Design and Evaluation of Improvement Method on the Web Information Navigation – A Stochastic Search Approach

Benjamin P.-C. Yen
School of Business
The University of Hong Kong, Hong Kong

Y.-W. Wan
Graduate Institute of Global Operations Strategy and Logistics Management
National Dong Hwa University, Hualien, Taiwan

Abstract

With the advent of fast growing Internet and World Wide Web (the Web), more and more companies enhance the business competitiveness by conducting electronic commerce. At the same time, more and more people gather or process information by surfing on the Web. However, due to unbalanced Web traffic and poorly organized information, users suffer from slow communication and disordered information. To improve the situation, information providers can analyze the traffic and Uniform Resource Locator (URL) counters to adjust the information layering and organization; nevertheless, heterogeneous navigation patterns and dynamic fluctuating Web traffic complicate the improvement process. Alternatively, improvement can be made by giving direct guidance to the surfers in navigating the Web sites. In this paper, information retrieval on a Web site is modeled as a Markov chain associated with the corresponding dynamic Web traffic and designated information pages. We consider four models of information retrieval based on combination of the level of skill or experience of the surfers as well as the degree of navigation support by the sites. Simulation is conducted to evaluate the performance of the different types of navigation guidance. In addition, we evaluate the four models of information retrieval in terms of complexity and applicability. The paper concludes with a research summary and a direction for future research efforts.

Keywords: Stochastic shortest path, Web, navigation

1. INTRODUCTION

Information plays an indispensable role in the world. The network and information systems have changed the way people communicate with each other as well as expedited the process to obtain the information that matches their interests. Everyday hundreds of millions of transactions flow through the network all over the world. Any information can be transferred from one place to another only within few seconds. Together with the growth of the needs of information, the web pages on the Internet grow explosively during the past few years and such increase is expected to be more acute going forward.

The abundance of the URL's apparently creates a great value for all the visitors by allowing them to retrieve comprehensive information from the Web. Web page owners, at the same time, benefit from the advertisement opportunities when visitors surf their web pages. However, any lack of organization of the voluminous information may encumber the searching performance. Visitors spend astounding amount of time in navigating through the useless or redundant pages. To tackle the problem, Web page owners need to invest to update and reorganize the information for their Web pages on an on-going basis. As a result, there are questions concerning both the visitors and Web page owners that need to be answered: What do Web page owners need to take into account in addition to the Web pages (i.e. information content) in the design of Web site? How can be enhanced on the Web site to improve information retrieval? How to justify the requirement, necessity, and cost for such improvements? How to balance the cost and benefit of such improvements? In answering these questions, research has been conducted in three areas: (1) Web site customization based on the user access information; (2) agents based intelligence search for information retrieval and discovery; and (3) intelligence browser and the collection of user information.

Perkowitz and Etzioni [1-2] propose an Artificial Intelligence approach to create the Adaptive Web Site, which can improve the site organization based on the user access log with the assumption that each originating computer corresponds to a particular user. Yan *et al.* [3] propose the use of access patterns to generate hyperlinks, which are captured in the access log and analyzed offline in an interval basis, to improve the information access. Wang *el al.* [4] propose a personalized filtering model to filter and rank the product information with linear functions on the user preference. Chen and Kuo [6] propose a personalized information retrieval system based on the user profile modeled as the Semantic Relevance (SR) and Co-occurrence (CO) of keywords to capture the real meaning of user query.

The research in the second area focuses on seeking for the information on the Internet. Cheung *et al.* [5] propose a model of four-level classification tool of learning the behavior of both information user and information source. Chang *el al.* [7] present a Site Traveling Algorithm (STA) to discover the relevant information, in which the relevance of the retrieved document is evaluated with the content popularity and richness (CPR). Yang *el al.* [8] present the development of intelligent personal Internet agent based on automatic textual analysis of Internet document and hybrid simulated annealing algorithm. Tu and Hsiang [9] propose an Interactive Information Retrieval (IIR) agent architecture to handle group knowledge and preference, and to keep track of the individual user profile. Teng *et al.* [10] propose a scalable method for parallel processing in both information crawling/gathering and processing.

The third area of research concerns navigation assistance for user during the browsing process. Joachims *et al.* [11] introduce the Web-Watcher, based on a learning approach with user feedback to improve the quality of interactive navigation advice. Similarly, Liaberman [12] introduces the intelligent agent, *Letiza*, which works with conventional web browsers to keep track of the user browsing behavior and interests. Furthermore, Berghel *et al.* [13] present a web browser called the “Cyberbrowser” to customize the information access to the content within the web page, which include keyword and sentences extraction according to user selection. Lin *et al.* [14] describe an approach for capturing user access patterns on the Web by addressing the limitation of the Web servers which only recognize the proxy servers instead of the individual users. Richardson [15] does a comparison on the existing tools to gather the access information on the Internet, such as visitor counter and guest book.

The literature above shows the modeling of the information retrieval mostly is based on database models and data mining techniques. The analysis centers on the user behavior patterns largely for the global Web information retrieval. There are few studies for the analysis on site structure for information retrieval as optimization problems. There are two types of models for problem formulation – deterministic models and stochastic models.

Gibson *et al.* [16] and Chakrabarti *et al.* [17] define the Web sites as “authorities” and “hub” in isolation and conclude that a respected authority is a page that is referred to by many good hubs and a useful hub is a location that points to many valuable authorities. Donato *et al.* [18] mine the inner structure of the Web graph and propose a series of measurements on the Web. They find that graph does not exhibit self similarity within its components and their inner structure is quite distinct. Chakrabarti *et al.* [19] develop algorithms that exploit the hyperlink structure of the Web for information discovery and categorization, the construction of high quality resource lists, and the analysis of on-line hyperlinked communities. Eiron and McCirley [20] investigate the construction of data models of the Web that capture the hierarchical nature of the Web and some crucial features of the link graph. Yen [21] defines four types of Web page accessibility models proposes the guidelines to balance accessibility and popularity. Kleinberg *et al.* [22] describe two algorithms that operate on the Web graph, addressing problems from Web search and automatic community discovery.

Sarukkai [23] uses a Markov Chain model based on the user access information for link prediction and path analysis. Levene *et al.* [24] derive Zipf’s rank frequency law from an absorbing Markov chain model of surfers’ behaviour assuming that less probable navigation trails are, on average, longer than more probable ones. Levene and Loizou [25,26] formulate a hypertext database as a graph and propose a probability approach to find the trail to match the query. Kumar *et al.* [27] propose a stochastic model of the web graph to show additional properties of the random graph. Levene *et al.* [28] extend the evolutionary model of the Web graph by including a non-preferential component and viewing the stochastic process in terms of an urn transfer model. Zin and Levene [29] propose that information on the topology is important for useful exploration and can also help to reduce the feeling of disorientation that users may experience.

From the review above, it should be noted that there lacks the consideration of dynamic factors (such as dependent reverse links and user familiarity with Web navigation) of assessing information from the Web in analyzing and evaluating the access models. In this research, we propose a graph-based structure with stochastic properties (such as traffic conditions and navigation information) and various dynamic policies to guide the users in accessing information. The proposed approach captures the essence of the surfing process: A surfer surfs among web pages of a site according to the structure of pages as reflected from the hyperlinks, waiting for the loading of new web pages as he triggers hyperlinks. The process is similar to an entity jumping from the source to the destination node of a graph, with web pages as nodes, hyperlinks as arcs, loading times as distances among nodes, and some stochastic properties and dynamic policies as the surfing behavior and some as the navigation guidance provided by web pages. All these web structures and routing policies are modeled discrete-time Markov chains, and the expected time for a surfer to arrive at his destination is calculated based on both dynamic and stochastic shortest paths.

The level of skill or experience of the surfers as well as the degree of guidance supported by the sites is two of the major issues for information retrieval on the Web. Four models are considered in our research to reflect these two issues: inexperienced surfers on guidance-less sites (ISL), where totally inexperienced surfers surfing among web pages that do not provide any navigation guidance, leading to surfers randomly moving among pages; experienced surfers on guidance-less sites (ESL), where the web pages do not provide guidance but the surfers are experienced, i.e., they only trigger unvisited pages as they search for their required information; sites with the Mean-Path Guidance (MPG), where the navigation guidance of a site is in terms of static values of the mean loading times of web pages; and sites with the Known-First-Arc Guidance (KFA), where the navigation guidance of a site considers the real-time loading time of the next page with mean loading times for future pages.

The rest of the paper is organized as the following. Section 2 contains the problem description and formulation. Section 3 sets out the discussion on the four searching models of navigation guidance. Section 4 demonstrates the simulation result and comparison on complexity and applicability for the four proposed models. The paper concludes with a research summary and a direction for future research efforts.

2. PRELIMINARY - PROBLEM DESCRIPTION

Consider a situation where data are distributed over a computer network that is composed of switches and cables. A switch contains information about (i) the cable capacity; and (ii) the estimated time for the packet going from each neighboring switch to the destination (typically in the form of look-up table). Whenever a data-packet arrives at a switch, the switch to visit next needs to be determined based on the real time information on the network capacity. A routing strategy is used to utilize the real-time information to form dynamically a path of minimized costs for data delivery. We, therefore, proposed to improve the path for data delivery by way of adding cables (the aggressive approach) and/or deleting cables (the passive approach) in between the respective switches to improve the data delivery performance. By the same token, we propose to improve the structure of the Web site by adding or deleting hyperlinks on the Web pages, so that the Web surfers can navigate to the destination page most efficiently. In this section, we first model the Web navigation as graph traverse problem. We further classify the problems based on various Web structure properties. The model is extended based on the "explorative transformation".

2.1 Problem Formulation

A Web site comprises a number of Web pages and each Web page may have a number of hyperlinks connecting to other pages. Each Web page is associated with a loading time, which varies in proportion to the page content and the network traffic conditions. Vertices and arcs are denoted as follows:

Vertices: $V=\{v_1, v_2, \ldots v_n\}$. Each v_i (i=1, 2, … n) refers to one page..

Arcs: $E =\{[v_i, v_j]$ or e_{ij} | There is a hyperlink in page i pointing to page $j\}$. The arc connecting v_i to v_j denotes a hyperlink from page i to page j.

The hyperlinks are directed, so is the network. Since each Web page has a loading time, the corresponding vertex in the network is assigned a weight representing such loading time. Naturally, the loading time of a Web page is directly determined by the size of the page contents, such as text, images and sound/video clips, and network conditions. Among these factors, the page size (x_i) is the most dominating one. Hence, it is assigned as the weight of vertex to reflect the download time:

$w: V \rightarrow R^+$

$w(v_\mathrm{i}) = x_i$

Therefore, the network is denoted as $\vec{G} = (V, E, w)$, which is a *directed graph* in which a weight is associated with each vertex. For example, let A be the start and L the destination page for a surfer on a site of web structure as shown in Figure 1. There are many ways to reach the destination *L* from *A*. Not only that there are multiple paths from *A* to *L*, there are loops in the seemingly directed graph. All web browsers provide the function to return to the previous page, i.e., a path from *A* to *L* can actually be of the form *A*-*C*-*A*-*C*-*G*-*L*. The length of the path depends not only on the sequence of nodes, but also on whether the cache function is enabled or not. If it is enabled, the length of loading time for each reverse link becomes zero; otherwise, each reverse link bears the same loading time for the previous page.

In short, the actual time taken for a surfer to move from *A* to *L* depends on the web structure, the experience of the surfer, and the type of navigation guidance provided for the surfer. For a given web structure, the time is determined by the latter two factors. For the web structure as shown in Figure 1, suppose that based on historical record, on average *A*-*C*-*G*-*L* is the shortest path from *A* to *L*. If the surfer is inexperienced and the web does not provide any navigation guidance, the surfer may take a random walk on the network, with the page-back function as the means to move in the opposition direction of the directed arcs. If the surfer is experienced, his search will be more careful; pages visited before will not be re-visited unless there is no choice. With navigation guidance provided by the web, the surfer may be guided to select *A*-*C*-*G*-*L* based on the long-term average loading times of pages, or different paths according to the traffic jam during navigation. In case the surfer needs more than one destination, the problem becomes multi-destination, i.e., we need to find the shortest collection of paths that cover all the destination nodes. If the hyperlinks on web pages can be changed, no matter as a static web design problem that changes once for a long while, or a dynamic problem that hyperlinks are highlighted according to traffic condition, then effectively the web structure is changed with the change in hyperlinks.

<Insert Figure 1 here>

2.2 Problem Properties

We elaborate the model for the Web-based information retrieval by two dimensions – structure properties and navigation properties. The *structure properties* concern the static properties of the Web site structure, including the server capacity and the cache mechanism. The *navigation properties* refer to the dynamic aspects of information retrieval, including single or multiple root pages and destination pages, constraints on the navigation path, etc. Both dimensions have significant impact on problem modeling and problem solving.

Structure properties. The sever performance might be inversely proportional to the number of the users making simultaneous page requests. This is valid both for an individual page and for a number of pages forming the whole or part of a Web site. One particular characteristic for the Web site structure is the “conditional reverse link” – the hyperlink visited enables the corresponding reverse link. If we consider the cache function to be enabled, the length of loading time for each reverse link becomes zero; otherwise, each reverse link bears the same loading time as the previous page. The cache, depending on its size constraint or time limit constraints, can store both the address and the content of a Web page. Furthermore, hyperlinks can be bi-directional (i.e. the links in both directions are valid in the original graph) and multiple (i.e. multiple identical or non-identical links between nodes).

Navigation properties. Navigation can start from a single root page (entry page) or multiple root pages (direct page address). Similarly, the destination page can also be a single page or a group of the pages. Navigation can be constrained by path length or browsing time. Objectives of navigation may include minimizing retrieval time, maximizing information quality (completeness and relevancy) and minimizing path length (radical distance form the root and number of pages visited). The decision making during navigation process can be of static or dynamic. In the static model, all the nodes along the navigation path are determined at once from the beginning of the navigation process; however, in the dynamic model, the next node to visit is only determined one at a time from the immediate proceeding node.

2.3 Problem Transformation

In addition to the *structure* and *navigation properties*, we adopt the explorative expansion approach to address the cache capability of a navigation process. The *explorative expansion* approach for conditional reverse link is based on the graph traversal (either breath first or depth first search) on a link by link basis. The process starts with the root node as an initial component. For each traverse on the new link, we consider the source node, the destination node, and the link between them. A component is defined as a graph. A new expansion component is constructed for the destination part and it is the combination of a replica of the original component, the links of both direction as the newly explored links, and the destination node if it is not included in the original component. The exploring link will connect the source node in the original component to the destination node in the new expansion component.

For example, Figure 2 (a) shows an original Web site graph. To start the expansion process, we construct an initial component with only root node (i.e. node *A*) as shown on the top of Figure 2 (b). If we explore the link e_{AB} in the next, we construct the new expansion component with node A, node B, the link e_{AB} and the link e_{BA}. A link is added between the source node (i.e. node A) in original component to the destination node (i.e. node B) in the newly generated expansion component. The whole process will complete after all the links are explored. Figure 2(b) is the complete expansion of the original Web graph in Figure 2(a). The reverse link in the expansion components represents the "back" function. All the distances in the expansion components are zero and the length of a link between any two components is the same as that in the original Web graph. For the purpose of simplicity, no subscript is added in the expansion components to differentiate one node from another.

The process can be described as an algorithm as follows.
For simplicity, we assume the original Web graph $G^0=(V, A)$.

```
Step 1.   The original component $G^e = (V^e, A^e)$ only consists of the root node, i.e. $G=(\{v_0\},\{\})$
Step 2.   The set of the links to be explored $A^x=A$.
Step 3.   $\{e_{ij} \mid v_i \in V^e$ and $e_{ij} \in A^x\}$
Step 4.   $\{G' \mid$ Qualified components with $v_i$ but without $e_{ij}\}$
Step 5.   Construct the new expansion component $G'' = G' + \{e_{i2j2}, e_{j2i2}\} + \{v_{j2}\}$
Step 6.   If exists already, then $G = G + \{e_{i1j2}\}$
Step 7.   Otherwise, include new expansion component $G = G' + G'' + \{e_{i1j2}\}$
Step 8.   Go to Step 4 if there is any $G'$
Step 9.   Go to Step 3 if there is any $e_{ij}$
Step 10.  $A^x = A^x - e_{ij}$
Step 11.  Go to Step 3 if $A^x \neq \emptyset$
```

(Remark: In Step 6, 7, and 8, i_1 and j_1 are the node subscripts in the original component; i_2 and j_2 are the node subscripts in the new component)

The number of the expansion components is dependent on the number of the links in the original graph. Two extreme examples, namely chain structure and star structure, represent the lower bound (i.e. $n\times(n+1)/2=(n^2+n)/2$) and upper bound (i.e. $n^2\times n=n^3$), respectively, where n is the number of nodes. In this study, we focus on cases with the assumption of single-destination and disabled cache.

<Insert Figure 2 here>

3. SEARCHING MODELS

The level of skill or experience of the surfers as well as the degree of navigation support by the sites may vary. An inexperienced surfer may browse randomly and load repeated pages, while an experienced one may browse systematically and only load a page again if necessary. An immature site may not provide any navigation information for surfers, while a well-designed one may provide navigation information about the site structure and the expected loading time of the respective pages. A sophisticated site may even provide real-time navigation information changing simultaneously with the traffic of the Web.

For simplicity, we assume that the cache function is disabled and that the surfer has only one destination in mind. Enabled cache function and multiple destinations are intended for future extension works. We consider four simulation models based on the combination of the level of skill or experience of the surfers as well as the degree of navigation support by the sites: inexperienced surfers on guidance-less sites (ISL), experienced surfers on guidance-less sites (ESL), sites with the Mean-Path Guidance (MPG), and sites with the Known-First-Arc Guidance (KFA). The classification of the four models is based on the navigation guidance and repeatable navigation as shown in the Figure 3. The information retrieval on the Web site in these four models can be described as discrete Markov Chains as follows [23]. A discrete Markov chain model can be defined by the tuple (S, $\boldsymbol{T}$, λ). $\boldsymbol{S}$ corresponds to the state space (set of pages of Web site), $\boldsymbol{T}$ is a matrix representing transition probabilities from one state to another (i.e. from one page to another page), and λ is the initial probability distribution of the states in S. The detail is discussed in the following the subsections. The modeling is mainly based on the work of Glover *et al.* [30], Shier and Witzgall [31], Psaraftis and Tsitsiklis [32], Geetha and Nair [33], Polychronopoulos and Tsitsiklis [34] and Cheung [35].

<Insert Figure 3 here>

3.1 Inexperienced Surfers On Guidance-less Sites (ISL)

Consider a totally inexperienced surfer on a site without any navigation guidance. The surfer moves randomly, possibly back and forth, among the pages, and picks links in a page arbitrarily. The movement of such a surfer can be modeled by a random walk on a connected graph, and the page search process can be modeled as a discrete-time Markov chain [23].

Refer to the site structure in Figure 1. Let $X_n = s$ if the surfer is at page s after the nth move (page loading). Take $X_1 = A$, because A is the root page. In general, if a surfer starts from page s with probability p_s, the following argument still goes through by taking weighted average of outcomes from page s with probability $P(X_1 = s) = p_s$. Let $S = \{A, B, \ldots, L\}$ be the state space of $\{X_n\}$. From the description on the above paragraph, $\{X_n\}$ is a discrete-time Markov chain. The transition probabilities can be found from the number of links on a page. For example, if the surfer is on page G of the site shown in Figure 1, he will next visit sites C, K, J, and L with probability 0.25 (providing that he decides to continue his surfing). It is straightforward to show that $\{X_n\}$ is an absorption chain with L as the absorbing state.

For any page $s \neq L$¸ let N_s be number of visits to page s before visiting page L and T_s be the loading time of page s. $E[N_s]$ is found from the first passage time argument from state A to state L for the chain $\{X_n\}$ (please see [36], pp 152 and pp. 172) and the expected searching time of L = $\sum_{i \neq L} E[N_s]E[T_s]$.

3.2 Experienced Surfer On Guidance-less Sites (ESL)

Consider an experienced surfer visits, for the first time, a site that does not provide any navigation guidance. The surfer randomly picks up *unvisited* links in a page. As far as possible, he will not re-visit a page that he has previously visited. However, he still needs to re-visit some pages when he goes into a dead end during the process of searching for his destination.

Such a search behavior cannot be modeled by the random walk in Section 3.1 above. We can still formulate it as a discrete-time Markov chain, but the size of the state space will be astronomically big: for N pages, to keep track of the identification of the pages visited, the state space is of size $O(N\times 2^N)$. The expected searching time will be the first-passage time from first entering page A, to any state such that page L is visited for the first time. The size of the chain precludes any sensible study through this approach on sites of practical size. Fortunately, we can still easily build up simulation models to estimate the expected searching time for this approach. In each replication of simulation, the simulation program traces the pages visited by the surfer, with the probability of visiting an unexplored page changed along the course. The mean time to reach page L across all simulated replications is an estimate of the mean time for ESL.

3.3 Sites with Mean-Path Guidance (MPG)

From this section onwards we consider sites that provide various types of navigation guidance. The navigation guidance may be static or dynamic, ranging from long-term mean to real-time loading time, with all possible combinations of means and exact values lying between the two extremes. Because of the navigation guidance, the difference in the expected searching time between the experienced and inexperienced surfers is minimal and hence is ignored.

In this section, we consider navigation map guidance constructed from the mean path (loading) time. To do so, we define the loading time of the destination page as the length of the directed arc. For example, the *length* of arc $A\rightarrow B$ is $E(T_B)$, which is the expected time to load page B. Other arcs are treated similarly. After the directed arcs are formed using the mean loading time, we get a directed graph with positive cycle length. Standard algorithms, such as Dijkstra's algorithm (or its variations) [37] can be used to find the shortest path from A to L.

In a site that provides the Mean-Path Guidance, a surfer is given a sequence of pages identified using the above procedure. The average time taken for the surfer to reach his destination is the value as identified by the Mean-Path Guidance. While the calculation is simple and straightforward, without considering the real-time page loading times, the path from the Mean-Path Guidance may be suboptimal. For example, take the destination page to be page L in Figure 1 and suppose that the Mean-Path Guidance suggests the path A-C-G-L. However, at any epoch, due to the instantaneous traffic, the actual loading time for an alternative path A-B-E-L could be less than that of the suggested path A-C-G-L.

3.4 Sites with Known-First-Arc (KFA)

The Mean-Path Guidance can be readily extended to the Known-First-Arc Guidance, which considers real-time loading time. At page *A*, a surfer may choose from three pages, namely, pages *B*, *C*, and *D*. If the system can estimate the loading time of each of pages *B*, *C*, and *D*, then in the calculation of the shortest path from *A* to *L*, the actual exact loading time can be used in arc *A*-*B*, *A*-*C*, and *A*-*D*, while the mean loading time are used in the remaining arcs of the paths. Such a method is called the *Known-First-Arc* Guidance. This Known-First-Arc Guidance is considered more advanced than the Mean-Path Guidance, and the difference is even more noticeable when loading time have large variance.

The Know-First-Arc Guidance can be applied repeatedly. Suppose a surfer moves to page *C* based on the Known-First-Arc approach. At *C*, the loading times of pages *A*, *F*, and *G* become known quantities at the moment when the surfer leaves page *C*. A system can repeatedly apply the Known-First-Arc approach to determine the loading time of the next page to visit, with an objective to moving to page *L* in the shortest possible time. The guidance provides by this method is dynamic. Generally speaking, once the surfer loads the root page and enters his detention, the system will guide him through the site to the destination by informing him dynamically which page to load next.

Assume that each time the actual exact loading time are random draws of the corresponding T_s. The Known-First-Arc Guidance discussed above is exactly the *Dynamic Stochastic Shortest Path* (*DSSP*) problem considered in Cheung [35] that studies the formulation of a dynamic shortest path in a network and proposes a routing policy to compute the expected path cost by mimicking the classical label-correcting approach. The *DSSP* allows the surfers to retreat from a wrongly chosen path, when real-time rather than the excepted mean loading time is revealed. However, the surfer may cycle around pages before he reaches the destination. Refer to Figure 4 below which shows part of the site structure in Figure 1. For simplicity, we use arcs with two arrows to represent the two directional flows. Here we assume that the loading time of all pages are *i.i.d.* (Independent and Identically-Distributed) random variables, distributing uniformly in {1, 2, 3, 10}, whether or not a page is visited for the first time. Suppose that the surfer wants to go from page *A* to *L*. At the moment when the surfer leaves *A*, whether the surfer will visit page *B* or *C* next depends on the current loading times of page *B* and page *C* at that time. If the loading time of page *B* is of one unit and that of page *C* of 10 units, then it is more desirable to load page *B* next, in which case it is possible for the surfer to be directed to page *A* or page *E* later on. If, however, page *C* is loaded next, then the surfer will be directed to page *G* later on. Similarly, at page *E*, the surfer may be directed next to page *B* or page *L*.

The movement of a surfer based on the *DSSP* can be modeled as a discrete-time Markov chain. Provided that there are not many links from one page to another, the first-passage analysis of such a chain is feasible for pages of all reasonable sizes. Consider the same example in Figure 1 going from page *A* to page *L*. Define a Markov chain $\{X_n\}$ with the same state and state space as in Section 3.1 above. The transition probabilities are found from *DSSP*. Let $S(i)$ be the set of successor pages that are possible to visit next when the surfer is at page *i*. It is straightforward to find $p_{ij} = P(X_{n+1} = j|X_n = i)$ and $E[T_j|\{X_n\}$ moves from i to $j]$ from the joint distribution of $\{T_j, j \in S(i)\}$. Hence, the time from *A* to *L* is given by $\sum_{k=1}^{N_{AL}} T_k$, where N_{AL} is the first-passage time (number of transitions) from page *A* to page *L*, and T_k is the time taken in loading the *k*th page. N_{AL} can be expressed as the sum of the $i \to j$ transitions before reaching page *L*, and giving the $i \to j$ transition, the expected loading time is given by the set of conditional expected loading time $\{E[T_j|\{X_n\}]\}$. Consequently, we can compute the expected searching time $E\left[\sum_{k=1}^{N_{AL}} T_k\right]$.

<Insert Figure 4 here>

4. EXAMPLE, EVALUATION, AND EXTENSION

In this section, we compare the expected searching time of the four models discussed in Section 3. While some of these models can be reviewed analytically, we have examined all four by simulation. We simulate the page searching performance of surfers of different skill levels on sites of different degrees of guidance support. In our simulation, the loading times are random. Such an approach is a first-order approximation that captures the variation of the loading time in relation to the traffic of the Web site. We illustrate the models with a site structure shown in Figure 1. The nodes are the pages and the arcs are the hyperlinks of a page. We take page *A* as the root and page *L* as the destination, and we will compare the total expected time for a surfer to get to page *L* after page *A* has been loaded. All loading time are assumed to be independent. We also compare the four models on complexity and applicability. The implementation issue of Known-First-Arc Guidance approach is also discussed at the end of the section.

To highlight the effect of searching and navigation guidance, we only consider the page loading times and ignore time taken to read information of a page. It is easy to incorporate the page reading time in the simulation models. However, this additional factor only blurs our focus – the percentage differences of times among models are certainly reduced by long page reading times.

The arcs in Figure 1 appear as directed, however, in reality there are possibly loops traced by the surfer as he searches from page *A* to page *L*. The page-return function of any browser ensures the looping possibility in ISL, ESL, and KFA. On the other hand, no matter for any web structure, with intrinsic loops or not, MPG simply suggests one straight path without any loop for a surfer. Taking all these considerations together, it does not change the inferences and insights deduced from the numerical run even if we take the web structure as shown in Figure 1. Along the same line, a different set of parameter values for the numerical runs certainly changes the values of numerical results, but it will not change the general trend of the inferences and insights gained from simulation.

4.1 Numerical Examples

The underlying processes of all the four methods can be modeled as discrete-time Markov chains. Generally, the one-step transition probability matrix of MPG is the easiest to deduce, and that of ESL the most tedious. Consider web structure as shown in Figure 1. By definition, MPG suggests a single path and its one-step transition probabilities are equal to one for arc along the suggested path and are zero otherwise. For ISL, if page j being adjacent to page i, $P(X_1 = j|X_0 = i) = 1/n$, where n is the number of pages adjacent to page i. For KFA, conceptually, given the distribution of the loading times and the web structure, it is possible to deduce the probability that a path is shorter at a transition, which effectively gives the one-step transition probabilities. The deduction is simply a matter of notation and is not shown here. Similarly, should we put down a discrete-time Markov chain of $12*2^{12}$ states according to the current page and collection of pages visited or not, we can list the one-step transition probabilities of ESL. Certainly there is no point to do so. In fact, the simulation programs only follow the random mechanism of each model to generate the next move of a surfer. It is not necessary to explicitly know the one-step transition probability matrices.

In our simulation, all page loading times are assumed to be independent and identically distributed (i.i.d.) random variables distributing uniformly in {1, 2, 3, 10}. With this choice of loading time distribution, the Mean-Path Guidance will definitely direct a surfer to the path either *A-C-G-L* or *A-B-E-L*, and the Known-First-Arc Guidance may do the same depending on the instantaneous loading time of pages *B* and *C* at the moment when the surfer leaves page *A*. Meanwhile, the *DSSP* gives the same result as the Known-First-Arc Guidance at the moment when the surfer leaves page *A*. However, at pages *B*, *E*, and any subsequent visits of page *A*, the page to visit next is determined by the real-time loading time as discussed in Section 3.4 above.

The underlying processes of the four models are of different sizes and complexities. They require different number of runs to achieve the same degree of accuracy; e.g., in MPG, the mean time to reach page *L* is given by a close-form expression without doing any simulation. However, to ensure that all simulation results are under the same set of random variates generated, which is a form of variance reduction by common random numbers, the four models are simulated for the same number of runs. With nearly 50,000 states in ESL, it takes a large number of runs to ensure enough precision and confidence on the estimates for the model. Consequently, a million replications are carried out for the four models. The simulation results are shown in Table 1 below.

<Insert Table 1 here>

As expected, an inexperienced surfer without navigation guidance takes the longest time to get to his destination. Our results show a ratio of more than 13 times between this random search and the guided searches. An experienced surfer without guidance can reduce his searching time by more than 50% when compared to an inexperienced surfer. The surfer only revisits a previous page if he has exhausted all the possible options in the current page. However, without the knowledge of site structure, it is inevitable for a surfer to visit a page repeatedly. Consequently, the mean searching time of unguided surfers is more than six times of that of guided surfers. The two guided searches give very similar results, in terms of the mean searching time. The mean searching time in Mean-Path Guidance is longer than that in *DSSP*, because Mean-Path

Guidance does not use any real time information.

4.2 Evaluation

All four models work similarly in terms of providing (or not) navigation guidance for the purpose of information retrieval; however, they differ in terms of applicability and complexity. *Complexity* refers to the performance of the model in the various scenarios, such as conditional reverse links, motivation guidance, decision-making, algorithm complexity, and structure modification. *Applicability* concerns the capability of problems types, such as cache option, multiple root nodes, multiple destination nodes, and concurrency. The following sets out the comparison of the four models based on these two aspects.

A. Complexity

(1) *Conditional reverse links*. The original graph needs to be extended by explorative transformation for the models ISL, ESL and MPG; however, it only adds the reverse links dynamically to the graph for model KFA.

(2) *Navigation guidance*. The first two models, ISL and ESL, do not have the guidance, whereas the other two, MPG and KFA, benefit from the guidance of site structure and loading time for navigation decision.

(3) *Decision-making*. Both ISL and MPG models focus on the static decision-making, i.e. one-pass decision process. Meanwhile, ESL and KFA models concern dynamic information for decision-making.

(4) *Algorithm complexity*. For ISL model, the main task is to compute the first passage time; the complexity for each node is $O(N^2)$ and the total complexity is $O(N^3)$. Because of the state space of the Markov chain for ESL model, the total complexity is $O(N^4 \times 2^N)$. The complexity of MPG model depends on that for Dijkstra's algorithm, which is $O(N^2)$. KFA model cannot be solved in polynomial time unless it is acyclic. If we take into account the conditional reverse links, the complexity of all four models are non-polynomial.

(5) *Structure modification*. The structure modification involves addition or modification of links/nodes. For non-guidance models, ISL and ESL, are not effected by the changes in structure formulation; guidance-based models, MPG and KFA, need to update the guidance information. Meanwhile, static models, ISL and MPG, need to re-compute the solution by taking into account the new information; but dynamic models, ESL and KFA, only consider

the new information during every dynamic decision process.

B. Applicability

(1) *Cache option*. The cache option decides the loading time for the conditional reverse links, which range between 0 (cache enabled) and the loading time of the original node (cache disabled). The cache option (either enabled or disabled) is applicable for all four models. There may be time (life span) constraints on the cache content. These time constraints, however, are only applicable to KFA model.

(2) *Multiple root nodes*. The navigation may start from different root nodes in different sessions. All four models are capable of handling multiple root nodes (without any further modification).

(3) *Multiple destination nodes*. Unlike the case of multiple root nodes, the multiple destinations need to be covered in the same session. In ISL model, navigation is independent from the destination nodes and the calculation is additive as a linear function. Similarly, MPG model can also take the destination nodes as independent in the graph and calculate the result as a linear function. On the contrary, the calculation becomes much more complicated for ESL and KFA models, which might take non-additive calculation as non-linear functions.

(4) *Concurrency*. The concurrency involves both multiple roots and destinations for different session (for different users) simultaneously. Since KFA model needs real-time information for step-by-step decision-making, the multiple sessions might interfere with each other. For the other models (ISL, ESL, and MPG), the sessions can be independent.

The summary of the comparison is listed in Table 2. We may also include other evaluation criteria in order to investigate the impact of adding links on accessibility efficiency of an individual Web page or that of the Web site as a whole. From the comparison result, the suitable models can be adopted and adapted based on the problem properties.

<Insert Table 2 here>

4.3 Implementation Approach

We have illustrated in Section 4.1 that navigation guidance can significantly reduce the searching time of the surfers. The guidance can be implemented in (at least) two forms: (1) an explicit navigation map is provided with the shortest path highlighted for surfers to follow; and (2) sites structure is dynamically generated to suit the needs of surfers. The implementation of the first form is straightforward and its detail is deliberated omitted. We will discuss as follows the implementation issues of the second form.

Refer to Figure 1 for the structure of the site for illustration. Remember that page *A* is the root and page *L* is the destination. The structure indicates the hierarchical arrangement of information: Page *A* contains links to pages *B*, *C*, and *D*, in the order of the links arranged in page *A*; other pages and links are interpreted in the same fashion. Given the known destination page *L* provided by the surfer, the system provides navigation guidance based on the real-time information at the moment when the surfer finishes a page. Suppose that the guidance directs the surfer to page *C*, and, by the same token, next to page *G*. So that for the surfer, the site structure is as if that in Figure 5, all the relevant links being arranged in the most convenient top (right) position for the surfer.

In Section 4.1 above, we use simulation to evaluate the performance of the different guidance methods. In real-life, it is hard to simulate on real time for each surfer to determine his best virtual site structure. Fortunately, there are polynomial time algorithms to calculate the shortest paths. The standard shortest path algorithms for deterministic arc lengths give the shortest path for the Mean-Path Guidance, as long as we set the arc lengths to their mean values. There is a family of similar algorithms to determine the shortest paths with real-time information on path length. In the following, we give the algorithm suggested in Cheung [35].

<Insert Figure 5 here>

Any site structure is an undirected graph as shown in Figure 1. Let (G, N) be such a graph, where G = {*A*, …, *L*} be the set of nodes and N be the set of arcs of the graph. Node *L* is the destination. Let

$S(i)$ be the set of successor nodes of node i;

$B(j)$ be the set of predecessor nodes of node j;

T_{ij} be the (random) cost of arc (i, j); (T_{ij} is the loading time of page j in our application);

$\bar{V}_i$ be the expected distance from i to L.

As stated above, all the arc costs are assumed to be independent. Then

$$\bar{V}_L = 0,$$

$$\bar{V}_i = E[\min_{j\in S(i)}(T_{ij} + \bar{V}_j)], \quad \forall i = A, \ldots, K. \quad (1)$$

Note that T_{ij}'s are known quantities whenever we compute the expected for node i. $\{\bar{V}_i\}$ are found by solving the equation set simultaneously, which can only be done numerically. However, Cheung suggested the following modified generic Label-correction method. It is an approximation in alternate to what we give in Section 3.4 above. Let

Q be the set of nodes (typically in the form of queue) whose distance from node L have been known;

$\hat{V}_i$ be an estimate of $\bar{V}_i$;

$V_i^{\max}$ = maximum value of $\min_{j\in S(i)}(T_{ij} + \hat{V}_j)$.

Step 1. Initialize $\hat{V}_i = \infty$, $\forall$ $i \in$ G, $i \neq L$; $\hat{V}_L$.

Step 2. Initialize Q.

Step 3. Remove a node i from Q and compute $\hat{V}_i = E[\min_{j\in S(i)}(T_{ij} + \hat{V}_j)]$.

Step 4. For each node $j \in B(i)$, if $V_j^{\max} < T_{ji} + \hat{V}_i$ and if $i \notin Q$, then add i to Q.

Step 5. Repeat steps 3 to 4 until Q = ϕ.

The ways to initialize Q (in Step 2) and to add i to Q are implementation specific. See Cheung [35] for an implementation example that reduces computational effort.

5. CONCLUSIONS AND FUTURE DIRECTIONS

With the advent of the Internet technology, the information crawling/gathering on the Web is highly demanded and important in various applications. For example, users need to surf on the Web for sourcing in procurement process. However, dynamic traffic and poorly organized Web pages lead the users to navigate through irrelevant and redundant pages. In this paper, we adopt a stochastic searching approach to provide users with dynamic guidance for information access on the Web. We consider four models: inexperienced surfers on guidance-less sites, experienced surfers on guidance-less sites, sites with the Mean-Path Guidance, and sites with the Known-First-Arc Guidance (which are generalized as sites with *Dynamic Stochastic Shortest Path* Guidance (DSSP)). From the simulation result, we conclude that providing navigation guidance reduces web page search time, and dynamic guidance improves the access performance.

The results provide insights to design and administrate web pages. Navigation guidance can take different forms. They can be explicit information built in as text or images of a page, or implicit information hidden under menu bars and choice options. In all cases the design of the site, from its structure to presentation, is important. For small sites or sites that generally entertain limited number of surfers, the real-time traffic information of the site is not really essential on the access performance. On the other hand, for trans-continental portals that possibly entertain huge number of surfers simultaneously, the real-time information can direct and divert surfers according to needs. The simplest example may be a page directing surfers to different downloading sites according to the real-time utilization and traffic of sites. The realization of the insights gained from this study would be an interesting synergy of web page design and information technology.

We can further extend this research in the following directions:

(1) *Multiple-destination*. The dynamic guidance algorithm needs to be extended to cover a minimum spanning tree, if the user wants to visit multiple pages in a Web site.

(2) *Oscillation avoidance*. Since the dynamic guidance takes into account the traffic condition, we may need to add a "stopping rule" or a learning mechanism to avoid the oscillation in selecting the remaining path.

(3) *Enabled cache*. The traverse information can be kept in the cache for "go-back" function. In this case, we need to add a reverse link with zero length or change its length to zero if it exists in the original graph.

(4) *Dynamic information content*. If the pages are generated dynamically, such as ASP (Active Server Page), we can split/merge the pages and re-organize information links/content.

(5) *Personalized information space*. The dynamic information content can be further extended for personalization that each user will surf on the customized Web information space based on his preference of information and security requirement.

REFERENCES

[1] Perkowitz, M and Etzioni, O. (2000) Adaptive Web Sites. *Communication of ACM*, Vol. 43, No. 8, August.

[2] Perkowitz, M and Etzioni, O. (2000) Towards adaptive Web sites: conceptual framework and case study. *Artificial Intelligence*. Vol. 118, No.1-2, pp. 245-75

[3] Yan, T.W., Jacobsen, M., Garcia-Molina, H., and Dayal, U. (1996) From user access patterns to dynamic hypertext linking. *Computer Networks & ISDN Systems*, vol.28, no.7-11, pp.1007-14.

[4] Wang, Z., Siew, C.K., and Yi, X. (2000) A new personalized filtering model in Internet Commerce, *Proceedings of SSGRR (Scuola Superiore G. Reiss Romoli)*, Rome, Italy.

[5] Cheung, D.W., Kao, B. and Lee, J. (1998) Discovering user access patterns on the World Wide Web. *Knowledge-Based Systems*, vol.10, no.7, pp.463-70.

[6] Chen, P.M and Kuo, F.C. (2000) An Information Retrieval System based on User Profile. *The Journal of System and Software*, vol. 54, pp3-8.

[7] Chang, C.H., Hun, C.C. and Hou, C.L. (1998) Exploiting hyperlinks for automatic information discovery on the Web. *Proceedings of 10th IEEE International Conference on Tools with AI*, pp.156-63.

[8] Yang, C.C., Yen, J. and Chen, H. (2000) Intelligent Internet Searching Agent Based on Hybrid Simulated Annealing. *Decision Support Systems*. Vol.28 pp.269-277

[9] Tu, H.C. and Hsiang, J. (2000) An Architecture and Category Knowledge for Intelligent Information Retrieval Agents. *Decision Support Systems*. Vol. 28, pp.255-268

[10] Teng, S.-H., Lu, Q., Eichstaedt, M., Ford, D. and Lehman, T. (1999), Collaborative team crawling: information gathering/processing over Internet, *Hawaii International Conference on System Sciences: HICSS32*.

[11] Joachims, T., Freitag, D. and Mitchell, T. (1997) WebWatcher: A Tour Guide for the World Wide Web, *Proceedings of IJCAI-97*, Nagoya, Janpan, pp770-775

[12] Liaberman, H. (1995) Letizia: an agent that assists Web browsing. *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI-95)*. vol. 1, pp.924-929

[13] Berghel, H., Berleant, D., Foy, T., and McGuire, M. (1999) Cyberbrowsing: information customization on the Web. *Journal of the American Society for Information Science*, vol.50, no.6, pp.505-13

[14] Lin, I.-Y., Huang, X.M. and Chen, M.S. (1999) Capturing user access patterns in the Web for data mining. *Proceedings 11th International Conference on Tools with Artificial Intelligence*, pp.345-8.

[15] Richardson, O. (2000) Gathering accurate client information from World Wide Web sites. *Interacting with Computers*. Vol.12, no.6, pp.615-22.

[16] Gibson, D., Kleinberg, J. and Raghavan, P. (1998) Structural Analysis of the World Wide Web. *WWW Consortium Web Characterization Workshop*, November.

[17] Chakrabarti, S., Dom, B., Kumar, S.R., Raghavan, P., Rajagopalan, S., Tomkins, A., Kleinberg, J.M. and Gibson, D. (1999) Hypersearching the Web, *Scientific American*, June.

[18] Donato, Debora, Leonardi, Stefano, Millozzi, Stefano, and Tsaparas, Panayiotis (2005) Mining the Inner Structure of the Web Graph, *8th International Workshop on the Web and Databases (WebDB)*, June 16-17, 2005, Baltimore, Maryland.

[19] Chakrabarti, S., Dom, B., Kumar, S.R., Raghavan, P., Rajagopalan, S., Tomkins, A., Gibson, D. and Kleinberg, J.M. (1999) Mining the Web's Link Structure. *IEEE Computer*, 32(8): 60-67

[20] Eiron, Nadav, and McCurley, Kevin S. (2004) Links in Hierarchical Information Networks, *Lecture Notes in Computer Science*, Vol. 3243, pp. 143-155, Springer.

[21] Yen, B.P.-C (2006). The Design and Evaluation of Accessibility on Web Navigation, *Decision Support Systems*, Vol 42/4, pp 2219-2235.

[22] Kleinberg, J., Kumar, S.R., Raghavan, P., Rajagopalan, S. and Tomkins, A. (1999) The Web as a graph: Measurements, models and methods. *International Conference on Combinatorics and Computing*.

[23] Sarukkai, R.R. (2000). Link Prediction and Path Analysis using Markov Chains. *Computer Network*, vol. 33 pp377-386.

[24] Levene, M., Borges, J. and Loizou, G. (2001) Zipf's law for web surfers. *Knowledge and Information Systems an International Journal*, 3, 120-129.

[25] Levene, M. and Loizou G. (1999) A probabilistic approach to navigation in Hypertext. *Information Sciences*, 114, 165-186.

[26] Levene, M. and Loizou G. (1999) Navigation in Hypertext is easy only sometimes. *SIAM Journal on Computing*, 29, 728-760.

[27] Kumar, R., Rajagopalan, S., Sivakumar, D., Tomkins A. and Upfal, E. (2000) Stochastic models for the web graph. *Proceedings of the IEEE Symposium on Foundations of Computer Science*.

[28] Levene, M., Fenner, T., Loizou, G. and Wheeldon, R. (2002) A stochastic model for the evolution of the web. *Condensed Matter Archive*, cond-mat/0110016 v2.

[29] Zin, N. and Levene, M. (1999) Constructing web views from automated navigation sessions. In *ACM Digital Library WOWS*, Berkeley, Ca., August, pp. 54-58.

[30] Glover, F., Klingman, D. and Phillips, N. (1985) A New Polynomially Bounded Shortest Path Algorithm, *Operations Research*, 33, 65-73.

[31] Shier, D. and Witzfall, C. (1981) Properties of Labeling Methods for Determining Shortest Path Trees, *Journal* of *Research* of the *National Bureau* of Standards, 86, 317-330.

[32] Psaraftis, H.N. and Tsitsiklis, J.N. (1993) Dynamic Shortest Path in Acyclic Networks with Markovian Arc Costs, *Operations Research*, 41, 91-101.

[33] Geetha, S. and Nair, K.P.K. (1993) On Stochastic Spanning Tree Problem, *Networks*, 23 675-679.

[34] Polychronopoulos. G. H. and Tsitsiklis, J.N. (1996) Stochastic Shortest Path Problems with Recourse, *Networks*, 27, 133-143.

[35] Cheung, R. K. (1998) Iterative Methods for Dynamic Stochastic Shortest Path Problems, *Naval Research Logistics*, 45, 769-789.

[36] Wolff, R. W. (1989) *Stochastic Modeling and the Theory of Queues*, Prentice-Hall, New Jersey.

[37] Ahuja, R.K., Magnanti, T.L. and Orlin, J.B. (1993) *Network Flows - Theory, Algorithms, and Applications*, Prentice-Hall International, New Jersey.

Table 1. Simulation Results

Models	Mean Searching Times[*1]
Inexperienced surfer on guidance-less Sites (ISL)	161.50
Experienced surfer on guidance-less Sites (ESL)	73.47
Mean-Path Guidance (MPG)	12.01
Known-First-Arc or DSSP guidance (KFA)	11.95

[*1] The loading time of page *A* for the first time is excluded.

Table 2. Summary of comparison for searching models

	ISL	ESL	MPG	KFA
Complexity				
Conditional reverse links	Extend graph	Extend graph	Extend graph	Add links
Navigation guidance	No guidance	No guidance	Guidance	Guidance
Decision-making	Static	Dynamic	Static	Dynamic
Algorithm complexity *[1]	Polynomial	Non-polynomial	Polynomial	Non-polynomial
Structure modification	Re-compute	No changes	Update/ Re-compute	Update
Applicability				
Cache option *[2]	Not applicable	Not applicable	Not applicable	Applicable
Multiple roots	Applicable	Applicable	Applicable	Applicable
Multiple destinations	Linear	Non-Linear	Linear	Non-linear
Concurrency	Independent	Independent	Independent	Dependent

[*1] If taking into account conditional reverse links, the complexity becomes non-polynomial for all four models.
[*2] In the case of time-constraint cache option

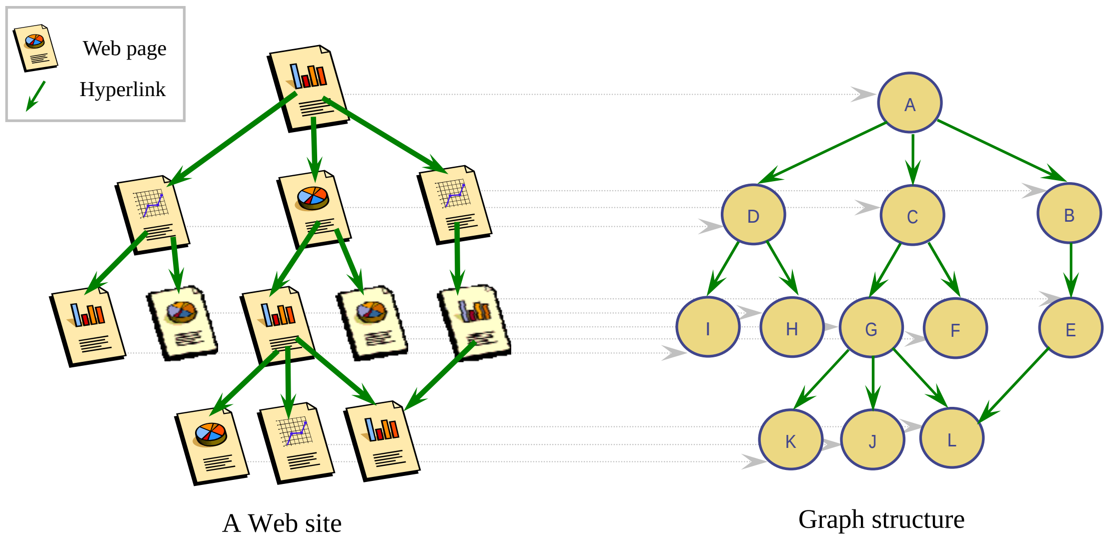
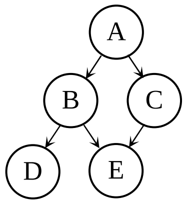
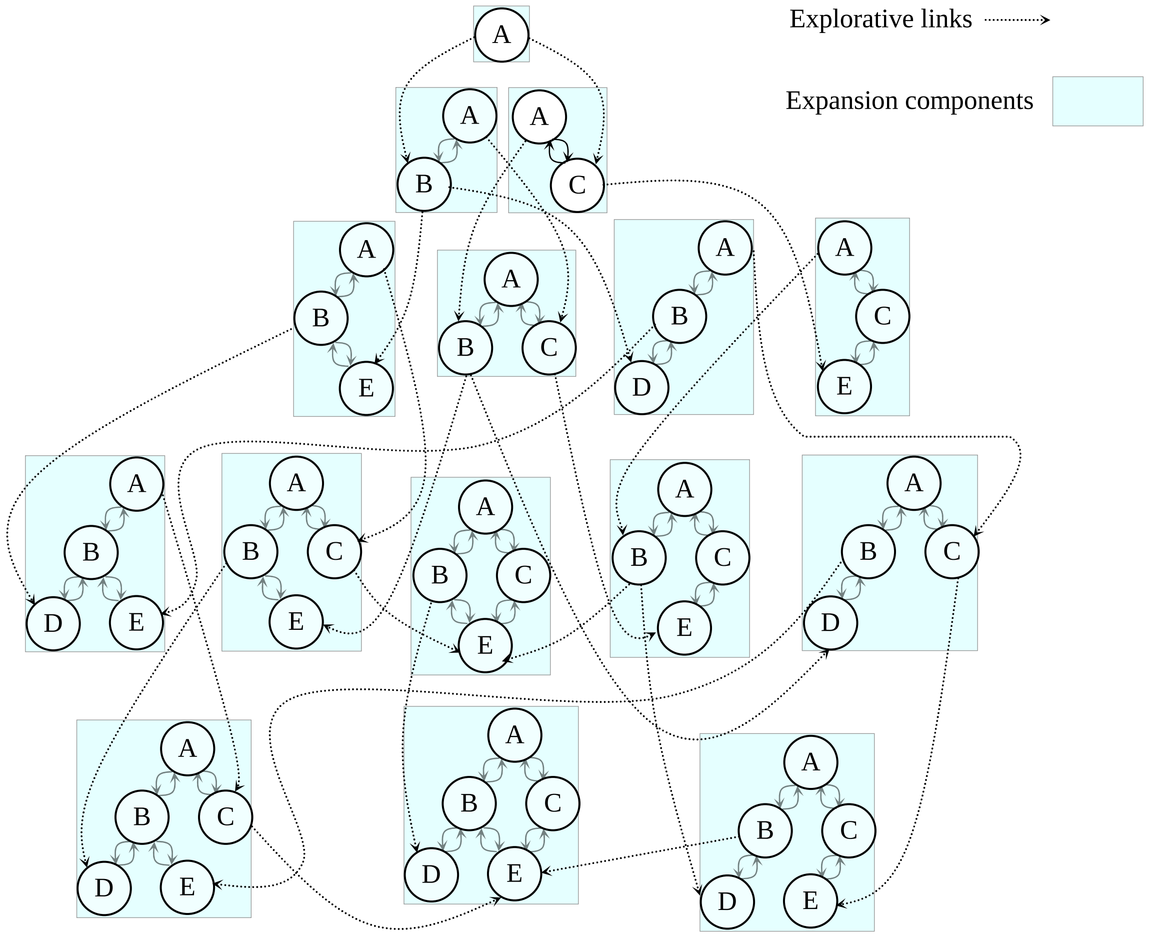


Figure 1 Example – a Web site and its graph structure



(a) Original graph



(b) Expansion graph

Figure 2 Example of explorative expansion

Navigation Guidance / *Repeatable Navigation*	**No**	**Yes**
No	**ESL** Experienced surfers on guidance-less sites	**MPG** Sites with the mean-path guidance
Yes	**ISL** Inexperienced surfers on guidance-less sites	**KFA** Sites with the known-first-arc guidance

Figure 3 Classification of the four models

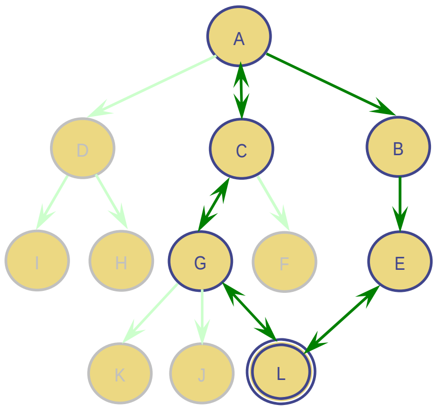


Figure 4 Cycling in *DSSP* Guidance

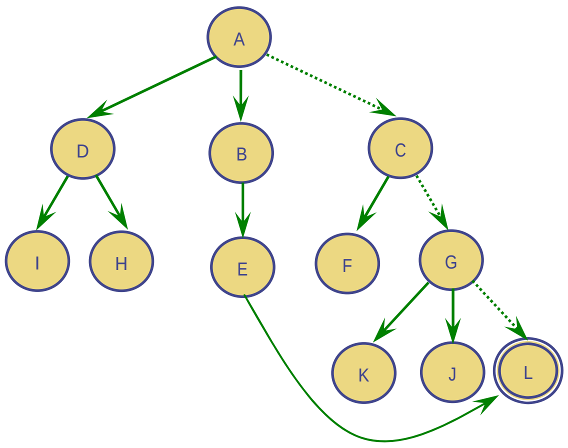


Figure 5 The Virtual Site Structure Experienced by a Surfer