On Proxy Agents, Mobility, and Web Access *

Anupam Joshi

Department of Computer Science and Electrical Engineering
University of Maryland, Baltimore County
Baltimore, MD 21250
E-mail: joshi@cs.umbc.edu

With the emerging need for ubiquitous access to information, web access from mobile clients is gaining increasing importance. Unfortunately, the underlying protocols of the Web are not designed to support operations from a resource poor platform in a low bandwidth, disconnection prone environment. Efforts to create systems to support mobile browsing have typically been proxy based. However, such solutions have recently been criticized due to their non scalability. Developments in ad-hoc networking also threaten the viability of proxy based approaches. In this paper, we critically examine the proxy based approach and its place in mobile access to the web in particular, and networked multimedia information in general. We argue for a hybrid approach that uses both proxy based and end–end techniques as appropriate, suggest when each is more appropriate, and present a preliminary implementation.

1. Introduction

With the advent of dynamic and executable content, integration of security mechanisms, and emerging metadata standards, it is clear that the Web is transforming into the basis of a globally distributed computing and information access system. Concurrent to the explosion of the Web is the advent of mobile computing, which provides users ubiquitous access to the resources and services of the networked information infrastructure. This has enabled new applications in the areas like tele-medicine, public information services, battlefield awareness and education etc. For example business users with palmtop/laptop type devices (aka Road Warriors) constitute a large and growing segment of users. Their

* This work was supported in part by an IBM faculty development award and NSF awards IIS9801711 and IIS9875433.

work typically involves accessing and modifying corporate information repositories with multimedia data over low bandwidth connections typified by wireless or phoneline access. More often than not, the applications they use are web enabled and use the browser as a "thin" client. Given the emergence of the Web as the "kernel" for distributed information and computation access, a system to permit mobile web access can allow a variety of networked resources to be accessed from mobile clients.

As was discovered fairly early on[17,19,21], the web (more specifically, the HTTP protocol) is designed to work in wired, high bandwidth environments, and does not operate particularly well when the access point is a mobile host. The reasons behind this such as low/variable bandwidth, disconnections, etc.[19, 10] are quite well known. Further, the mobile host is typically resource poor. Even though the very high end laptop machines can now deliver performance comparable to low-end to moderate desktops, most "thin and light" subnotebooks and PDA/PCS type devices are constrained in terms of CPU, power, memory, disk, display capabilities etc.

These problems are compounded by the current model of web information access, where the user has to navigate the information space to find information that s/he needs. Bandwidth, a precious resource in the mobile scenario, is thus wasted by transmitting a lot of useless data across the wireless channel in the browsing process. This problem relates to *personalization* of the web space, *i.e.* locating the information needed by a user and possibly filtering it. Given the synchronous nature of the HTTP protocol, a continuous connection is also required throughout the information retrieval process. Disconnections typically require that the information be fetched again. More generally, the problem is a lack of *asynchronous operation* – *i.e.* not allowing the mobile user to disconnect in the process of information retrieval. Further, web servers have no knowledge about the client's resources and assume that it can handle the data sent by them. For example, a user may follow a hyper-link to a VRML world on a machine that does not have the capability of handling a VRML plugin. The client will retrieve the data from the server, and then not be able to use it. This results in consumption of available bandwidth and wastage of time while transferring data, and potentially even the plugin, across the wireless network. The cause is a *capability mismatch* in terms of handling content between the client and the server. Moreover, the mobile client expends battery resources in receiving this data, which it cannot use. With the ever increasing use of different types of multimedia content on the web, this phenomenon represents a growing problem.

An argument can be made that these bandwidth and resource related problems are transient, and eventually wireless network speeds as well as the resources available on mobile platforms will increase. That may well be the case, yet the speeds on wired networks and the resources (memory, CPU speed, etc.) on static hosts will also increase. As has been amply demonstrated over the last decade, software catches up and uses all available hardware resources – Existing applications will evolve, and novel applications will be created, to use these enhanced capabilities. Thus while the absolute performance measures of mobile systems will undoubtedly improve, what we call the *bandwidth gap* and *resource gap* will remain.

In this paper, we discuss the relative merits of the proxy based vs end–end approaches to supporting mobile web access. We argue that given the likely diversity in "mobile" technologies, ranging all the way from Bluetooth like devices to satellite based WANs, an architecture which combines the two approaches and uses each as appropriate is needed. We present an implementation, and present ideas on where to use proxies and where an end-end approach is more suited.

2. Related Work

A considerable amount of work has been done in the area of information access from mobile platforms, using mostly the client-proxy-server model. Due to space limitations, we present here some of the larger efforts in enabling web access from mobile computers. Other related work done earlier includes the TeleWeb system of Schilit *et al.*, the notion of stream transducers advanced by Brooks *et al.*[7], location specific personalization[33], IBM's WebExpress[8], and Rover[14].

Significant work in this area has been done by the Daedalus group at Berkeley. In GloMop [22,11], the proxy performs *distillation* of the document received from the server before sending it to the client. Distillation is defined here as a highly lossy, real-time, datatype-specific compression that preserves most of the semantic content of the document. For instance, GloMop performs transcoding of motion JPEG to sub-sampled H.261 for video data. A more formal model for proxy functionality (TACC), alongwith an overview of their system, is described in [6]. More recently, this group has used a similar approach to create a split browser[12] for the Palmpilot PDA. Note however that their approach is essentially proxy based.

The Mowgli system [26] consists of two mediators located on the mobile host and the mobile-connection host which use the Mowgli HTTP protocol to communicate with each other, reducing the number of round-trips between the client and server. Mowgli reduces the data transfer over the wireless link in three ways: data compression, caching, and intelligent filtering.

The notion of web intermediaries to affect transcoding and personalization related functionalities is also the focus of IBM's WBI[3] system.

In the work of Noble *et al.*[29], the proxy is developed in the context of what the authors term *agile, application aware adaptation*. Basically, they allow an application to register with the OS its expectations about a resource and the variability it can tolerate. The Odyssey system monitors resources, and informs the applications via upcalls when the resource value strays outside the bounds decided by the application. The application can then adapt its behavior. For web browsing in particular, a module called Cellophane on the client transforms HTTP requests from Netscape into file operations on Odyssey web objects and selects fidelity levels for images which are forwarded to a distillation server. However, this approach is specific to the Odyssey file system and requires a modified version of the Net BSD kernel. This also requires the addition of a module on the client.

Another line of work has sought to support disconnected operation using local caching. Several commercial offline browsers download documents into a local disk for later viewing. The user specifies a set of links, similar to the bookmarks file, which the application downloads and caches. Some of these run a proxy web server on the portable machine to deliver cached documents. Offline cache, a Netscape Netcaster feature, allows users to download information from an information broadcast channel, save it in a cache on their system's hard drive, and view it at a later date. The common problem of these systems is that they propose to conserve one scarce resource (connection bandwidth) in the mobile scenario by using up another scarce resource (disk space).

The use of "personalization" techniques and recommender systems, which has recently gained popularity in information retrieval, can also help the wireless web access effort by limiting the information flow on the wireless channel. We briefly describe some work in the recommender systems area from the IR community. GroupLens [25] is a system to help people find usenet news articles they will like in the huge stream of available articles. News reader clients display predicted scores and make it easy for users to rate articles after they read them. Rating servers, called Better Bit Bureaus, gather and disseminate the ratings. The rating servers predict scores based on the heuristic that people who agreed in the past will probably agree again. The Referral web [23] is based on the idea that searching for a piece of information is equivalent to searching the social network for an expert on the topic together with a chain of personal referrals from the searcher to the expert. The Savvy Search[13] meta-search engine is designed to query other search engines likely to return useful results and responding to fluctuating load demands on the Web. Savvy Search learns to identify which search engines are most appropriate for particular queries, reasons about resource demands and represents an iterative parallel search strategy as a simple plan. Phoaks[34], Siteseer[31] and Fab[2] are also recent recommender systems.

3. Proxy based vs. End–End approaches to Mobile Web Access

There exists a large body of work which handles the problem of *capability mismatch* for multimedia content in wireless web access. The solution, typically, has been to use a client-proxy-server model(e.g. [4,11,7]). The proxy transcodes multimedia formats, most often images, according to some predefined rules, usually in some manner that trades quality for bandwidth.

However, as PDA or thin-light notebooks become more popular, the proxy's functionality will increase. For example, some proxies now seek to deal with videos as well as images. This includes work done at Berkeley, as well as our own recent work[4]. We have also examined other questions that relate especially to PDA type mobile clients, such as what to do with active content and HTML fonts / styles which the PDAs typically cannot handle. There are also proxy approaches that re-render the HTML in a format appropriate for the PDA before transmitting. This added functionality of the proxy increases the computational resources it requires.

We argue that a purely proxy based solution will become increasingly non scalable, especially with the number of users connecting wirelessly expected to grow. We note that limited functionality proxy systems have been developed recently that are quite scalable. A good example is Inktomi's traffic server, or the proxy developed by the Daedalus project at Berkeley for dial in connections. However, with the advent of palmtop type devices, the transformation needed by the client, and hence the computational resources needed by the proxy to affect it, increase significantly. While workstation clusters supporting proxies can possibly be deployed to provide computational resources, it is not clear that this *proxy only* approach provides the best solution to the problem. The role of proxies has been recently questioned elsewhere as well – there is some debate as to whether proxy based solutions are really needed to provide networking services to mobile clients[1].

Moreover, the proxy based approach typically assumes that the data is being served by a host on the wired side. This means that a proxy can be run on some host with lots of MIPS on the wired side which is on the path from the server to the mobile client. Most often, this is at the mobile support station. Clearly, in ad-hoc networks that will be engendered by Bluetooth like devices, such an assumption would be fallacious. The alternative is to make the server itself provide data in a format that is most suited for mobile access. This represents an instance of an end-end approach. End-end approaches are well known in networking and systems literature. In the web context, dual versions (graphics heavy vs. text only) of web pages kept at servers represent an end-end approach. To the best of our knowledge, Seshan *et al.*[32] were one of the first to present the notion that the Web clients could use network performance parameters to download documents from a server at different "fidelities", and explicitly mentioned that this was something beyond text only pages. Implicit in their paper was the idea that the server would indeed have different fidelities to present to the client.

Probably the most obvious functionality that can be off-loaded from a proxy is the one found in it most often – transcoding multimedia. The basic assumption underlying transcoding is that the servers will have a document at a fixed "resolution" of a given media, and so the proxy needs to appropriately modify the multimedia content to suit the client side resources and bandwidth. However, recent developments tend to invalidate this assumption. For example, with bandwidth constrains being faced even in wired networks when dealing with videos, new formats (e.g. wavelet based codes) that allow for efficient progressive transmittal of images and videos, and/or reconstruction of data at arbitrary resolutions[36] from the same bitstream are being developed. Moreover, with disk storage increasingly inexpensive, it is not altogether unrealistic to expect that content providers will react to the market and make available multimedia content in multiple fixed reso-

[1] There was a most interesting panel on this theme at MobiCom98

lutions. This is but a natural extension of the "text only" versions of websites that we often see. Maintenance of multiple resolution content is a potential concern, as is the related question of how many fixed resolutions are needed. However, one can broadly divide the accessing devices and their network connections into 4-5 broad categories (e.g. desktop over LAN/DSL/CableModem, desktop over dialin, Laptop over LAN, Laptop over wireless/dialin, PDA/PCS Phone over wireless/dialin). Thus 4-5 "resolutions" of the data will suffice to cover a large majority of client/connection types. This, coupled with the easy (and often free) availability of tools to effect the transformation makes the problem of maintaining different "resolution" content easly managable. For example the mapblast driving directions site (http://www.mapblast.com/) provides driving directions for the in special formats for palmpilot and WinCE devices, in addition to the regular HTMLized format. More recently, AOL has announced differentiated content service for those connecting via high speed links (cable modems, DSL), SprintPCS has announced web access over PCS phones, and Ameritrade has announced access to their online brokerage over PCS phones. It is thus clear that given that given the market, content providers will create and maintain content at multiple resolutions.

With an end-end approach, the server can also carry out more complex transformations (such as summarizing text, converting text to audio, identifying keyframes from a video etc.) based on the client device needs. These will be computed offline and stored. In fact, most of these are too compute intensive to be done on-the-fly at the proxy, and so not possible in a proxy only approach. For example, creating a summary (i.e. identifying key frames) of a video stream of about four seconds took several minutes of time on an SGI O2 machine. For longer videos, the time involved can be several hours.

The end-end approach is predicated upon the client being able to "identify" itself to the server. There has been a very recent initiative (CC/PP from W3C, http://www.w3.org/TR/NOTE-CCPPexchange) to use RDF based HTTP extensions as a standard mechanism for this. The WAP forum (http://www.wapform.org) Wireless Application Environment also provides for mechanisms where state information allows a server to send content optimized for narrowband delivery to a client. Alternatively, the client must have a way to express its preferences for particular resolutions to the server. This is similar in concept to expressing preferences for languages or formats as envisaged in HTTP/1.1 .

The functionality of information personalization has, to the best of our knowledge, not been much explored in the context of supporting mobile web access. The exceptions include work on location dependent information retreival[33]. We have shown[19,24] how personalization can be used to minimize information flow on the wireless network, and have typically included this as a part of proxy functionality. However, this can also be taken away from proxies and passed on to servers. The idea that web sites (more generally, information sources) should "adapt" based on the user has been discussed in the last few years. Besides our own work on web mining[20,27], there is also Etzioni's description of adaptive web sites[30], and the Webminer project of Han *et al.*[38]. Once a user is identified with particular information access patterns, the server can tailor the information returned in response to their requests. The same result can be achieved by involving the user directly in the process (for example, by providing rankings or other feedback), although clearly this is more burdensome for the user. An even more primitive form of this idea can be seen in most "personalized" search engines, where users are asked to explicitly provide keywords that match their interest.

Since the functionality most associated with proxies is seemingly redundant, what then would be their *raison d'etre*? The obvious reason is that we may still want to leave this functionality in, since it is not clear that every server will be able to provide data in the format and resolution that the client needs, or tailor its information content. Moreover, the server may not maintain content appropriate for all possible client/network speed combinations. However, there are other functionalities that the proxy can provide. For one, it can provide disconnection management features. This typically involves maintaining state/checkpoint information for the client, and providing allowable points of disconnection. Second it can provide transcoding of a different kind – transforming multimedia content coded to handle the channel characteristics (e.g. error rates) of the wired networks to codes that are better suited to the wireless channel. A related functionality is protocol translation related. Several specialized protocols have been developed for TCP/IP over a wireless channel, and the proxy could be used, as in Mowgli[26], to transform a standard TCP/IP stream into the specialized formats. Further, by decoupling the data delivery from the server to the client, it can mitigate the effects of low proxy-MH bandwidth on fast servers that have fat pipe connections to the proxy. Also, by caching the transcoded data and hence replicating it, the proxy can help increase availability.

The proxy can also provide information location services. In other words, rather than ask the user to browse the web space to locate the right information, the proxy can accept from user some *query like* formulation of the information needed, and then locate this information from various web sources. This idea is similar to the notion of *web portals* that many vendors are trying to push. In our work, we have combined these functionalities of information location and disconnection management, along with information filtering [16,19], to provide asynchronous operation. In other words, the process of accessing information from the web becomes one where the user asks the proxy for some information, retrieves that information, and rates this information. The proxy maintains state, and provides allowable points of disconnection in these operations.

In summary, the architecture best suited for mobility combines both proxy based and end-end approaches. Some tasks, such as complex transcodings which require significant computational resources are best done in an end–end manner. This would include tasks like summarizing videos, or finding keywords in a document etc. . Others, such as maintaining state for an MH or shared state for collaborative filtering applications, clearly are best handled by a proxy. Other tasks can be done either end-end or at the proxy. For example, if appropriate resolution images are not available at the server then the proxy can transcode images. In the sections that follow, we present a system which implements this combined approach, and then some experimental results.

4. A Simple Implementation

We have implemented a system to combine end-end and proxy based approaches by extending our proxy based Mowser (**Mo**bile Bro**wser**) and W^3IQ (Web Intelligent Query) systems[17,24,18]. Mowser is a transcoding proxy to handle multimedia content, W^3IQ is a proxy that provides for Personalization and Asynchronous Operation.

4.1. Mowser

Mowser runs as a process on the wired side, and is set as the proxy server by the user in his/her browser. Since there are no readily available mechanisms in IPv4 to automatically obtain QoS parameters, the user is asked to set these via a CGI+perl form which asks user to answer simple questions about the type of connection and resources available on the MH. The user can then browse the web as s/he would with any web client. On receiving a request from the MH, the proxy uses the preferences set by the MH to obtain and/or transform the data so as to serve it in the most suitable resolution. Default preferences are used if no values had been set by the MH.

4.1.1. End-End Framework: Using Server Capabilities

Traditionally[17,11,8], the request from the client is passed as is to the target server, while the return stream's multimedia content is altered. In our work, we alter the request as well so as to add information that will enable the server to send content in a resolution appropriate for the client if possible.

Since there are no explicit mechanisms to enable this in HTTP as yet, we extend and adapt the content negotiation scheme of HTTP/1.1 to demonstrate the idea. The basic idea of content negotiation is to allow a client to get a document in some particular language or character encoding etc. The server can automatically send the right representation if the client sends the preferred representations as part of the request. We extend this idea to multiple multimedia resolutions for a document. For example, an image file may be made available in varying resolutions by the content provider on the server. We request the server to send the image file which has the resolution appropriate to the present QoS and client parameters by including the preference in the request. HTTP/1.1 content negotiation requires that the variants of a resource type have different mime (sub)types. For example, in our system we use image/x-sgif, image/x-mgif, and image/x-lgif to respectively denote images with low, medium and high resolution. Clearly, this solution is cumbersome and can lead to a proliferation of (sub)types. We simply use it to demonstrate end-end content negotiation coexisting with proxy based transcoding using existing browsers and servers. With emerging standards from W3C (CC/PP) and WAP to support this, better implementations of the end-end framework will be possible.

To effect this, any HTTP GET request received from the MH is munged to an HTTP 1.1 request. The Accept headers stored for the MH are then appended to the outgoing stream to request for the file in the resolution most suited for the MH. Multiple alternatives are identified in the request. For example, the request from a WinCE type palmtop could be augmented by the following accept header – *Accept: image/x-sgif, image/x-mgif;q=0.8, image/gif;q=0.5, image/x-lgif;q=0.2.* A Host header is added to complete the HTTP 1.1 request.This process is transparent to the user and works even if the request comes from an HTTP/1.0 compliant browser, like most present commercial systems.

The proxy uses another end-end approach to reduce data transfer. When the proxy receives a GET request from a client, it sends a HEAD request to the WWW server to get information about the content type of the file. It GETs the data only if the client can handle it. This adds the overhead of an additional HEAD request. This overhead is minimal, and is acceptable given the significant saving of time/bandwidth for constrained MHs such as PDAs when dealing with large images or videos. The content-type of the file is used rather than file extensions since they can be misleading.

4.1.2. Multimedia – Images and Videos

In case the server is not able to provide the multimedia content at a resolution appropriate for the MH, the proxy monitors and transcodes the server response. For images, this happens in a manner similar to most other proxies – we use netpbm to scale and/or dither the image. Care is taken to ensure that imagemaps are not scaled.

Unlike image data where transcoding steps are obvious, video data represents a great challenge. Simple sub-sampling, as proposed in the Glomop system [11], is still not adequate as some clients may not have enough computational resources to do software decoding of MPEG or H.261. Mowser uses two approaches to handle video.

Visual Summaries – End-End Framework

The problem of bandwidth usage is faced by video databases when the user tries to interactively browse/query them. This aspect of interactive querying of Video DBs has been formalized by Bolle[5], and there has been significant work in creating summaries of videos (e.g. [9,37]) which can be used in this process. We have developed techniques[1,15] to create hierarchical summaries of a video. The structure of video is a hierarchy of the movie or episode, made up of frames, segments, shots, and scenes. Each segment consists of sequence of scenes, each scene consists of several shots, and each shot is composed of several frames which have similar properties. Frames are grouped together based on features obtained from visual properties such as chrominance and luminance. Each group is classified as one shot. We pick the frame that is closest to center of each group to be the Representative Frame (Rframe). Fuzzy techniques are used since frames can belong to different groups to different degrees.

The Rframes at a particular level of hierarchy are grouped together in temporal sequence creating a “film strip” – a snippet of the whole video (see figure 3). A new MIME subtype (video/x-rmpg) is used to identify these Rframe based summaries. Since the film strip is simply a gif file, it can be handled as such by the client. We note that the process of extracting features from frames and then grouping them requires significant time and so has to be an offline transformation done at the server.

Scalable Codes – Proxy based

The Image and Video Coding Community, driven by MPEG4 proposals and the bandwidth needed by video streams on wired networks, is creating new coding methods which are scalable (resolution, temporal) and which seek to minimize computational complexity of the encoding and decoding process. These codes are also resilient in presence of high channel BERs which characterize the wireless environments. We have developed a system[36](not integrated into Mowser yet) which uses SLCCA/3DSLCCA codes developed by Zhuang *et al.*[35]. Specifically, the proxy system downloads image and video data normally from the remote server. However, it transcodes this data into SLCCA/3DSLCCA for images and video respectively, and streams it to the Mobile Client. A plug-in has been created to decode this content. Also, given the code's error resilient nature, retransmission is turned off and the bit/burst errors are handled by the decoder. At present, we use a H.223 simulator for this system.

In general, if the remote server was using scalable codes, then the proxy could get the information at the right scale, and simply transcode to handle wireless channel characteristics.

4.1.3. Active Content

For mobile hosts with limited memory and computational resources, it may not be possible to execute any Java applets, JavaScript, VBScript, etc. This would likely be the case for PDAs. When such a preference is set, the document to be transmitted is parsed and all the active content is eliminated before sending it to the MH. Often though, Java (and Javascript) are used to provide functionality that can be duplicated using CGI callbacks or server parsed HTML. The proxy can request for a CGI version instead of the JavaScript from the server. In other words, replace active content of a page with equivalent dynamic content. Like all content negotiation, it assumes that the server provides alternative versions

(CGI based vs Java based) of a particular URL.

4.1.4. HTML

As it has evolved, HTML has introduced tags for several formatting, font and style options. Depending on the display capability of the client, some of these tags may not be useful (e.g. italics or color specifications on a plampilot). In fact, W3C is now proposing a compact HTML standard, and should it become widely adopted, the right format could be obtained by content negotiating with the server. At the moment, our proxy parses and eliminate all the tags that the MH does not support. For some PDAs, we eliminate all tags and simply send back a plain text document. Moreover, embedded links to objects that the client cannot handle (e.g. sound on a palmpilot) are re-written so that the client does not seek to even fetch them.

However, this kind of manipulation of tags and embedded multimedia often throws off the formatting and placement that the page creator intended, and the end result is not visually pleasing. In ongoing work, we are exploring the use of XML and XSL to mitigate this problem. In particular, the server will present the document as XML, and will have different XSLs for different client categories (workstations, laptops, PDAs etc.). Those clients with browsers capable of directly displaying the XML/XSL combination would have no problems. For thin mobile clients, the proxy would interpret this and translate it to appropriate HTML.

4.2. W^3IQ: Asynchronicity and Personalization

As mentioned earlier, personalization refers to tailoring the information content returned in response to a request based on who the end-user is. Clearly, this can cut down on the network traffic generated today by users who search the web to obtain the information they want, and can especially be useful in the mobile context. The fact that users accessing the web over wireless links tend to be looking for specific information, and not just browsing, provides further support for personalization systems in the context of mobile web access.

Personalization can be done in an end-end manner if the server can tailor its information content based on the user. Otherwise a proxy can obtain the information from the server, and filter it to extract the relevant information.

Proxy based Recommender System

The W^3IQ proxy provides functionality for both collaborative filtering and disconnected operation. Broadly, its operation can be described as follows: The user connects to the proxy, makes a request, and disconnects. The proxy then tries to assemble a set of URLs that would satisfy the request, and returns them to the user. Should the user be disconnected, the results of his prior request(s) are made available at reconnection. These results are URLs that the system feels will contain the information the user requested. Users then prioritize the URLs provided by the system by rating them, depending on their perception of the relevance of the URL. The user feedback allows the system to infer a "semantic" match between the user's query terms and the information in the document. This helps the system create a user's profile which is used to control the information s/he would receive in response to future queries. The ranking information, the URLs, and the corresponding query are all stored by the W^3IQ proxy as metadata. Over time, then, the proxy builds a repository of ranked URLs which can be used to answer future queries by this user or other users.

The system allows a proxy to pass the query to other W^3IQ proxies in the network, to garner any information related to the query in their local metadata. This allows all proxies to share information (specifically, ratings in regard to particular query terms) about URLs obtained by one proxy.

The process of evaluating a query proceeds as follows: To answer a user's query, the proxy first looks at its local metadata to see if this user has queried on the same terms before, and if so uses the ranked URLs to respond. If not, it queries its peer proxies about their metadata from all users in an order determined by the user's "closeness" to other proxies. If this fails, or not enough results have been returned so far, then the proxy passes the query to Lycos, and uses its results.

The closeness of a user to a particular proxy (or more specifically, its rating metadata) is determined by the difference in rankings for the URLs returned from a common query. In other words, the difference between the rank given to a URL by a user and its (aggregate) ranking in a proxy is computed and averaged over several URLs. The lesser the difference in rankings, the closer the user to (the rating metadata at) the proxy. A user qualifies for the grouping after rating a minimum number of URLs. Thus the "closer" W^3IQ servers are queried first for a new query by a user. Note that peer servers may be closer to a user that the local server of the user. This provides for "global aggregation" of resources using collaborative filtering techniques[24]. For a new user, the cache of URLs on the local server is queried first and the peer servers are searched in an arbitrary order.

Reusing previously obtained URLs which were highly ranked by other users with similar interests means that URLs the user likely wants will be returned. The proxy thus minimizes the bandwidth it uses over the mobile link as well as the power it forces MH to consume by sending information which is relevant to the user's query. By coming up with the closeness notion, we try to make our system net friendly. In other words, when a "new" user appears in the system, his query is broadcast to all peers to obtain information. Over time, as the peer information sources best for this user are determined, the broadcast mode is replaced by a selective communication mode to the "chosen" peers.

Web Mining to Support End-End Personalization

End-End personalization depends on the information server being adaptive[30]. In our work[20,28], we have created a system that mines a server's access logs for patterns of traversal. Specifically, we determine browsing "sessions" (URLs from the site accessed by the same host within a defined timespan). We define a similarity metric between two sessions based on how much overlap there is between the URLs visited. When comparing URLs, this metric takes into account the document hierarchy within the server. Given the sessions and distance between them, we use relational fuzzy algorithms to cluster them. This provides associations between URLs visited – in other words URLs belong to the same cluster when they are visited in the same session. Given such information, the server can then use cookie based mechanisms to associate a user with a pattern, and tailor the information sent back.

5. Experimental results

To test the extended Mowser system, two different client machines were used. One was connected to our departmental network (a combination of switched and shared FastEthernet) using a 100BaseTx NIC. The other had a wireless NIC (a Proxim RangeLan2). The wireless network basestations (Proxim RangeLan2 APIIs) had a 10 Mpbs connection to the departmental network. The wireless network is capable of 1.6 Mbps/channel, and at the time of testing there was only one mobile client in the network. Measurements we did using ftp showed that the wired client got approximately 7 times the bandwidth to the server than the wireless client (427.21 KBps vs. 59.6 KBbs). The access times similarly showed a factor of approximately 7.

A test page was created with two images and a video, besides some text. The images had both large and small versions, and the video had mpeg and Rframe representations. The page was served by an Apache 1.3.1 web server capable of content negotiation which was configured to understand the new mime (sub)types we had created. The clients declared different preferences to the proxy, and accessed the same web page. For the wired host, the proxy sent accept headers to allow large gif files and mpeg movies. Therefore, we received large image files and the entire video. For the wireless host, the preferences set limited the size of image and video files, and so the proxy sent accept headers requesting for small images and representative frames of mpeg videos. Hence, we received smaller versions of the images and only the representative frames for the movie file. The links embedded in the test page did not specify the extension of the image/video files, and the data was available in multiple resolutions (as denoted by file suffixes) to the server.

For the video, the wireless client took an average of 5.38 seconds to transfer the entire MPEG, but only needed 0.35 seconds on average to get the Rframe version – 15 times faster! Note that this is a very short video, playing for about 4 seconds. The different pages seen are illustrated in figures 1 and 2 respectively. Figure 3 illustrates the rframe version of the mpeg video. As described earlier, the time required to create the Rframe version is so large that this transformation is best done off-line.

Where the server cannot make appropriate resolution content available, the proxy scales it down either by size or color as specified by the user. To illustrate this for images, we turned off the end-end negotiation feature of our system by not adding the appropriate accept headers to the clients request. The server then returned the default images. Measurements using the time command indicate that the process of reducing the image of the bronze retriever statue took the proxy about 0.4 seconds on a lightly loaded Sun Ultra 10 machine. This involves converting the original gif image to pnm, scaling and color quantizing,

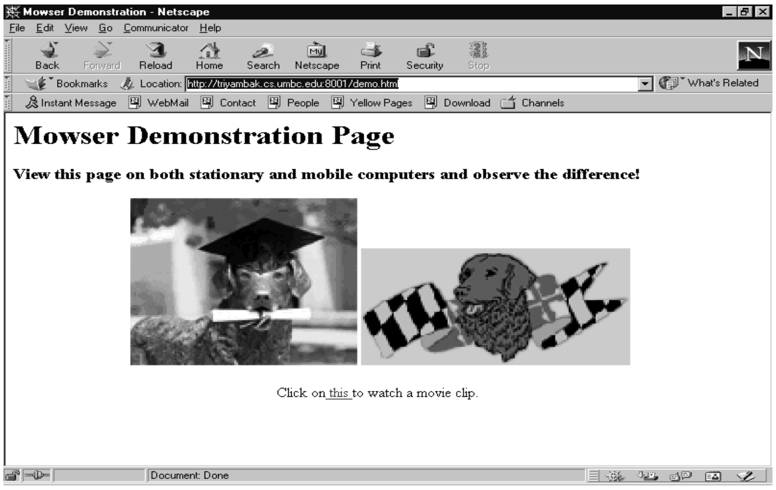

Figure 1. Response on a Resource rich Client

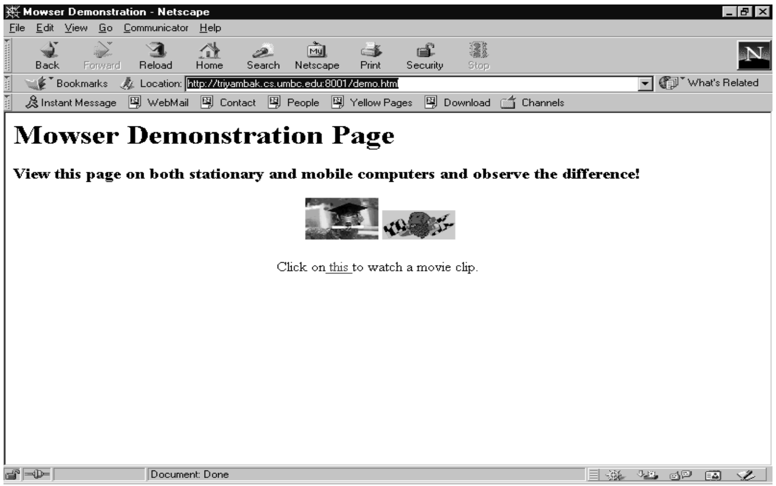

Figure 2. Response on a Resource poor Client

and reconverting to gif.

To get a clearer sense of the tradeoffs involved in doing image transcoding at the proxy vs end-end, we measured the time required by the proxy to provide a client with a variety of web pages from both commercial (CNN, ABC) and academic (UMBC, Berkeley) sites without content negotiation. Measurements were made for 25+ pages which had varying amounts of images, some of which required transcoding. CNNs main page took 68 seconds, and had 41 images, some of which did not have to be transcoded. ABC's web page took 321 seconds with

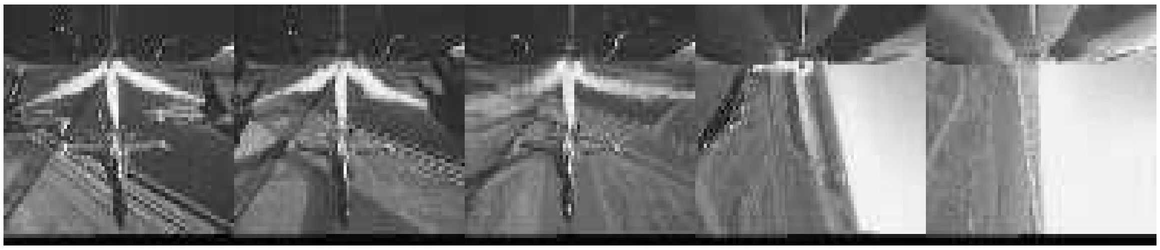

Figure 3. Rframes from a Blue Angels Video Clip

135 images. However, pages such as these receive tens or hundreds of thousands of hits per day. UMBC's main page took 17 seconds with 29 images. According to web logs, this page was accessed more than six thousand times in a particular day. We note that our system has not been optimized for speed, and we use freeware image conversion tools (netpbm). Despite this caveat, for such pages it makes sense for the server to keep the content in 4-5 different resolutions and make it appropriately available in an end-end manner. Where the page is infrequently accessed, the content can be kept in a single resolution with the proxy transcoding as needed.

The $W3IQ$ system has been used by a small group of students and this author. We observed that the remote Lycos server is less frequently queried as the URL caches of the W^3IQ servers grow, causing a considerable decrease in data transfer between the W^3IQ servers and the Lycos server and improving the response for the Mobile client. Also, as the system determines users' closeness values, the traffic amongst peer decreases, and the system is able to quickly retrieve information likely to be useful to the user. This further improves the response time. The fact that such collaborative filtering systems improve retrieval has also been underscored by many other developed and deployed systems, such as GroupLens[25] for usenet news.

6. Discussion

In this paper, we have presented a system to support web access from mobile platforms. Our system combines both proxy based and end-end approaches, using each as appropriate. The idea of using transcoding at the proxy to support mobility is not new per se. Many proxy based systems [11],[26],[39] have been developed to provide web access to mobile users. However, they typically transcode the image data received from the WWW server before sending it to the mobile client. The TACC model[6] takes a step towards a more general proxy architecture. Our framework transcoding, personalization and asynchronous access to support mobility, and is related to the TACC architecture. However, it goes beyond it in terms of incorporating both proxy based and end-end approaches. We also indentify in this paper when proxy and end-end approaches are better, and present experimental results to validate our ideas.

In ongoing work, we seek to add a third component to our system. Often the information the user seeks is not directly available, but can be computed from available data. This is particularly relevant in domains like Electronic Commerce, Finance, and Scientific/Engineering Systems. We are working to integrate software components into our system which will distribute themselves across the wired and wireless hosts and do the computations necessary to generate information on the fly.

7. Acknowledgements

The author would like to thank the anonymous referees and guest editors for their comments. These have enabled him to significantly improve the paper.

References

[1] S. Auephanwiriyakul, A. Joshi, and R. Krishnapuram. Fuzzy shot clustering to support networked video databases. In *Proc. IEEE FUZZ-IEEE 98/WCCI98*, May 1998.

[2] M. Balabanovic and Y. Shoham. Content-based, collaborative recommendation. *Communications of ACM*, 40:66–72, 1997.

[3] R. Barrett, P. Maglio, and D. Kellem. A confederation of agents that personalize the web. In *First Intl. Conf. on Autonomous Agents*, Marina Del Ray, CA, 1997.

[4] Harini Bharadvaj, A. Joshi, and Sansanee Auephanwiriyakyl. An active transcoding proxy to support mobile web access. In *Proc. IEEE Sumposium on Reliable Distributed Systems*, October 1998.

[5] R. Bolle, B. Yeo, and M. Yeung. Video query: Beyond the keywords. Technical report, IBM T. J. Watson Labs, October 1996.

[6] E.A. Brewer, R.H. Katz, Y. Chawathe, A. Fox, S.D. Gribble, T. Hodes, G. Nguyen, T. Henderson, E. Amir, H. Balakrishnan, A. Fox, V. Padmanabhan, and S. Seshan. A network architecture for heterogeneous mobile computing. *IEEE Personal Communications Magazine*, 5(5):8–24, 1998.

[7] C. Brooks, M. S. Mazer, S. Meeks, and J. Miller. Application-specific proxy servers as http stream transducers. In *Proc. WWW-4, Boston, http://www.w3.org/pub/Conferences/WWW4/Papers/56Application-Specific*, May 1996.

[8] IBM Corporation. Ringing in wireless services: Web access without wires. http://www.ibm.com/Stories/1997/08/wireless4.html.

[9] E. Elmagarmid, H. Jiang, A. Helal, A. Joshi, and M. Ahmed. *Video Database Systems: Issues, Products and Applications*. Kluwer Academic, 1997.

[10] G. Forman and J. Zahorjan. The challenges of mobile computing. *IEEE Computer*, 27:38–47, April 1994.

[11] A. Fox and E. A. Brewer. Reducing www latency and bandwidth requirements by real-time distillations. In *Proc. Fifth International World Wide Web Conference*, May 1996.

[12] A. Fox, I. Goldberg, S.D. Gribble, D.C. Lee, A. Polito, and E.A. Brewer. Experience with top gun wingman: A proxy-based graphical web browser for the usr palmpilot. In *Proc. IFIP International Conference on Distributed Systems Platforms and Open Distributed Processing (Middleware '98)*, 1998.

[13] Adele Howe and Daniel Dreilinger. Savvysearch: A meta-search engine that learns which search engines to query. http://www.cs.colostate.edu/ howe/pubs.html, January 1997.

[14] A. D. Joseph, A. F. deLespinasse, J. A. Tauber, D. K. Gifford, and M. F. Kaashoek. Rover: A Toolkit for Mobile Information Access. In *Proc. 15th Symposium on Operating Systems Principles*. ACM, December 1995.

[15] A. Joshi, S. Auephanwiriyakul, and R. Krishnapuram. On fuzzy clustering and content based access to networked video databases. In *Proc.8th IEEE Workshop on Research Issue in Data Engineering*, 1998.

[16] A. Joshi, C. Punyapu, and P. Karnam. Personalization & asynchronicity to support mobile web access. In *Proc. Workshop on Web Information and Data Management, ACM Conference on Information and Knowledge Management*, November 1998.

[17] A. Joshi, R. Weerasinghe, S. P. McDermott, B. K. Tan, G. Benhardt, and S.Weerawarna. Mowser: Mobile platforms and web browsers. *Bulletin of the IEEE Technical Committee on Operating Systems and Application Environments*, 8(1), 1996.

[18] A. Joshi, S. Weerawarana, and E. N. Houstis. Disconnected Browsing of Distributed Information. In *Proc. Seventh IEEE Intl. Workshop on Research Issues in Data Engineering*, pages 101–108. IEEE, April 1997.

[19] A. Joshi, S. Weerawarna, and E. N. Houstis. On disconnected browsing of distributed information. In *Proceedings of the seventh International workshop on Research Issues on Data Engineering*, pages 101–107. IEEE Press, 1997.

[20] Anupam Joshi and R. Krishnapuram. Robust fuzzy clustering methods to support web mining. In S. Chaudhuri and U. Dayal, editors, *Proc. ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, June 1998.

[21] R. Katz. Adaptation and Mobility in Wireless Information Systems. *IEEE Personal Communications*, 1(1):6–17, 1994.

[22] R. H. Katz, E. A. Brewer, E. Amir, H. Balakrishnan, A. Fox, S. Gribble, T. Hodes, D. Jiang, G. T. Nguyen, V. Padmanabhan, and M. Stemm. The bay area research wireless access network (barwan). In *Proceedings Spring COMPCON Conference*, 1996.

[23] H. Kautz, B. Selman, and M. Shah. Combining social and collaborative filtering. *Communications of ACM*, 40:63–65, 1995.

[24] R. Kavasseri, T. Keating, M. Wittman, A. Joshi, and S. Weerawarana. Web Intelligent Query - Disconnected Web Browsing using Cooperative Techniques. In *Proc. 1st. IFCIS Intl. Conf. on Cooperative Information Systems*, pages 167–174. IEEE, IEEE Press, 1996.

[25] Joseph A. Konstan, Bradley N. Miller, David Maltz, Jonathan L. Herlocker, Lee R. Gordon, and John. Riedl. Applying collaborative filtering to usenet news. *Communications of ACM*, 40:77–87, 1997.

[26] M. Liljeberg, M. Kojo, and K. Raatikainen. Enhanced services for world-wide web in mobile wan environment. http://www.cs.Helsinki.FI/research/mowgli/mowgli-papers.html, 1996.

[27] O. Nasraoui, R. Krishnapuram, and A. Joshi. Mining web access logs using a fuzzy relational clustering algorithm based on a robust estimator. In *to appear in Proc. WWW8*, August 1999.

[28] O. Nasraoui, R. Krishnapuram, and A. Joshi. Relational clustering based on a new robust estimator with applications to web mining. In *Proc. Intl. Conf. North American Fuzzy Info. Proc. Society (NAFIPS 99)*, New York, 1999.

[29] B. D. Noble, M. Satyanarayanan, D. Narayanan, J. E. Tilton, J. Flinn, and K. R.Walker. Agile application-aware adaptation for mobility. In *Proceedings of the 16th ACM Symposium on Operating System Principles.*

[30] M. Perkowitz and O. Etzioni. Adaptive web sites: Automatically synthesizing web pages. In *Proc. National Conference of the American Association of Artificial Intelligence (AAAI)*, 1998.

[31] J. Rucker and M.J. Polanko. Personalized navigation for the web. *Communications of ACM*, 40:73–75, 1997.

[32] S. Seshan, M. Stemm, and R. Katz. Spand: Shared passive network performance discovery. In *Proc. 1st Usenix Symposium on Internet Technologies and Systems (USITS '97*, 1997.

[33] M. Spritzer and M. Theimer. Scalable, secure, mobile computing with location information. *Comm. ACM*, 36:27–27, 1993.

[34] L.G. Terveen, W.C. Hill, B. Amento, D. McDonald, and Creter J. Phoaks. Web page: http://www.acm.org/sigchi/chi97/proceedings/paper/lgt.htm.

[35] J. Vass, B-B. Chai, and X. Zhuang. 3dslcca - a highly scalable very low bit rate software-only wavelet video codec. In *Proc. IEEE Workshop on Multimedia Signal Processing*, 1998.

[36] J. Vass, J. Yao, S. Zhuang, A. Joshi, and X. Zhuang. Multimedia information access in wireless environments. Technical Report TR-9901, CECS Department, University of Missouri, 1999.

[37] M. Yeung and B. Yeo. Efficient matching and clustering of video shots. In *Proc. International Conference on Image Processing*, 1995.

[38] O.R. Zaine, M. Xin, and J. Han. Discovering web access patterns and trends by applying olap and data mining technology on web logs. In *Proc. Advances in Digital Libraries Conf. (ADL'98)*, pages 19–29, April 1998.

[39] B. Zenel. *A Proxy Based Filtering Mechanism for The Mobile Environment.* PhD thesis, Department of Computer Science, Columbia University, N/A.