When Cookies Aren't Enough:

Tracking and Enriching Web Activity with Recer

Matthew Chalmers

Computing Science, University of Glasgow, Glasgow G12 8QQ, U.K.

http://www.dcs.gla.ac.uk/~matthew

Abstract

We describe recent work on the path model, an extension of recommender systems that focuses on the temporal order of information activity. URLs and words displayed in a web browser, as well as filenames, words and symbols used in the *xemacs* editor, are timestamped and logged in *paths*. Recommendations are made by comparing the representation of the user's ongoing activity with the rest of their path, and with the paths of selected others. The prototype system described thus affords access to and reuse of program files, class names, variables, URLs and words consistently used in similar contexts. We treat heterogeneous data uniformly, as symbols. The *Recer* system relies on recurrent patterns of symbol use that emerge from activity, and that form adaptive and subjective categorisations. We describe the system's implementation and our early experience with it, and contrast our approach with that of traditional IR. We discuss ways in which such recommender systems are set in a wider system of activity tracking, profiling, and commerce, and discuss the tradeoffs between of locality and privacy of information with its publicity and economics.

1. Introduction

The majority of work on information access and visualisation focuses on finding structure inside data, based on content analysis. The words on the page and the colours in the image are analysed, and the terms and symbols produced serve as indices for search and retrieval. This mainstream approach has become familiar to a widespread audience primarily through World Wide Web search engines such as AltaVista. The techniques of information retrieval (IR) at the core of this approach have changed relatively little over the past decades, as can be seen in comparing older [Salton] and more recent [Yates] introductory textbooks in this field.

A major drawback of this approach is the limited ability to handle the heterogeneous data so typical of today's information environment. The content on the Web is a rich mixture of text, images, sounds, programs, data files and, of course, links. IR's response to this is to add tags and labels to the content associated with a web page or site, forming so–called 'metadata'. Metadata schemes such as PICS [Resnick96] are intended to offer an external, consistent and objective categorisation scheme for description of this heterogeneous mix. Metadata schemes have met with limited success for many reasons, not the least of which is the difficulty of obtaining consistency and objectivity from a huge number of authors, each of whom has their own view as to how their web pages should be represented. In their hands, metadata is just more data, to use as they will.

Another limitation of this kind of content–based formalisation relates to the reader rather than the author. As Tim Berners-Lee pointed out, search engines "are notorious for their ineffectiveness… A web indexer has to read a page of hypertext and try to deduce the sorts of questions for which the page might provide the answer" [Berners-Lee]. As readers and their contexts of use vary, any text is open to an infinite variety of happily contradictory interpretations. This stands in contrast to the way that indexers represent information formally, finitely and fixedly.

There has been a significant body of research on the application of visualisation techniques to information retrieval, stemming from early work such as [Korfhage], [Lin] and [Chalmers92]. This work is directed towards improving traditional IR's reliance on simple interfaces, such as ranked lists of search results, and offering more sophisticated interfaces involving 2D and 3D graphics, colour, animation, sound and so forth, so that our perceptual skills and spatial memory can be brought to bear.

Information visualisation techniques generally use similar representational and analytic techniques to that of IR, focusing on content. They then work to re–present the multidimensional content in a space of lower dimensionality, such as a 2D map. Thus they are prone to the same limitations as traditional IR, in focusing on patterns in topical similarity rather than on a deeper notion of meaning and relatedness. They aim to give 'the big picture' of a body of information, but as a result can be too abstract and general to suit any particular person in any particular context. The big picture can be reductionist, and so fail to be a useful, practical tool.

By letting people take advantage of their everyday skills of perception, visualisation aims to make complex information more comprehensible and useful. One drawback of its approach is its rather limited view of perception. Information visualisation tends to take into account only low level perception, dealing more with the eye than the brain. While recognition of patterns of spatial and aural form are undoubtedly powerful abilities that we should make the most of, higher level issues of linguistic and social perception are at least as important, as has been recognised in closely related but more mature fields such as cartography. There, a reductionist view of information content has been superseded by a combination of 'low level' perceptual physiology with 'high level' contemporary semiology [MacEachren].

The importance of the subjectivity of association and interpretation is also evident if one looks back to perhaps the most seminal document in information systems research: Vannevar Bush's *As We May Think*, recently republished as [Bush] and the subject of at least one retrospective symposium [Simpson]. *As We May Think* is often held as foundational to hypermedia, the World Wide Web, and information retrieval. Bush does not propose that the collection of links used in connecting hypermedia objects is a solution in itself. A 'sea of links' merely replicates the problem of having a sea of tuples or documents. Bush proposed that human activity in the form of 'trails' or 'paths' was the solution i.e. particular human–selected sequences of links associating information objects, and not the vast set of possible associations that hypermedia, databases and libraries provide. Bush's scheme of information representation, his abstract model that might be instantiated in photographic film or computer memory, involved only identifiers of documents, and links between identifiers to form paths. Paths might then be individually numbered and archived. A path, as a representation, does not directly involve content but relies instead on the human interpretation of content. The accessibility, meaning or use of an object is primarily determined by the author's chosen context of the path. A link between two objects thus only has meaning within the context of a particular path of a particular person. The hypermedia and retrieval systems that declare an origin in Bush's work generally do not work with the phenomena of context and person as Bush proposed: context and subjectivity are not part of their underlying model.

2. A Complementary Approach

There are a number of recently developed information access systems that operate in complementary fashion to IR, mixing the different types of data typical of the Web, and not solely relying on content in attempting to fixedly categorise content. In this they face up to an aspect of the complexity of the real world task at hand, in that the bodies of information that we deal with in our everyday activities simply are heterogeneous in content. They are called collaborative filters [Goldberg], or recommender systems [Resnick97]. While recognising the historical precedence of Goldberg et al., we will generally use the latter term since this paper addresses reflexive as well as collaborative use of information.

The models underlying recommender systems directly involve the people involved in information access, and these models adapt with ongoing use rather than being static analyses made prior to, and hence independently of, use. They thus take account of the subjectivity of information use quite explicitly. Recommenders circumvent the complexity of analysing the content of heterogeneous objects by instead using profiles or patterns in people's choices of object identifiers. Such systems present to each user objects rated highly by people with similar profiles. Objects already in the user's profile are generally excluded from recommendation. Each symbol in a recommender system is 'mutually indexed,' in that its representation and manipulation is in terms of relative patterns with regard to other symbols in the same system, as expressed in people's activity, rather than in terms of an external and absolute categorisation scheme. Profiles generally are unordered sets, so recommendations are specific to the user but not specific to their current activity or information need.

http://www.arosabergbahnen.ch/	18.0
http://www.arosabergbahnen.ch/Grafiken/collage.jpg	15.0
http://www.klosters.ch/images/tn_gotschna.jpg	9.0
http://www.klosters.ch/gotschna.html	9.0
http://194.158.230.224:9090/telenet/CH/180/2.g-html	9.0
http://www.arosa.ch/main.html	9.0
http://www.arosa.ch/skiauswahl.html	9.0
http://ad.adsmart.net/src/goski/mountains^1?adtype=ac&bgcolor=F...	9.0
http://www.arosabergbahnen.ch/home.html	9.0
http://www.arosa.ch/	9.0

Figure 1. A recommendation list from an early version of our system. Starting to choose a day for a ski trip, web pages were accessed with detailed weather reports for the mountains of Switzerland, including the *telenet* service of a local university. Recommended URLs were drawn from six sequences of past activity in three people's paths, and were mostly for web sites of ski resorts near Zürich, such as Arosa and Klosters. The numbers in the right hand column are weighted tally values.

Extending the recommender system approach to take more account of the temporal order of activity is at the core of the path model, which is the focus of this paper. A *path* is a time–ordered history of a person's use of information objects. The use of symbols such as URLs accessed in a web browser is logged over time, and the log serves as the representation of that user's activity. An earlier prototype for recommending and visualising URLs was described in [Chalmers98]. The range of influences on the path model's development is broad, including urban design theory and *As We May Think*, as discussed in [Chalmers99a] and [Chalmers99b] respectively.

Unlike most recommender systems, the context of activity is an essential part of the approach. Each path entry, i.e. each recorded instance of a symbol's use, is associated with the temporally neighbouring entries in that path. We then use statistical patterns of symbol recurrence in representing relevance and information need, by treating the most recent path entries as an implicit request for recommendations. While we will discuss system operation in greater detail in a later section, we briefly summarise the basic procedure here.

The system periodically takes the symbols recorded in recent path entries and searches for past occurrences. This search can cover one's own path and/or a selection of others'. The system collects the context of each past occurrence of each recent symbol i.e. a window containing the path entries immediately following the past occurrence. It tallies the symbols collected from these windows and then, in a final 'exclusion step,' discards any symbols that were recently used. The remaining symbols are presented as a ranked recommendation list. The system then sleeps for a time before waking, collecting a new set of recent symbols and beginning again. To afford some privacy, path logging can be turned on and off at will. An example recommendation list is shown in Figure 1, and is discussed in section 4.

We take account of IR research's persistent warnings against topical, impersonal and static notions of relevance, voiced in recent work such as [Wang] and [Schamber], and that of earlier decades e.g. [Saracevic] and [Cooper]. In the latter one reads that it "is really documents with high utility, and not merely [topically] relevant documents, that the user wants to see". When past occurrences are temporally near to each other, their contexts or windows overlap. Symbols within overlaps get higher tallies. Highly ranked symbols tend to be, therefore, those symbols most consistently used just after the periods most similar to recent activity. We assume that consistency of use in similar contexts is the best indicator of appropriate (literally, utile) recommendations. Just as the stereotypical recommender system draws recommendations only from the most similar people to oneself, path systems draw recommendations from the most similar contexts to one's own.

Like recommender systems, the path model is dependent on external systems to initialise and extend the set of symbols recorded and used in recommendations. Paths would be useless without people's use of Web search engines and file browsers, and indeed editors, mail tools, magazines, journals and so forth. We see this not as a weakness, but as realistic and pragmatic acceptance of the interdependence and holism of tools, and their embedding in the wider world. We interleave information tools in everyday use, and the sets of objects they operate on overlap. One of the improvements to the system of [Chalmers98] described

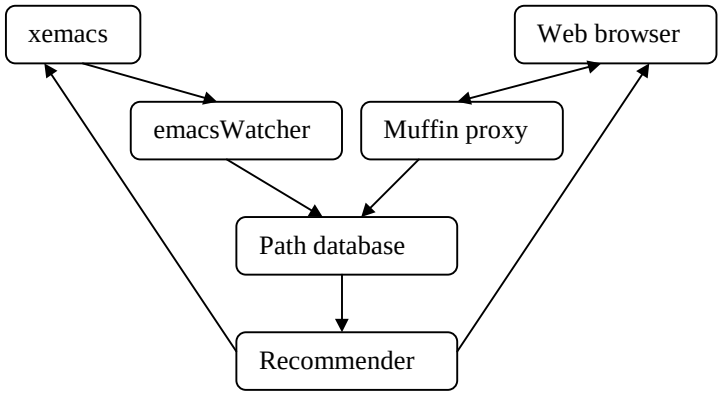


Figure 2. In the *Recer* system, symbols used in *xemacs* buffers are sent out to a logger program that enters timestamped path entries into the database. A similar logger has been inserted inside the *Muffin* web proxy, so that URLs and words inside web documents can be timestamped and logged. The recommender program looks at the most recent symbols in the user's path, and displays the resulting recommendation lists. Clicking on a URL in such a list triggers display in the web browser, and clicking on a file triggers *xemacs* to load it.

in this paper is that we now record activity in the *xemacs* text editor, in the same path as that of the web browser. Web activity can thus potentially trigger recommendations for local files, and vice versa.

Also, we have begun adding the content of documents to paths. We see titles and words as instances of a more general class: symbols. We are responding to simple phenomena such as titles consisting of single words and, conversely, document titles and URLs being mentioned in a text, and therefore being 'content terms.' Secondly, we are following the commonplace practice of IR of handling terms and titles uniformly, for example in representing query terms and documents by term vectors. (A more detailed comparison of IR, recommenders, paths and workflow is given in [Chalmers99b].)

3. System Implementation

Most of our system, *Recer*, is written in Java, although some parts involve Unix shell scripts and Lisp. We also use a commercial relational database as the low–level repository of paths. An overview of the system is given in Figure 2, below.

We are experimenting with two modes of activity logging. The first records only the URLs and filenames that identify the documents active in an editor and a browser. In the *xemacs* editor, a timer triggers a check every five seconds for a change between editor buffers. When one is detected, and there is a filename associated with the new buffer, the filename is written out to a temporary log file along with a timestamp roughly representing the time of the buffer switch. A shell script, *emacsWatcher*, periodically pipes any new lines added to the log file into a Java program that enters a path entry for each via a JDBC connection. The name of the database table to write to, e.g. *path_matthew*, is determined by the username property of the Java environment. In the *Muffin* web proxy, a Java program freely available from *muffin.doit.org*, each time a new page is accessed we insert a path entry for the URL in question. This first mode does not log content, and we can either use the simple display shown in Figure 1 or the display described later in this section.

In the second logging mode we create a line of text once every second, containing the nearest complete word or symbol to the *xemacs* insertion point, the current time, and the filename associated with the current editor buffer. This is only written out when the symbol differs from the previous recorded symbol. Each time a new page is accessed in the Web browser, the proxy records the URL and all unique terms in the page, including contained URLs. Again, all path entries for the same person are written to the same table.

In a path table, each entry or row is a triple: (timestamp, symbol, source). In the case of a content symbol, the second element represents the symbol itself while the third element represents the document or file it was found in. Otherwise, the third is a duplicate of the second. For efficiency, we use the hash function of [Pearson] to make 32 bit fingerprints for the symbol and the source. A separate lookup table stores

(fingerprint, string) pairs. Also, each path table has a primary key based on both the timestamp and the symbol, and an index based on this key.

We treat the user's most recent path entries as an implicit request for recommendations. Periodically, the recommender takes these entries as a set, and searches the database for past occurrences of each set member. (At this stage it also collects the set of symbols used later in the exclusion step.) The search can be over any combination of path tables i.e. the set of paths is a resource shared amongst those who create paths. By offering selectivity of which paths (or rather, whose paths) to draw from, we allow people to use knowledge of their colleagues' skills and interests to steer the recommendation process.

The recommender's interface is split into two parts. The first, as shown in Figure 4 below, displays recommended symbols. The second, not shown, has buttons for selecting paths to draw from and for triggering an immediate recommendation, as well as four sliders which set periods of time explained below: the recent period, the window after each past occurrence, the period used for excluding symbols from recommendations, and the sleep period between recommendations.

For each of the most recently used symbols, and from each selected path, the recommender collects the context of each past symbol occurrence i.e. the entries within the window after each past occurrence. The recommender tallies and collates these symbols, and removes any that are amongst the set of excluded symbols. It then splits this ranked list into three: those symbols that appear to be remote URLs, local filenames, and 'everything else.' In the case of no content being logged, the third list is empty. These ranked lists are presented as scrollable lists as in the example of Figure 4 (discussed later). Clicking on a URL sends it to the web browser for display—and consequent logging. Clicking on a local filename triggers xemacs to load it. (Admittedly, this inhibits loading local files directly into the web browser, and files specified by URLs directly into *xemacs*.) Note that a URL or filename can be recommended even if no–one has actually visited it. If a URL occurs frequently enough in the content of files or pages associated with recent work then it can gain enough tallies to be recommended. In the *Everything else* list, clicking pops up a window which shows a list of the URLs and filenames which were recorded as containing the selected symbol. A click in this pop–up list triggers display as before. We do not identify the paths (and people) that contributed most to recommendations. No–one has so far objected to the idea, but then no–one has experienced it. Later we may make a less conservative trade–off between the utility of finding out about contributors and the potential invasiveness in making explicit exactly who used what, and when [Bellotti].

The system thus recommends to the user symbols that were frequently used in similar contexts but that it has not observed recently in the user's path. Symbols representing heterogenous objects, drawn from a variety of sources, are interwoven in this process even though, in display and later interaction, they may later be repartitioned. The recommendation process is triggered automatically and periodically, and takes a varying amount of time dependent on the number of past occurrences, but on average takes roughly 30 seconds. By using temporary tables on the database server for all intermediate results, we minimise communication costs and shift most of the computation away from the user's workstation.

Statistics for recommendation and visualisation thus form categorisations or abstractions over path entries that are not fixed *a priori*. Instead, recurrence statistics are made anew for each person at the time of, and based on the context of, each recommendation operation. They are also ephemeral, in that once a new path entry is made after a recommendation e.g. upon clicking on a recommended URL, the old recurrence statistics are no longer valid. Adaptation can evince itself even if one refrains from logged activity for a time, since one's recommendations may vary as other people's activities extend their paths.

Unlike our previous implementations, we have decoupled the periods used to represent recent activity and to define the set of excluded symbols, affording greater user control at the expense of another interface slider. By lengthening the exclusion period, the likelihood of recommending common symbols can be decreased. In many IR systems, such symbols would appear in a stop list and be excluded at the start of processing, rather than at the end of processing, as here. Stop lists involve an assumption of the validity or utility of a contextually independent categorisation: prior to or independently of use, one can partition symbols into those that are useless in all situations and those which are not. We may gain overall by trading such assumptions for computation i.e. for the increasingly expendable resources of CPU cycles and disk space. Exclusion and recent sets afford the required effect without relying on this assumption, as may be made clear from three cases.

Figure 3. Four snapshots of an interactive 'map' of URLs, scaled down from actual screen size, and with the frame and controls of the surrounding visualisation tool cropped. The URLs visualised are those collated prior to the exclusion and ranking which led to the list in Figure 1. Access of web pages with detailed weather (*wetter* in German) reports for the mountains of Switzerland led to recommendations for URLs for web sites of ski resorts near Zürich, such as Arosa, Davos, and Klosters, as well as URLs for other weather pages. Each snapshot shows the same positions, based on co–occurrences, but shows in a lighter colour those URLs that match the given search string e.g. *arosa* in *www.arosa.ch*. The *Klosters* snapshot also shows three URLs which have been 'popped up' to show textual detail (although illegible here due to scaling down of the original snapshots).

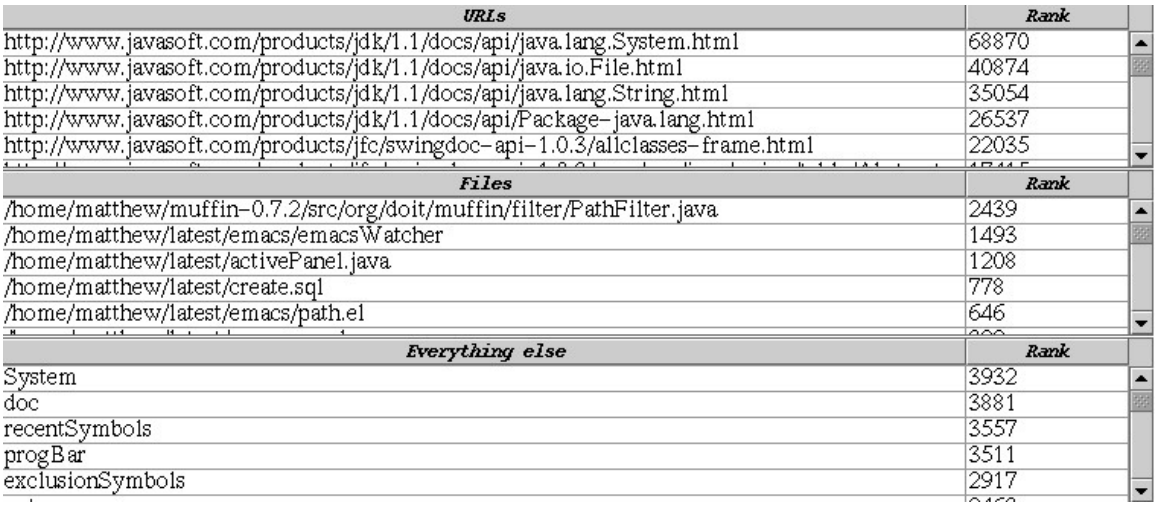
In the first case, if a symbol is not contextually specific then it is likely to occur within any sample period. Hence it is likely to occur in many past periods—but also in the last minutes or hours, and so it will be in the exclusion set. Second, if a symbol is very specific in its use and it occurs in the recent period, then it is in the exclusion set and is likely to occur only in similar periods to the recent period—at least under the definition of similarity being used here. This contextual specificity will be useful in finding good recommendations. Third, if it is specific to current activity but does not appear in the recent period then it is likely to be recommended, as one would wish.

We are just beginning to build up paths that include content, and so can only offer preliminary assessments. While programming, we have found that an exclusion period of ten or fifteen minutes tends to cut out most common Java keywords from the top of the recommendation list. When looking at textual web documents in particular, the 'stop symbols' tend to swell the set of apparently relevant periods of time used in collation, but the existence of topic–specific symbols raises the rankings of more useful files and URLs. Cumulative statistical effects in this way tend to push down documents that have less in common with recent activity. We have already noticed crossover between languages, such as when Java keywords such as *if* and *then* match with shell scripts and plain text. Another effect is that the sampling of editor activity may lead to partial words being logged e.g. *int* may be logged while typing the word *international.* If such overlaps and crossovers consistently and frequently occur, then we assume they are meaningful and deliberate. An example might be a programming textbook, which would often interleave code and text. If overlaps or crossovers are accidental, however, we assume they will be infrequent, and so gain low tallies.

4. Examples

In a first example, we discuss the set of recommendations shown in Figure 1, made without content logging and for the first author while he was resident in Zürich, Switzerland. We also refer to Figure 3.

This example is intended to demonstrate how recommendations, such as ski information given weather pages, might not be obviously useful until one considers the author's particular situation and the history behind it. If the mountain weather was good, he might ski. One checked the weather before getting details of piste conditions and cable car times. In this case, the author got the Klosters information from his own path, and the Arosa information from a colleague's path. Never having been to Arosa, or to the Arosa web site, the recommendation was therefore both novel and relevant. The example also offers a contrast with content–based tools whose likely recommendation, we suggest, would simply have been for yet more weather pages. Lastly, the recommendations include mixed data types: JPEG images as well as pages of HTML. This demonstrates the ability to handle media usually indexed and searched by disjoint systems.



URLs	Rank
http://www.javasoft.com/products/jdk/1.1/docs/api/java.lang.System.html	68870
http://www.javasoft.com/products/jdk/1.1/docs/api/java.io.File.html	40874
http://www.javasoft.com/products/jdk/1.1/docs/api/java.lang.String.html	35054
http://www.javasoft.com/products/jdk/1.1/docs/api/Package-java.lang.html	26537
http://www.javasoft.com/products/jfc/swingdoc-api-1.0.3/allclasses-frame.html	22035

Files	Rank
/home/matthew/muffin-0.7.2/src/org/doit/muffin/filter/PathFilter.java	2439
/home/matthew/latest/emacs/emacsWatcher	1493
/home/matthew/latest/activePanel.java	1208
/home/matthew/latest/create.sql	778
/home/matthew/latest/emacs/path.el	646

Everything else	Rank
System	3932
doc	3881
recentSymbols	3557
progBar	3511
exclusionSymbols	2917

Figure 4. Recommendations involving content are split into three lists: URLs, local files and everything else. Clicking on a URL or filename triggers display in the web browser or *xemacs*. While working on the recommender, relevant documentation and system components of varied types have been recommended, including HTML, Java, shell scripts, Lisp, SQL, and variable and class names.

As with [Chalmers98], co-occurrence statistics were used in the layout algorithm of [Chalmers96], and the results—such as that of Figure 3—browsed using the visualisation tool of [Brodbeck]. The intention is to show more of the context and structure which lies behind recommendation lists. Layouts involve a relatively small number of symbols, which tend to form 'strands.' Each strand is a sequence of symbols collected prior to symbol tallying i.e. one of the periods of past activity used in the recommendation process. Strands cross at multiply accessed symbols, and so symbols used at diverse periods of time, but adjacent to these recurrences, are brought near to each other. Such a visualisation is made in under one minute on a Silicon Graphics O2.

In Figure 4, we see the recommendations offered via the second logging mode while working with *xemacs* on the recommender itself, in particular on the file *recer.java.* Only one path was used. URL recommendations are for JDK and Swing user interface toolkit documentation read during previous editing. Files recommended for reuse are other components of the recommender system, and are of various types: Java files, a shell script, a file of SQL statements and an Emacs lisp file. The *Everything else* recommendations are for the class *System*, often used in *println* statements to output (copious) debugging information, and variable names used elsewhere in the *recer* class such as *doc* and *recentSymbols*. If several programmers' paths were combined in this way, we believe the current system would recommend to each user, in his or her current work, software components that were used by others in similar work but that the user did not know about. Our system would then help support software reuse within work groups.

5. The Value of Activity

The recording of paths mean that people are tracked and logged: where they go on the web, and where they go in their local file system. Paths are similar to mobile phones and credit cards in that individuals have their activity recorded and monitored, but in this section we will discuss differences with regard to the costs and benefits of such tracking.

Tracking web activity is commonplace. Commercial web sites record details of their visitors and customers, for example the machine address, the web browser used, and the URL previously accessed. Further details can be gained via cookies and registered logins. The visitor may gain specialised or restricted information, and the site gains valuable customer profiling and marketing information. However, this information is generally limited by the fact that such a site can gain only a narrow view of the activity of each visitor. Once a visitor leaves the site, they become invisible. The wider picture of the interests, preferences and activities of that visitor is unavailable to the site. This situation may become increasingly

complex in the near future, as web proxies and cryptographic techniques such as Anonymizer.com, the Janus ReWebber, and Lucent's ProxyMate become more widely used. (A recent issue of the Communications of the ACM was devoted to web privacy [Connor].) One issue important for this discussion is the way that these services and techniques can offer anonymity as well as individuality. One can use them to visit a site with an identity that is specific to that site and to you, and yet which masks your true identity and location. Therefore one can, for example, register with a site, have a login and password particular to that site, and yet remain anonymous. Such anonymity may change the balance of the costs and benefits of activity tracking. Visitor identity and activity are valuable to web sites, and visitors may now be able to demand more in return for that information.

Tracking in path systems is somewhat different. Tracking is done on the client side, which means that site to site transitions are unproblematic. The database that stores users' activity can be kept locally, and be more under the control of those users. Being local, technical and legal means of controlling how this database is used can be complemented with social, informal means. More sensitive information can be entered there, such as the names and content of local files, the names of people one communicates with, and records of where one moves. As we saw earlier in the example of Figure 4, this local information can improve the quality of recommendations because the recommender has a more complete view of user activity. Recommendations can be based not just on the activity on one site, but many sites. They can be based not just on web activity, but also on more local sources of information. If one wishes to obtain recommendations for useful information, one need not go to commercial sites. Friends and colleagues can get them from each other. This may also shift the balance away from commercial sites, who may need to offer their visitors more valuable information if they are to compete with locally derived information.

Ultimately, one must ask whether better or more tailored information is enough. If information is valuable, then it is worth money. If I can be anonymous, sites may pay me to tell them who I am. Since I could use different identities on different sites, they can pay me to tell them which other identities I use, and so let them build up cross–site activity profiles. These possibilities are reflections on the way that our clicks, choices, selections, purchases and other actions are made less transient and more public, and so our activity become a commodity. It becomes clearer than ever that everyday activity with computers and information is part of the world of value, money and economics.

6. Ongoing and Future Work

We are extending the recommender interface to allow the user to add to or remove symbols from the 'recent' set, allowing the modelling of information need to be as active or as passive as required. We have yet to link in the new content–based recommendation data to our visualisation tools, and we would like to support more sophisticated software browsing than simple loading of files in *xemacs*. We would also like to log the classes loaded in programs run by users, rather than just their editing. This is part of a wider goal of better representing user activity. Programs being run, mail being read, mobile computers changing spatial position—these and other traces of activity seem to be recordable and able to be interwoven in paths and recommendations. We also wish to display where recently used symbols matched in people's paths, to show their relative lengths and the pattern of hits, and hence to help in selecting paths to use in future.

As mentioned earlier, we have just begun building up experience with content logging. Two issues for future work here are handling the varied flow of symbols coming in from different logger programs, and the general issue of scalability. Web documents with many contained symbols tend to get high ratings, as we log not the symbols viewed by the user but all the symbols in the file. The two may not be the same, but we have so far avoided the obvious approach of scaling based on document length and giving weights to symbols. This would, we feel, only temporarily sidestep the deeper problem of more accurately tracking what the user really reads and writes. (And of course we recognise the distinct difference between this problem and that of assessing what the user thinks.) For experimental purposes one might use eye tracking, but we are focusing on further work inside *xemacs* and the web browser. Also, as logging continues, and as we extend the community of users of our prototype beyond its developers, the path database will grow. We will then be able to observe how the speed and character of recommendations change.

In terms of future applications of paths, we have begun work with one of our local museums, developing a system for visualising the exhibition spaces and the artifacts there, displaying information about selected artifacts, and recommending others based on recorded activity. In the long run, we would hope to expand

such recommendation to other galleries and museums in Glasgow, and on to the city itself. In the latter case we would recommend a mixture of museum artifacts, web information and city spaces. Another area of application is shopping and finance: if I recently bought X, Y and Z, in a particular street, what and where next?

In reflecting on the system, one becomes aware of the limited work on underlying theories of representation, design and use in systems such as this, involving varied media, varied tools, strong subjectivity and adaptation, and social interaction. We aim to build on early work such as [Chalmers99a] and [Chalmers99b] in exploring the informatics' links across disciplinary boundaries to topics such as economics, contemporary philosophy, semiology, and architecture. Our aim is informatics theory that links design and use in a manner that does justice to the complexity of the phenomena involved.

7. Conclusion

We have described a prototype information access system, emphasising the contextual and subjective character of information activity. Focusing on patterns of use rather than patterns in content lets us handle heterogeneous data. We deliberately blur the distinction between words, titles and URLs, and also between tools and between local and remote data, in order to experiment with a more holistic stance than is generally taken in IR. This can be seen as an attempt to work *with* the way that everyday activity continually moves and shifts amongst tools, amongst data types, and amongst information sources, rather than in spite of it. This early work can also serve as an example of a complementary approach to that of traditional IR, and hence as a means to highlight some of the assumptions that we sometimes take for granted.

Our 'metadata,' as in other recommender systems, are statistics of usage patterns within a set of identifiers. Path–based systems involve categorisation that is ephemeral, subjective and adaptive with the context of use. We aim to minimise abstractions made objectively or *a priori.* We shift our focus from the document to the identifier or symbol, which we still assume to be objective in that it is contextually independent, shared and persistent—even though it might refer to almost any type of object. It is this referential uniformity that gives recommender systems their flexibility with regard to heterogeneous data, affording application in many domains including software development and reuse. We also shift our focus from the notion of topical relevance that IR itself has recognised as being primitive, to a notion of utility based on patterns of activity. We hope that this chapter helps to raise our level of discourse and design more utile systems, but we also must recognise that we have barely begun to understand how to handle information's symbolic and economic significance.

7. References

[Baeza-Yates] Baeza-Yates, R. & B. Ribiero-Neto, *Modern Information Retrieval*, Addison Wesley, 1999.

[Bellotti] Bellotti, V. & Sellen, A. Design for Privacy in Ubiquitous Computing Environments. In: G. de Michelis, C. Simone and K. Schmidt (Eds.) *Proc. Third European Conference on Computer Supported Cooperative Work, (ECSCW'93),* pp. 77-92. Kluwer, 1993.

[Berners-Lee] Berners-Lee, T. World-Wide Computer. Comm. ACM, 40(2):57-58, February 1997.

[Brodbeck] Brodbeck, D., Chalmers, M., Lunzer, A. & Cotture, P. Domesticating Bead: Adapting an Information Visualization System to a Financial Institution, *Proc. IEEE Information Visualization 97,* Phoenix, Oct. 1997, 73–80.

[Bush] Bush, V. As We May Think, *Atlantic Monthly*, July 1945. Reprinted in *ACM Interactions* 3(2), March 1996, 37–46.

[Chalmers92] Chalmers, M. Chitson, P. Bead: Explorations in Information Visualisation, *Proc. ACM SIGIR'92,* Copenhagen, published as a special issue of *SIGIR Forum*, ACM Press, pp. 330–337, June 1992.

[Chalmers96] Chalmers, M. A Linear Iteration Time Layout Algorithm for Visualising High–Dimensional Data. *Proc. IEEE Visualization 96,* Oct.–Nov. 1996, 127–132.

[Chalmers98] Chalmers, M., Rodden, K. & Brodbeck, D., The Order of Things: Activity-Centred Information Access, *Proc. 7th Intl. World Wide Web Conf. (WWW7),* Brisbane, 1998, 359–367.

[Chalmers99a] Chalmers, M. Informatics, Architecture and Language, to appear in *Social Navigation: Footprints in the Snow*, Munro, A., Hook, K. & Benyon, D. (eds.), Springer, 1999.

[Chalmers99b] Chalmers, M. Comparing Information Access Approaches, to appear in *J. ASIS*.

[Connor] Connor, L. (ed.) *Communications of the ACM*, 42(2), Special Issue on Internet Privacy, Feb. 1999.

[Cooper] Cooper, W. On Selecting a Measure of Retrieval Effectiveness. *J. ASIS*, 24:87-100, 1973.

[Goldberg] Goldberg, D. et al. Using Collaborative Filtering to Weave an Information Tapestry, *Comm. ACM 35(12)*, December 1992, 61–70.

[Korfhage] Korfhage, R., To See or Not to See: Is That the Query? *Proc. ACM SIGIR*, Chicago, 1991, 134–141.

[Lin] Lin, X., D. Soergel & G. Marchionini., A Self–Organising Semantic Map for Information Retrieval, *Proc. ACM SIGIR*, Chicago, 1991, 262–269.

[MacEachren] MacEachren, A. *How Maps Work: Representation, Visualization and Design*, Guilford Press, 1995.

[Pearson] Pearson, P. Fast Hashing of Variable–Length Text Strings, *Comm. ACM*, 33(6):677–680, June 1990.

[Resnick96] Resnick P, Miller J. PICS: Internet Access Controls without Censorship. Comm. ACM, 39(10):87-93, October 1996.

[Resnick97] Resnick, P. & Varian, H. (eds.) Special Issue of *Comm. ACM* on Recommender Systems, 40(3), March 1997.

[Salton] Salton, G., *Automatic Text Processing*, Addison Wesley, 1989.

[Saracevic] Saracevic, T. The Concept of "Relevance" in Information Science: A Historical Review. In: *Introduction to Information Science*, 111-151, Bowker, New York, 1970.

[Schamber] Schamber, L., Eisenberg, M. & Nilan, M. A. Re-examination of Relevance: Towards a Dynamic, Situational Definition. *Information Processing & Management*, 26:755-776, 1990.

[Wang] Wang, P. The Design of Document Retrieval Systems for Academic Users: Implications of Studies on Users' Relevance Criteria, *Proc. ASIS 97*, 162-173, 1997.