

The Navigation Problem in the World-Wide-Web

M. Levene

Department of Computer Science
University College London
Gower Street, London WC1E 6BT, U.K.
email: mlevene@cs.ucl.ac.uk

Abstract: Herein we build statistical foundations for tackling the navigation problem users encounter during web interaction, 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 belongs to; such knowledge can be used to adapt the system to the user's goal.

Herein 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. The rest of the paper is organised as follows. In Section 2 we give a brief overview of hypertext and 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 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 3.1 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 3.2 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 Subsection 3.3 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. We summarise in Section 4.

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).

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, i.e. a URL, describing where the resource resides and how to retrieve 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.

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.

As stated at the beginning of this section 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, \dots, 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

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

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.)

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 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.

3 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 accepting into the web view only trails whose overall probability is above some *cut-point* $\lambda \in [0, 1)$.

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 $p(T) > \lambda$.

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} .

3.1 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 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 decision only on the page currently being browsed.

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 [BL99]. 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.

3.2 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. The generic algorithm is composed of the following three steps:

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 [ZL99]. 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.

3.3 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 i th page to the j th page was followed, and m_i is the number of visits to the i th page.

4 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. 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.

Bibliography

- [BL99] J. Borges and M. Levene. Data mining of user navigation patterns. In *Proceedings of Workshop on Web Usage Analysis and User Profiling (WEBKDD), in conjunction with ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 31–36, San Diego, Ca., 1999. Long version submitted for publication.
- [BR99] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. ACM Press and Addison-Wesley, Reading, Ma., 1999.
- [CT91] T.M. Cover and J.A. Thomas. *Elements of Information Theory*. Wiley Series in Telecommunications. John Wiley & Sons, Chichester, 1991.
- [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.
- [HU79] J.E. Hopcroft and J.D. Ullman. *Introduction to Automata Theory, Languages and Computation*. Addison-Wesley, Reading, Ma., 1979.
- [KS60] J.G. Kemeny and J.L. Snell. *Finite Markov Chains*. D. Van Nostrand, Princeton, NJ, 1960.
- [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.
- [Nie90] J. Nielsen. *Hypertext and Hypermedia*. Academic Press, Boston, Ma., 1990.
- [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.