Personalizing Web Sites for Mobile Users

Corin R. Anderson
University of Washington
Seattle, WA, USA
corin@cs.washington.edu

Pedro Domingos
University of Washington
Seattle, WA, USA
pedrod@cs.washington.edu

Daniel S. Weld
University of Washington
Seattle, WA, USA
weld@cs.washington.edu

ABSTRACT

The fastest growing community of web users is that of *mobile* visitors who browse with wireless PDAs, cell phones, and pagers. Unfortunately, most web sites today are optimized exclusively for desktop, broadband clients, and deliver content poorly suited for mobile devices — devices that can display only a few lines of text, are on slow wireless network connections, and cannot run client-side programs or scripts.

To best serve the needs of this growing community, we propose building *web site personalizers* that observe the behavior of web visitors and automatically customize and adapt web sites for each individual mobile visitor. In this paper, we lay the theoretical foundations for web site personalization, discuss our implementation of the web site personalizer PROTEUS, and present experiments evaluating its behavior on a number of academic and commercial web sites. Our initial results indicate that automatically adapting web content for mobile visitors saves a considerable amount of time and effort when seeking information "on the go."

Keywords

Adaptive web sites, personalization, wireless web

1. INTRODUCTION

The fastest growing community of web users is that of *mobile* visitors — people who browse the web with wireless PDAs, cell phones, and pagers. Ninety-five percent of cell phones sold today are "web-ready" and authorities predict that the number of wireless Internet devices will outnumber desktop computers by 2003. Despite this trend, however, few web sites today cater to mobile visitors, instead, optimizing their content exclusively for desktop clients. Unfortunately, mobile devices are not as capable as their desktop counterparts, instead limited by small screens, low-bandwidth networks and slower processors. Thus the user experience for mobile visitors at these "one-size-fits-all" web sites suffers. To address this problem, we propose building *web site personalizers* that automatically adapt and personalize a web site to each individual mobile visitor.

A number of companies [1, 23, 10, 6] have taken the first step to bring web content to wireless devices: syntactic translation. Syntactic translation recodes the web content in a rote manner, usually tag-for-tag or following some predefined templates or rules. This method enjoys some success, particularly for mobile clients that have at least some graphical display (*e.g.*, a Palm Connected Organizer, but not text-only pagers). However, this approach essentially produces a scaled down version of the original web site: all the original content and structure designed for the desktop, broadband visitor, but in miniature form. The mobile visitor must wade through a morass of possibly irrelevant links and content to find the single gem of information for which he or she is looking; browsing such a site with a small screen and a low-bandwidth network connection only exacerbates this problem. Syntactic translation is not a flawed approach — quite to the contrary, it is a necessary component of a successful mobile web site. What it lacks is an awareness of the particular needs of each visitor — syntactic translation simply perpetuates the "one-size-fits-all" philosophy from the original web site.

Our solution to this problem begins with the observation that mobile web visitors exhibit a variety of browsing behaviors: random surfing, task completion (*e.g.*, buying stocks), information-goal seeking (*i.e.*, answering questions), etc. Information-goal seeking is of particular interest because it is generally *predictable*: visitors tend to have similar information goals in the future as in the past. Some example goals include: "What is the current stock price of MSFT?"; "Are there any Pentax K-mount zoom lens on auction at eBay?"; "What office is Dan Weld in?". This browsing behavior is predictable because visitors generally follow the same set of links, view the same set of pages, to attain these goals each time, and attempt to do so in a direct and efficient manner. In addition, visitors tend to view pages with similar content as pages viewed in the past (*e.g.*, a photography aficionado may frequently view pages containing words "zoom lens" and "f-stop", although the URLs requested may differ). By mining the past interactions with the web site for these behaviors (the navigational patterns and the content viewed), we can automatically *personalize* the web content for each individual visitor. We envision *web site personalizers* that act on behalf of a mobile visitor to adapt[1] web content as the visitor browses. A web site personalizer is an intermediary between the web site and the visitor and may be situated on the web server, on the visitor's device, or at a proxy server in between. A web site personalizer can:

- Make frequently-visited destinations easier to find. For example, if a visitor always follows a link $p_a \rightarrow p_b$, then a personalizer could make the link easier to find

[1]We use adapt, adaptation and personalize, personalization interchangeably throughout this paper.

WWW10, May 1-5, 2001, Hong Kong.
ACM 1-58113-348-0/01/0005.

by making the anchor text boldface. In another example, if the visitor always follows the same path of links $p_a \rightarrow p_b \rightarrow p_c \rightarrow p_d$, then a personalizer could add a link directly from p_a to p_d, saving the visitor the time to follow each link and download each page over the wireless network.

- Highlight content that interests the visitor. A personalizer can build a model of each visitor's interests, for example, "technology stocks" and "world news", and correspondingly alter the presentation of each page to make interesting content more salient.
- Elide uninteresting content and structure. Just as a visitor may be keenly interested in certain content at a site, the visitor frequently will be *un*interested in other content and links at the site. A personalizer can leverage the same visitor model to elide uninteresting content, for example, by replacing large swathes of content with a link to a new page bearing the elided content.

The key observation behind web site personalizers is that a great deal of information about visitors is readily available in the form of access logs (either at the web site or at an intermediary web proxy). That is, without any additional effort by either the site administrator or the mobile visitor, a personalizer can improve a site for the individual visitor. A web site personalizer adapts the site for the mobile visitor in a two-step process:

1. The personalizer mines the access logs to build a *model* for each visitor. This model includes navigational browsing behavior (which links the visitor follows), as well as content interests (what sorts of words or images appear on the pages the visitor views most often). This model can also include "out-of-band" information, such as visitors' geographical location, demographics, etc.
2. The personalizer transforms the site to maximize the *expected utility* [12] for a given visitor. The expected utility of a personalized web site is a measure of how much benefit the visitor will receive by browsing the site; the personalizer computes this value based on the visitor model derived in the first step.

The focus of this paper is on a principled approach to automatically personalizing web sites for mobile clients. We extend the pioneering work of Perkowitz and Etzioni on adaptive web sites [20, 21] and Bickmore and Schilit's Digestor [3] system by focusing on each individual visitor and by building and leveraging a deeper semantic model of the web site. Specifically, this paper makes the following contributions:

- **Precise problem statement.** Section 2 provides clear details of the web site personalization problem, including input and output characteristics.
- **Principled utility model.** Central to our approach to personalization is a model of the *expected utility* the visitor will receive by viewing a particular personalized web site. Our model takes a probabilistic approach to estimating what parts of the site the visitor will see, and defines utility in part based on what content the visitor has viewed in the past. We present the basis of this model in section 3.2 and additional implementation-specific details in section 4.3.
- **Heuristics for efficient search.** Section 3 describes how we characterize web site personalization as a search process. However, the naïve implementation of this search is computationally intractable for any but the smallest sites. We therefore present in section 4.3.3 a number of heuristics that trade off optimality of the solution for search run-time.
- **Experimental evaluation.** Following the framework we propose, we implemented the web site personalizer PROTEUS and applied it to a number of sites on the web. Section 5 presents our experimental evaluation of PROTEUS, comparing our personalizations to strict syntactic translation.

2. WEB SITE PERSONALIZATION: PROBLEM STATEMENT

In this section we precisely formulate the web site personalizer problem. In short, a web site personalizer acts as an intermediary between a web site and its many visitors, automatically adapting content to meet the needs of each visitor. We tend to view a personalizer as associated with only one web site, although it could just as easily be situated on a proxy server and adapt many sites, or exist on the browsing device and serve only one visitor. In contrast to manual personalization approaches, such as those employed at Yahoo! and Excite, automatic personalization does not require explicit effort from the web visitors to customize the content. The inputs to the personalizer are the web site and data about the visitors, and an evaluation function that measures the quality of a particular personalized site. More precisely, the inputs are:

- A web site W, represented as the URL of its main entry page U. From U, a personalizer can crawl the site to determine any number of features: the graph of pages and links at the site; the content of every page; the URL naming hierarchy of the web content; etc. The exact features the personalizer mines from the site depends on the implementation; we describe our personalizer's site model in section 3.1.
- A set of visitors $V = \{v_0, \ldots, v_m\}$. An individual visitor v_i is represented as a tuple (R, D) whose elements are, respectively, the visitor's history of page requests (*i.e.*, links followed) as captured by a proxy server and the visitor's demographics. A single request r_j is a tuple (u_s, u_d, t, c) containing the source page u_s, destination page u_d, time of the request t, and web client device c (*i.e.*, the IP address of the requesting computer or mobile device). From R we can reconstruct the sequence of pages viewed, by selecting the u_d from each request and ordering by time, and can subsequently apply any number of sequence prediction algorithms (*e.g.*, [17, 5]). The demographics D is an n-tuple of values that embody all the available information about the visitor external to the web experience: the visitor's age, gender, city of residence, annual income bracket, etc. We represent each of these data items generically as d_i. In summary:

$V = \{v_0, \ldots, v_m\}$	A set of m individual visitors
$v_i = (R, D)$	A visitor is represented as his or her history and demographics
$R = \langle r_0, \ldots, r_t \rangle$	The history is a sequence of requests ordered by time
$r_i = (u_s, u_d, t, c)$	A request is the originating page, destination page, time, and client
$D = (d_0, \ldots, d_n)$	Demographic information is an n-tuple of data items

- Current visitor v and requested URL u. A web personalizer acts as an intermediary for many visitors but adapts content for each visitor individually. v is the specific visitor to whom the personalizer must presently return a page, and u is the content that the visitor has requested.

- An evaluation function $\mathcal{F}$. $\mathcal{F}$ measures the quality of the web site for a specific visitor, starting at a particular page, as a scalar (larger values indicate higher quality). We emphasize that the quality of a site depends on the particular page the visitor has requested — it is this page that defines the visitor's current view into the site. Thus, $\mathcal{F}$ takes as input the web site W, the requested URL u, and the visitor model v, and returns a real-valued scalar: $\mathcal{F}(W, u, v) \mapsto \mathcal{R}$.

The output of a personalizer is a customized version of the requested content u that maximizes $\mathcal{F}(W, u, v)$. Although a personalizer may choose to adapt an entire site *en masse* for the visitor, the only adapted content returned to the visitor is that for page u. Of course, the personalizer may record the steps it took to personalize the site and reuse these steps for future requests.

3. PERSONALIZATION AS SEARCH

In this section we present the framework of our approach, providing precise definitions wherever appropriate. In the next section we describe how we implemented this system as a functioning web site personalizer.

We first describe our approach briefly. In essence, our web site personalizer performs a search through the space of possible web sites. The initial state is the original web site of unadapted pages. The state is transformed by any of a number of adaptation functions, which can create pages, remove pages, add links between pages, etc. The value of the current state (*i.e.*, the value of the web site) is measured as the expected utility of the site for the current visitor. The search continues either until no better state can be found, or until computational resources (*e.g.*, time) expire. We provide more detail on our approach below.

3.1 State representation

Each state in our search space S is an entire web site, W. Although an actual implemented system (such as ours, as we discuss in section 4) may choose to adapt only a single page at a time, we model the entire web site to allow adaptations to be made anywhere in the site. The web site W is modeled as a directed graph whose nodes are pages, $p_0, \ldots, p_n$, and whose arcs are hypertext links, $l_0, \ldots, l_m$. A link l_k is a triple (p_s, p_d, a) where p_s is the source page (*i.e.*, the page on which the link appears), p_d is the destination page[2], and a is the anchor text. Each page p_i is modeled as a hierarchy of web content, much in the same way the parse tree of an HTML document confers a hierarchy of HTML tags. p_i is thus represented as the root of this hierarchy, and is a *content node*. A content node c is a pair (C, B) where C is a sequence of children $\langle c_1, \ldots, c_k \rangle$ of c and B is a behavior that c imparts on its children. The elements of C may be either plain text or (recursively) content nodes. The behavior B is the action that affects the human-viewable content. For example, if c were a "`<strong>`" node, then B would render its children in boldface; or if c were an "`<a>`" node, then B would render its children as a hypertext link. Summarizing:

$S =$	$\{W_0, W_1, \ldots\}$	Each state in the search space is a web site
$W =$	$(\{p_0, \ldots, p_n\}, \{l_0, \ldots l_m\})$	A web site is a directed graph of pages and links
$l_k =$	(p_s, p_d, a)	A link has a source, destination, and anchor
$p_i =$	c_i	A page is its root content node
$c_i =$	$(\langle c_{i1}, \ldots, c_{ik} \rangle, B)$	A content node is a sequence of children and node behavior; or
$c_i =$	*text*	A content node is plain text

3.2 State Evaluation

We estimate the quality of the personalized web site as the expected utility of the site from the point of view of the requested page. Intuitively, the expected utility is the sum of the utility the visitor receives by browsing each page in the site, discounted by the difficulty of reaching each page[3]. For example, following a link at the top of the current page may not be difficult, but reaching a page many links "away" will require scrolling (to find the links) and waiting for intermediate pages to download over the wireless network. The graph nature of a web site naturally suggests a recursive traversal to evaluate the site, starting from the current page. At each step in the traversal (*i.e.*, at each page in the site), we define the utility of the page as the sum of its *intrinsic* utility — the utility of the page, in isolation — and the *extrinsic* utility — the utility of the linked pages[4]. We make these concepts more precise below.

3.2.1 *Web site model for evaluation*

We transform the search state model slightly to make evaluation easier. We observe that, while adaptation requires detailed internal structure of each web page, an evaluation function that is based on a human visitor's view should not be exposed to this internal structure, but should see only a linear sequence of text and links. Thus, instead of using the tree-based model, we use only the *leaves* of the tree in their left-to-right order.

The leaves of a page p_i and their ordering impose a linearization of the content, which we subsequently decompose into a sequence of "screens" $\langle s_{i0}, \ldots, s_{im} \rangle$, each of which represents the web content that can be seen in one window of the visitor's browser. A single screen s_{ij} is composed of web content (*i.e.*, the text and graphics that are displayed),

[2]p_d may also be the distinguished page "$\langle$*external*$\rangle$" which represents any page outside the current web site.

[3]We will actually compute the utility at a finer granularity than each web page, but the intuition from the utility of an entire page is the same.

[4]In many ways, the intrinsic and extrinsic utilities are analogous to the authority and hub weights, respectively, from Kleinberg's work [13].

which we denote as T_{ij}, and a set of links $l_{ij1}, ..., l_{ijk}$ that appear on the screen. In summary:

p_i	$= \langle s_{i0}, \ldots, s_{im} \rangle$	A page is a sequence of screens
s_{ij}	$= (T_{ij}, \{l_{ij1}, \ldots, l_{ijk}\})$	Each screen contains web content and links

3.2.2 *Expected utility*

Because our goal is to maximize *expected* utility, we must be able to calculate the expectation that the visitor will view any given piece of content in the site. To this end, we model the visitor as having only a fixed set of *navigation actions* at his or her disposal, but any number of which the visitor may select (*i.e.*, the actions are not mutually exclusive). Specifically, if the visitor is at screen s_{ij}, then the set of actions available is:

$$A = \{a_{\text{scroll}}, a_{l_{ij1}}, \ldots, a_{l_{ijk}}\}$$

That is, the visitor may: scroll down to the next screen (assuming that s_{ij} is not the last screen of the page); or follow any link that appears on the screen. We reiterate that the visitor will generally perform a number of these actions at any screen, and that the actions are not independent. For example, the visitor may follow three different links (each time returning to this screen with the browser's "back" button) and scroll to the next screen of content. Thus, to model the visitor behavior, we maintain independent probabilities that the visitor will take each action, and these probabilities will generally not sum to one. In section 4 we describe how we choose these probabilities in practice.

Recall from section 2 that the evaluation function $\mathcal{F}$ takes as input the web site, the page requested, and the current visitor model, and calculates the quality of the site. If we let $\hat{p}$ be the personalized page for the requested URL u and $U_V(\hat{p})$ be the utility of $\hat{p}$ for visitor v, then:

$$\mathcal{F}(W, u, v) = E[U_V(\hat{p})]$$

That is, as we observed earlier, the evaluation of W is the product of a recursive traversal through the site, and this equation states that $\hat{p}$ is the root of that recursion. Similarly, because only the first screen of $\hat{p}$ is initially visible to the visitor, we calculate the expected utility of $\hat{p}$ (or any p_i, in fact) as the expected utility of its first screen:

$$E[U_V(p_i)] = E[U_V(s_{i0})]$$

The expected utility of a single screen s_{ij} is the sum of its intrinsic and extrinsic utilities:

$$E[U_V(s_{ij})] = E[IU_V(s_{ij})] + E[EU_V(s_{ij})]$$

The intrinsic utility of a screen measures how useful the screen's content is towards fulfilling the visitor's information goal, in isolation of the rest of the web site. Typically, the intrinsic utility will depend on the visitor model — the past history and demographics. A more detailed description of intrinsic utility depends on particular assumptions regarding visitor interests and goals; see section 4 for a discussion of the method we used in PROTEUS.

The extrinsic utility, on the other hand, measures the value of a screen by its connections to the rest of the web site. As we noted earlier, the visitor may reach other parts of the web site by taking one or many navigation actions from the current screen. Associated with each of these actions is a probability that the action will be taken, denoted by $P(\text{action})$ (see section 4 for a discussion of how PROTEUS estimates these probabilities). In addition, actions may impose a cost to the visitor (*i.e.*, a negative utility); these costs are γ_s and γ_l for scrolling and following a link, respectively. These costs subtract directly from the expected utility of the action, and represent the cost to the visitor (in time, money, etc.) of taking the action. In summary, if we let d_{ijk} be the destination page of link l_{ijk}, then the extrinsic utility of screen s_{ij} is a sum weighted by probabilities:

$$\begin{aligned} E[EU_V(s_{ij})] &= P(\text{scroll})(E[U_V(s_{i,j+1})] - \gamma_s) + \\ &\quad \sum_k [P(l_{ijk})(E[U_V(d_{ijk})] - \gamma_l)] \qquad (1) \end{aligned}$$

We can see that equation 1 introduces the recursive component of the evaluation function, by referencing the utility of other pages and screens. The recursion is halted when the expected utility of a screen or page is less than the cost of reaching that content (*i.e.*, when $E[U_V(s_{i,j+1})] < \gamma_s$ or $E[U_V(d_{ijk})] < \gamma_l$).

The equations given above and a formula for intrinsic utility completely determine the utility of an adapted page $\hat{p}$. However, the equations given, evaluated verbatim, would not be computationally tractable — they call for a screen-by-screen decomposition of potentially every page in the entire web site. For small sites, this fine-grained analysis may be possible, but many sites have hundreds if not thousands of static web pages, as well as potentially limitless dynamic web pages — far too many to examine individually. Fortunately, the evaluation can be made computationally much simpler with the aid of a few assumptions. We describe those assumptions and under what conditions they hold in the next section.

4. IMPLEMENTATION

We have implemented PROTEUS, a web site personalizer based on the framework described in the previous section. While most of the details of this system should be clear from the framework, a few issues are implementation-specific; we describe those issues here. We conclude this section with a few words about the performance of PROTEUS.

4.1 Search control

We implemented a simple, steepest-descent search control method for PROTEUS (see Table 1). Throughout the search, PROTEUS maintains the best-known state so far, *BestState*, and a current search seed *SearchSeed*. At each iteration in the search, PROTEUS produces all states one "step" away from the search seed, and replaces the search seed with the best of these new states. The search stops after a fixed number of iterations and PROTEUS returns the best-known state.

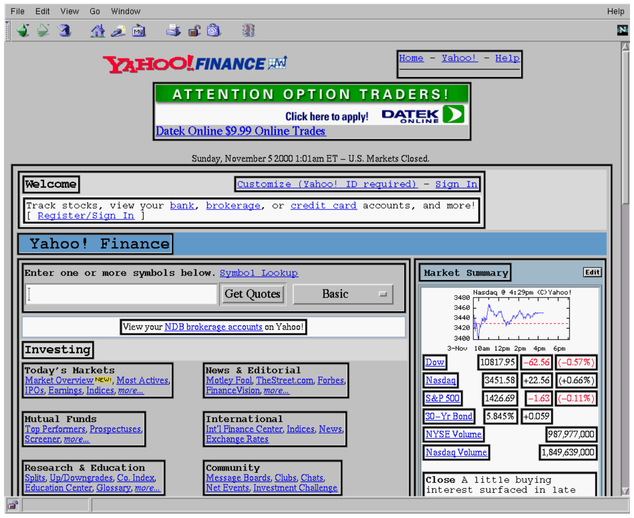
4.2 Search operators

While the framework described in the previous section allows for transformation to occur anywhere in the site, we have restricted PROTEUS to make changes only that involve the requested page, $\hat{p}$. That is, any transformation to the site must act on $\hat{p}$ — add content to $\hat{p}$, remove content from $\hat{p}$, etc. Changes made elsewhere in the site, such as creating a new web page or adding links between two other pages,

```
Inputs:
$W$      Unadapted web site

Search($W$)
  $BestState \leftarrow W$
  $BestUtility \leftarrow$ Expected utility of $W$
  $SearchSeed \leftarrow W$
  For $K$ iterations:
    $S \leftarrow$ Generate new states from $SearchSeed$
    $s \leftarrow$ State in $S$ with greatest expected utility
    $SearchSeed \leftarrow s$
    If expected utility of $s > BestUtility$
      $BestState \leftarrow s$
      $BestUtility \leftarrow$ Expected utility of $s$
  Return $BestState$
```

Table 1: Search control.



ATTENTION OPTION TRADERS!
Click here to apply! DATEK ONLINE
Datek Online $9.99 Online Trades

Sunday, November 5 2000 1:01am ET – U.S. Markets Closed.

Welcome
Customize (Yahoo! ID required) - Sign In
Track stocks, view your bank, brokerage, or credit card accounts, and more!
[Register/Sign In]

Yahoo! Finance

Enter one or more symbols below. Symbol Lookup
Get Quotes Basic

View your NDB brokerage accounts on Yahoo!

Investing

Today's Markets
Market Overview NEW!, Most Actives, IPOs, Earnings, Indices, *more...*

News & Editorial
Motley Fool, TheStreet.com, Forbes, FinanceVision, *more...*

Mutual Funds
Top Performers, Prospectuses, Screener, *more...*

International
Int'l Finance Center, Indices, News, Exchange Rates

Research & Education
Splits, Up/Downgrades, Co. Index, Education Center, Glossary, *more...*

Community
Message Boards, Clubs, Chats, Net Events, Investment Challenge

Market Summary Edit

Dow	10817.95	−62.56	(−0.57%)
Nasdaq	3451.58	+22.56	(+0.66%)
S&P 500	1426.69	−1.63	(−0.11%)
30−Yr Bond	5.845%	+0.059	
NYSE Volume			987,977,000
Nasdaq Volume			1,849,639,000

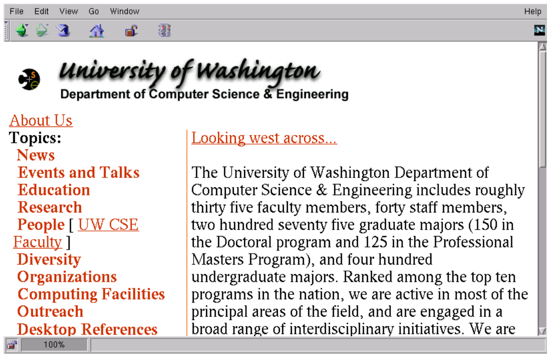
Close A little buying interest surfaced in late

Figure 1: Content blocks. Solid black borders outline blocks of content on which elide-content **and** swap-siblings **may operate. Note that these blocks may be nested.**

are not allowed. We made this choice to limit the branching factor of the search.

We presently support three transformation operators: elide-content, swap-siblings, and add-shortcut. Two of these operators, elide-content and swap-siblings, act on only certain subtrees of the original $\hat{p}$, specifically, only subtrees that correspond to block-level content. For instance, content in a `<div>` tag can be manipulated, as can entire lists (`<ul>`, `<ol>`), but a single emphasized word (`<em>`) cannot. In addition, subtrees can be nested. Figure 1 shows a typical decomposition into these content blocks for Yahoo!'s finance portal (`finance.yahoo.com`). PROTEUS automatically identifies the content blocks using a number of heuristics.

The elide-content operator replaces a subtree of $\hat{p}$ with a link to the original content in a fashion similar to Digestor [3] (figure 2). Content elision trades off screen space and page complexity of $\hat{p}$ for the cost the visitor may incur to view the elided content (*i.e.*, to follow the link). Note that elided content is still available — the visitor can always reach the original content by following the replacing link.



Department of Computer Science & Engineering

About Us
Topics:
News
Events and Talks
Education
Research
People [UW CSE Faculty]
Diversity
Organizations
Computing Facilities
Outreach
Desktop References

Looking west across...

The University of Washington Department of Computer Science & Engineering includes roughly thirty five faculty members, forty staff members, two hundred seventy five graduate majors (150 in the Doctoral program and 125 in the Professional Masters Program), and four hundred undergraduate majors. Ranked among the top ten programs in the nation, we are active in most of the principal areas of the field, and are engaged in a broad range of interdisciplinary initiatives. We are

Figure 3: Shortcut links. PROTEUS has created a new shortcut link on the this page: "UW CSE Faculty". The link is placed near the most-common starting point of the path it shortens.

The second operator, swap-siblings, permutes two subtree siblings in $\hat{p}$'s hierarchy. The result of swap-siblings is to swap the positions of the subtrees' content in the rendered page — the former right-sibling's content in placed above the former left-sibling's content. Like elide-content, swap-subtree is allowed to swap only preidentified content blocks. In addition, swap-subtrees may not swap any blocks that are implicitly ordered, such as ordered lists (`<ol>`) or text (`<p>`).

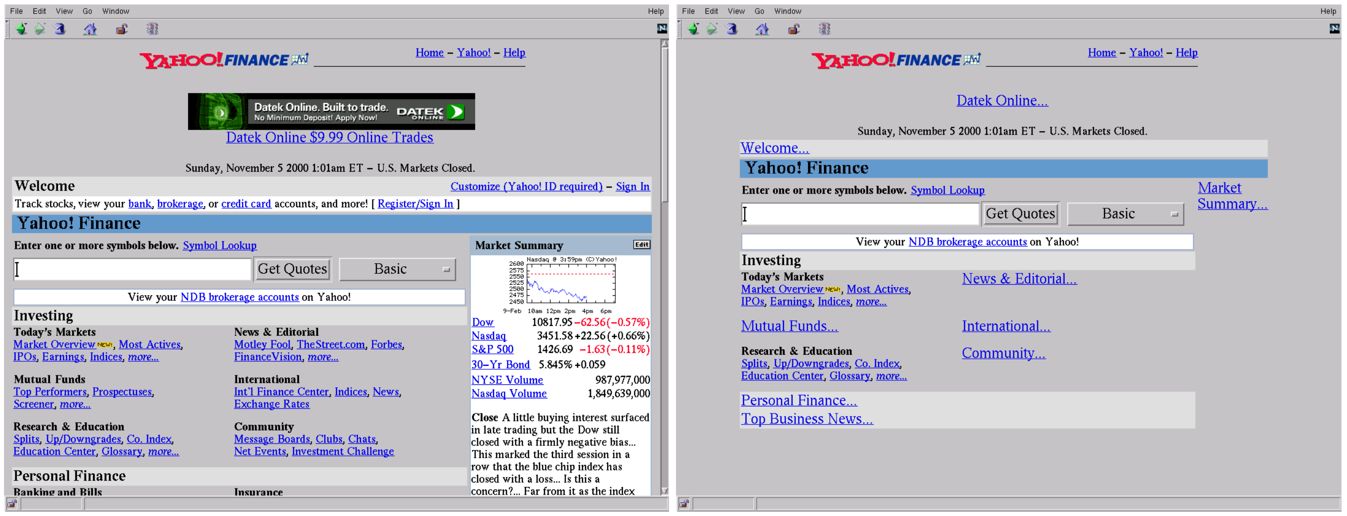
The final operator, add-shortcut, creates a new link from $\hat{p}$ to some other page in W (figure 3). Because W may have an arbitrarily large number of pages, add-shortcut does not consider *all* new links from $\hat{p}$. Instead, add-shortcut considers only those pages that can be "reached" by following at most k links from (the original version of) $\hat{p}$. Thus, add-shortcuts may create a link from $\hat{p}$ to another page that effectively "shortcuts" a longer path of links. For instance, if the visitor previously followed the path $\hat{p} \rightarrow p_a \rightarrow p_b \rightarrow p_c \rightarrow p_d$, add-shortcut will create a link directly from $\hat{p}$ to p_d. Furthermore, PROTEUS places the new link $\hat{p} \rightarrow p_d$ next to the original link $\hat{p} \rightarrow p_a$, based on the assumption that, if the visitor previously reached p_d by first going through p_a, and the visitor wants to find p_d again, then the visitor will look towards the link to p_a first. Of course, placing $\hat{p} \rightarrow p_d$ near $\hat{p} \rightarrow p_a$ is possible only if the visitor actually took the path $\hat{p} \rightarrow p_a \rightarrow \cdots \rightarrow p_d$ before. If the visitor has not established this path before, then PROTEUS places the link based on the paths *other* visitors at the site have taken. That is, if there are many paths from $\hat{p}$ to p_d, PROTEUS will choose the most popular path and place the shortcut link near the first step along that path. The anchor text of the link is chosen heuristically as either the destination's `<title>` or `<h1>`.

4.3 Web site evaluation

As we stated in the previous section, we evaluate the search states by calculating the expected utility of the web site for the visitor. The previous section described the framework for this evaluation, but left some details to the actual implementation of the system. Below, we describe these choices in PROTEUS.

4.3.1 *Navigation action probabilities*

Section 3.2.2 presented a set of navigation actions that a visitor can take when viewing a screen s_{ij}. PROTEUS estimates the probabilities that the visitor takes each of these



Home – Yahoo! – Help

Datek Online. Built to trade. No Minimum Deposit! Apply Now! DATEK ONLINE

Datek Online $9.99 Online Trades

Sunday, November 5 2000 1:01am ET – U.S. Markets Closed.

Welcome — Customize (Yahoo! ID required) – Sign In

Track stocks, view your bank, brokerage, or credit card accounts, and more! [Register/Sign In]

Yahoo! Finance

Enter one or more symbols below. Symbol Lookup

View your NDB brokerage accounts on Yahoo!

Investing

Today's Markets: Market Overview, Most Actives, IPOs, Earnings, Indices, more...

News & Editorial: Motley Fool, TheStreet.com, Forbes, FinanceVision, more...

Mutual Funds: Top Performers, Prospectuses, Screener, more...

International: Int'l Finance Center, Indices, News, Exchange Rates

Research & Education: Splits, Up/Downgrades, Co. Index, Education Center, Glossary, more...

Community: Message Boards, Clubs, Chats, Net Events, Investment Challenge

Personal Finance

Banking and Bills — Insurance

Market Summary

Nasdaq @ 3:59pm (C)Yahoo!

Dow	10817.95	−62.56(−0.57%)
Nasdaq	3451.58	+22.56(+0.66%)
S&P 500	1426.69	−1.63(−0.11%)
30−Yr Bond	5.845%	+0.059
NYSE Volume		987,977,000
Nasdaq Volume		1,849,639,000

Close A little buying interest surfaced in late trading but the Dow still closed with a firmly negative bias... This marked the third session in a row that the blue chip index has closed with a loss... Is this a concern?... Far from it as the index

Home – Yahoo! – Help

Datek Online...

Sunday, November 5 2000 1:01am ET – U.S. Markets Closed.

Welcome...

Yahoo! Finance

Enter one or more symbols below. Symbol Lookup

Market Summary...

View your NDB brokerage accounts on Yahoo!

Investing

Today's Markets: Market Overview, Most Actives, IPOs, Earnings, Indices, more...

News & Editorial...

Mutual Funds...

International...

Research & Education: Splits, Up/Downgrades, Co. Index, Education Center, Glossary, more...

Community...

Personal Finance...

Top Business News...

Figure 2: Elided content. On the left is an unadapted web page. On the right a number of blocks of content have been elided and replaced with hypertext links.

actions by measuring the frequency with which the visitor took the action in the past. For example, the probability that the visitor follows a link $p_s \to p_d$ is the quotient of the number of sessions[5] in which the visitor viewed p_d sometime after p_s divided by the number of sessions in which the visitor viewed p_s (*i.e.*, the probability of the link $p_s \to p_d$ is simply the probability the visitor will reach p_d sometime after p_s).

The probability for scrolling is derived empirically and is held constant. Proteus presently uses a probability of 0.85 that the visitor will scroll to the next screen, although we are in the process of determining this number empirically.

In addition to the action probabilities, actions also impose a cost on the visitor. Through empirical evaluation, we set the cost of scrolling, γ_s, at 0.01 and the cost of following a link, γ_l, at 0.05. These values tended to work acceptably in practice, although we found that our results were largely insensitive to their exact values. In practice, the dominant term in the expected utility equation is the product of probabilities of taking chains of actions. That is, for all but the most probable links the visitor would follow, the contribution of a remote page to expected utility is already vanishingly small, irrespective of the cost of following the link. The situation is similar for scrolling through the screens of the current page.

4.3.2 Intrinsic Utility

Our implementation measures intrinsic utility of a screen as a weighted sum of two terms, which relate to how the screen's content matches the visitor's previously viewed content, and how frequently the visitor viewed this screen. If T_{ij} is the viewable content on screen s_{ij}, then:

$$IU_V(s_{ij}) = \omega_{sim} \cdot sim_V(T_{ij}) + \omega_{freq} \cdot freq_V(s_{ij}) \qquad (2)$$

[5]A *session* is a sequence of page views that occur closely in time, and together serve to answer a single information goal. The visitor will have many browsing sessions represented in their request history. Proteus sessionizes the browsing data by collecting sequential accesses that are separated by not more than one minute from the previous access.

$sim_V(T_{ij})$ is the similarity between T_{ij} and a model representing the visitor's web content interest. Specifically, we are concerned only with the textual content of the page, and ignore any graphical elements. Thus, we model the visitor and the requested content as *word vectors*, vectors in an n-dimensional space, with each word that appears on any web page as a different dimension. Further, we scale both vectors according to a TFIDF scheme [22], which weights words proportionally to how often they appear in the document (in this case, the particular screen or the visitor's history), and inversely proportionally to how often the words appear in *any* document.

Let $w_{T_{ij}}$ be the word vector for content T_{ij}, and consider a particular word k. The value of $w_{T_{ij}}(k)$ is the number of times word k appears anywhere in T_{ij}, divided by the number of *other pages* on which word k appears. For instance, if the word "Microsoft" appeared eight times in T_{ij}, but appeared in a total of 119 documents, then $w_{T_{ij}}(\text{"Microsoft"}) = 8/119$. The numerator is called the "term frequency" while the denominator is called the "document frequency."

Let w_V be the visitor's word vector, and again consider a word k. Instead of simply summing the number of times the visitor has ever encountered the word k before, we compute a weighted sum in the following manner. Any words on a page at the end of a browsing session receive a high weight; if the visitor stopped browsing after that page, then that page, more likely than not, answered the visitor's needs.[6] On earlier pages in a browsing session, any words near the *links* selected receive a moderate weight, while the other words on the page receive little or no weight. We assign moderate weight to link anchor words because these words caught the visitor's eye while browsing; these words must

[6]Of course, a visitor may end a browsing session without ever satisfying his or her information goal, and, worse, this model would reinforce whatever dead-end the visitor stopped at. However, our experience indicates that, for a wide range of information-seeking goals, the last page in a session *does* satisfy the visitor's task, and does contain pertinent content. An interesting line of future work would be to improve how the content model is populated from the access logs.

have some significance to the visitor. The term frequency for a word k is thus the weighted sum of all the occurrences of word k in the visitor's history. The document frequencies, by which the entries in w_V are divided, are the same as for $w_{T_{ij}}$: the number of web pages on which word k appears.

The similarity between T_{ij} and V is thus finally computed as the dot product of the word vectors normalized by their lengths:

$$sim_V(T_{ij}) = \frac{w_{T_{ij}} \cdot w_V}{||w_{T_{ij}}|| \cdot ||w_V||}$$

$freq_V(s_{ij})$ measures the number of times the visitor viewed screen s_{ij}. Calculating this value is easy, by simply counting the number of times the visitor requested *page* p_i in the visitor's link history L and dividing by the number of screens in p_i (we assume that the visitor viewed each screen on every page visited in the past). We balanced the weights ω_{sim} and ω_{freq} in equation 2 to trade off each additional page view by the visitor for roughly 1% of text similarity. As with the action cost values, this choice of value seems to work well in practice, but we are in the process of determining it empirically.

4.3.3 Approximating Expected Utility

At the end of the previous section, we noted that calculating expected utility using the given equations verbatim would be computationally intractable — the equations imply a need to recur through the entire web site, evaluating each screen of content. Fortunately, we can make this evaluation more tractable with two heuristics. The first heuristic is to assume that the cost of scrolling a page of text is much smaller than that of following a link ($\gamma_s \ll \gamma_l$). In this case, the cost of viewing, say, the second screen on a distant page is dominated by the cost of reaching that page ($\gamma_l + \gamma_s \approx \gamma_l$). Thus, we may treat all pages but $\hat{p}$ as single-screen pages and can ignore the recursion due to scrolling on these pages.

The second heuristic places a bound on the number of pages considered when evaluating $\hat{p}$. We implemented a priority queue into which pages p_i and screens s_{ij} are placed and ordered by the maximum probability that the visitor will reach the respective content. For example, the first screen on $\hat{p}$ has probability 1.0, the second screen probability 0.85, and a link very frequently followed from the second screen (say, in nine sessions out of ten) has probability $0.85 \times 0.9 = 0.765$. We can tune exactly how much computation is spent evaluating $\hat{p}$ by setting a threshold on the probability — any content that has a lower probability of being viewed will simply be ignored in our computation. This threshold gives us a direct "knob" that we can turn to trade off performance for accuracy: the lower the threshold, the more accurate the evaluation, at the expense of recurring through more of the site.

4.4 Performance

PROTEUS is written in Python and uses a MySQL database server. PROTEUS runs as a proxy through which mobile clients connect. To adapt a remote web site, PROTEUS retrieves the requested page and manipulates it locally, retrieving additional pages from the remote site on-demand. Our system can generate each alternative personalization in 10 milliseconds and produces an average of 40 adaptations per search iteration. Evaluating each personalization requires roughly 500 milliseconds using a probability threshold of 0.001. In our experiments we ran PROTEUS for 20 iterations of search for each page, which typically produced a personalized page in four to seven minutes.

With these rates, PROTEUS can successfully adapt a web site offline in a reasonable amount of time, but is not yet fast enough to produce personalized content at "click-time." However, we are confident that two simple enhancements can substantially increase the speed of our system. First, the single most-expensive operation is retrieving content from the remote host. Our system caches as much information locally as it can, but ensuring that the local information is accurate, and collecting new content from remote hosts, consumes a great deal of time. If PROTEUS were installed on the remote web site directly, this overhead would be eliminated. Second, although Python is an excellent prototyping language, its performance in certain areas, such as string manipulation, is less than ideal. We are confident that reimplementing PROTEUS in C++ would yield two orders of magnitude of speedup — enough for real-time adaptation.

5. EVALUATION

In this section we present the results of an experiment that provides insight into the effectiveness of our system. In the experiment, we track ten test subjects' browsing habits on their desktop workstations and then compare how effectively these subjects can use a suite of personalized and non-personalized web sites on a wireless Palm Connected Organizer. We measure visitor effort in terms of both time to attain the goal and the amount of navigation (number of scrolling actions and links followed) the visitor must take. In the following subsections, we present our method of collecting data, details of our experimental setup, and our results and analysis.

5.1 Data collection

A key element in our experiment is a body of detailed access logs for the ten test subjects' browsing behavior. To produce these logs, we instrumented every subject's desktop browser to make all its requests through a proxy, and to not cache any web content. Thus, pages the subjects viewed and links followed were recorded in the logs[7]. We then asked the test subjects to perform a suite of information-seeking tasks that we provided each day. The tasks dictate a starting page and we directed the subjects to attain their goals by browsing exclusively at the given site. The complete list of questions used in our experiment appears in Appendix A. We illustrate two example questions here:

- "Find the current stock price for MSFT, starting at `finance.yahoo.com`". We varied the particular stock ticker symbol among a number of computer-related technology stocks (MSFT, YHOO, AMZN, etc.).

- "Find the make and model of the editor's choice digital camera at `cnet.com`". The variable is the consumer electronics device, which we selected from among several choices.

[7]The only page views missing from the logs were pages visited using the browser's Forward and Back buttons. However, because every new page request included the referring document, we could reconstruct the entire sequence of page requests, with the only exception of "loops" of Forwards and Backs.

The tasks in the seed suite were drawn randomly from a distribution of parametric questions (*i.e.*, the tasks contain variables that permit many similar but not identical questions) and represent a coherent model of visitor interest (*i.e.*, the goals were typical of a visitor with a stable set of interests).

5.2 Evaluation with a mobile device

In our experiment we asked the test subjects to browse the web with a wireless Palm Connected Organizer[8]. The subjects were given a suite of information-seeking goals to achieve, drawn from the same distribution as the goals during the original seeded browsing phase. Note that some of the goals in this test phase were identical to the goals from the seeded-browsing phase. This duplication is acceptable in our experiments, because visitors frequently *will* have the same goals on a number of occasions, for example, when the answer to a question changes with time (*e.g.*, "What is the stock price of MSFT *today*?"). During this experiment, we measured the number of navigation actions taken and amount of time required to meet each information-seeking goal.

We asked test subjects to answer the suite of questions twice: once, on the unmodified web site, and again on a *personalized* version of the target site. We personalized the target site for each visitor by allowing PROTEUS to first build a model of that visitor, based on the subject's past seeded browsing data (their desktop, broadband browsing data), and then to create adapted pages for the testing suite. We did not personalize every page of every web site because of the sheer volume of the task — all the sites in our study contained dynamic web content and potentially an unlimited number of pages. However, because our current implementation is not yet fast enough to adapt a single page in real-time, we personalized the sites before the subjects performed their tests. We chose the specific pages for PROTEUS to adapt by using our human judgment of where the subjects would likely visit during the test. Note that we have not influenced the *personalization* at all — simply, we have selected the subset of pages that PROTEUS will personalize, purely for the sake of efficiency. On average, PROTEUS personalized twenty-one pages for each subject, which represented 38% of the subject's page views during the experiment. Because PROTEUS personalized only a subset of the actual pages viewed, our results present a lower bound on the benefit the visitor would receive and tend to understate the full impact personalization would have for a mobile visitor.

A priori, we anticipated two results from this experiment. First, we anticipated that the subjects' behavior on the personalized site would require fewer navigation actions and less time than on the equivalent unmodified site. Second, we anticipated that subjects' behavior would become more efficient as they navigated the web sites on the Palm device. To mitigate this "subject-training" effect in our results, we alternated for each visitor which version of the web sites we presented first — personalized or unmodified. Thus, both versions of the web sites received equal advantage in our experiment as a whole.

Figures 4 and 5 compare links followed and time spent to attain each goal on the personalized versus unmodified web sites (the graph of the scrolling actions follows the same trends as the graph for time). Along the y-axis is the amount of effort — time in seconds or number of links — while along the x-axis is the location of each goal listed chronologically. Results for the unmodified sites appear as the left, darker column while results for the personalized sites are given in the right, lighter column. These graphs show that, for a majority of the sites, PROTEUS's personalizations appear quite useful: PROTEUS's addition of shortcut links and elision of unnecessary content reduced both the time required and the amount of visitor navigation at the sites. However, the study also illustrates a number of weaknesses in our implementation. These are not fundamental flaws in our approach, but rather implementation issues that must be addressed by a successful personalizer:

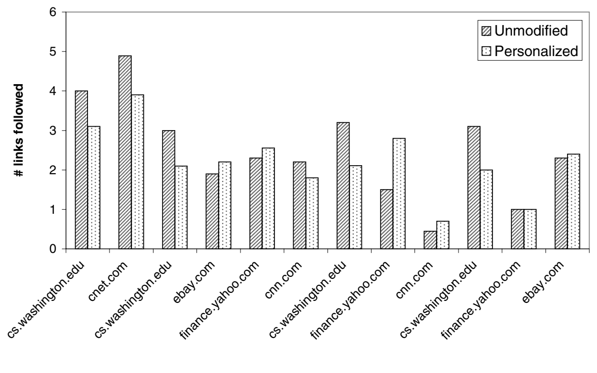


Figure 4: Links followed.

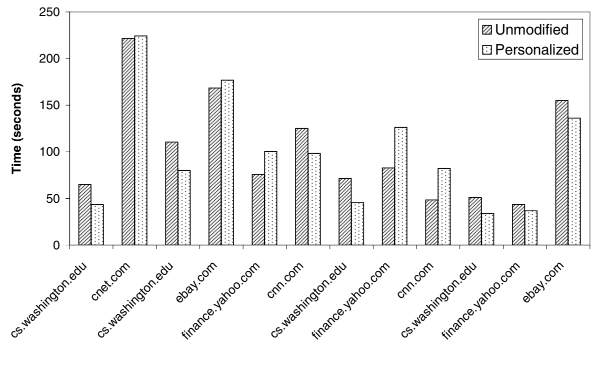


Figure 5: Time required.

Overly aggressive content elision. For a number of personalized pages, particularly those for the `cnet.com` and `finance.yahoo.com` domains, PROTEUS elided blocks of content that contained the links for which the visitor was looking, thereby requiring more effort to attain the information-goal. PROTEUS's visitor model incorporates both content and structural preferences, but it is clear that the weights of these preferences must be tuned carefully. In future work we are investigating how to incorporate a *confidence* model that predicts how accurate each component of the model will be on each page. For example, on a page containing predominantly links, the confidence in the structural model component is much greater than for the content (*i.e.*, word-based) component. That is, the personalizations on a page of links should depend more strongly on the probability the visitor will follow each link, and less strongly on the textual content of each anchor word. Additionally, the confidence in the structural component depends on the number of times

[8]We connected a Palm VII to a Ricochet wireless modem for network connectivity and used EudoraWeb [23] as the web browser on the Palm.

the visitor has viewed the page. If the visitor views the page frequently, and follows the same two links each time, then the personalizer has much higher confidence that those links are very important aspects of the page.

Inconspicuous elision links. We designed our method of eliding content explicitly to tolerate when PROTEUS elided content incorrectly: PROTEUS creates a link in the place of the removed content. However, the subjects in our study often could not find the link to the elided content, usually for one of two reasons. First, the link anchor text for elided content was taken from the first few words of the removed content. For the examples in Figure 2 the anchors are intuitive (*e.g.*, "Mutual Funds..."). However, when the elided content block contained, for instance, an advertisement at the top, PROTEUS selected the text from the advertisement as the link anchor — clearly, an unintuitive choice. Unfortunately, automatically finding a brief summary of a block of text, much less a page of content, is an open research problem [16]. A second reason elision links were difficult to find was that their visual appearance was no different from the other links on the page. Thus, the visitors could not tell when a link led to elided content, or when the link simply led to another page. A simple solution to this problem is to add a unique annotation to elision links. However, we are also investigating other approaches to making links more salient.

Over-estimated navigation probabilities. At graphically intense pages, such as `cnn.com` or `cnet.com`, the visitor can easily find his or her desired link when using the desktop browser. However, on the mobile device, such a page typically appears as a morass of links and text, often indistinguishable from one another, and the visitor has great difficulty in locating the link of interest. Unfortunately, PROTEUS calculates the probability that the visitor follows a link simply as how often the visitor has followed the link before while browsing on the *desktop*, when the visitor is unencumbered by the display constraints of the mobile device. Thus, PROTEUS tends to overestimate the link probabilities and, instead of adapting the page to reduce the visual complexity, will search for (what it views as) more useful personalizations, such as adding new shortcut links. In another line of future work, we are expanding the link probability estimate to take link *salience* into account, which will discount the value of visually complex pages and encourage PROTEUS to create simpler pages.

6. RELATED WORK

As we have discussed previously, a closely related line of research is Perkowitz and Etzioni's adaptive web sites [20, 21]. Our work differs from theirs on three important points:

- Adaptive web sites find singular transformations that appeal to all visitors at the site, while PROTEUS personalizes the web content for *each* visitor. Perkowitz and Etzioni provide a number of reasons to take their *en masse* approach. However, as we noted in section 1, mobile web visitors keenly need personalized web sites.

- The only transformation their adaptive web sites produce is synthesizing *index pages* — hubs of links to other pages in the site. PROTEUS considers a much wider range of adaptations that specifically aim to personalize every page on a site.

- Their evaluation metric for their transformations is based strictly on the navigational usage patterns of past visitors — the pages requested. Their model does not include any notion of a visitor's content interest, and cannot be used to determine how relevant novel pages or links are to the visitor. Our approach, based on expected utility, explicitly measures visitors' content interest with intrinsic utility and is applicable even to pages that the visitor has never seen before.

Two other systems similar to our own work are Digestor [3] and Pythia [8]. Neither system personalizes web content, but both concentrate simply on transforming content for display on a small screen with limited network bandwidth. Digestor uses a steepest-descent search similar to ours, to find an optimal page. However, Digestor rates the quality of a web page not on the visitor's expected utility, but only on how much screen space the page occupies (smaller pages have higher quality). This quality metric has two significant weaknesses. First, different elements on a page will have different value to the visitor, and should not be measured simply by how large they are. Second, this metric encourages the system to create degenerate pages — a blank web page receives the highest quality value. Both these weaknesses are addressed by a richer visitor model, such as we define for PROTEUS. In contrast to Digestor and PROTEUS, Pythia does not use search, and instead performs a *distillation* of every image on the requested web page to reduce the screen size and file transmit cost of the content. Pythia allows the visitor to subsequently select images to restore to a larger size in a refinement step. Pythia's approach is well suited to improving how images are transmitted and viewed on mobile devices, but is unable to improve the textual content or navigational structure in a web page. PROTEUS addresses both of these issues and, in fact, would be beneficially complemented by Pythia's approach.

The Web Browser Intelligence (WBI) [2, 15] project proposes an architecture of pluggable intermediaries that exist between web servers and web clients. These intermediaries generate, transform, and monitor the content they see in connection with requests made from web visitors, and can be used either individually or in chains. In many ways, our approach to web site personalization can be viewed as a particular transformation intermediary, providing highly personalized content for mobile clients. An interesting line of future research would be to reimplement PROTEUS in the WBI framework and investigate what new capabilities are available as a member of WBI's federation (in particular, what other intermediaries are available that could be usefully composed with PROTEUS).

Countless other ideas have been researched and systems been implemented that attack all or part of the web site personalization problem. We mention briefly several systems and projects that are most relevant to our approach. The Daily Learner [4] is an agent that learns a Palm VII user's preference for news content, by monitoring exactly which stories the user requests from both the Palm device and the user's corresponding desktop computer. Mobasher *et al.* [18] describe how to mine web usage patterns and web content to personalize the visitor experience, specifically, to recommend new content the visitor may like to see. The PersonalClipper [9] allows visitors to build their own custom views of web sites by recording navigational macros using a VCR-metaphor and selecting components of the target page to view with the mobile device. In contrast to PROTEUS, the PersonalClipper requires the visitor to *manually* create these navigational macros and place the result on the personal clipping. Letizia [14], WebWatcher [11], and adaptive web site agents [19] are agents that help guide web visitors by suggesting pages that the visitor may like to see. Letizia relies on the visitor's past behavior at the site, while WebWatcher suggests pages that the site's audience, as a whole, have favored in the past. Adaptive web site agents combine both approaches and integrate an animated avatar with which visitors interact. Our work differs from this earlier agent-based work in that we concentrate on personalizing the web site for each visitor, instead of merely suggesting which links to follow next, and in the depth to which we mine the web logs for useful access patterns.

7. FUTURE WORK

The framework for web site personalizers that this paper presents opens the door to a wide range of future research. We are pursuing three specific threads in ongoing work. First, we are looking at separating the web content model into data, structure, and presentational elements [7]. As separate elements, the web site personalizer can transform each aspect of the site independently for both a computational cost savings and for a expected utility performance gain. For instance, when the data on the site is separated from its presentation, the web site personalizer can very precisely tune what content the visitor will see — largely independent of how the content is currently presented.

Second, we are expanding the set of transformations that the PROTEUS can make to the site. Specifically, we are investigating a transformation that can aggregate blocks of content from many pages into a single view for the visitor. Such a transformation can allow PROTEUS to effectively create "portal pages" that integrate information from several separate resources on a web site. A single portal page that satisfies a wide range of information-seeking goals at a site would save the visitor a great deal of costly navigation.

In a third line of research, we are leveraging the whole population of visitor models described in section 2 to improve the site evaluation function for the current visitor. Along this line of work, we first cluster visitors based on models of their browsing behavior, and then use the history of *other* cluster members to influence the utility model of the current visitor. Clusters are formed by modeling visitors' browsing habits with Markov models, and clustering these models [5]. By incorporating the models of many closely related visitors, the web site personalizer can mitigate the sparseness of data for any single visitor and increase the accuracy of the expected utility estimate. Early results from this work show promise in finding cohesive clusters that can greatly improve the site evaluation for every visitor.

8. CONCLUSIONS

As the community of mobile web clients grows, so grows the need for web sites to cater to visitors off the desktop and off broadband network connections. This paper provides evidence that web site personalizers are an effective means of retargeting existing content to mobile clients in a way that is adapted for each visitor and that requires no additional work from the web site designer. We underscore the following contributions of our work:

- We propose a precise and broadly-encompassing definition of the automatic web site personalization problem, extending previous proposals;
- We apply the ideas of web site personalizers to automatically produce adapted web content for mobile visitors;
- We introduce a model with which we calculate the expected utility of a web site, a model that is fitted to each individual visitor to the site;
- We propose heuristic methods to trade off the precision of approximating expected utility with the computational expense in measuring it;
- We describe PROTEUS, an implemented web site personalizer, with which we evaluated our approach; and
- We present experimental evidence indicating that our web site personalization method improves the user experience for mobile web visitors, by reducing the time and effort required to attain their information-seeking goals.

Acknowledgements

The authors cannot offer enough thanks to the people who supported us in our work: Cathy Anderson, Adam Carlson, Oren Etzioni, Zack Ives, Tessa Lau, and Steve Wolfman. This research was funded in part by an NSF CAREER award to the second author, National Science Foundation grants #IRI-9303461, #IIS-987218, #IIS-9978567, and #IIS-9874759, Office of Naval Research grant #N0014-98-1-0147, and a grant from the Royalty Research Fund.

9. REFERENCES

[1] AvantGo, Inc. *AvantGo*. `http://www.avantgo.com/`.

[2] R. Barrett, P. P. Maglio, and D. C. Kellem. How to personalize the web. In *Proceedings of ACM CHI 1997 Conference on Human Factors in Computing Systems*, 1997.

[3] T. W. Bickmore and B. N. Schilit. Digestor: Device-independent access to the world wide web. In *Proceedings of the Sixth International World Wide Web Conference*, 1997.

[4] D. Billsus, M. J. Pazzani, and J. Chen. A learning agent for wireless news access. In *Proceedings of the 2000 Conference on Intelligent User Interfaces*, 2000.

[5] I. V. Cadez, D. Heckerman, C. Meek, P. Smyth, and S. White. Visualization of navigation patterns on a web site using model based clustering. In *Proceedings of the Sixth International Conference on Knowledge Discovery and Data Mining*, 2000.

[6] Digital Paths LLC. *DPWeb*. `http://www.digitalpaths.com/prodserv/dpwebdx.htm`.

[7] M. Fernandez, D. Florescu, J. Kang, A. Levy, and D. Suciu. Catching the boat with Strudel: Experiences with a web-site management system. In *Proceedings of ACM SIGMOD Conference on Management of Data*, 1998.

[8] A. Fox and E. Brewer. Reducing WWW latency and bandwidth requirements by real-time distillation. In *Proceedings of the Fifth International World Wide Web Conference*, 1996.

[9] J. Freire and B. Kumar. Web services and information delivery for diverse environments. In *Proceedings of the VLDB Workshop on Technologies for E-Services*, 2000.

[10] ILINX, Inc. *Palmscape 3.0.* `http://www.ilinx.co.jp/en/products/ps.html`.

[11] T. Joachims, D. Freitag, and T. Mitchell. WebWatcher: A tour guide for the World Wide Web. In *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence*, 1997.

[12] R. L. Keeney and H. Raiffa. *Decisions with Multiple Objectives: Preferences and Value Trade-Offs.* Wiley, New York, NY, 1976.

[13] J. Kleinberg. Authoritative sources in a hyperlinked environment. In *Proc. 9th ACM-SIAM Symposium on Discrete Algorithms*, 1998.

[14] H. Lieberman. Letizia: An agent that assists web browsing. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, 1995.

[15] P. P. Maglio and R. Barrett. Intermediaries personalize information streams. *Communications of the ACM*, 43(8), 2000.

[16] I. Mani and M. Maybury. *Advances in Automatic Text Summarization.* MIT Press, 1999.

[17] H. Mannila, H. Toivonen, and A. I. Verkamo. Discovery of frequent episodes in event sequences. *Data Mining and Knowledge Discovery*, 1:259–289, 1997.

[18] B. Mobasher, H. Dai, T. Luo, Y. Sun, and J. Zhu. Combining web usage and content mining for more effective personalization. In *Proceedings of the International Conference on E-Commerce and Web Technologies (ECWeb)*, 2000.

[19] M. J. Pazzani and D. Billsus. Adaptive web site agents. In *Proceedings of the Third International Conference on Autonomous Agents*, 1999.

[20] M. Perkowitz and O. Etzioni. Adaptive web sites: an AI challenge. In *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence*, 1997.

[21] M. Perkowitz and O. Etzioni. Towards adaptive web sites: Conceptual framework and case study. *Artificial Intelligence Journal*, 118(1–2), 2000.

[22] G. Salton and M. J. McGill. *Introduction to Modern Information Retrieval.* McGraw-Hill, New York, NY, 1983.

[23] QUALCOMM, Inc. *Eudora Internet Suite 5.0.* `http://www.eudora.com/internetsuite/eudoraweb.html`.

APPENDIX

A. USER STUDY QUESTIONS

- "What is the top editor's choice for ⟨*device*⟩ at cnet.com?", where ⟨*device*⟩ was one of: a "semipro" digital camera; an ultralight laptop; a Palm OS handheld; a CD-RW; a flat panel display.
- "What is the best price for ⟨*device*⟩ at cnet.com?", where ⟨*device*⟩ was one of: a "semipro" digital camera; an ultralight laptop; a Palm OS handheld; a CD-RW; a flat panel display.
- "What is the current stock value of ⟨*ticker*⟩? Start at finance.yahoo.com", where ⟨*ticker*⟩ is in the set MSFT, YHOO, AMZN, ATHM, PALM, HAND, OMNY, AVGO, MCOM, RIMM, INSP, EBAY.
- "What is the 52-week range for ⟨ticker⟩? Start at finance.yahoo.com", where ⟨*ticker*⟩ is in the set MSFT, YHOO, AMZN, ATHM, PALM, HAND, OMNY, AVGO, MCOM, RIMM, INSP, EBAY.
- "What stocks were ⟨*upgraded/downgraded*⟩ to a ⟨*strong buy, buy, hold, sell*⟩ rating today? Start at finance. yahoo.com"
- "Find the research profile for ⟨*company*⟩. Start at finance.yahoo.com" ⟨*company*⟩ is one of: Microsoft, Yahoo, Amazon.com, Excite@Home, Palm, Handspring, OmniSky, AvantGo, Metricom, Research In Motion, InfoSpace, eBay.
- "What is the average analyst rating (recommendation) for ⟨*company*⟩. Start at finance.yahoo.com" where ⟨*company*⟩ is one of: Microsoft, Yahoo, Amazon.com, Excite@Home, Palm, Handspring, OmniSky, AvantGo, Metricom, Research In Motion, InfoSpace, eBay.
- "Find ⟨*item*⟩ for sale and report the highest bid (or report that none is for sale). Start at www.ebay.com.". ⟨*item*⟩ was one of: an Olympus D-340R digital camera; a Nikon Coolpix 950; a Nikon Coolpix 990, a Palm Vx, a Samsung Syncmaster 150mp flat panel display, a Pentax K-mount wide-angle lens, a Pentax K-mount 70-200 zoom lens, a Pentax K-mount telephoto (300mm or longer) lens; a 1900 Morgan Dollar; a 1938 Walking Liberty half-dollar, S mint mark; a 1976 proof quarter-dollar, S mint mark; a US commemorative "Hudson" half dollar circa 1935.
- "⟨*When/Where*⟩ is the lecture for CSE ⟨*class*⟩? Start at www.cs.washington.edu", where ⟨*class*⟩ was one of: 505, 531, 533, 573, 589, 594.
- "Is ⟨*room*⟩ available at ⟨*time*⟩ on ⟨*date*⟩? ⟨*room*⟩ was either Sieg 322 or Sieg 114 and ⟨*time*⟩, ranged from 9:00 to 4:00 (by hour), and ⟨*date*⟩ ranged from November 13th to November 17th.
- "What office is ⟨*person*⟩ in? Start at www.cs.washington. edu", where ⟨*person*⟩ was one of sixteen selected graduate students or faculty at the University of Washington.
- "What is the topic of the upcoming colloquium, if any? Start from www.cs.washington.edu."
- "Is the colloquium in Sieg or in Kane, if any? Start from www.cs.washington.edu."
- "What is the top breaking news for today at cnn.com?"
- "What is the top business news for today at cnn.com?"
- "What's the latest news about ⟨*topic*⟩ at cnn.com?", where ⟨*topic*⟩ was chosen from: the first resident mission to the International Space Station (ISS); violence in the Mideast; the attack on the USS Cole; the U.S. presidential race; the Microsoft hacker attack; the Singapore Airlines plane crash in Taiwan.