The illustration caption within:

SEARCH:
Rain Gear
Waterproof
Cover

# PERSONALIZED SEARCH

*A contextual computing approach may prove a breakthrough in personalized search efficiency.*

PETER AND MARIA HOEY

CONTEXTUAL COMPUTING REFERS TO THE ENHANCEMENT of a user's interactions by understanding the user, the context, and the applications and information being used, typically across a wide set of user goals. Contextual computing is not just about modeling user preferences and behavior or embedding computation everywhere, it's about actively adapting the computational environment—for each and every user—at each point of computation.

With respect to personalized search, the contextual computing approach focuses on understanding the information consumption patterns of each user, the various information foraging strategies [3] and applications they employ, and the nature of the information itself. Focusing on the user enables a shift from what we call "consensus relevancy" where the computed relevancy for the entire population is presumed relevant for each user, toward personal relevancy where relevancy is computed based on each individual within the context of their interactions. The benefits of personalized search can be significant, appreciably decreasing the time it takes people— novices and experts alike—to find information.

Here, we review the evolution of the field of information retrieval (IR) [4], setting the stage for examining how a search can be personalized, with particular emphasis on the Web. We then describe the Outride system, and review a set of experiments.

The field of IR has evolved from analyzing the letters and words that make up the content of docu-

ments to the integration of intrinsic document properties like citations and hyperlinks to the incorporation of usage data. Content-based approaches such as statistical and natural language techniques provide results that contain a specific set of words or meaning, but cannot differentiate which documents in a collection are the ones really worth reading.

This need gave rise to a set of methods we refer to as "author relevancy" techniques. By computing what the most respected authors deem important, citation and hyperlink approaches provide an implicit measure of importance. However, these techniques can create an authoring bias where the meaning and resources valued by a group of authors determine the results for the entire user population. Imagine for a moment if the Java programming language was called something different. A query for the term "java" on the Web would produce a different set of results, likely about coffee, which is probably closer to most users' expectations. Additionally, a ranking bias can occur when, for a given topic, the authoring community values a different set of resources than the general population. A typical example of this is link promotion where a set of highly interconnected sites is created by a small set of authors in an attempt to appear to become the most relevant resources on a particular topic.

Usage-based IR methods add to the previous research by leveraging the actions of users to compute relevancy. A usage rank is computed from the fre-
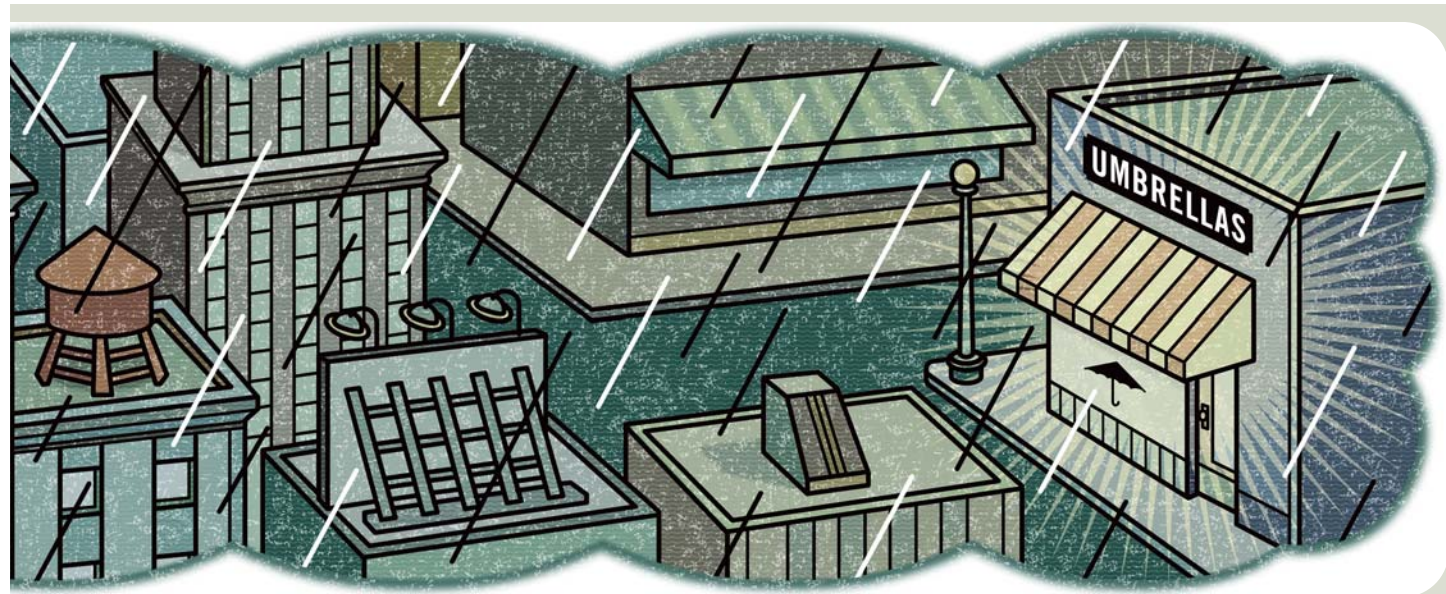
THE MAGNITUDE OF THE DIFFERENCE BETWEEN THE OUTRIDE SYSTEM AND THE OTHER ENGINES IS COMPELLING,

quency, recency, and/or duration of interaction by users. This provides a direct measure of what is relevant at any point in time to the users of the information system. Typically, the usage rank for a page is combined with content and link-based ranking methods. Although impossible to infer from link and content approaches, usage techniques readily compute changes in relevancy over time. These temporal changes include: ephemeral events such as record-breaking usage driven by interest in the comet Shoemaker-Levy; emerging trends such as the growth in usage of MP3s; seasonal favorites like the popularity of flowers around Valentine's Day; and faddish events such as the rise and fall of the NCSA Mosaic Web browser.

Interestingly, the retrieval process can be infused with different granularities of usage data—individual, group/social, and census—enabling a system to fallback to a coarser level of usage data in the face of uncertainty. The latter forms create a kind of social relevancy, where the notion of importance is defined by the usage of a community of users. Very few usage-based systems have been developed, with the most notable exception being Direct Hit's collaborative filtering-inspired approach that monitors which search results people select.

What's curious about these approaches is that relevance is measured as a function of the entire population of users. One can view this as an attempt to optimize the consensus relevancy for any given topic. For any query, relevancy is computed identically for all users without acknowledging that *relevance is rela-tive* for each user. Further, none are able to differentiate based upon who is searching, their current context, interests, and/or prior knowledge. What's needed is a way to take into account that different people find different things relevant and that people's interests and knowledge change over time. What's needed is a way to compute personal relevancy.

## The Outride Approach

We posit that at least two different computational techniques need to be combined to personalize search: *contextualization* and *individualization*. By contextualization, we mean the interrelated conditions that occur within an activity. Individualization means the totality of characteristics that distinguishes an individual. Contextualization includes factors like the nature of information available, the information currently being examined, the applications in use, when, and so on. Individualization encompasses elements like the user's goals, prior and tacit knowledge, past information-seeking behaviors, among others. These elements are used to build a user model to personal relevancy computationally, as we will describe. It is this focus on the user and their context within the application of search that makes personalized search a compelling area to explore within the framework of contextual computing.

It is worth mentioning upfront that since the following techniques alter the search experience, careful integration of these features into the user interface is required. In particular, the interface needs to provide a way to explain what the system is doing to personalize

ESPECIALLY GIVEN THAT MOST SEARCH ENGINES ARE LESS THAN 10% BETTER THAN ONE ANOTHER.

{ BY JAMES PITKOW, HINRICH SCHÜTZE, TODD CASS, ROB COOLEY, DON TURNBULL, ANDY EDMONDS, EYTAN ADAR, AND THOMAS BREUEL }

the experience as well as to undo the personalization.

The primary ways to personalize a search for an active searcher are *query augmentation* and *result processing.* Figure 1 shows the architecture of the Outride system where the personalization engine sits between a user interface and an intra/Internet search engine. Once a user has entered a query, the query can be compared against the contextual information available to
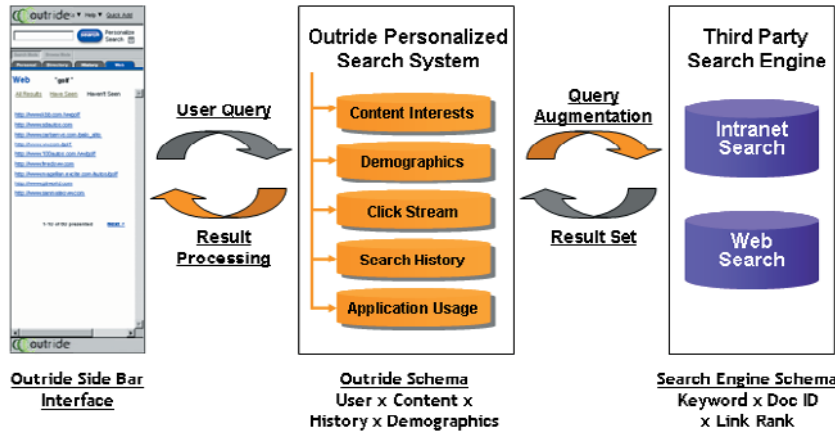


Figure 1. Outride system architecture.

determine if the query can be refined to include other terms. For example, if a user is looking at a series of pages on car information and searches for "contour," the system may augment the query by adding the term "car" or "ford" to provide the user with results about the Ford Contour car.

In much the same manner, the user model can be used to perform query augmentation where the similarity between the query term and the user model is computed. If the query is on a topic the user has previously seen, the system can reinforce the query with similar terms, or suggest results from prior searches. If it is a new topic, chances are the system should not augment the query, or if it does, it can help define what the topic is *not* about by providing a diverse set of results to the user. The final output of query augmentation is a more precise query that can be shown to the user and submitted to a search engine for processing.

Once the search engine has processed the query, the results can be individualized. Information can be filtered based upon information in the user's model and/or context. For instance, if the model contains demographic information, the system can point people directly to local restaurants and entertainment or prevent minors from seeing adult content. As with query augmentation, the user model can re-rank search results based upon the similarity of the content of the pages in the results and the user's profile. With this, a Java programmer gets information related to developers while a teacher of programming languages
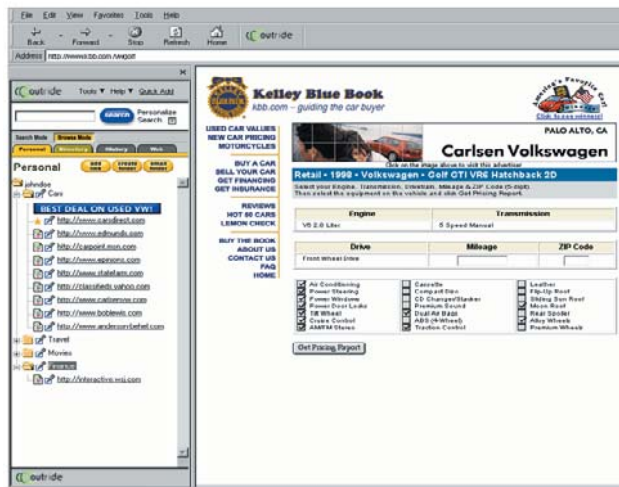
gets information about Java tutorials and overviews.

Another useful result processing method re-ranks the results based upon the frequency, recency, or duration of usage, providing users with the ability to identify the most popular, faddish, and time-consuming pages they've visited. For example, a feature we call "Have Seen, Have Not Seen" provides a quick way to identify new information and return to information already seen. This enables users to effectively say, "You know what I know, show me what I do not know," and conversely, "Show me only what I already know." If usage information is aggregated across multiple profiles, re-ranking can be done across specific user groups ("Show me the pages hikers use the most") or even the entire population ("What's the coolest page on the Internet right now?").

## The Outride Personalized Search System

The Outride system was designed to be a generalized architecture for the personalization of search across a variety of information ecologies. Figure 2 shows the Outride client integrated into the sidebar of Internet Explorer as a 64KB component. As part of the browser, the component supports direct manipulation and has access to all user interactions.

The sidebar is partitioned into four separate information spaces via a tabbed interface: a personal hierar-



chy of each user's links (Personal), a catalog of links (Directory), the user's surf history (History), and search results

Figure 2. Outride client integrated into Internet Explorer sidebar.

from the entire Web (Web). The partitioning was designed to directly support the conceptual model and tasks people utilize in searching the Web. Two complementary modes—browse and search—are supported for each information space. The scope of activities defined by the current mode is limited by the

## Testing Methodology and Results

Outride, with eTesting Labs as an independent tester, designed a series of empirical tests to measure if the Outride system makes searches faster and easier to complete. Instead of measuring precision and recall, the elapsed time to successfully complete a search and

information space; for example, users can limit a search to their personal links, their surf history, or the Directory.

User models are computed from the content in these information spaces in the Outride sidebar. The models are based upon the ontology of the Open Directory Project (ODP) where each user has their own weighting across the top 1,000 categories of the ODP. Upon download of the sidebar component, if the user imports a set of favorite links, the system fetches the pages and classifies them into the ODP adjusting the weights accordingly. If no links are imported, the user starts out with no content weighting. As the user clicks around the Web each click is captured by the sidebar, classified, and the user model is updated accordingly. The last 1,000 unique clicks of each user are stored in their surf history.

Query augmentation is performed by integrating various clues provided by the instrumentation of the interface and user models. If a user is browsing the ODP, the category name and its contents are compared to the query to see if they are similar. Likewise, the title and contents of the currently viewed Web page is checked. Our initial investigations found that while these cues provide meaningful data, comparing the query to the users' content profile using vector methods provide better results. Only queries exceeding certain similarity thresholds are augmented automatically by the system.

Result set processing is performed across an expanded set of results, typically 1,000, from the backend search engine. Filtering by "Have Seen, Have Not Seen," and usage-based re-ranking are straightforward to implement. To re-rank search results based upon the user profile, the titles and other metadata from the pages are compared via vector methods against the user profile. Although far from optimal, if we were to re-rank results based upon the context of each result page it would require indexing the entire Web—an option we felt better handled via a partnership.

We found the combination of query augmentation and result processing with a contextually designed interface to be quite effective in making search easier and faster.

the number of interface actions (mouse clicks or keyboard entries such as entering a search term) were used as metrics. While not true for all search tasks, we believe these metrics more accurately reflect how people generally search for information on the Web: They stop once sufficient information is identified. Moreover, time and actions directly measure the real costs incurred by users in finding information.

Creating a representative set of search tasks and topics with which to measure search systems is not a solved problem. The topics and tasks used in the study were developed in tandem with eTesting Labs and were based on our mutual prior experiences in studying Web search behaviors through questionnaires, lab studies, client monitoring, and proxy log analysis. It must be noted that several tasks measure functionality provided by Outride but not the other search engines. The search activities of Web searching; searching for a previously accessed Web page; searching for a page not accessed before; and searching through a set of preferred pages were performed across the topics of football, car shopping, and travel planning. Each participant performed 12 search tasks with a particular search engine and with Outride, alternating tasks between each system.

In order to have consistency throughout the tests, a default user model was used for all participants. The profile included a set of contextually related preferred links and a surfing history related to the search tasks. Due to testing limitations, however, the profile could not be based on each participant's actual prior Web use. The contents of the profile were briefly overviewed to the participants to illustrate the personalized aspects of the system.
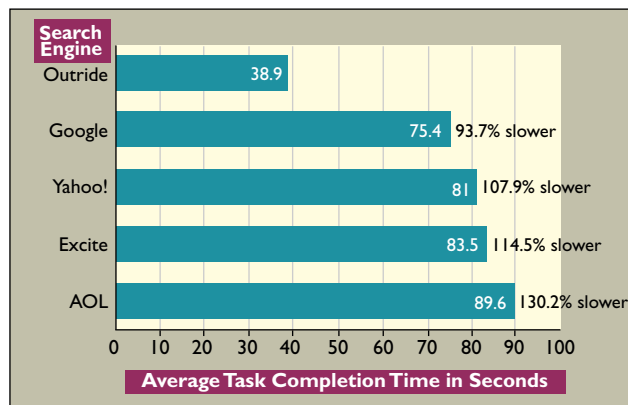
Some 48 novice (less than 8 hours per week of Internet usage) and experienced (over 8 hours per week of Internet usage) Web users were introduced to Outride and one of the following search engines: AOL Search, Excite, Google, and Yahoo. Google was used as the underlying search engine by the Outride system. The one-hour test was administered by a trained eTesting proctor and consisted of an introduction, a tutorial, a review, and the 12 search tasks. Each search task was limited to three minutes.

Figure 3 shows that participants found the answers more quickly with Outride than with any other

engine. On average, participants took 39 seconds to complete the tasks using Outride compared to 75 seconds using Google. As shown in Table 1, participants needed fewer actions when using Outride, completing tasks in 11 user actions compared to 21 user actions using the next fastest engine.

Table 2 shows that novice and experts were able to successfully find information the fastest with Outride. Novice Outride users were able to complete tasks in 45 seconds, compared to 76 sec-

**Figure 3. How long to get an answer?**



(% slower from Outride enabled search)
Source: ZDLabs/eTesting, Inc. Oct. 2000

negatively influenced the outcome, as the profile did not represent the test participants' actual surfing patterns, nor were the participants intimately familiar with the content of the profiles. Third, some of the gains are likely due to the user interface since the Outride sidebar remains visible to users across all interactions, helping to preserve context and provide quick access to core search features. For example, while search engines require users to navigate back and forth between the list of search

**Figure 4. Novices versus experts.**
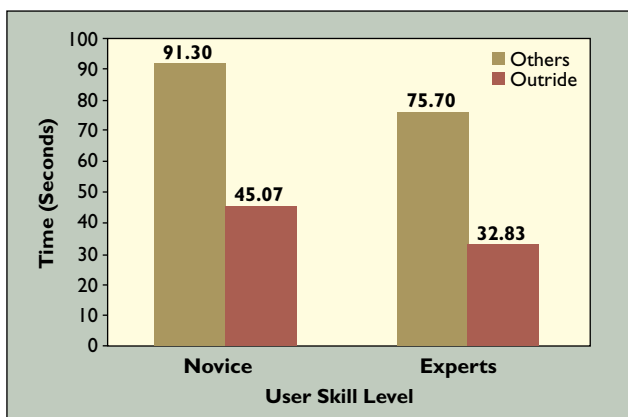


(Average time to complete task)
Source: ZDLabs/eTesting, Inc. Oct. 2000

| Search Engine | User Actions | Difference (%) |
|---|---|---|
| Outride | 11.2 | |
| Google | 21.2 | 89.6 |
| Yahoo! | 22.4 | 100.5 |
| AOL | 23.1 | 107.0 |
| Excite | 23.3 | 108.5 |
| Average | 22.5 | 101.4 |

Source: ZDLabs/eTesting, Inc. Oct. 2000

| Engine | Expert Time | Rank | Novice Time | Rank | Average | Rank | % Difference |
|---|---|---|---|---|---|---|---|
| Outride | 32.8 | (1) | 45.1 | (1) | 38.9 | (1) | 0% |
| AOL | 92.3 | (5) | 87.0 | (4) | 89.6 | (5) | 130.2% |
| Excite | 75.7 | (3) | 91.3 | (5) | 83.5 | (4) | 114.5% |
| Google | 72.5 | (2) | 78.4 | (3) | 75.4 | (2) | 93.7% |
| Yahoo! | 85.1 | (4) | 76.9 | (2) | 81.0 | (3) | 107.9% |

(in seconds, with placement in parenthesis)
Source: ZDLabs/eTesting, Inc. Oct. 2000

**Table 1. Results of user actions' study.**

onds using Yahoo. Expert users were even faster, requiring 33 seconds using Outride, and 73 seconds using Google. Interestingly, novices using Outride were faster than experts using other engines. Paradoxically, for AOL and Yahoo, novice users were faster than expert users.

While the results may seem overwhelmingly in favor of Outride, there are some issues to interpret. First, some of the scenarios contained tasks directly supported by the functionality provided by the Outride system, creating an advantage against the other search engines. Indeed, Outride features are specifically designed to understand users, provide support by the conceptual model and tasks users employ to search the Web, and to contextualize the application of search. This is the goal of contextual computing and why personalizing search makes sense.

Second, while the use of default profiles could have provided an advantage for Outride, it also could have

results and specific Web pages, Outride preserves context by keeping the search results open in the sidebar of the Web browser, making the contents of each search result accessible to the user with a single click.

**Table 2. Overall timing results.**

Still, the magnitude of the difference between the Outride system and the other engines is compelling, especially given that most search engines are less than 10% better than one another [2].

## Future Directions

Personalized search opens the door to a new set of challenges and opportunities. One difficult problem is modeling a user's changing interests over time. Although power laws of recency and frequency have been shown to sufficiently model human memory [1] and can be applied to information consumption behaviors, there will always be times when exceptions arise. Carefully designed interfaces can help alleviate inaccurate personalization and allow users

to control the extent of the personalization.

Additionally, much of the success of query augmentation rests in correctly detecting user context switches, which is not an easy task. While not optimal, incorrect estimation can be mitigated by enabling the user to readily undo the personalization.

Privacy issues are plentiful in systems that store models based upon user's interactions with information, mandating that users be able to inspect and modify their models. One side effect of Outride modeling interests based upon the ODP and having the ODP be browsable in the client is that users can turn portions of their model on and off with a single click as they navigate their folders of the ODP hierarchy.

Finally, one of the benefits of consensus search is that all users get the same results, which fosters result sharing and the use of search results for navigation—an issue that can also be addressed at the interface level.

## Conclusion

We have presented here a new type of IR system that personalizes the search experience for each user across their interactions. We have shown initial evidence to support our firm conviction that the contextualized computing approach toward the personalization of search is the next frontier toward significantly increasing search efficiency. **C**

**REFERENCES**
1. Anderson, J.R. *Cognitive Psychology and Its Implications.* Freeman, San Francisco, CA, 1980.
2. eTesting Labs. Google Web Search Engine Evaluation; www.etestinglabs.com/main/reports/google.asp
3. Pirolli, P. and Card, S.K. *Psychological Review* 106, 4 (1999), 643–675.
4. Salton, G. and McGill, M.J. *Introduction to Modern Information Retrieval.* McGraw-Hill, New York, 1983.

**JAMES PITKOW** (pitkow@parc.com), PARC, Inc., Palo Alto, CA.
**HINRICH SCHÜTZE,** Novation Biosciences, CA.
**TODD CASS,** Intelligent Markets, Inc., San Francisco, CA.
**ROB COOLEY,** KXEN, Inc. San Francisco, CA.
**DON TURNBULL,** University of Texas, Austin.
**ANDY EDMONDS,** Vivendi, Encino, CA.
**EYTAN ADAR,** Hewlett-Packard Laboratories, Palo Alto, CA.
**THOMAS BREUEL,** PARC, Inc., Palo Alto, CA.