Web Interaction and the Navigation Problem in Hypertext written for Encyclopedia of Microcomputers

Mark Levene
University College London
Gower Street
London WC1E 6BT, U.K.
email: mlevene@cs.ucl.ac.uk

George Loizou
Birkbeck College
Malet Street
London WC1E 7HX, U.K.
email: george@dcs.bbk.ac.uk

Abstract

The web has become a ubiquitous tool, used in day-to-day work, to find information and conduct business, and it is revolutionising the role and availability of information. One of the problems encountered in web interaction, which is still unsolved, is the *navigation problem*, whereby users can "get lost in hyperspace", meaning that when following a sequence of links, i.e. a *trail* of information, users tend to become disoriented in terms of the goal of their original query and the relevance to the query of the information they are currently browsing.

Herein we build statistical foundations for tackling the navigation problem based on a formal model of the web in terms of a probabilistic automaton, which can also be viewed as a finite ergodic Markov chain. In our model of the web the probabilities attached to state transitions have two interpretations, namely, they can denote the proportion of times a user followed a link, and alternatively they can denote the expected utility of following a link. Using this approach we have developed two techniques for constructing a web view based on the two interpretations of the probabilities of links, where a *web view* is a collection of relevant trails. The first method we describe is concerned with finding frequent user behaviour patterns. A collection of trails is taken as input and an ergodic Markov chain is produced as output with the probabilities of transitions corresponding to the frequency the user traversed the associated links. The second method we describe is a reinforcement learning algorithm that attaches higher probabilities to links whose expected trail relevance is higher. The user's home page and a query are taken as input and an ergodic Markov chain is produced as output with the probabilities of transitions giving the expected utility of following their associated links. Finally, we characterise typical user navigation sessions in terms of the entropy of the underlying ergodic Markov chain.

1 Introduction

The World-Wide-Web (known as the web) has become a ubiquitous tool, used in day-to-day work, to find information and conduct business, and it is revolutionising the role and availability of information. (Currently the web contains over a billion web pages; see [LG99] for a recent analysis of the amount of information on the web and its distribution.) Although current search engines have access to large off-line databases that are frequently updated with new online data, they are still deficient in narrowing down the list of "hits" to a manageable number and in ranking the results in a meaningful way by using contextual knowledge. In

addition, search engines do not address the problems encountered during *navigation* (colloquially known as "surfing") which often lead users to "getting lost in hyperspace" meaning that when following links users tend to become disoriented in terms of the goal of their original query and the relevance to the query of the information they are currently browsing; we refer to this problem as the *navigation problem*. Moreover, current search technology does not make adequate use of past knowledge about the individual user who is using the system or of past experience gained by the group of users he/she belongs to; such knowledge can be used to adapt the system to the user's goal.

Searching for information on the web will normally proceed through the following steps:

1) *Input Query* − the user inputs a query which specifies the user's goal.

2) *Information Retrieval* − the system invokes a search engine which provides the user with a ranked list of pages according to their relevance to the user's query.

3) *Navigation* − the user *repeats* the following two tasks:

   (a) The user chooses a page to *browse*.

   (b) The user follows a *link* which is embedded in the page he/she is browsing.

4) *Query Modification* − the user *returns* to (1) to refine and/or modify the original goal.

Hereafter we concentrate on the navigation step which is not adequately dealt with in current browser technology. In particular, we summarise our recent work which addresses the navigation problem in the context of a probabilistic model of the web, and survey work related to the navigation problem. The term *web interaction*, which we take to mean the combination of contextual search and navigation, is central to our approach. Within a navigation session contextual information may include: domain knowledge with respect to the user's goal, other web pages visited during the session and knowledge about the user and his/her preferences. Herein we will assume that the user's goal is specified via a query and that our knowledge about the user is summarised by a collection of past navigation sessions.

We now introduce the main points of our formalism via an example. Consider the web topology shown in Figure 1, where each node is annotated with its URL, U_i, which is the unique address of the page P_i represented by the node (see Section 2). In addition to the URL U_i each node contains the score which is a measure of the relevance of the page P_i to the input query.

A *trail* of information through the database graph consists of a sequence of pages visited by the user in a navigation session. For example, with respect to Figure 1 four possible user trails starting from P_1 are:

1) P_1, P_2,

2) P_1, P_3, P_4, P_1, P_2,

3) P_1, P_3, P_5, P_6, P_1 and

4) $P_1, P_3, P_5, P_6, P_3, P_4$.

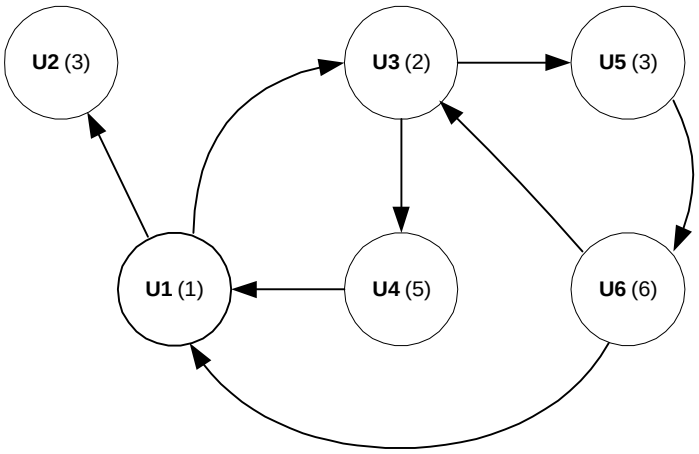

Figure 1: An example web topology

In our formal model we view the web as a finite automaton called a *Hypertext Finite Automaton* (HFA), whose states are web pages and transitions are links. In a HFA all states can be initial and final, since we allow navigation to start and finish at any page. Taking this view the words accepted by a HFA are the possible trails through the web. We enrich the semantics of HFA by attaching probabilities to state transitions resulting in *Hypertext Probabilistic Automata* (HPA) [LL99c]. The transition probabilities can have two interpretations in our model. Firstly they can denote the proportion of times a user followed a link, and secondly they can denote the expected utility of following a link. We further develop the notion of HPA by viewing them as finite *ergodic Markov chains* [KS60]. In order to realise this view we consider the user's home page as an artificial starting point for all navigation sessions and assume that there is a positive probability (however small) of jumping to any other relevant web page. We thus would modify Figure 1 by adding the user's home page and links from it to all other pages. The probabilities of these links are the initial probabilities of the Markov chain. Moreover, we assume that the user is following links according to the transition probabilities and when completing a navigation session returns to his home page. Thus we would further modify Figure 1 by adding links from its pages to the artificial home page. The probabilities attached to these links denote the probability of concluding the navigation session after visiting a particular web page. The resulting HPA can be seen to be an ergodic Markov chain.

A *web view* is a collection of trails which are either the result of user navigation over a period of time, or are relevant trails with respect to the user's input query. We have developed two techniques for constructing a web view based on the two interpretations of the probabilities of links. Our first technique for constructing a web view is within the area of *web data mining*, which is concerned with finding frequent user behaviour patterns. A collection of trails is taken as input and an ergodic Markov chain is produced as output with the probabilities of transitions corresponding to the frequency the user traversed the associated links. For example, with respect to the four trails from Figure 1, corresponding to four navigation sessions, the probability of going from P_1 to P_3 is $3/5$, the probability of going from P_1 to P_2 is $2/5$ and the probability of going from P_3 to P_4 or to P_5 is $1/2$. In constructing the Markov chain we can also take into account the situation when the user decides to traverse a link based on the previous pages he/she has visited during the navigation

session.

Our second technique for constructing a web view is a reinforcement learning algorithm that attaches higher probabilities to links whose expected trail relevance is higher. The user's home page and a query are taken as input and an ergodic Markov chain is produced as output with the probabilities of transitions giving the expected utility of following their associated links. The relevance of an individual trail is calculated as the average of the relevances of the pages in the trail. For example, with respect to the four trails from Figure 1, the average relevances of the first, second, third and fourth trails are, respectively, 2.00, 2.40, 2.60 and 3.17. So, in this case the probability of going from P_1 to P_3 is again higher than the probability of going from P_1 to P_2 (since following the former link results in trails with higher on average relevance), and similarly the probability of going from P_3 to P_5 is higher than the probability of going from P_3 to P_4.

The rest of the paper is organised as follows. In Section 2 we give an overview of hypertext and introduce the navigation problem. In Subsection 2.1 we review the history of hypertext, and in Subsection 2.2 we present our formal model of hypertext where initially we view a hypertext database as a finite automaton and then extend this view to that of a probabilistic finite automaton. In Section 3 we review work on the navigation problem. In Section 4 we introduce the notion of a *web view* which is a subgraph induced by a collection of trails within the topology of the hypertext database. In Subsection 4.1 we detail our first technique for constructing a web view, which is in the area of web data mining, whose transition probabilities correspond to our first interpretation in terms of the proportion of times a user followed a link. In Subsection 4.2 we detail our second technique for constructing a web view, based on a reinforcement learning algorithm, whose transition probabilities correspond to our second interpretation in terms of the expected utility of following a link. In Section 5 we utilise our view of a hypertext database as an ergodic Markov chain by characterising typical user navigation sessions in terms of the entropy of the Markov chain. In Section 6 we review related work. Finally, in Section 7 we discuss the potential impact of web interaction.

2 Hypertext as an Underlying Model for Web Navigation

The foundations of the web are rooted in the area of *hypertext* [Nie90], which breaks from the traditional organisation of text as a linear sequence of words dictating to the reader the order in which the text should be read; we often refer to the reader of a hypertext as the user. Hypertext organises documents in a nonsequential (or nonlinear) order. It presents the reader with several different options of reading a document, the choice of how to read the document being made at the time of reading. Let us call a textual unit of information a *page*. A *hypertext database* consists of a set of pages which are *linked* together according to the authors' specifications, i.e. a hypertext database is a directed graph (digraph), where the nodes are the pages and the arcs are the links. Every link connects two nodes, the starting node which is called the *anchor* node (or simply the anchor) and the finishing node which is called the *destination* node (or simply the destination).

Links can either be *hard* (equivalently *static*) or *soft* (equivalently *dynamic*); cf. DeRose's taxonomy of links [DeR89]. A hard link is one, which given the anchor node, explicitly specifies the address of the destination node. A soft link is one, which given the anchor node, implicitly specifies the address(es) of the destination node(s) via a *script* that computes the set of destination nodes at the time the link is followed. The advantage of soft links is that

the addresses of the destination nodes are not fixed and thus the *dangling* link problem is avoided. (A dangling link is one which is referencing a page at a nonexistent address.)

Creating hypertext can be viewed as a dynamic process whereby readers can also take on the role of authors by adding their own pages and links to the database. The example of the web is very instructive in this case, since it can be viewed as a continuously evolving hypertext database.

The web is undoubtedly the largest hypertext database available providing readers with an almost unlimited source of data. Without going into any detail, each unit of information on the web is known as a resource, and each resource has a unique identifier describing where the resource resides and how to retrieve it. (The mechanism used is that of a *Unified Resource Locator*, or simply URL, which specifies the type of resource and a unique path for locating it.) Every web user has a *home page*, which is a hypertext page authored by the user, providing information and links created by the user. Thus the home page essentially connects the information provided by the user to the larger body of information available on the web, via the links that can be followed from the home page. Any other user *visiting* this home page can also follow these links.

Authors of web pages can create document pages using the *Hypertext Markup Language* (HTML) [Aro94]. HTML provides the facilities for formatting a document, including images in documents, linking a document to other documents and interacting with user input through forms. A recent proposal of a markup language, which supersedes HTML, is the *Extensible Markup Language* (XML) [MFDG98]. XML is a metalanguage that allows you to design your own document types, with their individual structure, as opposed to HTML, which is a regular markup language in the sense that it defines a specific way of describing the information content of document pages. In particular, HTML caters for the report style document type with headings, paragraphs, lists and the like, with some provision for hypertext and hypermedia. XML allows you to customise the information according to the application, in order to cater for many classes of documents where the markup is descriptive and the tags are more informative than just for formatting purposes as in HTML. (There is a large amount of online information on the web concerning XML and its recommended standard.)

Apart from querying the database users are most often browsing through pages of the hypertext database while traversing links. This process of following a *trail* of information in a hypertext database is called *navigation* (or alternatively *link following*). During the navigation process users may become "lost in hyperspace", meaning that they become disoriented in terms of what to do next and how to return to a previously browsed page. This is one of the main unsolved problems confronting hypertext, which is known as the *navigation problem*. It is the problem of having to know where you are in the database digraph representing the structure of a hypertext database, and knowing how to get to some other place you are searching for in the database digraph. In other words, readers may lose the context in which they are browsing and need assistance in finding their way. We discuss ways of tackling the navigation problem in Section 3.

2.1 Historical Review

The inspiration for hypertext comes from the *memex* machine proposed by Bush [Bus45] (see [NK91] for a collection of essays on Bush and his memex). The memex is a "sort of mechanized private file and library" which supports "associative indexing" and allows navigation whereby

"any item may be caused at will to select immediately and automatically another". Bush emphasises that "the process of tying two items together is an important thing". By repeating this process of creating links we can form a *trail* which can be traversed by the user, in Bush's words "when numerous items have been thus joined together to form a trail they can be reviewed in turn".

Bush also envisaged the "new profession of trailblazers" who create new trails for other memex users, which allows for sharing and exchange of knowledge. Trigg [Tri91] emphasises that Bush views the activities of creating a new trail and following a trail as being connected. Trails can be authored by trailblazers based on their experience and can also be authored by memex which records all user navigation sessions.

Many years after the publication of his original 1945 paper, Bush revisited and extended the memex concept in [Bus91a] written in 1959 and in [Bus91b] written in 1965. In particular, he envisaged that memex could "learn form its own experience" and "refine its trails". By this Bush means that memex collects statistics on the trails that the user follows and "notices" the ones which are most frequently followed. Oren [Ore91] calls this extended version *adaptive memex*, stressing that adaptation means that trails can be constructed dynamically and given semantic justification; for example, by giving these new trails meaningful names.

The term "hypertext" was coined by Ted Nelson in 1965 (see [Nel80]), who considers "a literature" (such as the scientific literature) to be a *system of interconnected writings*. The process of referring to other connected writings, when reading an article or a document, is that of *following links*. Nelson's vision is that of creating a repository of all the documents that have ever been written and thus achieving a universal hypertext database. Nelson views his hypertext system, which he calls Xanadu, as a network of distributed documents that should be allowed to grow without any size limit and such that users, each corresponding to a node in the network, may link their documents to any other documents in the network. Xanadu can be viewed as a generalised memex system, which is both for private and public use. Nelson's vision of hypertext is in fact materialised to a large degree in the web, since he also views his system as a means of publishing material by making it universally available to a wide network of interconnected users.

2.2 Formal Model of Hypertext

As stated at the beginning of Section 2 a hypertext database is a digraph whose nodes are the pages and arcs are the links. We give semantics to a hypertext database in terms of a class of finite automata [HU79], which we call *Hypertext Finite Automata* (HFA). The alphabet of the HFA is in a one-to-one correspondence with the page set of the hypertext database, and to each state of the HFA there corresponds a single page. We will assume for now that the state set of the HFA and the page set of the hypertext database are also in a one-to-one correspondence. In addition, all the states of the HFA are both initial and final, due to the fact that we can start our navigation at any page and finish at any page. The state transitions of the HFA occur according to the links of the digraph of the hypertext database, namely the state transition from state s_i to state s_j, labelled by symbol (page) P_i, is given by

$$s_i \xrightarrow{P_i} s_j$$

and corresponds to a link from page P_i to page P_j. Our interpretation of this state transition is that a user browsing P_i decides to follow the link leading to page P_j. At the end of the

navigation session, after some further state transitions, the user will be in state, say s_k, browsing page P_k.

A word that is accepted by a HFA, which we call a *trail* of the HFA, is a sequence of pages

$$P_1, P_2, \ldots, P_n$$

which were browsed during a navigation session, starting at page P_1, then following links according to the state transitions of the HFA and ending at page P_n. The language accepted by a HFA is the set of trails of the HFA. In other words, the language accepted by a HFA is the set of all possible trails a user could follow, which are consistent with the topology of the hypertext database.

Let xy denote the concatenation of the words x and y. Then a word y is a *subword* of a word w if $w = xyz$ for some words x and z, and a word w is the *join* of words xy and yz if $w = xyz$ and y is not the empty word.

In [LL99c] we provide a characterisation of the set of languages accepted by a HFA, as the subset of regular languages closed under the operations of subwords and join. This result is intuitive in terms of web navigation since subwords correspond to subtrails, and the join of two words corresponds to the join of two navigation trails, where the second trail completes the first one.

We now formulate simple queries over HFA as follows. A *trail query* (or simply a query) is an expression of the form

$$k_1 \text{ AND } k_2 \text{ AND } \ldots \text{ AND } k_n,$$

where the k_i are keywords.

A trail, T, which is accepted by a HFA *satisfies* a trail query if for all the k_i there is a page P_j in T such that k_i is a keyword of P_j. (We omit to further specify the notion of a *keyword* and refer the reader to [BR99] which discusses how keywords can be extracted from a page of text; see also [GRGK97].)

In [LL99b] we show that checking whether a HFA accepts a trail satisfying a trail query is NP-complete. The proof of this result utilises a duality between *propositional linear temporal logic* [Eme90] and a subclass of finite automata. In temporal logic terminology the condition that k_i is a keyword of page P_j is the assertion that "sometimes" k_i, viewed as a condition on P_j, is true. Therein we also defined a more general class of queries which supports the additional temporal operators "nexttime" and "finaltime", and more general Boolean conditions. In the context of hypertext the natural interpretation of "time" is "position" within a given trail. So, "sometimes" refers to a page at some position in the trail, "nexttime" refers to the page at the next position in the trail, and "finaltime" refers to the page at the last position in the trail.

In [LL99b] we have shown that only for restricted subclasses of queries is the problem of checking, whether a HFA accepts a trail satisfying a query, polynomial-time solvable. Such a subclass essentially prescribes a one-step at a time navigation session using the "nexttime" operator. Current navigation practice where links are followed one at a time conforms to this subclass. These time-complexity results have led us to investigate a probabilistic approach to navigation in hypertext by adding probabilities (or equivalently weights) to the automaton's state transitions (or equivalently links), resulting in *Hypertext Probabilistic Automata* (HPA).

We interpret the probabilities attached to links in two separate ways:

1) The HPA models a user's (or group of users) navigation behaviour patterns, and the transition probability denotes the proportion of times that the user (or group of users) followed the link from its anchor node.

2) Given a query the HPA models the expected trail relevance, and the transition probability denotes the expected utility of following the link.

We further develop the notion of HPA by viewing them as finite *ergodic Markov chains* [KS60]. We consider the user's home page as an artificial starting state of any navigation session and assume there is a positive probability (however small) of jumping to any other relevant web page. These probabilities can be viewed as the initial probabilities of the Markov chain. The user then follows links according to the topology of the web and the transition probabilities, eventually returning to his/her home page at the end of the navigation session. The probability of a trail T, denoted by $p(T)$, is thus defined as the product of the initial probability of the first page of the trail together with the transition probabilities of the links in the trail.

We utilise this Markov chain model in Section 4, where we introduce web views as a set of trails forming a subgraph of the web topology, and in Section 5, where we investigate the entropy of user navigation.

3 A Review of the Navigation Problem in Hypertext

We have already introduced the navigation problem in Section 2. It is the problem of disorientation during the process of link following when a reader loses track of the context and is unsure how to proceed in terms of satisfying his/her original goal. Herein we review previous work on the navigation problem, which has mostly been concerned with the construction of navigational aids based on the topology of the hypertext database digraph.

The *browser* is the component of a hypertext system that helps users search for and inspect the information they are interested in by graphically displaying the relevant parts of the hypertext database and by providing contextual and spatial cues with the use of *orientation tools*. A simple orientation tool is the *link marker* which acts as a signpost to tell the user what links can be immediately followed and what links have just been traversed. Another useful orientation tool is the *book mark*, allowing readers to mark a page to which they can return to on demand when feeling lost [Ber88]. Readers may also mark pages which were already visited in order to avoid repetition; such marks are called *bread crumbs* [Ber88].

Maps, *webs* and *overview diagrams* give readers a more global context by displaying to them links which are at a further distance than just one link from the current position. Moreover, by highlighting the reader's history the reader can *backtrack* his/her steps to a previously browsed page. Maps can be displayed using a *fisheye-view* that selects information according to its *degree-of-interest*, which decreases as the page under consideration is further away from the currently browsed page [Fur86, TD92]. A set of tools that aid the construction of maps by performing a structural analysis of the database digraph is described in [RBS94]. One such tool is a hierarchical structure that can be imposed on the hypertext database where its root is chosen to be a central node whose distance to other nodes is relatively small. Another tool creates semantic clusters within the database digraph by identifying strongly

connected components of the digraph. A more sophisticated orientation tool is the *guided-tour* which actively guides users through the database digraph by suggesting interesting trails that users can follow [MI89].

In [Zel89] it is argued that trails should be first-class citizens in a hypertext system. Three types of trail are identified: (i) a *sequential* trail, which conforms to the syntactic definition of a trail as a sequence of nodes, (ii) a *branching* trail, which is a trail formed by the reader making a choice at various nodes when several branches may be followed, and (iii) a *conditional* trail, which is a trail formed by the system making a choice at various nodes according to some pre-specified test. Zellweger [Zel89] also proposes mechanisms by which these types of trail can be supported within a hypertext system by providing authoring facilities, path visualisation tools and playback control allowing the user to trace the nodes in a trail.

In a recent paper Bernstein [Ber98] described a variety of link patterns that can be found in hypertext, which are of larger granularity than trails. These link patterns include cycles, contours (each of which is an aggregation of several cycles), neighbourhoods (each of which contains a subgraph of associated nodes) and sieves (each of which contains several choices for the reader to follow). Identifying such patterns can help us understand the hypertext within a richer vocabulary of its structure.

In [Van89] navigational strategies are suggested as a means of tackling the navigation problem with the aim of reaching a target node. In particular, five strategies are identified, which may be associated with specific graph topologies such as a linear path, a cycle or a hierarchy: (i) the *identifier* strategy, which associates a unique identifier with each node in the database digraph allowing the user to recognise the target node, (ii) the *path* strategy, which gives the user explicit instructions at each node of how to get to the target node, (iii) the *direction* strategy, which gives the user more general instructions than those pertaining to the path strategy indicating the direction of the target node, (iv) the *distance* strategy, which utilises a distance metric between nodes to indicate how far the target node is, and (v) the *address* strategy, which refines the direction strategy by establishing coordinates describing the location of the target node.

In [Fur97] issues of navigation in web topologies are explored in terms of a *viewing graph* which is a small subgraph of the hypertext structure in which the user is currently navigating. Navigability is the property of being able to find the shortest path to a target node from the node currently being browsed by making decisions based solely on local information visible at the current node. This implies that at each node in the viewing graph sufficient information must be available to guide the user to the correct target node via the shortest route. Moreover, the information available at each node must be compact. Under this definition of navigability, navigation on the web is, in general, not effective, due to the fact that local information at nodes is limited. Ways of improving navigation on the web include: organisation of information into classification hierarchies and the ability to make local decisions through similarity-based measures between nodes of close proximity.

In [CRS85] users' navigational behaviour is characterised by measuring various indices. Essentially Canter et al. [CRS85] propose several types of trail: (i) a *path*, which is defined as a trail without recurring nodes, (ii) a *loop*, which is defined as a cycle where the only recurring node is the start node, (iii) a *ring*, which is defined as a trail that eventually returns to the start node with potentially many recurring nodes and (iv) a *spike*, which is defined as a trail with a forward "journey" and a return "journey" that retraces the path taken on the forward

"journey". Two metrics are defined: (i) the ratio NV/NT of the number of different nodes visited and the total number of nodes in the hypertext system (this measure is irrelevant in the context of the web unless we can restrict ourselves to some meaningful subset of its pages) and (ii) the ratio NV/NS of the number of different nodes visited and the total number of nodes visited, which includes multiple visits. These measures are utilised by Canter et al. [CRS85] to characterise the differences between various navigational strategies. For example, goal oriented *browsing* may be characterised by several large rings composed of many long loops and a medium NV/NS, *searching* may be characterised by ever increasing spikes with few loops in order to find a target node and a low NV/NS, and *wandering* (or random surfing) may be characterised by medium sized rings, where the user revisits nodes in an unstructured "journey", and a low NV/NS.

In [KH95] the activity of user navigation is compared to the activity of wayfinding through a physical space. Both activities include user tasks such as being aware of one's current location, planning a route to follow and executing the plan. Research into wayfinding in physical spaces is based upon the assumption of the existence of cognitive maps encoding the user's knowledge about the space he/she is navigating through. Such spatial knowledge can be classified into the representations of: place, route and survey knowledge, which concerns the spatial layout of the salient places. Various tools to help solve the disorientation problem have been developed which are inspired by the spatial metaphor. These include: differentiation of regions, maps, guided-tours, definition of prominent nodes, called *landmark nodes*, fisheye-views, history lists, history trees and summary boxes.

A recent activity that may be of assistance in tackling the navigation problem is that of mapping the web, known as *mapping cyberspace* [JO99]. For this purpose a spatial model of the web must be formulated. Jiang and Ormeling [JO99] distinguish between three views of cyberspace. In the first view the web is taken to be a physical network that can be mapped according to the physical location of its nodes. The induced cybermaps are thus superimposed on a map of the world, depicting for example, the distribution of hosts, the volume of internet traffic and internet growth. In the second view the web is taken to be a digraph, as in our mathematical model of the web, where a link provides the unit of distance. The induced cybermaps are thus graphs depicting, for example, trails between nodes indicating the growth of the web and connectivity patterns which give rise to social networks. In the third view the web is taken to be a virtual world in which humans interact. In such a cyberspace various activities evolve such as online shops, gaming networks, and subspaces where we can meet and chat to other people. In such an environment cybermaps are needed for navigation and orientation purposes.

4 Web Views

We define a *web view* as a collection of trails which are either the result of user navigation sessions over a period of time, or are relevant trails that satisfy a user's trail query. Thus a web view is a subgraph of the digraph of the hypertext database induced by a collection of trails.

We limit the trails in a web view by two threshold parameters, as follows:

1) *support* $\alpha \in [0, 1)$; accept into the web view only trails whose initial probability is greater than α.

2) *confidence* $\beta \in [0,1)$; accept into the web view only trails whose product of transition probabilities is greater than β.

Alternatively, we accept into the web view only trails whose overall probability is above some *cut-point* $\lambda \in [0,1)$, with $\lambda \geq \alpha \cdot \beta$.

Let $\mathcal{M}$ be an ergodic Markov chain modelling the semantics of the hypertext under consideration, in our case modelling the web. Then a web view over $\mathcal{M}$ constrained by λ is the set of all trails T in $\mathcal{M}$ such that $p(T) > \lambda$. (An alternative formalisation of a web view using the support and confidence thresholds can also be given.)

In the next two subsections we will describe two different techniques for constructing web views based on our two interpretations of the transition probabilities of the Markov chain $\mathcal{M}$.

4.1 Data Mining of User Navigation Patterns

Our first technique for constructing a web view is within the area of *web data mining*, which is concerned with finding frequent user behaviour patterns. In $\mathcal{M}$ the high probability trails, i.e. those having probability above the cut-point, correspond to the user's preferred trails.

We assume that we have at our disposal web log data; for example, collected by the user's browser, from which it is possible to infer user navigation sessions. It is customary to define a navigation session as a sequence of page visits (i.e. URL requests) by the user where no two consecutive visits are separated by more than a prescribed amount of time, which is normally not more than half an hour.

When sufficient such log data is available we pre-process this data into a collection of trails, each trail being represented as a sequence of URLs. Moreover, we assume that the start and end URL of all trails correspond to the user's home page. We note that a trail may appear more than once in this collection, since the user may follow the same trail on two or more different occasions. We then build an ergodic Markov chain (or equivalently HPA), say $\mathcal{M}$, whose initial probabilities correspond to the frequency the user visited a page present in any one of the input trails, and whose transition probabilities correspond to the frequency that a link was followed in any one of the input trails. We observe that the states of $\mathcal{M}$ are the pages the user visited and the topology of $\mathcal{M}$, i.e. its underlying digraph, is induced by the links the user followed. In constructing $\mathcal{M}$ we have implicitly assumed that when the user chooses a link to follow he/she does not base his/her decision on the previous pages visited during the navigation session. That is, we have assumed that $\mathcal{M}$ is a first-order Markov chain. This assumption can be relaxed so that N (with $N \geq 1$) previous pages including the current one are taken into account; the case with $N = 1$ is the first-order case when the user bases his/her decision only on the page currently being browsed. The parameter N is called the *history depth*.

Given a history depth $N > 1$, a higher-order Markov chain can be reduced to a first-order Markov chain by aggregating states (see [Bil61]). The drawback of such a higher-order Markov chain is the increase in the number of states, which is expected to be $n \cdot b^{(N-1)}$, where n is the number of states in the first-order Markov chain and b is the average number of out-links embedded in a page. Thus there is a trade-off between the history depth and the complexity of the Markov chain measured by its number of states. The decision on whether the gain in accuracy by adopting a higher-order Markov chain is significant can be aided by statistical techniques [Cha73, MGZ89].

Once the HPA $\mathcal{M}$ has been constructed from the collection of trails, which have been pre-processed from the log data, we employ a Depth-First Search (DFS) to find all the trails in $\mathcal{M}$ starting from the user's home page and having probability above the cut-point λ. We have run extensive experiments with synthetic and real data to test the performance of the DFS algorithm [BL98, BL01]. It transpires that for a given cut-point there is a strong linear correlation between the size of $\mathcal{M}$, measured by its number of states, and the running time of the algorithm, measured by the number of links it traverses. Moreover, for a given cut-point, the number of mined trails increases linearly with the size of $\mathcal{M}$. On the other hand, the number of mined trails increases exponentially with the decrease in the cut-point.

The DFS algorithm has two main drawbacks. Firstly, since it is an exhaustive search it will, in general, return too many trails. Secondly, if we increase the cut-point to reduce the number of trails, then on average the returned trails are short, and therefore may not be very interesting.

These observations have led us to develop two heuristics for mining high quality trails [BL99]. Our first approach, which we call the *fine-grained* heuristic, limits the numbers of trails returned via a stopping parameter, which is between zero and one, that determines an upper bound on the sum of probabilities of the returned trails. The method used to implement this heuristic is to explore trails whose probability is above the cut-point, one by one and in decreasing order of probability. When the stopping parameter is zero then the fine-grained heuristic reduces to the DFS algorithm and as the parameter gets closer to one less trails are returned. Our initial results show that for a given cut-point the number of trails decreases almost linearly with the increase in the stopping parameter, indicating that the stopping parameter provides good control over the number of trails. Our second approach, which we call the *inverse-fisheye* heuristic, is a method of obtaining longer trails while controlling their number. This is obtained by having a dynamic cut-point which is high at the initial stage of the exploration in order to limit the number of trails, and decreases in subsequent stages in order to allow further exploration of the selected trails. The user specifies a maximum exploration depth, which limits the length of the trails returned. Our initial results show that if the initial cut-point is not too low and the decrease in the cut-point at each step is gradual then we can reduce the number of trails while increasing their average length.

Previously Chen et al. [CPY98] have proposed a method to convert log data into maximal forward references which can then be processed by association rule techniques for discovering rules between data items in large databases [AMS$^+$96]. A different approach has been put forward by Schechter et al. [SKS98] with the aim of using web log data to predict the next URL to be requested from a user. Their algorithm essentially constructs a *suffix tree* [Apo85] generated from user paths within the log data, where a suffix of a path is added to the tree only if the occurrence count of the first node in the path is greater than a predefined threshold. Then an algorithm is devised which finds the maximal prefix of a path in the said tree that matches the current user path, and then uses this found path to predict the next URL. Techniques for capturing the user navigation profile based on clustering similar user paths were proposed by Shahabi et al. [SZAS97] using K-means clustering and by Ridgeway and Altschuler [RA98] using a clustered discrete Markov chain model.

4.2 Automated Navigation from User Queries

Herein we view the specification of the goal of a navigation session in terms of a query, which normally would be a set of keywords. We also assume as before that the navigation session starts from a fixed web page, say the user's home page. Starting from the home page we are interested in constructing a web view of trails which are highly relevant to the query. To this end we construct a HPA whose link probabilities represent the expected relevance of a trail resulting from following those links. The relevance of a trail is calculated as the average of the relevances of the pages in the trail, where the relevance of an individual page in a trail is the score of the page with respect to the input query. We note that we may compute the relevance of a trail by functions other than the average, for example by eliminating or penalising duplicate pages in a trail, or by applying a discount factor to pages which are further away from the start page.

The method used to construct this web view is based on a *sample-credit-update* loop, a common concept in reinforcement learning [SB98]. The generic algorithm is composed of the following three steps [ZL99]:

1) Starting from the user's home page a sample of trails is taken according to the topology of the web and the link probabilities of the HPA. (Initially the link probabilities are uniform random, i.e. the probability of choosing one link out of m out-links is $1/m$. We call a HPA with such uniform link probabilities a *random* HPA.)

2) Links are credited according to the relevance of the trails passing through these links and the probabilities of links are normalised.

3) The web view is updated according to a learning rate which is between zero and one, which combines the old and new link probabilities.

We have run extensive experiments with synthetic and real data to test the performance of the web view construction algorithm. Our results show that starting from a random HPA the expected trail relevance is significantly increased by the algorithm until the final HPA is output, once the link probabilities have converged within a small error.

5 Computing the Entropy of User Navigation

Herein we utilise the view of a HPA as a finite ergodic Markov chain, say $\mathcal{M}$, where all user navigation sessions start from the user's home page and eventually return to this page. Over a period of time we assume that the empirical distribution of the Markov chain probabilities, induced by the user navigation sessions, stabilises in accordance with the actual transition probabilities. The entropy of the Markov chain is central to this approach, since once the empirical distribution stabilises, the entropy of a *typical* trail is "close" to the entropy of the Markov chain as a consequence of the *Asymptotic Equipartition Property* (AEP) [CT91]. Such a typical trail can then be seen to represent the user's navigation behaviour over a period of time.

In [LL99a] we developed an iterative method for computing this entropy by considering a long navigation session, which can be viewed as the concatenation of shorter sessions each starting from the user's home page. Therein we show that the empirical entropy converges

from below to the true entropy, i.e. the empirical entropy is always an underestimation of the true entropy. The empirical entropy of a navigation trail of length t is given by

$$H(\mathcal{M}, t) = -\sum_{i=1}^{n}\sum_{j=1}^{n} \frac{m_{i,j}}{t} \log \frac{m_{i,j}}{m_i},$$

where n is the number of states of $\mathcal{M}$, $m_{i,j}$ is the number of times the link from the ith page to the jth page was followed, and m_i is the number of visits to the ith page.

Although the Markov chain assumption is somewhat controversial, it is justifiable as a first attempt to obtain analytic results to aid our understanding of the navigation problem. The recent empirical results of Pirolli and Pitkow [PP99], based on web log data that summarise user navigation sessions, support our initial choice of a (first-order) Markov chain model as opposed to a higher-order Markov chain model, for the following reasons. Firstly, navigation sessions are typically short, i.e. they do not tend to exhibit long range dependencies. In this context Huberman et al. [HPPL98] have suggested a "universal law of surfing", backed-up by evidence from web log data, which predicts that typical trails are short. Secondly, the experimental results of Pirolli and Pitkow [PP99] suggest that a first-order Markov chain model is substantially more stable over a period of time than a higher-order Markov chain model and is thus more reliable. Finally, as indicated in Subsection 4.1 higher-order Markov chains can be reduced to first-order Markov chains by aggregating states [Bil61] and therefore the techniques we present in [LL99a] extend to higher-order Markov chain models. In fact, we have extended our basic technique to higher-order Markov chains of bounded order utilising *dynamic Markov modelling* [CH87]; this is an adaptive context modelling method in which a finite-state model is built dynamically.

6 Related Work

An application of our method for computing the entropy of user navigation sessions is that of ranking web pages according to their importance or relevance. In its raw form we have the PageRank algorithm [PBMW98] (cf. [CGP98]), which is a special case of our Markov chain model, where the initial probabilities are uniform random over the state space and the transition probabilities are also uniform random over the set of out-links from a given state. The stationary distribution, say π, of this Markov chain summarises the relative weights of pages. In the more general case when the distribution of probabilities is not uniform random, indicating user preference, π will contain a more accurate weighting of the pages with respect to a given user or group of users.

The PageRank algorithm is utilised in the *Google* search engine [BP98] in order to improve traditional information retrieval methods [GRGK97]. It has also been used to measure the quality of a search engine by collecting a set of URLs gathered through a long random walk on the web according to the weights of the PageRank distribution and then testing the proportion of these pages that are indexed by a particular search engine [HHMN99].

A related but different approach is that of the *HITS* (Hypertext Induced Topic Search) algorithm [Kle98], which identifies high quality pages for a given topic by analysing the linkage between *authorities*, which are pages that are focused on the topic, and *hubs*, which are pages that contain useful links to relevant pages on the topic. The roots of this approach originate from the area of citation analysis [PN76, Gel78], whose aim is to measure the influence of

research in a given subfield using citation data; see also [Lar96], where co-citation analysis is used to cluster related web pages. The output of the HITS algorithm is a densely linked focused subgraph of hubs and authorities which is called a *web community*. The process of enumerating web communities is described in [KRRT99a, KRRT99b]. The HITS approach is based on analysing the topology of the web and does not take user navigation into account. In our Markov chain model, which is motivated by the navigation problem, the user's behaviour provides the basis of the formalism. The HITS algorithm is utilised in the *Clever* search engine [CDK+99]. It has also been used as the basis of an algorithm to find related web pages to a given page [DH99].

A straightforward technique for integrating the link structure of the web into a search engine query was proposed by Yuwono and Lee [YL96]. Therein the score of a page being considered, obtained with respect to the query, is augmented by a weighted sum of the scores of pages that have a direct link to this page. An alternative approach, using an algorithm based on spread of activation to simulate users' surfing patterns, was considered in [PPR96] (cf. [HPPL98]). In this algorithm a decay factor is utilised in order to reduce the influence of pages according to their distance from an initial set of activated pages.

A reinforcement learning technique, based on Q-learning [WD92], for learning the structure of a hypertext was proposed by Joachims et al. [JFM97]. In Q-learning a state is chosen such that the discounted sum of future rewards is maximised; in this application the states are web pages and the reward is the score returned for the web page the user is browsing, with respect to keywords the user specifies as his/her search criteria. More recently Rennie and McCallum [RM99] utilise Q-learning for the "spidering" task of finding relevant web pages on a particular topic.

A recent emerging subarea is that of *adaptive hypertext and hypermedia* [Bru96, BKV98], whose aim is to build a model of an individual user of a hypertext system and apply this model for the purpose of adapting the system to that user. A useful distinction is between *adaptive presentation*, which deals with adapting the contents of a page according to the user's knowledge and goal, and *adaptive navigational support*, whose aim is to help the user find the most relevant trails to follow by adapting the choice of links that the user can traverse.

Most adaptive presentation techniques deal with *text adaptation* which is concerned with tailoring the contents of a page to a particular user. Thus the contents of a page change according to the user browsing it. An effective technique used to implement adaptive presentation is that of *conditional text*. The text in a page is divided into several chunks, each one being associated with a relevant condition.

The most common techniques for adapting link presentation in the context of adaptive navigational support are: *direct guidance* which aims at suggesting the "best" link to follow, *adaptive ordering* which sorts the links according to some criteria which are useful to the user, *hiding* which restricts the number of allowable links by hiding links that are not relevant to the navigation session, and *annotation* which augments the links with useful comments relating to the pages that can be reached by following them.

7 Concluding Remarks

We have presented a statistical foundation for navigation in the web and hypertext structures based on a formal model in terms of hypertext probabilistic automata and ergodic Markov

chains. Using this approach we have developed two techniques for constructing a web view, which is a collection of user relevant trails. The first technique utilises web log data to construct a web view of user navigation trails, where a trail having higher probability is considered to be more relevant. The second technique utilises the user query in order to construct a web view of automatically generated trails, where a link having higher probability leads to a trail whose average relevance is higher.

We are currently working on combining the two interpretations of probability as the frequency of user traversal and the relevance to a given query. For this purpose, after a web view is constructed with respect to a user query, the transition probabilities, representing the expected relevance of following the corresponding links, can be modified to take into account the user's navigation behaviour according to a web view which is constructed from user web log data.

We are also developing an algorithm which, given a user query as input and a starting page for user navigation, rather than outputting a web view outputs the trail with the highest relevance to the query; we call this algorithm the *best trail* algorithm [LZ98].

The potential impact of web interaction on improving the quality of user interaction with the web is huge. On the one hand, it can benefit users in terms of being able to find useful information and on the other hand it can be utilised in electronic commerce by the suppliers of information.

References

[AMS+96] R. Agrawal, H. Mannila, R. Srikant, H. Toivonen, and A.I. Verkamo. Fast discovery of association rules. In U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, pages 307–328. AAAI Press, Menlo Park, Ca., 1996.

[Apo85] A. Apostolico. The myriad virtues of subword trees. In A. Apostolico and Z. Galil, editors, *Combinatorial Algorithms on Words*, volume F12 of *NATO ASI Series*, pages 85–96. Springer-Verlag, Berlin, 1985.

[Aro94] L. Aronson. *HTML, Manual of Style.* Ziff-Davis Press, Emeryville, Ca., 1994.

[Ber88] M. Bernstein. The bookmark and the compass: Orientation tools for hypertext users. *SIGOIS Bulletin*, 9:34–45, 1988.

[Ber98] M. Bernstein. Patterns of hypertext. In *Proceedings of ACM Conference on Hypertext*, pages 21–29, Pittsburg, Pa., 1998.

[Bil61] P. Billingsley. Statistical methods in Markov chains. *The Annals of Mathematical Statistics*, 32:12–40, 1961.

[BKV98] P. Brusilovsky, A. Kobsa, and J. Vassileva, editors. *Adaptive Hypertext and Hypermedia.* Kluwer, Dordrecht, 1998.

[BL98] J. Borges and M. Levene. Mining association rules in hypertext databases. In *Proceedings of International Conference on Knowledge Discovery and Data Mining*, pages 149–153, New York, NY, 1998.

[BL99] J. Borges and M. Levene. Heuristics for mining high quality user web navigation patterns. Research Note RN/99/68, Department of Computer Science, University College London, 1999.

[BL01] J. Borges and M. Levene. Data mining of user navigation patterns. In B. Masand and M. Spiliopoulou, editors, *Web Usage Mining*, Lecture Notes in Computer Science (LNCS 1836), pages 92–111. Springer-Verlag, Berlin, 2001.

[BP98] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. In *Proceedings of International World Wide Web Conference*, pages 107–117, Brisbane, 1998.

[BR99] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. ACM Press and Addison-Wesley, Reading, Ma., 1999.

[Bru96] P. Brusilovsky. Methods and techniques of adaptive hypermedia. *User Modeling and User-Adapted Interaction*, 6:87–129, 1996.

[Bus45] V. Bush. As we may think. *Atlantic Monthly*, 76:101–108, 1945.

[Bus91a] V. Bush. Memex II. In J.M. Nyce and P. Kahn, editors, *From Memex to Hypertext: Vannevar Bush and the Mind's Machine*, pages 165–184. Academic Press, San Diego, Ca., 1991.

[Bus91b] V. Bush. Memex revisited. In J.M. Nyce and P. Kahn, editors, *From Memex to Hypertext: Vannevar Bush and the Mind's Machine*, pages 197–216. Academic Press, San Diego, Ca., 1991.

[CDK+99] S. Chakrabarti, B. Dom, S.R. Kumar, P. Raghavan, S. Rajagopalan, A. Tomkins, D. Gibson, and J.M. Kleinberg. Mining the web's link structure. *IEEE Computer*, 32:60–67, 1999.

[CGP98] J. Cho, H. Garcia-Molina, and L. Page. Efficient crawling through URL ordering. In *Proceedings of International World Wide Web Conference*, pages 161–172, Brisbane, 1998.

[CH87] G.V. Cormack and R.N.S. Horspool. Data compression using dynamic Markov modelling. *The Computer Journal*, 30:541–550, 1987.

[Cha73] C. Chatfield. Statistical inference regarding Markov chain models. *Applied Statistics*, 22:7–20, 1973.

[CPY98] M.-S. Chen, J.S. Park, and P.S. Yu. Efficient data mining for traversal patterns. *IEEE Transactions on Knowledge and Data Engineering*, 10:209–221, 1998.

[CRS85] D. Canter, R. Rivers, and G. Storrs. Characterizing user navigation through complex data structures. *Behaviour and Information Technology*, 4:93–102, 1985.

[CT91] T.M. Cover and J.A. Thomas. *Elements of Information Theory*. Wiley Series in Telecommunications. John Wiley & Sons, Chichester, 1991.

[DeR89] S.J. DeRose. Expanding the notion of links. In *Proceedings of ACM Conference on Hypertext*, pages 249–257, Pittsburg, Pa., 1989.

[DH99] J. Dean and M.R. Henzinger. Finding related pages in the world wide web. In *Proceedings of International World Wide Web Conference*, pages 1467–1479, Montreal, 1999.

[Eme90] E.A. Emerson. Temporal and modal logic. In J. Van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B, chapter 16, pages 997–1072. Elsevier Science Publishers, Amsterdam, 1990.

[Fur86] G.W. Furnas. Generalized fisheye views. In *Proceedings of ACM*, pages 16–32, Boston, Ma., 1986.

[Fur97] G.W. Furnas. Effective view navigation. In *Proceedings of ACM Conference on Human Factors in Computing Systems*, pages 367–374, Atlanta, Georgia, 1997.

[Gel78] N.L. Geller. On the citation influence methodology of Pinski and Narin. *Information Processing & Management*, 14:93–95, 1978.

[GRGK97] V.N. Gudivada, V.V. Raghavan, W.I. Grosky, and R. Kasanagottu. Information retrieval on the world wide web. *IEEE Internet Computing*, 1:58–68, 1997.

[HHMN99] M.R. Henzinger, A. Heydon, M. Mitzenmacher, and M. Najork. Measuring index quality using random walks on the web. In *Proceedings of International World Wide Web Conference*, pages 1291–1303, Montreal, 1999.

[HPPL98] B.A. Huberman, P.L.T. Pirolli, J.E. Pitkow, and R.M. Lukose. Strong regularities in world wide web surfing. *Science*, 280:95–97, 1998.

[HU79] J.E. Hopcroft and J.D. Ullman. *Introduction to Automata Theory, Languages and Computation*. Addison-Wesley, Reading, Ma., 1979.

[JFM97] T. Joachims, D. Freitag, and T. Mitchell. WebWatcher: A tour guide for the World Wide Web. In *Proceedings of International Joint Conference on Artificial Intelligence*, pages 770–775, Nagoya, Japan, 1997.

[JO99] B. Jiang and F. Ormeling. Mapping cyberspace: Visualising, analysing and exploring virtual worlds. Working Paper Series 11, Centre for Advanced Spatial Analysis, University College London, 1999.

[KH95] H. Kim and S.C. Hirtle. Spatial metaphors and disorientation in hypertext browsing. *Behaviour and Information Technology*, 14:239–250, 1995.

[Kle98] J.M. Kleinberg. Authoritative sources in a hyperlinked environment. In *Proceedings of ACM-SIAM Symposium on Discrete Algorithms*, pages 668–677, San Francisco, 1998.

[KRRT99a] R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins. Extracting large-scale knowledge bases from the web. In *Proceedings of International Conference on Very Large Data Bases*, pages 639–650, Edinburgh, 1999.

[KRRT99b] R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins. Trawling the web for emerging cyber-communities. In *Proceedings of International World Wide Web Conference*, pages 1481–1493, Montreal, 1999.

[KS60] J.G. Kemeny and J.L. Snell. *Finite Markov Chains*. D. Van Nostrand, Princeton, NJ, 1960.

[Lar96] R.R. Larson. Bibliometrics of the world wide web: An exploratory analysis of the intellectual structure of cyberspace. In *Proceedings of Annual American Society for Information Science Meeting*, pages 71–78, Baltimore, Md., 1996.

[LG99] S. Lawrence and C.L. Giles. Accessibility of information on the web. *Nature*, 400:107–109, 1999.

[LL99a] M. Levene and G. Loizou. Computing the entropy of user navigation in the web. Research Note RN/99/42, Department of Computer Science, University College London, 1999.

[LL99b] M. Levene and G. Loizou. Navigation in hypertext is easy only sometimes. *SIAM Journal on Computing*, 29:728–760, 1999.

[LL99c] M. Levene and G. Loizou. A probabilistic approach to navigation in hypertext. *Information Sciences*, 114:165–186, 1999.

[LZ98] M. Levene and N. Zin. An adaptive algorithm for navigation in web-like databases. Research Note RN/98/36, Department of Computer Science, University College London, 1998.

[MFDG98] S. Mace, U. Flohr, R. Dobson, and T. Graham. Weaving a better web. *Byte*, 23:58–68, 1998.

[MGZ89] N. Merhav, M. Gutman, and J. Ziv. On the estimation of the order of a Markov chain and universal data compression. *IEEE Transactions on Information Theory*, 35:1014–1019, 1989.

[MI89] C.C. Marshall and P.M. Irish. Guided tours and on-line presentations: How authors make existing hypertext intelligible for readers. In *Proceedings of ACM Conference on Hypertext*, pages 15–26, Pittsburg, Pa., 1989.

[Nel80] T.H. Nelson. Replacing the printed word: A complete literary system. In *Proceedings of IFIP Congress 80*, pages 1013–1023. Tokyo, 1980.

[Nie90] J. Nielsen. *Hypertext and Hypermedia*. Academic Press, Boston, Ma., 1990.

[NK91] J.M. Nyce and P. Kahn, editors. *From Memex to Hypertext: Vannevar Bush and the Mind's Machine*. Academic Press, San Diego, Ca., 1991.

[Ore91] T. Oren. Memex: Getting back on the trail. In J.M. Nyce and P. Kahn, editors, *From Memex to Hypertext: Vannevar Bush and the Mind's Machine*, pages 319–338. Academic Press, San Diego, Ca., 1991.

[PBMW98] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Working paper, Department of Computer Science, Stanford University, 1998.

[PN76] G. Pinski and F. Narin. Citation influence for journal aggregates of scientific publications: Theory, with application to the literature of physics. *Information Processing & Management*, 12:297–312, 1976.

[PP99] P. Pirolli and J.E. Pitkow. Distributions of surfers' paths through the world wide web: Empirical characterizations. *World Wide Web*, 2:29–45, 1999.

[PPR96] P. Pirolli, J.E. Pitkow, and R. Rao. Silk from a sow's ear: Extracting usable structures from the Web. In *Proceedings of ACM Conference on Human Factors in Computing Systems*, pages 118–125, Vancouver, 1996.

[RA98] G. Ridgeway and S. Altschuler. Application of discrete finite Markov process clustering for understanding web usage. In *Spring Research Conference on Statistics in Industry and Technology, Section on Physical and Engineering Sciences*, Santa Fe, New Mexico, 1998.

[RBS94] E. Rivlin, R. Botafogo, and B. Shneiderman. Navigating in hyperspace: Designing a structure-based toolbox. *Communications of the ACM*, 37:87–96, 1994.

[RM99] J. Rennie and A.K. McCallum. Using reinforcement learning to spider the web efficiently. In *Proceeding of the International Conference on Machine Learning (ICML)*, Bled, Slovenia, 1999.

[SB98] R.S. Sutton and A.G. Barto. *Reinforcement Learning*. MIT Press, Cambridge, Ma., 1998.

[SKS98] S. Schechter, M. Krishnan, and M.D. Smith. Using path profiles to predict HTTP requests. *Computer Networks and ISDN Systems*, 30:457–467, 1998.

[SZAS97] C. Shahabi, A.M. Zarkesh, J. Adibi, and V. Shah. Knowledge discovery from users web-page navigation. In *IEEE Workshop on Research Issues in Data Engineering*, Birmingham, U.K., 1997.

[TD92] K. Tochtermann and G. Dittrich. Fishing for clarity in hyperdocuments with enhanced fisheye-views. In *Proceedings of ACM Conference on Hypertext*, pages 212–221, Milano, Italy, 1992.

[Tri91] R.H. Trigg. From trailblazing to guided tours: The legacy of Vannevar Bush's vision of hypertext use. In J.M. Nyce and P. Kahn, editors, *From Memex to Hypertext: Vannevar Bush and the Mind's Machine*, pages 353–367. Academic Press, San Diego, Ca., 1991.

[Van89] H. Van Dyke Parunak. Hypertext topologies and user navigation. In *Proceedings of ACM Conference on Hypertext*, pages 43–50, Pittsburg, Pa., 1989.

[WD92] C.J.C.H. Watkins and P. Dayan. Q-learning. *Machine Learning*, 8:279–292, 1992.

[YL96] B. Yuwono and D.L. Lee. Search and ranking for locating resources on the world wide web. In *Proceedings of IEEE International Conference on Data Engineering*, pages 164–171, New Orleans, Lo., 1996.

[Zel89] P.T. Zellweger. Scripted documents: A hypermedia path mechanism. In *Proceedings of ACM Conference on Hypertext*, pages 1–13, Pittsburg, Pa., 1989.

[ZL99] N. Zin and M. Levene. Constructing web views from automated navigation sessions. In *Proceedings of ACM Digital Library Workshop on Organizing Web Space (WOWS)*, pages 54–58, Berkeley, Ca., 1999.