. because of the framesets in the site. Although the referrer heuristic we use has been customized to cope with framesets (Berendt et al. 2001), the rules it uses seem to be still suboptimal.

2. There are many users that perform parallel sequences of activities. The referrer heuristic identifies them and places them into different real sessions. This contradicts our assumption that such activities should belong to the same real session, and thus with the way the "real sessions" are built for this experiment. Hence, **h-ref** might perform better when parallel activities of the same user are regarded as distinct sessions.

Impact of session interleaving. Session interleaving is reflected when comparing the performance of the heuristics without user identification to their performance when the users are known. Degradation is approximately 20% for the time-based heuristics and is rather uniform, so that the relative scores among the different heuristics do not change.

Interestingly, the **h-ref** heuristic is much less prone to interleaving sessions than the temporal heuristics. For the gradual measures, its performance drop does not exceed 10 percentage points. Possible explanations are:

1. There are many users accessing the site simultaneously through the same proxy and agent. Since these users are unlikely to issue the same requests, the referrer heuristic may place their activities into separate sessions.

2. There are many consecutive short sessions of the same or of different users from the same proxy and agent, and the elapsed time between them is low. While the temporal heuristics tend to place these real sessions into the same constructed session, **h-ref** identifies the gaps in the referrers and builds separate constructed sessions for them.

A closer inspection of the dataset revealed that simultaneous users with the same IP address and agent made up only 1% of the log sessions. Thus, the second explanation is more likely than the first one.

Finally, the referrer heuristic has rather good performance for the gradual measure on similarity if no user identification is available. Only the temporal heuristic **h2-10** shows a slightly better performance. The reader may recall that the gradual measure on similarity is based on the number of reconstructed real session fragments, whereby foreign elements in the constructed session result in lower scores (cf. denominator of the degree of similarity function in Definition 9). This indicates that the referrer heuristic builds many real session fragments, and its constructed sessions do not contain many elements beyond these fragments.

Summary of the results. The measures proposed in Section 5 reflect the types of error pertinent in Web server logs, when no proactive strategies are available.

No heuristic achieves the performance of the cookie-based proactive strategy. However, in the presence of cookie identifiers, the time-based heuristics reconstruct fragments of real sessions to an extent of more than 90%. Thus, these heuristics are an acceptable alternative to proactive sessionization, if the KDD applications do not require completely reconstructed sessions. In Section 5, we provided examples of such applications.

Whenever faithful reconstruction of the real sessions *in their entirety* is indispensable, reactive heuristics are no appropriate choice. The heuristic on total session duration tends to place many real sessions into the same constructed session, apparently because of the necessarily conservative estimation of the session-duration threshold. The heuristic on page-stay time is less prone to this segmentation error. However, both heuristics have rather unstable performance among the different measures.

The referrer heuristic performs poorly because it splits real sessions into small fragments. However, its performance does not deteriorate much in the absence of cookie identifiers. In fact, its (rather low) performance is quite stable across all categorical and gradual measures. Finally, the session fragments it reconstructs are comparable in size to those built by the temporal heuristic **h2-30**, which has more unstable performance.

These observations on the referrer heuristic indicate that it is an appropriate method of choice when (1) there are no cookie identifiers for the users and (2) the Web server log contains many interleaved requests from hosts that are likely to be proxies or anonymizers.

The referrer heuristic is the method of choice when multiple Web server logs must be merged but the server clocks are not synchronized. In that case, the time-based heuristics are not applicable. As for the case of a single Web server log, the objectives and requirements of the KDD application determine the appropriateness of this heuristic: if session fragments are sufficient, the referrer heuristic can reconstruct them to a certain extent. If they are not sufficient, then proactive strategies should be installed.

7. Conclusions

In this study, we have proposed a formal framework for the comparison of session reconstruction heuristics according to a set of measures, and have applied it to evaluate heuristic performance for a real frame-based Web site.

Three main contributions can be distinguished: first, we provide comparative results on the impact of proactive strategies like cookie-based identification upon the process of session reconstruction. Our experiments show that cookie identifiers increase the quality of reconstructed sessions up to 20% but do not reach the quality achieved by Web servers with embedded proactive session identification mechanisms.

Second, our experiments showed that there is no best heuristic for all cases. Even for a simple application, two variations in the method of assessing reconstruction quality led to significantly different precision scores among the heuristics. Third, to alleviate this ambiguity, we proposed a set of measures that reflect the demands of different KDD application types, and we evaluated the heuristics according to those measures. The performance results for each measure can serve as an aid to the analyst for the appropriateness of the heuristics for each type of application.

One subject of future work is the quantification of the impact of caching. Caching prevents the registration of all requests by the server and thus blurs the picture of user behavior; this is dangerous for KDD applications that analyze navigational behavior, e.g. for usability testing. Furthermore, we want to extend our framework to capture the particularities of KDD applications with respect to further types of error, beyond segmentation and interleaving errors dealt with in this study. Finally, we want to incorporate heuristics and evaluation measures upon them into a generic data preparation toolkit for a variety of KDD applications.

References

Berendt, B., B. Mobasher, M. Spiliopoulou, J. Wiltshire. 2001. Measuring the accuracy of sessionizers for web usage analysis. *Proceedings of the Workshop on Web Mining, First SIAM International Conference on Data Mining, Chicago, IL.* 7–14.

Berendt, B., M. Spiliopoulou. 2000. Analysis of navigation behaviour in web sites integrating multiple information systems. *The VLDB Journal* **9** 56–75.

Catledge, L., J. Pitkow. 1995. Characterizing browsing behaviors on the world wide web. *Computer Networks and ISDN Systems* **26** 1065–1073.

Cooley, R., B. Mobasher, J. Srivastava. 1999. Data preparation for mining world wide web browsing patterns. *Journal of Knowledge and Information Systems* **1** 5-32.

Cooley, R., P. Tan, J. Srivastava. 2000. Discovery of interesting usage patterns from Web data. B. Masand, M. Spiliopoulou, eds. *Advances in Web Usage Analysis and User Profiling.* LNAI 1836, Springer, Berlin, Germany. 163–182.

Geyer-Schulz, A., M. Hahsler, M. Jahn. 2001. A customer purchase incidence model applied to recommender systems. *Proceedings of the WebKDD'01 Workshop at the ACM SIGKDD, San Francisco, CA.* 35–45.

Hand, D., H. Mannila, P. Smyth. 2001. *Principles of Data Mining.* MIT Press, Cambridge, MA.

Mayer-Schönberger, V. 1997. The Internet and privacy legislation: cookies for a treat? *West Virginia Journal of Law & Technology* **1**. http://www.wvu.edu/~wvjolt/Arch/Mayer/-Mayer.htm .

Padmanabhan, B., Z. Zheng, S.O. Kimbrough. 2001. Personalization from incomplete data: What you don't know can hurt. *Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA.* 154–163.

Pyle, D. 1999. *Data Preparation for Data Mining.* Morgan Kaufmann Publishers Inc., San Francisco, CA.

Reuters Limited. 2001. EU sticking to tough spam law. *Wired News*, 6 December 2001. http://www.wired.com/news/politics/0,1283,48894,00.html

Spiliopoulou, M., L.C. Faulstich. 1999. WUM: a tool for Web utilization analysis. *Proceedings EDBT Workshop WebDB'98*, LNCS 1590, Springer, Berlin, Germany. 184–203.

Tan, P., V. Kumar. 2002. Discovery of Web robot sessions based on their navigational patterns. *Data Mining and Knowledge Discovery* **6** 9–35.

World Wide Web Committee Web Usage Characterization Activity. 1999. *W3C Working Draft: Web Characterization Terminology & Definitions Sheet.* http://www.w3.org/-1999/05/WCA-terms/ .

Zheng, Z., B. Padmanabhan, S. Kimbrough. 2003. On the existence and significance of data preprocessing biases in Web usage mining. *INFORMS Journal on Computing* **15** xxx–xxx.

Accepted by Amit Basu; received November 2001; revised March 2002; accepted May 2002.