

Clickstream Clustering using Weighted Longest Common Subsequences

Arindam Banerjee and Joydeep Ghosh*
{abanerje,ghosh}@ece.utexas.edu
Dept. of Electrical and Computer Engineering
University of Texas at Austin
Austin, TX 78712

Abstract

Categorizing visitors based on their interactions with a website is a key problem in web usage mining. The clickstreams generated by various users often follow distinct patterns, the knowledge of which may help in providing customized content. In this paper, we propose a novel and effective algorithm for clustering webusers based on a function of the longest common subsequence of their clickstreams that takes into account both the trajectory taken through a website and the time spent at each page. Results are presented on weblogs of www.sulekha.com to illustrate the techniques.

keywords : web usage mining, clickstream, subsequence, clustering.

1 Introduction

With the rapid increase in web-traffic and e-commerce, understanding user behavior based on their interaction with a website is becoming more and more important for website owners. Identifying the nature of each user visiting a website may enable that site to provide customized content for the users, thereby making it more *sticky* and enhancing user experience. The business implications of such an ability are huge, specially for portals, personalized content providers and e-tailers.

The sequence of pages visited by a user within a particular website is his “*clickstream*” in that site for one session. The purpose of clustering users based on their clickstreams in a particular website is to find groups of users with similar interests and motivations for visiting that website. If the site is well designed there will be strong correlation between the similarity among user clickstreams and the similarity among the users’ interests or intentions. Therefore, clustering of the former could be used to predict groupings for the latter.

A lot of research has been done in the area of Web Usage Mining [1] which directly or indirectly addresses the issues involved in the extraction of web navigational patterns [2], ordering relationships [3], prediction of web surfing behavior [4], and clustering of web usage sessions [5] based on web logs, possibly supplemented by web content or structure information. A fundamental difficulty in all these problems is that the raw data is in the form of sequences, and not vectors. As a result, well developed mining techniques for vector type data cannot be applied directly. The specific problem of web usage clustering has been studied over the past few years. In [5], the authors looked at clustering web usage sessions using a generalization-based approach and making use of the hierarchical clustering algorithm called BIRCH. In [6], the authors introduced the idea of a path feature space and path angles, and finally clustered the paths using the k-means algorithm. However, a definitive solution is yet to emerge.

Categorizing visitors from their clickstream is a difficult problem since typically the number of possible sequences of page-visits is huge. Moreover the time spent at each page may differ significantly from user to user. The time spent by a user in a particular page is a clear indication of the user’s interest in the page¹, and thus should not be ignored. Using the webserver logs, we have developed a formal definition of a *path* corresponding to a session in a website

*contact author; phone 512 471 8980; fax 471 5907

¹however one needs to set a heuristic upper bound on the time spent on any page by any user to take care of outliers, distracted users, etc.

and a way to compute a similarity measure between any two paths for a given *resolution*, taking into account both the trajectory taken and the time spent at each page.

For getting the user paths from webserver logs, we need to first extract the user sessions from the weblog files. Extracting user sessions from weblogs, specially in the absence of cookie information, is a difficult task and we will not address it here. We shall assume that the sessions have already been extracted either using cookies or by some reasonable heuristics [7]. Then we define a path generated in a session and propose a method for finding the similarity between all pairs of paths that have been obtained from the extracted sessions. This method first determines the intersection of any two paths under consideration as defined by the longest common subsequence(LCS) between the two sequences of pages visited by each user. Then, a similarity value is computed over this LCS as a function of the closeness in the times spent on the corresponding pages weighted by a certain importance factor. Using these similarity values, a graph is constructed whose nodes are the paths. An edge connecting two nodes has a weight equal to the similarity value between the corresponding paths. Clustering of the sessions is then performed in this similarity space using efficient graph-based techniques.

For large websites, if paths are considered at a webpage level of resolution, many of the similarity values turn out to be zero since very few paths have actual page overlaps. This may not reflect user behavior properly. Therefore we introduce the idea of concept-based paths so that paths through conceptually similar pages have a non-null intersection even without any direct webpage overlap. This procedure is effective in yielding meaningful clusters of user paths. Also, this reduces the computations by a significant amount.

The paper is organized as follows. In section 2, we define a path through a website formally and briefly discuss the algorithm for finding the intersection between two such paths. We propose a path similarity measure between two paths making use of their intersection and show some properties of the measure in section 3. The formation of the similarity graph from these pairwise similarity values and methods for obtaining the path clusters from this graph are discussed in section 4. In section 5 we extend the idea of a path to a concept based path and explain the reasons for doing so. In section 6, we discuss sample results. We study the flexibility and other issues of the algorithm in section 7. We draw some conclusions in section 8.

2 Path Intersection

A session in which a user visits n pages can be represented as a n -length sequence of ordered pairs given by $\pi = [(\alpha_1, \tau_1^\alpha) (\alpha_2, \tau_2^\alpha) \cdots (\alpha_n, \tau_n^\alpha)]$, where $\vec{\alpha} = \langle \alpha_1, \alpha_2, \cdots, \alpha_n \rangle$ is the sequence of webpages visited, α_j being the j th page visited in the session, and $\vec{\tau}^\alpha = \langle \tau_1^\alpha, \tau_2^\alpha, \cdots, \tau_n^\alpha \rangle$ being the corresponding sequence of times spent on a page, τ_j^α being the time spent on α_j . We shall call the sequence $\vec{\alpha}$ as the *path* corresponding to the session under consideration. Next, we shall outline a dynamic programming algorithm for finding the intersection between two such paths.

Formally, given a sequence $\vec{\alpha} = \langle \alpha_1, \alpha_2, \cdots, \alpha_n \rangle$, another sequence $\vec{\gamma} = \langle \gamma_1, \gamma_2, \cdots, \gamma_l \rangle$ is a subsequence of $\vec{\alpha}$ if there exists a strictly increasing sequence $\langle j_1, j_2, \cdots, j_l \rangle$ of indices of $\vec{\alpha}$ such that for all $i = 1, 2, \cdots, l$, we have $\alpha_{j_i} = \gamma_i$. Given two sequences $\vec{\alpha}, \vec{\beta}$, we say that $\vec{\gamma}$ is a *common subsequence* of $\vec{\alpha}$ and $\vec{\beta}$ if $\vec{\gamma}$ is a subsequence of both $\vec{\alpha}$ and $\vec{\beta}$. We are interested in finding the maximum-length or longest common subsequence(LCS) given two paths or sequences of page-visits $\vec{\alpha} = \langle \alpha_1, \alpha_2, \cdots, \alpha_n \rangle, \vec{\beta} = \langle \beta_1, \beta_2, \cdots, \beta_m \rangle$,

The LCS has a well-studied optimal sub-structure property as given by the following :

Theorem 1 Let $\vec{\alpha} = \langle \alpha_1, \alpha_2, \cdots, \alpha_n \rangle$ and $\vec{\beta} = \langle \beta_1, \beta_2, \cdots, \beta_m \rangle$ be sequences, and let $\vec{\gamma} = \langle \gamma_1, \gamma_2, \cdots, \gamma_l \rangle$ be any LCS of $\vec{\alpha}$ and $\vec{\beta}$.

1. If $\alpha_n = \beta_m$, then $\gamma_l = \alpha_n = \beta_m$ and $\vec{\gamma}_{l-1}$ is a LCS of $\vec{\alpha}_{n-1}$ and $\vec{\beta}_{m-1}$.
2. If $\alpha_n \neq \beta_m$, then $\gamma_l \neq \alpha_n$ implies $\vec{\gamma}$ is a LCS of $\vec{\alpha}_{n-1}$ and $\vec{\beta}$.
3. If $\alpha_n \neq \beta_m$, then $\gamma_l \neq \beta_m$ implies $\vec{\gamma}$ is a LCS of $\vec{\alpha}$ and $\vec{\beta}_{m-1}$.

where $\vec{\alpha}_{n-1} = \langle \alpha_1, \alpha_2, \cdots, \alpha_{n-1} \rangle, \vec{\beta}_{m-1} = \langle \beta_1, \beta_2, \cdots, \beta_{m-1} \rangle$ and $\vec{\gamma}_{l-1} = \langle \gamma_1, \gamma_2, \cdots, \gamma_{l-1} \rangle$

The proof of this theorem can be found in [8]. Efficient recursive algorithms to compute the LCS exist using this property of the LCS [9]. We shall not go into the details of the algorithms since a lot of literature already exists on the topic [9, 10, 11]. We have implemented the $O(mn)$ algorithm presented by Wagner and Fischer [8, 12] with an added module that outputs the subsequence of indices of the two sequences that match in forming the longest

common subsequence. For example, if $\vec{\alpha} = \langle A, B, C, B, D, A, B \rangle$ and $\vec{\beta} = \langle B, D, C, A, B, A \rangle$, their LCS is $\vec{\gamma} = \langle B, C, B, A \rangle$.² The indices over $\vec{\alpha}$ and $\vec{\beta}$ forming the LCS are $\langle 2, 3, 4, 6 \rangle$ and $\langle 1, 3, 5, 6 \rangle$ respectively.

Thus, for each pair of paths $\vec{\alpha}$ and $\vec{\beta}$, we obtain two one-to-one functions $l^\alpha(\cdot)$ and $l^\beta(\cdot)$ both of which take in a particular index i of the longest common subsequence $\vec{\gamma}$, and generate the corresponding indices $l^\alpha(i)$ and $l^\beta(i)$ in the sequences $\vec{\alpha}$ and $\vec{\beta}$ respectively. For the above example, $l^\alpha(1) = 2, l^\alpha(2) = 3, l^\alpha(3) = 4, l^\alpha(4) = 6$ and $l^\beta(1) = 1, l^\beta(2) = 3, l^\beta(3) = 5, l^\beta(4) = 6$.

3 Path Similarity Measure

In this section, we propose a path similarity measure between any two paths in the website based on their longest common subsequence. The similarity measure we propose here takes into account the time spent as in [6] but with a very important refinement. In [6] the product of the times spent in the corresponding pages had been used as a similarity measure. This returns the same similarity values for $t_1 = 1, t_2 = 100$ and $t_1 = 10, t_2 = 10$. Our measure gets rid of such problems and adds a lot of flexibility to the formulation. In a previous work [7], we had looked at finding the path similarity from substring matches. As we shall show, the complexity of the proposed approach, using subsequence matches, is lower than that of the previous work.

For every pair of paths $\vec{\alpha}$ and $\vec{\beta}$, we have obtained the functions $l^\alpha(\cdot)$ and $l^\beta(\cdot)$ using the LCS extraction algorithm. Note that since the functions are one-to-one, the cardinality of the range sets of both the functions are the same and is equal to the length of the LCS. We shall denote this cardinality by L and the LCS by $\vec{\gamma} = \langle \gamma_1, \gamma_2, \dots, \gamma_L \rangle$. Thus, for $i = 1, \dots, L$, $l^\alpha(i)$ and $l^\beta(i)$ represent the indices in $\vec{\alpha}$ and $\vec{\beta}$ respectively for which $\alpha_{l^\alpha(i)} = \beta_{l^\beta(i)} = \gamma_i$. Then, we focus on the sequences $\vec{\tau}_{l^\alpha(i)}^\alpha = \langle \tau_{l^\alpha(1)}^\alpha, \tau_{l^\alpha(2)}^\alpha, \dots, \tau_{l^\alpha(L)}^\alpha \rangle$ and $\vec{\tau}_{l^\beta(i)}^\beta = \langle \tau_{l^\beta(1)}^\beta, \tau_{l^\beta(2)}^\beta, \dots, \tau_{l^\beta(L)}^\beta \rangle$, which are the time subsequences corresponding to the page subsequences forming the LCS of the two sequences. Next we compute the total time spent in the LCS by each user as $T_{LCS}^\alpha = \sum_{i=1}^L \tau_{l^\alpha(i)}^\alpha$ and $T_{LCS}^\beta = \sum_{i=1}^L \tau_{l^\beta(i)}^\beta$. We also note that the total time spent by each user is given by $T^\alpha = \sum_{j=1}^n \tau_j^\alpha$ and $T^\beta = \sum_{j=1}^m \tau_j^\beta$. The similarity measure between paths $\vec{\alpha}$ and $\vec{\beta}$ has two components :

1. The **Similarity** Component : This component computes how similar the two paths are in the region of their overlap, which in this case is their LCS. Since they had spent $\tau_{l^\alpha(i)}^\alpha$ and $\tau_{l^\beta(i)}^\beta$ time in the page γ_i , their similarity on this page can be computed using the min-max similarity given by :

$$s_i = \frac{\min(\tau_{l^\alpha(i)}^\alpha, \tau_{l^\beta(i)}^\beta)}{\max(\tau_{l^\alpha(i)}^\alpha, \tau_{l^\beta(i)}^\beta)} \quad (1)$$

Then, the average similarity between the paths over the entire overlap region is given by

$$S' = \frac{1}{L} \sum_{i=1}^L s_i = \frac{1}{L} \sum_{i=1}^L \frac{\min(\tau_{l^\alpha(i)}^\alpha, \tau_{l^\beta(i)}^\beta)}{\max(\tau_{l^\alpha(i)}^\alpha, \tau_{l^\beta(i)}^\beta)} \quad (2)$$

2. The **Importance** Component : This component computes how important the overlap region is to each of the paths. The importance is estimated as the fraction of total time spent on this region, which for the path $\vec{\alpha}$ is $\frac{T_{LCS}^\alpha}{T^\alpha}$ and for the path $\vec{\beta}$ is $\frac{T_{LCS}^\beta}{T^\beta}$. Then, the overall importance both of the paths attach to the overlap region should be some mean of these two fractions – in our implementation, we have used the geometric mean of the two fractions. Hence, the contribution of the importance component is given by

$$S'' = \left(\frac{T_{LCS}^\alpha}{T^\alpha} \times \frac{T_{LCS}^\beta}{T^\beta} \right)^{\frac{1}{2}} \quad (3)$$

So, the total similarity between paths $\vec{\alpha}$ and $\vec{\beta}$ is given by

$$S_{\vec{\alpha}\vec{\beta}} = S' \times S'' = \left(\frac{1}{L} \sum_{i=1}^L \frac{\min(\tau_{l^\alpha(i)}^\alpha, \tau_{l^\beta(i)}^\beta)}{\max(\tau_{l^\alpha(i)}^\alpha, \tau_{l^\beta(i)}^\beta)} \right) \times \left(\frac{T_{LCS}^\alpha}{T^\alpha} \times \frac{T_{LCS}^\beta}{T^\beta} \right)^{\frac{1}{2}} \quad (4)$$

²also $\langle B, D, A, B \rangle$

The proposed similarity measure has the following properties :

1. In case of **no overlap** between the paths, $\mathcal{S}_{\bar{\alpha}\bar{\beta}} = 0$: If there is no overlap, $L = 0$. Hence $\mathcal{S}' = 0$ as the summation was over $i = 1, \dots, L$. For the same reason, $T_{LCS}^\alpha = T_{LCS}^\beta = 0$ and hence $\mathcal{S}'' = 0$. So, $\mathcal{S}_{\bar{\alpha}\bar{\beta}} = \mathcal{S}' \times \mathcal{S}'' = 0$.
2. In case of **identical** paths, $\mathcal{S}_{\bar{\alpha}\bar{\beta}} = 1$: In this case $L = n = m$ and $\tau_{l^\alpha(i)}^\alpha = \tau_{l^\beta(i)}^\beta, i = 1, \dots, L$. As a result, $s_i = 1, \forall i$ and hence $\mathcal{S}' = 1$. Again, as $L = n = m, T_{LCS}^\alpha = T_{LCS}^\beta = T^\alpha = T^\beta$ and hence $\mathcal{S}'' = 1$. Therefore, $\mathcal{S}_{\bar{\alpha}\bar{\beta}} = 1$.
3. In general, $0 \leq \mathcal{S}_{\bar{\alpha}\bar{\beta}} \leq 1$: This property follows from the fact that since \mathcal{S}' is an average over terms each of which lies between 0 and 1, $0 \leq \mathcal{S}' \leq 1$. Again, since \mathcal{S}'' is the arithmetic mean of two fractions lying between 0 and 1, $0 \leq \mathcal{S}'' \leq 1$. Hence their product lies between 0 and 1.
4. The similarity measure is commutative as $\mathcal{S}_{\bar{\alpha}\bar{\beta}} = \mathcal{S}_{\bar{\beta}\bar{\alpha}}$.

4 The Similarity Graph and Clustering

In the previous section, a technique to find the similarity between any pair of paths was outlined. Thus, if there were P paths $\pi_1, \pi_2, \dots, \pi_P$ under consideration, $\binom{P}{2}$ similarity values are obtained. These are used to construct a weighted similarity graph $G_0 = (V_0, E_0)$, where $V_0 = \{\pi_1, \pi_2, \dots, \pi_P\}$ and the edge e_{ij} between the nodes π_i and π_j exists $\forall i, j, i \neq j$, and has a weight equal to the path similarity value between the paths π_i and π_j as computed from section 3. We shall represent the edge-weight of the edge e_{ij} by $|e_{ij}|$. Note that $|e_{ij}| = |e_{ji}|$ from property 4 of the similarity measure. Also, from property 3, we get that $0 \leq |e_{ij}| \leq 1, \forall i, j, i \neq j$. From G_0 , another undirected graph $G_\theta = (V_\theta, E_\theta)$ is constructed as follows :

1. Initialize : $V_\theta = E_\theta = \phi$, the null set.
2. Choose an **edge threshold** θ such that $0 \leq \theta \leq 1$.
3. for $i = 1$ to $(n - 1)$
 - for $j = (i + 1)$ to n
 - if $e_{ij} \geq \theta$
 - $V_\theta \leftarrow V_\theta \cup \{\pi_i, \pi_j\}$
 - $E_\theta \leftarrow E_\theta \cup e_{ij}$
 - end
- end
- end

Note that $|V_\theta| \leq |V_0|$ and $|E_\theta| \leq |E_0|$ and the equality occurs when $\theta = 0$. G_θ maybe a disconnected graph but will not have isolated vertices. This property is proved in Appendix A.

The idea behind the construction of G_θ is based on a number of observations which we shall discuss next. A lot of webusers show similar behavior while surfing and can be automatically categorized into meaningful groups. However, there are quite a few whose behavior does not follow any general group characteristics. As the result, the nodes in the similarity graph representing the paths taken by these users are only weakly connected to the other nodes of the graph. Clustering based on G_0 is undesirable since these outlier users will be forced into the nearest cluster. The thresholding approach described above takes out the outliers from G_0 before clustering. It is to be noted that for a path to be taken out from the graph, it has to be a *unique outlier* – if there is at least another path which has strong similarity to this path, both of them stay in G_θ . One can think of variations and generalizations on these lines but we shall not go into that in this paper.

The problem of clustering is formulated as the partitioning of the graph G_θ with *min-cut* and *balancing* constraints. A set of edges whose removal partitions the graph into k pair-wise disjoint subgraphs $G_l = (V_l, E_l)$ is called an edge separator ΔE . The objective is to find such a separator with a minimum sum of edge weights as given by

$$\min_{\Delta E} \sum_{e_{ij} \in \Delta E} |e_{ij}| \text{ where } \Delta E = \left(E_\theta - \bigcup_{l=1}^k E_l \right) \text{ and } \bigcup_{l=1}^k V_l = V_\theta \quad (5)$$

The above mentioned min-cut objective is to be achieved under a balancing constraint given by $k \times \max_l |V_l| \leq t$. For graphs with weighted vertices, the $|V_l|$ term is replaced by the sum of the vertex weights. The imbalance threshold $t \geq |V_\theta|$ – the higher it is, the more is the imbalance allowed and vice versa. Partitioning of G_θ into k parts gives k sets of relatively closely connected nodes. Since the nodes represent the paths, we end up in getting k clusters of the paths from clustering and a set $V' = (V_0 - V_\theta)$ of unique outliers from thresholding. An efficient and fast graph partitioning algorithm called Metis [13] is used for partitioning G_θ .

5 Concept-based Clustering

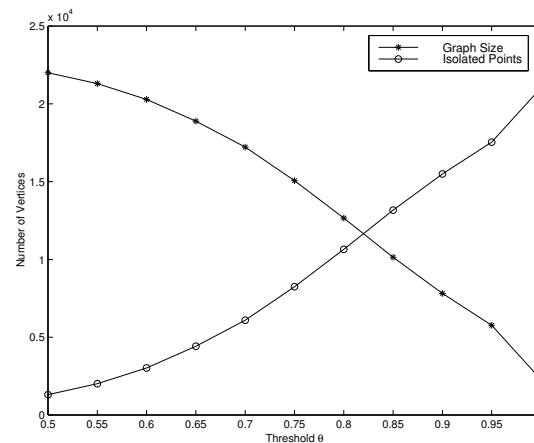
If paths are considered at the webpage level of resolution, paths tend to have very little similarity with one another. This is because at such a high resolution, there are very few exact webpage matches between the paths. This results in a lot of small or even zero length LCSs which in turn makes the number of paths to be clustered ($|V_\theta|$) pretty small even for small values of θ . This also makes the number of outliers ($|V_0 - V_\theta|$) large and all this is perhaps not very desirable. At this point we observe that though there not many exact webpage matches, a lot of webpages in the site are semantically equivalent. So, the webpages can be first grouped into categories based on suitable analytics and/or metadata information. We shall call each of these categories a *concept* and then new paths can be formed from the concept-category of the pages present in the original path. Even though there may not be exact webpage matches between two paths, if the paths are through similar concepts, e.g., sports, politics etc., we shall get a non-zero similarity value between the paths. Since most websites have their contents already categorized based on subject or topic into an ontology, this structure can be readily used to form the concepts. We have used this method successfully on the weblogs from www.sulekha.com, a highly trafficked community site and from www.ece.utexas.edu, the website of the department of ECE at UT, Austin. We shall briefly discuss the concept-categories for *sulekha*, followed by a few examples of conversion from webpage based paths to concept-based paths.

Being a very structured website, we identified the concept-categories to be the first level branchings from the root page of [sulekha.com](http://www.sulekha.com), the root page in itself being a category. We identified the following categories – home, which stands for www.sulekha.com/index.html; articles, authors, biztech, books, coffeehouse, contests, cooltools, creative, fun, games, movies, personal, philosophy, politics, sports, wo-men, each of which corresponds to a first level branching from home³ and misc, which stands for all the webpages in the site that do not fall in any of the previously mentioned categories.

When we convert the raw paths to concept-based paths, the average size of the path reduces and we get paths which can be easily understood. A few examples of this conversion are shown in Table 5.

Original path	Concept-based path
(/authors/ramesh_mahadevan.html,3) → (/articles/rm_phattas.html,75) → (/articles/rm_desidads.html,39)	(/authors,3) → (/articles,114) →
(/,85) → (/authors/arun_sampath.html,109) → (/philosophy/messages/1951.html,102) → (/philosophy/messages/1953.html,46) → (/,3) → (/philosophy/messages/1954.html,69)	(/,85) → (/authors,109) → (/philosophy,148) → (/,3) → (/philosophy,69)
(/,27) → (/articles/abbas_hindi.html,108) → (/wo-men/messages/1520.html,23) → (/wo-men/messages/1522.html,19) → (/wo-men/messages/1521.html,20) → (/wo-men/messages/1524.html,39)	(/,27) → (/articles,108) → (/wo-men,101)
(/,249) → (/fun/messages/2586.html,73) → (/fun/messages/2588.html,11) → (/fun/messages/2587.html,28) → (/movies/messages/1470.html,31) → (/movies/messages/1471.html,171) → (/books/messages/1562.html,194) → (/articles/dmobilus_vaccine.html,74) → (/articles/rananath_heavens.html,104)	(/,249) → (/fun,112) → (/movies,202) → (/books,194) → (/articles,178)

Original paths converted to Concept-based paths.



Variation of $|V_\theta|$ and $|V_0 - V_\theta|$ with change in θ .

³for example, the pages www.sulekha.com/books/foo.html, www.sulekha.com/books/foo/bar.html fall into the category books

Once these concept-based paths were formed, we treat them as ordinary paths and apply the already developed techniques from sections 3 and 4 to get the concept-based path clusters and outliers. We found that the results were much more meaningful than the simple page-based path case. In section 6, we present the clusters and a few outliers we have obtained from concept-based paths. Corresponding to each cluster, we have assigned a meaningful cluster label after looking at the general nature of paths in that cluster.

6 Sample Results

Due to lack of space, we just present the clustering results obtained after doing concept-based clustering on the *sulekha* website. The data used had 184 MB of raw logs, collected over a one month period. After certain standard preprocessing [7], the logs reduce to 43 MB. A total of 453,953 pages were accessed by 9,112 unique hosts. After session extraction using some time-out heuristics, a total of 37,753 sessions were extracted. Sessions with only a single page access were rejected since there is no way to compute the time spent on that page. Also, sessions with extremely large number of page accesses were rejected since these were most probably generated by crawlers/bots. This left 23,310 sessions for analysis. Before converting the original paths to concept-based paths, a session consisted of visits to 12 webpages on an average over a time span of 16 minutes. For concept-based paths, the average session consisted of 7 concept-based pages. Some examples of paths that fall in the same cluster, along with an interpretation of that cluster are presented. These clusters were obtained after partitioning the thresholded similarity graph G_θ with $\theta = 0.95$. The number of clusters generated is a user-defined parameter and was set to 20 in this case. The format of a path in the results is a sequence of `<category, time-spent-in-this-category>` tuples.

CLUSTER 1 :

users interested in contests

path : / 12 /movies 6 /contests 178.8
path : /contests 142.67
path : /coffeehouse 5 /contests 183
path : /contests 172
path : / 10 /contests 143.14

CLUSTER 2 :

users who glance through the articles

path : / 22 /articles 22
path : / 20 /articles 20
path : / 21 /articles 21
path : / 19 /articles 19
path : / 20 /articles 19

CLUSTER 3 :

users who spend time in authors and articles

path : / 148 /authors 6 /articles 77
path : /authors 290 /articles 290
path : /authors 295 /articles 295
path : / 33 /authors 90 /articles 475.75
path : / 32 /authors 91 /articles 425.57

CLUSTER 4 :

users interested in men-women relation-ships

path : / 6 /wo-men 221.43
path : / 10 /wo-men 197
path : / 4 /coffeehouse 32 /philosophy 65 /wo-men 1265.67
path : /wo-men 230
path : / 20 /wo-men 208.57

CLUSTER 5 :

users who read the articles

path : / 39 /articles 98 /misc 17 /articles 2649.9
path : / 9 /articles 2666
path : /authors 26 /articles 2561.5
path : /misc 20 /articles 77 /misc 32 /articles 43 /authors 16 /articles 2373.1
path : / 11 /articles 33 /authors 60 /articles 2538.625

CLUSTER 6 :

users interested in philosophy

path : /biztech 14 /philosophy 2202.4
path : /coffeehouse 12 /philosophy 1991.05
path : / 12 /philosophy 2339 / 138.29
path : / 14 /wo-men 69 /philosophy 1421.5
path : /philosophy 1905 /books 119.06

Next a few outliers are presented from $V' = (V_0 - V_\theta)$ for $\theta = 0.95$:

path : / 12 /coffeehouse 350 /fun 97 /personal 31 /wo-men 49 /books 1152
path : / 38 /philosophy 12 /books 26 /coffeehouse 20 /wo-men 101 /fun 6 /movies 24 /personal 53
path : / 22 /personal 771 / 1 /personal 27 /misc 484 /personal 56
path : / 14 /coffeehouse 9 /philosophy 59 /movies 23 / 131 /authors 5 /articles 291 /misc 83
path : /creative 12 /movies 9 /philosophy 124 /books 75 /personal 139

The change in the size of the sets V_θ and $(V_0 - V_\theta)$ with change in θ is shown in Fig. 5. As θ increases, only highly similar paths go into the clustering phase thereby resulting in extremely pure clusters. However, most paths are discarded as outliers. On the other hand, for lower values of θ , though the number of outliers is less, the quality of the clusters degrade. It is a trade-off and the value of θ needs to be decided depending on the application.

In order to study the results of clustering without taking the time factor into account, we assumed that the time spent in each of the concept-based pages by each user is unity and repeated the experiments. It was observed that a

lot of paths which were in different clusters came into the same cluster which was not always desirable. For example, the boxed paths in cluster 2 and cluster 5 in the above examples had a similarity of 1 and appeared in the same cluster. One can see that the behavior of these two users were clearly different as the former only glanced through the articles whereas the later actually read them. This extra information comes from the time component and turns out to be very important for a meaningful analysis of weblogs.

7 A Few Comments

Let us take a closer look at the techniques we have proposed in terms of the scalability and some of the design parameters involved.

- ⇒ **Scalability** : The path-similarity algorithm essentially computes the similarity between each pair of a total of P paths. The average-case complexity is $O(P^2\bar{L})$, where P is the number of paths under consideration, and $\bar{L} = E(YZ)$, where $Y = \max(X_1, X_2)$, $Z = \min(X_1, X_2)$, and X_1, X_2 are random variables for the length of a pair of paths taken jointly. This is better than the substring based approach [7] for which $\bar{L} = E(YZ^2)$. In comparison to the raw path with an average length of around 12-15 pages, the concept-based approach, in which group of pages map to a concept or *metapage*, reduces the average length of a path significantly, resulting in a significant reduction in complexity.
- ⇒ **Choice of θ** : The choice of θ determines the nature of G_θ and hence the nature of the clusters. For example, for $\theta = 1$, G_θ has nodes only corresponding to pairs of paths that are exactly equal and everything else is treated as an outlier; for $\theta = 0$, $G_\theta = G_0$ and the algorithm ends up pushing real outliers into nearby clusters. If one is interested in very pure clusters⁴ only, one shall keep the value of θ *high* where *high* will depend on the distribution of edge weights in G_0 . On the other hand, if one is interested in categorizing most of the paths in some group or the other, a *low* value of θ is desirable, once again depending on the distribution of edge weights in G_0 .
- ⇒ **Defining the Concepts** : We have defined the *concepts* as the branchings from the homepage of the website. This is the most simple and naive definition one can think of. However, there are ways of defining a concept in a more involved manner. One can cluster the documents in the website based on their content and define each cluster to be a concept. One can also take into consideration the ontology of the pages in the website and think of a multi-resolution concept. This is one aspect that needs further investigation.

8 Concluding Remarks

Current approaches to path clustering using correlations/associations as well as Markov-like approaches fail to use time information, and do not scale well with longer paths. A novel and noteworthy approach to this problem is taken by the Web Utilization Miner (WUM) system [2], in which individual paths are combined into an aggregated tree, and queries corresponding to desired path patterns are mapped onto the intermediate nodes of this tree structure. Again time information is not used.

In the paper, by defining a suitable similarity measure between any pair of paths, we map the paths into a similarity space. This technique enables one to naturally incorporate both path and time information. The idea of thresholding the similarity graph is particularly useful for data-mining applications since it is a good way of reducing the size of the data to be handled in the clustering stage, and being more robust against outliers. A fast, scalable and efficient graph-partitioning algorithm has been used to do the clustering. It avoids curse-of-dimensionality problems encountered in traditional clustering methods (K-means etc.) applied to high dimensional spaces. The superiority of graph-partitioning methods for clustering has also been demonstrated in other web analytics domains such as clustering of web documents [14]. Another contribution of this paper is the use of metadata such as pre-existing category labels. This captures the notion of *content* at a more abstract level. It is possible that extracting textual and meta-tag information from the contents of the pages being viewed can further enhance the quality of user clustering, provided the additional information complexity can be managed properly.

⁴purity may be defined by the completeness of the subgraph induced by the nodes in the cluster

Acknowledgements

This research was supported in part by the NSF under Grant ECS-9000353, and by a gift from Intel. We want to express our gratitude to Satya Prabhakar and Sangeeta Kshetry for providing the sulekha weblog data. All processing was done in a user-anonymous fashion to preserve the privacy of the visitors to sulekha.com.

A Property of G_θ

Property : There are no isolated nodes in the graph G_θ .

Proof : If possible let π_i be an isolated point in G_θ . Since $\pi_i \in V_\theta$, we must have some $\pi_j \in V_0, j \neq i$ such that $|e_{ij}| \geq \theta$. Again, since π_i is isolated, $\forall \pi_j \in V_0, j \neq i$, we must have $|e_{ij}| < \theta$. We note that both the conditions cannot hold good at the same time. Hence, there cannot be any isolated points in G_θ .

References

- [1] R. Cooley. *Web Usage Mining: Discovery and Applications of Interesting Patterns from Web data*. PhD thesis, Dept. of Computer Science, University of Minnesota, May 2000.
- [2] M. Spiliopoulou and L. C. Faulstich. WUM: A tool for web utilization analysis. In *Extended version of Proc. EDBT Workshop WebDB'98*, pages 184–203. Springer Verlag, 1999.
- [3] H. Mannila and C. Meek. Global partial orders from sequential data. In *Proc. 6th Intl. Conf. on Knowledge Discovery and Data Mining (KDD2000)*, pages 161–168, Aug 2000.
- [4] J. Pitkow and P. Pirolli. Mining longest repeating subsequences to predict world wide web surfing. In *Proc 2nd USENIX Symposium on Internet Technologies & Systems (USITS'99)*, Oct 1999.
- [5] Y. Fu, K. Sandhu, and M. Shih. A generalization-based approach to clustering of web usage sessions. In M. Spiliopoulou B. Masand, editor, *Web Usage Analysis and User Profiling*, pages 21–38. Springer, 2000.
- [6] C. Shahabi, A. M. Zarski, and V. Shah J. Adibi. Knowledge discovery from users web-page navigation. In *Proc. 7th Intl Conf on Research Issues in Data Engg*, pages 20–29, 1997.
- [7] A. Banerjee and J. Ghosh. Concept-based clustering of clickstream data. In *Proc. 3rd Intl. Conf. on Information Technology*, pages 145–150, Dec 2000.
- [8] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms*. MIT Press, 1990.
- [9] Vladimir Dančik. *Expected Length of Longest Common Subsequences*. PhD thesis, University of Warwick, 1994.
- [10] A. V. Aho, D. S. Hirschberg, and J. D. Ullman. Bounds on the complexity of the longest common subsequence problem. *J. Assoc. Comput. Mach.*, 23:1-12, 1976.
- [11] D. S. Hirschberg. Algorithms for the longest common subsequence problem. *J. ACM*, 24:664–675, 1977.
- [12] R. A. Wagner and M. J. Fischer. The string-to-string correction problem. *J. Assoc. Comput. Mach.*, 21:168-173, 1974.
- [13] G. Karypis and V. Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on Scientific Computing*, 20(1):359–392, 1998.
- [14] A. Strehl, J. Ghosh, and R. Mooney. Impact of similarity measures on web-page clustering. In *Proc. 7th Natl Conf on Artificial Intelligence : Workshop of Artificial Intelligence for Web Search (AAAI 2000)*, pages 58–64. AAAI, July 2000.