FORMING, VALIDATION, VERIFICATION AND UPDATE OF WEB CLIENT CLUSTERS USING PREFETCHING AND SOCKET CLONES METHODS

3.1 INTRODUCTION

With the web growing in leaps and bounds, the number of applications on the web has increased many folds (Xiao and Zhang 2001, Xie and Phoha 2001, Sit et al 2002). The number of people using these applications also has increased many folds. Thus there has been an urge in people to use these applications. The increased load on servers thus has been a cry of the moment. Solutions have been sought after to provide for an easy access to the increasing burden on these servers. Things have changed so much these days’ that people have started to buy commodities that are personalized; web surfing thus need not be left behind too. When web surfing is personalized it brings into focus prefetching (Xiao and Zhang 2001). Prefetching brings into question the ability of a particular server or a system to be able to predict the course of surfing. This means that web pages will be accessed in the immediate future.

But even before the question of service is discussed, it is imperative to finalize the clustering of these clients in question. How they should be clustered, what are the measures that can be used to cluster these users are all the things that will be analyzed. Web clients are clustered into groups so that there will be an underlying level of similarity between them in terms of access areas. The purpose of undertaking such an endeavor is multifaceted, namely reduced load on servers, faster retrieval of relevant data, customization, etc. Thus a reduction in overhead cost is achieved.
The objective is crystal clear, i.e. clusters of similar web clients can be serviced better. How relevant will it be for someone 15 years younger to get pop-ups on life cover, etc? Wouldn’t it then be more appropriate if they get a pop-up that pertains more to science and technology or sports? The basic objective of client clustering can thus be defined as customization. Customization can only be achieved if it is targeted to a particular section of the browsing community. Specific pages can be retrieved for particular sections of the web surfing community which actually will translate into reduction of time and overhead cost when it is done beforehand. Web pages can be prefetched if the server is able to predict what to prefetch. This study also discusses ways in which servers will be able to predict the pages that will be surfed in the immediate future. And in this way, servers can thus retrieve these pages either by storing them remotely or by storing them on board the system.

3.2 BACKGROUND

Xiao and Zhang (2001) proposes an approach for measuring similarity of interests among Web users, based on the interest items collected from Web user's access logs. A matrix-based algorithm is then developed to cluster Web users such that the users in the same cluster are closely related with respect to the similarity measure. As an application example, a Web document prefetching technique is proposed. It utilises the similarity measure and clusters obtained. Experiments have been conducted and the results have shown that the proposed clustering method is capable of grouping Web users with similar interests, and the prefetching method is practical.

Sit et al (2002) presents a new network support mechanism, called Socket Cloning (SC), in which an opened socket can be migrated efficiently between cluster nodes. With SC, the processing of HTTP requests can be moved to the node that has a cached copy of the requested document, thus
bypassing any object transfer between peer servers. A prototype has been implemented and tests show that SC incurs less overhead than all the mentioned approaches. In trace-driven benchmark tests, this system outperforms those approaches by more than 30% with a cluster of twelve web server nodes.

Hence the proffered approach aims at clustering the web clients to provide some sort of customization. But a basic challenge is how they should be clustered? What are the measures that can be used to cluster these users? This research study aims at providing a web clustering procedure for various clients which is economical and makes the web user comfortable while surfing. A matrix method (Xiao and Zhang 2001) is used to cluster users based on these similarity scoring features. Users are compared against each other; their similarities are measured; and a matrix is formulated. Then, the users that express a significant amount of similarity are actually considered whilst the others are dropped. The resulting matrix is a reduced form of the original matrix and it contains only significant similar sets of users for a particular measuring feature. The matrix is then decomposed and put through the tests to eventually give the sets of users that have a similar kind of interest. The basic objective of client clustering can thus be defined as customization. Customization can only be achieved if it is targeted to a particular section of the browsing community. Specific pages can be retrieved for particular sections of the web surfing community which actually will translate into reduction of time and overhead cost when it is done before hand. Web pages can be prefetched (Xiao and Zhang 2001) if the server is able to predict what to prefetch. This investigation also discusses ways in which servers will be able to predict the pages that will be surfed in the immediate future. Thus servers can retrieve these pages either by storing them remotely or storing them on board the system. The basic approach in this study is to collect the user access details from the server logs and analyze them to find
out access patterns that are distinguishable. These patterns are then stored to give the server an idea of what to retrieve when the particular user is surfing the web. Web clients are clustered into groups so that there will be an underlying level of similarity between them in terms of access areas. The clustered web clients are serviced using socket cloning method proposed by (Sit et al 2002). The purpose of undertaking such an endeavor is multifaceted, namely reducing the load on servers, faster retrieval of relevant data, customization, etc. Thus a reduction in overhead cost is achieved.

Xie and Phoha (2001) propose a novel approach for clustering web site users into different groups and by generating common user profiles. These profiles can be used to make recommendations, personalize websites, and for other users such as targeting users for advertising. By using the concept of mass distribution in Dempster-Shafer’s theory, the belief function similarity measure in this algorithm adds to the clustering task the ability to capture the uncertainty among web user’s navigation behavior.

3.3 SYSTEM MODEL

Given a set of $m$ users $U = \{u_1, u_2, \ldots, u_m\}$ who access $n$ web pages $P = \{p_1, p_2, \ldots, p_n\}$ of their subjective choices in any given time interval. There is some usage value associated with every user for a particular page denoted by $use(p_i, u_j)$. The value of $use(p_i, u_j)$ is 1 when page $p_i$ is accessed by user $u_j$. The value of $use(p_i, u_j)$ is 0 otherwise. The list of users is usually obtained from the system administrator.

3.3.1 Similarity Measures

Type #1 similarity
Type #1 similarity is defined based on the total common pages accessed and is given by sim1:

$$\text{Sim1}(u_i, u_j) = \sum_k (\text{use}(p_k, u_i) \times \text{use}(p_k, u_j)) / \sqrt{\sum_k \text{use}(p_k, u_i)^2 \times \sum_k \text{use}(p_k, u_j)^2}$$  \hspace{1cm} (3.1)$$

where, $\sum_k \text{use}(p_k, u_i)$ is the total number of pages that were accessed by the user $u_i$, $\sum_k \text{use}(p_k, u_j)$ is the number of pages accessed by the user $u_j$, $\sum_k (\text{use}(p_k, u_i) \times \text{use}(p_k, u_j))$ being the common number of pages accessed by the users $u_i$ and $u_j$. The measure of similarity is based on the number of common pages accessed between 2 users.

**Type #2 similarity**

The Type #2 similarity measure of similarity can be computed by counting the number of times these users actually access these common pages at all the sites visited. The similarity measure here is given by sim2:

$$\text{sim2}(u_i, u_j) = \sum_s \sum_k (\text{acc}_s(p_k, u_i) \times \text{acc}_s(p_k, u_j)) / \sqrt{(\sum_s \sum_k (\text{acc}_s(p_k, u_i))^2) \times (\sum_s \sum_k (\text{acc}_s(p_k, u_j))^2)}$$  \hspace{1cm} (3.2)$$

where, $\text{acc}_s(p_k, u_i)$ is the total number of times user $u_i$ accesses the page $p_k$ at the web site $s$.

**Type #3 similarity**

Type #3 similarity based on the most accurate measure of similarity would be the actual amount of time spent in viewing the pages involved. Then let $t(p_k, u_j)$ be the amount of time user $u_j$ spends on viewing page $p_k$. $t = 0$ if the page was not viewed at all, 1 if otherwise. Then type #3 is given by sim3:

$$\text{sim3}(u_i, u_j) = \sum_k (t(p_k, u_i) \times t(p_k, u_j)) / ((\sum_k (t(p_k, u_i))^2) \times \sum_k (t(p_k, u_j))^2)$$  \hspace{1cm} (3.3)$$

where, $\sum_k (t(p_k, u_i))^2$ is the square sum of the time the user $u_i$ actually spent on viewing the pages in question. The denominator here is the inner product
over time spent on actually viewing the common pages by users $u_j$ and $u_i$. In this method of scoring, even if two users access the same set of pages their similarity count might be less than 1, owing to the fact that the amount of time the user spent viewing these common pages differs to a large extent. This is justified by the fact that there are pages which are simply just used as a part of the navigation path that the user uses. These pages thus cannot be used as a criterion for measuring the similarity between the users in question. For credible similarity, measuring pages that are actually viewed with some intent of analysis only is taken into account.

**Type #4 similarity**

The next and final measure of similarity is type #4 which involves the actual path of navigation that users’ $u_j$ and $u_i$ implement. The natural angle between the paths is calculated, it’s actually the cosine between the 2 paths, i.e. $\cos(\theta Q^i Q^j)$, where the final formula will contain the inner products of the paths involved over the feature spaces of the paths $Q^i$ and $Q^j$. The similarity between web users is actually very much application dependent. In cases of subjective reasoning, if a more complicated similarity measure is needed, an applicable similarity scoring feature must be defined for the said case.

When a navigation path is taken into context the links that are involved in the navigation path are primary. In the example discussed below, the italicized words are the links in the web site that correspond to the connectivity graph under examination.
Six users are taken into consideration:


With the data given above for six different users, a similarity matrix is computed which results in a $m \times n$ matrix called the similarity matrix. Using the first measure of similarity $\text{sim}_1$ the similarity matrix $\text{SM}$ is given as follows:

**Table 3.1 Similarity statistic of six different users based on sim1 estimation**

<table>
<thead>
<tr>
<th></th>
<th>$U_1$</th>
<th>$U_2$</th>
<th>$U_3$</th>
<th>$U_4$</th>
<th>$U_5$</th>
<th>$U_6$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$U_1$</td>
<td>1</td>
<td>0.204</td>
<td>0.183</td>
<td>1</td>
<td>0.817</td>
<td>0.913</td>
</tr>
<tr>
<td>$U_2$</td>
<td>0.204</td>
<td>1</td>
<td>0.224</td>
<td>0.189</td>
<td>0.25</td>
<td>0.224</td>
</tr>
<tr>
<td>$U_3$</td>
<td>0.183</td>
<td>0.224</td>
<td>1</td>
<td>0.676</td>
<td>0.447</td>
<td>0.8</td>
</tr>
<tr>
<td>$U_4$</td>
<td>1</td>
<td>0.189</td>
<td>0.676</td>
<td>1</td>
<td>0.756</td>
<td>0.845</td>
</tr>
<tr>
<td>$U_5$</td>
<td>0.817</td>
<td>0.25</td>
<td>0.447</td>
<td>0.756</td>
<td>1</td>
<td>0.671</td>
</tr>
<tr>
<td>$U_6$</td>
<td>0.913</td>
<td>0.224</td>
<td>0.8</td>
<td>0.845</td>
<td>0.671</td>
<td>1</td>
</tr>
</tbody>
</table>
Looking at the above given matrix in Table 3.1, it is easy to compare it to the log given before it. The first and the fourth users have similar patterns of visitation. Hence the measure of similarity between them is 1. While the access paths of the first and the third users are not the same, the similarity measure is 0.183. Between the first and the sixth users, the degree of similarity is given as 0.913 which naturally indicates a higher degree of similarity between the users than in the case between the first and the third.

Matrices for all measures of similarity are computed. The data are considered raw and need to be preprocessed to extract items of similarity and statistical significance from them. Clustering generally brings together items of similar interest into consideration with respect to a particular measure of similarity. A similarity threshold is thus determined to rid the matrix of any insignificance. Notice in the above matrix that the leading diagonal is 1, and that the halves of the matrices on either side are the same. A threshold value of 0.7 is used to eliminate any insignificant relations that exist.

### 3.3.2 Permutation of the Matrix

With insignificant elements in the matrices being eliminated, the matrices now have clusters of users that have similar tastes with respect to a certain similarity measure. Thus, the higher the value of SM (i,j) the closer the users u_j and u_i would be. The matrix after the computation will thus have elements with similar interests closer to each other. The next step will be to find out a dividing point that decomposes the said matrices. To permutate the given matrices a variable called the GA (Global Affinity) is first found out. The GA is an indicator of how closely users with similar interests are situated in the matrices.
GA value of matrices is given by:

\[ GA(SIM) = \sum_{i=1}^{n} A(i, i-1) + A(i, i+1) \]  

(3.4)

Then, swapping of the rows is done to compute the GA. If it is found higher, then rows are swapped to maintain relative positions of the columns.

### 3.3.3 Decomposition of the Matrix

A dividing point in each matrix is to be calculated for the clustering of the users into 4 sub-matrices with similar kind of interests. The 4 parts will thus be called Upper left, Upper right, lower left and Lower right sub-matrices.

### 3.4 SERVICE MODES

These days, cluster based web based servers are an impending necessity as a solution to high traffic web hosting. The solutions that are around suffer from the maladies that arise from the dispatcher based systems. These systems have a major drawback of single-point-of-failure. Because all the traffic is centered on one point in the network, these points in the network become crucial to the proper functioning of the network in general. To overcome this big disadvantage Distributed Packet Writing was invented which uses more than 1 dispatcher.

This method has its variations. The Layer-7 methodology implements the TCP splicing and TCP handoff. In TCP splicing, the dispatcher (Cassalichio and Collajanni 2001, Sit et al 2002) itself is used as a proxy to reroute the HTTP response back to the client. In the TCP handoff methodology, only the endpoint is passed to the resultant node so the server
literally bypasses the dispatcher and directs the HTTP responses back to the client directly as shown in Figure 3.1.

![Figure 3.1 TCP splicing](image)

The network support mechanism that will actually aid the service to the clusters is called socket clone and it will be discussed shortly. A socket clone is an opened socket which can be migrated between clusters.

### 3.4.1 Socket Clone

A socket clone (Sit et al 2002) is an opened socket which can be efficiently migrated between cluster nodes. Thus subjective to the clusters, the needed processing is thus shifted to the cluster that mostly has some mechanism like a local cache for storage of prefetched documents.

The clone has three components as shown in Figure 3.2:

- SC client
- SC server
- Packet router
The SC client provides a system call interface to the web server in the node. If the web server is unable to handle the request, it deputizes another server to handle the request. It does so by using the system call given to it to clone the socket. The SC client packs all the information of the opened socket and passes it onto the remote node through a persistent connection. This is called an SC message. When the new server that was deputized receives the message, it creates a socket called a clone. The status is now worked backwards. The clone now becomes native to the new server and subsequent packets will go through its protocol stack. The output is generally sent to the Client. Usually, after successful cloning the SC server will inform the packet router to route the packets to the clone’s node. During the processing, though packets from the client reach the original node and are then routed to the clone’s node HTTP responses are sent directly to the clone’s node.

The clone and the original socket have the same state after cloning. The states change only after they have actually started sending out packets. As further processing of the requests from the client takes place their states to
be synchronized. This is done by a synchronizer which is probably a timer like device. A state of consistency has to be thus maintained to enable the original socket to process other requests further. To keep up the synchronization, the status of the original clone is usually constantly updated to keep it in consistent form. It is usually done in the form of inter-nodal communication and it rarely affects scalability or processing. Here, explicit synchronization is not needed: the router is in charge, it uses the acknowledgement packets received from the client to update the status of the original socket. Then it updates the status and also aids in continuing with the normal transaction. Thus a triangular mode of implementation is introduced.

3.5 PROPOSED ARCHITECTURE

Figure 3.3 Proposed system architecture

The client interests are tracked using its IP address and the major properties of the client choices are extracted as shown in Figure 3.3. This raw information is used to form the web client clusters based on the similarity measurements which are proposed earlier. The most frequent items are determined and formed as a cluster. These clusters are under the continuous process of verification, validation and updation (FVVU). Now, the clients are served effectively by the servers by using this clustered information in a customized manner. In this regard, TCP splicing methods and cloning methods are used to service various requests from clients.
3.6 RESULTS

3.6.1 Implementation of the Clone in a Web Server

The server software in a cluster node first establishes connections with the clients directly and parses the request. A mapping function is used to decide on which node would process the request. The fact that the clustering of clients is already accomplished is to be remembered here. The mapping function itself can be of many types:

- Load based
- Location based
- Cache based

If a particular server is chosen to handle the request, it then handles it. Else it simply clones the request and passes it onto the server that will actually service the request. For persistent HTTP requests, an efficient and scalable mechanism, for cloning the socket multiple times, is provided so that every request is served by the most appropriate server. In the event of a request being handled by a clone, then a server in the clone’s node will subsequently ignore the requests put to it. Thus it is now clear that clones are only meant for subjective processing and not used generally for routing or any general purpose. The main server usually handles the request and may choose to clone the request or handle it itself. Pipelining of requests is also allowed where a browser sends for a request even before a complete response from the previous request is got. For error checking parity bits are counted. After receiving the final response from the server the router is usually directed to process the next request.

3.6.2 Hybridization

It is important here to run through the entire length of the process before the final concept of hybridization is introduced. The system first makes
clusters and uses some form of enumeration for identification. These clusters will have activation flags which will operate on the binary values of 0 and 1. One (1) indicates the cluster being serviced and zero (0) on the contrary. If the value of the flag is 0, then no system resources are assigned. When the search criteria are given they are first checked and then related documents are pre-loaded. Facilities for dynamically modifying the clusters of users are also included. A router table like structure is maintained where cluster ids’ and their usual access caches are addressed. So when a client logs on to the network the matching cluster of the client is found based on previous records. Prefetching is done, however, in the event of the user trying to access a page that is not usual to the personalized interest. Then the user is almost immediately assigned to a matching cluster if found and documents pertaining to the new cluster are prefetched. In the event of simultaneous accesses, common pages are cached and are referenced simultaneously for the different clusters that are being served. Time slicing is done when servicing simultaneous requests. Each time, the slot is split into equal parts based on the number of clients that need to be serviced.

3.6.3 Performance Analysis

It is observed from Figure 3.4 that the performance of the proposed algorithm increased with the cloning of the client requests, by the dispatcher to an appropriate server. The performance of the system is good when TCP splicing is done on various servers that are running clusters. The utilization is good for various requests of web pages and the same is shown in Figure 3.4. Figure 3.5 shows the performance of the system with and without clustering. It is to be noted that the clustering is more successful when there is similarity between web client requests. Figure 3.6 displays the performance of the servers that cluster data. It is observed that the clustering is more successful because every cluster is utilized by more than 90 % and it is an indication of good clustering. The FVVU factor also affects the performance of the proposed system.
Figure 3.4 Performance of the system

Figure 3.5 Performance of the system with and without clusters
3.7 SUMMARY

This investigation has thus devised a robust and dynamic plan for clustering web clients. The implementation of the socket clone, along with the matrix based method of classifying clients, is a novel idea that has proved efficient by its performance. The elimination of unwanted clients from clusters has also proved to be a good plan on boosting the performance of the system as a whole. With systems like this in place safer, faster and more customized surfing of the web is to be expected in the not-too-distant future. However, the key performance factor of the system lies with Formation, Verification, Validation and Update (FVVU) of the clusters.