An automaton model of user-controlled navigation on the web

K. Lodaya and R. Ramanujam

The Institute of Mathematical Sciences
C.I.T. Campus, Chennai 600 113, India.
E-mail: {kamal,jam}@imsc.ernet.in

Abstract

We present a simple theoretical model of web navigation, in which each WWW user creates *style specifications* which constrain web browsing, search and navigation using the user's own judgement of the quality of visited sites. A finite state automaton is associated with each specification, which is presented in a two-level modal logic making up the acceptance condition for the automaton. We show that many interesting queries regarding the user's web search can be answered using standard automata theory.

1 Motivation

A classical application of automata theory is searching a given text for occurrence of patterns. Finite state automata are employed in lexical analysers and document processors for pattern search. Recently, the world of documents has grown very fast, thanks to the World Wide Web (WWW), where we do not speak of text, but **hyper**text. Here documents contain pointers or *links* to documents elsewhere. However, access to documents is not only via these links but also using *search engines* which, given a keyword, locate a document containing the keyword. In such a situation, a natural application of automata theory seems to be that of web navigation.

On the other hand, we could argue that the distributed nature of information on the net is irrelevant and we can consider the entire web as one single huge piece of text. Search engines like Alta Vista or Google in effect do that, act as finite state automata that access the entire web when they search for keywords. In this manner, one can argue that there is no need for any new automata theory for searching the web, but only cleverer methods for data representation.

While there is some measure of truth in this argument, we believe that there is need for automata theory in this context from a slightly different viewpoint, that of quality controlled navigation. Quality remains the biggest challenge of search engines [Hen99], and there is a perceived urgent need for ways by which web users can direct and control navigation based on *their own judgement* of quality of web sites.

In this paper, we present a simple (naïve) theoretical model of user controlled navigation on the web. In this model, each user prepares a style sheet in which she specifies her judgement of sites based on the information they contain, and sets down constraints that every navigation must satisfy. The style sheet induces a finite state automaton that can be thought of as her own navigation-cum-search engine, which uses some general purpose engine like Alta Vista to access the web. These automata can be seen as customized front-ends to standard search engines. Formally, they are finite state transition systems, and their behaviour is given by partitioning the set of possible runs (or *trajectories*, as they are called here) into good and bad runs based on the constraints specified in the style sheet.

We then have a situation where the input to an automaton is given as a finite graph whose nodes carry input information, and the finite state control dictates which nodes to visit, in what order. It defines a process which originates from a home location, migrates to various nodes on the web picking up information on the way and returns home eventually.

The main contribution of the paper is a framework for the theoretical study of quality controlled web navigation. The framework is built on standard concepts and hence offers hope for implementation. It must be emphasized that the paper presents only the initial steps of a theoretical framework, and it is hoped that more realistic models will follow. However, design features of such models (like whether links should be static or dynamic, bookmarks implicit or explicit, search distributed or not, whether transitive closure of paths on the web is needed) should be decided by empirical compulsions rather than theoretical elegance. Corpus studies like [SHMM98] should dictate how this theory should proceed.

Related work: User-controlled navigation has been studied by several researchers from the WWW community, under the name of *spiders, crawlers, robots, knowbots* and so on. Miller and Bharat [MB98] give a kind of style specification using C++. Mendelzon, Mihaila and Milo [MMM96] modify the query language SQL to operate on the web. Pazzani, Muramatsu and Billsus [PMB96] describe a system which piggybacks onto a Lycos browser.

However, the use of automata theory for modelling these systems appears to be new. Automata working on graph structures have been studied for long: the important topic of visiting all sites in a labyrinth is surveyed in [KUK93]. Thomas [Tho97] has recently emphasized the importance of studying automata on partially ordered structures. All these papers concentrate on the class of graphs accepted/traversed by such automaton models; but, as we will see, the approach followed here is distinct. Even if we were to consider the Web as graph input to the automaton, due to the keyword search facility, the automaton is not obliged to follow the paths in the graph.

2 The model

In this paper, automata behave in a somewhat non-standard fashion: there is one single finite graph processed by all automata. The behaviour of an automaton is some set of trajectories in this graph (as opposed to a set of graphs). Each node of the graph is supposed to contain some information, represented by a finite set of letters from a fixed alphabet. The finite state control moves on the graph not linearly, but according to a specified constraint, either using an edge or using a *search*; in general the automaton does not process the entire input graph, but its behaviour is given by trajectories that take it from a designated home location back again to the home location visiting nodes as given by a *navigation constraint*.

The input graph is meant to represent the internet, with URLs as nodes and links as the edge relation. Each automaton is thought of as a *search strategy* or a search engine defined by a user whose home location on the net defines the start node for the automaton. Every run of the automaton navigates the net in some way. The user controls this navigation by specifying requirements on the **quality** of information in accessed sites. Therefore, the (static) description of the net comes with a specification of information available at each node, and each user automaton includes a quality judgement of sites, based on information they contain. We use letters of a finite **information alphabet** to abstractly associate keywords with each node, and a set of propositional letters to denote quality judgement.

Which locations in the net are accessible to any specific user ? In principle, all. However, if the user can access arbitrary locations only by using a search engine, then the vocabulary used to query the engine effectively limits the range of locations accessed. This is reflected in the model below.

2.1 Navigators

A **web** is a tuple $W = (U, L, I, \iota)$, where U is a finite set of locations (URLs), $L \subseteq (U \times U)$ is the link relation, I is a finite information alphabet, $\iota : U \to 2^I$ is the content map.

We use $u, v, \ldots$ to refer to elements of U and $a, b, \ldots$ to refer to elements of I. Below, we will define a logical language in which we can talk about specific locations, connectivity between locations and navigation paths on the net. Let $P = \{p_0, p_1, \ldots\}$ be a countable set of propositional letters. The set of all formulas of the logic is denoted Φ, and we use ϕ, ϕ' etc to refer to formulas in Φ. P_ϕ denotes the set of propositions occurring in ϕ. For the rest of this section, fix a web $W = (U, L, I, \iota)$.

A **navigator** on W is a tuple $N_W = (h, Voc, \chi, \phi)$, where $h \in U$; Voc, the **vocabulary** of N_W, is a nonempty subset of I such that $\iota(h) \subseteq Voc$; $\chi : 2^{Voc} \to 2^{P_\phi}$ is a quality map and $\phi \in \Phi$. $U(N_W) \stackrel{\text{def}}{=} \{u \in U \mid (\iota(u) \cap Voc) \neq \emptyset\}$, is said to be the set of locations visible to N_W. $Val(N_W) : U(N_W) \to 2^{P_\phi}$ is the induced valuation defined by: $Val(N_W)(u) = \chi(\iota(u) \cap Voc)$.

h is the home location of the navigator N_W. Note that h is always visible to N_W. χ is a map that indirectly assigns judgements to locations in U, which is given by the induced valuation map.

Let $N_W = (h, Voc, \chi, \phi)$ be a navigator on W. The **subnet visible to** N_W is given by: $W' = (U', L', Voc, \iota')$, where $U' = U(N_W)$, $L' = L \cap (U' \times U')$ and $\iota' : U' \to 2^{Voc}$ is defined by: $\iota'(u) = \iota(u) \cap Voc$. W' is denoted as $W\lceil N_W$.

A **trajectory** of N_W is a *finite* sequence $\rho \in (U(N_W))^*$ of the form $u_0 u_1 \cdots u_K$, $K \geq 0$, where $u_0 = u_K = h$. Thus a trajectory is an itinerary of locations visited on the net. We do not explicitly represent bookmarking visited locations; as we will see below, the navigation constraint ϕ can refer to sites that have been visited already, and this provides an implicit way of referring to bookmarks in trajectories.

Let $Tr(N_W)$ denote the set of all trajectories of N_W. The **behaviour** of N_W is given by the set $Beh(N_W) \stackrel{\text{def}}{=} \{\rho \in Tr(N_W) \mid \rho \models \phi\}$, where the relation $\rho \models \phi$ is defined below. Thus, we need to give the syntax and semantics of the logic to complete the definition of the model.

The formulas of the logic are presented in two layers. The lower layer consists of **location formulas**, and their syntax is given as follows:

$$\Gamma ::= p \in P \mid \neg \alpha \mid \alpha_1 \vee \alpha_2 \mid \langle from \rangle \alpha \mid \langle to \rangle \alpha$$

The other logical connectives $\wedge$, $\supset$, $\equiv$ are defined as usual. The dual modalities are given by: $[from]\alpha \stackrel{\text{def}}{=} \neg\langle from\rangle\neg\alpha$ and $[to]\alpha \stackrel{\text{def}}{=} \neg\langle to\rangle\neg\alpha$.

The syntax of **navigation formulas** is given as follows, where $\alpha \in \Gamma$:

$$\Phi ::= \alpha \mid \neg\phi \mid \phi_1 \vee \phi_2 \mid \triangledown\alpha \mid \triangleright \phi \mid \alpha?\phi \mid \Diamond\phi$$

Location formulas specify how locations with specific quality of information are interlinked in the net, with $\langle from\rangle\alpha$ asserting the existence of a link from a node satisfying α to the current node, and $\langle to\rangle$ asserting the other way about. $\triangledown\alpha$ refers to an implicit bookmark to a site where α holds; it says that such a site has been visited in the trajectory at this stage, and hence that such information is available to the user. $\triangleright\phi$ asserts that the navigation given by ϕ continues along one of the links available at the current location (by a 'click'), whereas $\alpha?\phi$ directs a *search* for a location satisfying α, from where the navigation ϕ continues. $\Diamond$ represents eventuality, and its dual is given by $\Box\phi \stackrel{\text{def}}{=} \neg\Diamond\neg\phi$.

Let $N_W = (h, Voc, \chi, \phi)$ be a navigator, $W' = (U', L', Voc, \iota')$ the subnet of W visible to N_W and $Val(N_W)$ the valuation induced by N_W. Location formulas are interpreted over locations of W'. $W', u \models_l \alpha$ denotes that α holds in location u in net W'.

- $W', u \models_l p$ iff $p \in Val(N_W)(u)$.
- $W', u \models_l \neg\alpha$ iff $W', u \not\models_l \alpha$.
- $W', u \models_l \alpha \vee \beta$ iff $W', u \models_l \alpha$ or $W', u \models_l \beta$.
- $W', u \models_l \langle from\rangle\alpha$ iff there exists u' such that $(u', u) \in L'$ and $W', u' \models_l \alpha$.
- $W', u \models_l \langle to\rangle\alpha$ iff there exists u' such that $(u, u') \in L'$ and $W', u' \models_l \alpha$.

Let $\rho = u_0 u_1 \cdots u_K \in Tr(N_W)$. For $\phi \in \Phi$ and $k \in \{0, \ldots, K\}$, the notion $\rho, k \models \phi$ is again defined inductively.

- $\rho, k \models \alpha$, for $\alpha \in \Gamma$, iff $W', u_k \models_l \alpha$.
- $\rho, k \models \neg\phi$ iff $\rho, k \not\models \phi$.
- $\rho, k \models \phi_1 \vee \phi_2$ iff $\rho, k \models \phi_1$ or $\rho, k \models \phi_2$.
- $\rho, k \models \triangledown\alpha$ iff there exists $m : 0 \leq m \leq k : W', u_m \models_l \alpha$.
- $\rho, k \models \triangleright\phi$ iff $k < K$, $(u_k, u_{k+1}) \in L'$ and $\rho, k+1 \models \phi$.
- $\rho, k \models \alpha?\phi$ iff $k < K$, $W', u_{k+1} \models_l \alpha$ and $\rho, k+1 \models \phi$.
- $\rho, k \models \Diamond\phi$ iff there exists $m : k \leq m \leq K$ such that $\rho, m \models \phi$.

We use the notation $\rho \models \phi$ when $\rho, 0 \models \phi$. This completes the definition of navigator N_W. Note that formulas do not directly refer to the information contained in locations, but only to the quality of such information, as given by χ. $\alpha?\phi$ is more natural here, but we could have formulas $w?\phi$, where w is a keyword in Voc, without affecting our results.

3 Analysis

Given a web W and a navigator N_W on it, we can analyse W' and N_W to answer a number of queries regarding the 'search engine' given by N_W. The following queries seem natural: Does the navigation meet some basic quality requirement ? Is a site with some specific information visited at all ? Is it the case that every visited site of some quality has links only to sites having the same quality of information ? Is the navigation constraint consistent ? That is, is $Beh(N_W) \neq \emptyset$?

Many of these and other related questions can be answered by a simple technique [VW86]: construct a verifying automaton (an ordinary NFA) which captures the behaviour of the given navigator, and run standard algorithms on that automaton. Below we describe how we can associate such an automaton with every navigator.

Define as usual the subformula closure of any formula $\phi \in \Phi$, denoted $CL(\phi)$. The size of $CL(\phi)$ is linear in the length of ϕ. Let $\triangleright_N$ denote the set of all formulas in $CL(\phi)$ of the form $\triangleright\psi$. Similarly let $?_N$ contain all $\alpha?\psi$ formulas in $CL(\phi)$. Let $Nxt \stackrel{\mathrm{def}}{=} \triangleright_N \cup ?_N$.

Let $A \subseteq CL(\phi)$. A is said to be an atom iff it satisfies the following conditions:

- For all $\neg\psi \in CL(\phi)$, $\neg\psi \in A$ iff $\psi \notin A$.
- For all $\phi_1 \vee \phi_2 \in CL(\phi)$, $\phi_1 \vee \phi_2 \in A$ iff $\phi_1 \in A$ or $\phi_2 \in A$.
- $A \cap ?_N = \emptyset$ or $A \cap \triangleright_N = \emptyset$.
- If $A \cap Nxt = \emptyset$ then for all $\Diamond\psi \in A$, $\psi \in A$.

Assume that we are given a web $W = (U, L, I, \iota)$, and navigator $N_W = (h, Voc, \chi, \phi_0)$ on it. χ would be presented as a list of boolean formulas on Voc, one for each $p \in P_{\phi_0}$.

Let $|\phi_0| = m_0$. Let CL denote the subformula closure of ϕ_0. Fix an ordering $\preceq$ of CL such that if $|\phi_1| \leq |\phi_2|$ then $\phi_1 \preceq \phi_2$. Let $CL(i)$ denote the i^{th} formula in the enumeration, and let all formulas from Γ precede those from $\Phi - \Gamma$.

Step 1: Construct $W\lceil N_W = (U', L', Voc, \iota')$. This construction, which takes $O(|U|)$ time in the worst case, depends crucially on how ι is represented. Let $M = |U'|$. Fix an enumeration of locations in U'.

Step 2: Construct a $|(CL \cap \Gamma)| \times M$ boolean array μ such that $\mu(i, j) = 1$ iff $W', u \models_l \alpha$, where $CL(i) = \alpha$ and u is the j^{th} location in the enumeration of W'. (We will use the notation $\mu(\alpha, u) = 1$ in this case.) This construction can be done in $O(M.m_0)$ time. However, we can identify locations as follows: $u_1 \sim u_2$ iff for all $\alpha \in (CL \cap \Gamma)$, $\mu(\alpha, u_1) = \mu(\alpha, u_2)$. Thus we only need a boolean $m_0 \times 2^{m_0}$ array. Let $\mu(\alpha, [u])$ represent these elements.

Step 3: Let AT denote the set of all subsets of CL which are atoms. Let $Vis \stackrel{\text{def}}{=} \{\alpha \in \Gamma \mid \triangledown\alpha \in CL(\phi)\}$. The states of our verifying automaton $\mathcal{A}_N \stackrel{\text{def}}{=} (Q, \Rightarrow, I, F)$ keep track of these formulas in addition to the URLs visited:

- $Q = \{([u], A, S) \mid u \in L', A \in AT,\ S \subseteq Vis$ and for all $\alpha \in \Gamma$: $\alpha \in A$ iff $\mu(\alpha, [u]) = 1$ and if $\triangledown\alpha \in A$ then $\alpha \in S\}$.
- $I = \{([h], A, A \cap Vis) \mid A \in AT, \phi_0 \in A\}$.
- $F = \{([h], A, S) \mid A \in AT$ such that $A \cap Nxt = \emptyset\}$.
- $([u], A, S) \Rightarrow ([v], B, S')$ iff
  1. $A \cap Nxt \neq \emptyset$,
  2. if $\triangleright\phi \in A$ then $\phi \in B$ and $([u], [v]) \in [L']$,
  3. if $\alpha?\phi \in A$ then $\mu(\alpha, [v]) = 1$ and $\phi \in B$,
  4. if $\Diamond\phi \in A$ and $\phi \notin A$ then $\Diamond\phi \in B$, and
  5. $S' = S \cup (B \cap Vis)$.

For $q = ([u], A, S) \in Q$, we use the notation $q\lceil U$ to denote u. Let $\rho = u_0u_1 \ldots u_k \in U^*$. We say that ρ is **accepted** by A_N iff there exists a sequence $q_0q_1 \ldots q_k$ such that $q_0 \in I, q_k \in F$, and for all $i : 0 \leq i < k$, $q_i \Rightarrow q_{i+1}$ and $\rho = q_0\lceil U \ldots q_k\lceil U$. Let $Lang(\mathcal{A}_N)$ denote the set of all strings in U^* accepted by $\mathcal{A}_N$.

Theorem 3.1 *If $\mathcal{A}_N$ is the automaton associated with navigator N_W on web W, then $Beh(N_W) = Lang(\mathcal{A}_N)$.*

The construction takes $O(|U|) + O(M.m_0) + 2^{c.m_0}$, that is, $O(|U|) + 2^{c.m_0}$ time, where c is a (small) constant. We can then answer queries like the ones listed in the beginning of this section. Since the number of URLs in the web is large, the fact that the $O(|U|)$ factor is additive is crucial. It can then be seen as a preprocessing step for the algorithm. $M = |U'|$ is usually much smaller, and hence the multiplicative factor is more acceptable.

After elimination of useless states and transitions, we can have an efficient representation of $\mathcal{A}_N$ that helps us do reachability analysis, enumerate connected components, and so on. Then, checking whether a location is visited, or whether there exists a trajectory visiting all locations in a given set, is simple.

4 A web example

An example researcher who looks for grants dealing with work on genomes is considered by Pazzani, Muramatsu and Billsus [PMB96]. We consider a style sheet for such a researcher.

The web $W = (U, L, I, \iota)$ is given, and a relevant portion of the structure (U, L, ι) is shown in Figure 1. The style sheet is given by $N_W = (\texttt{Home}, Voc, \chi, \phi)$, where $Voc = \{$*grant, genome, institute, foundation, project, experiment, cloning, publist, appform*$\}$, and $I = Voc \cup \{$*chromosome-22, sex*$, \ldots\}$.

The set of propositions consists of $\{$*commercial, government, authority, hub, spam, submit, interesting, highstatus, money,* $\ldots\}$. For our example, we assume the only commercial site is `Gene Inc` and the only government site is `DBT`. *Authority* and *hub* [Hen99] refer to a URL having high in- and out-degree respectively. These are indicated in the figure by a few arrows coming into and going out of a node, respectively. *Spam*, *submit*, *interesting* are true in URLs which have *sex*, *appform* and $\{$*grant, genome*$\}$ respectively.

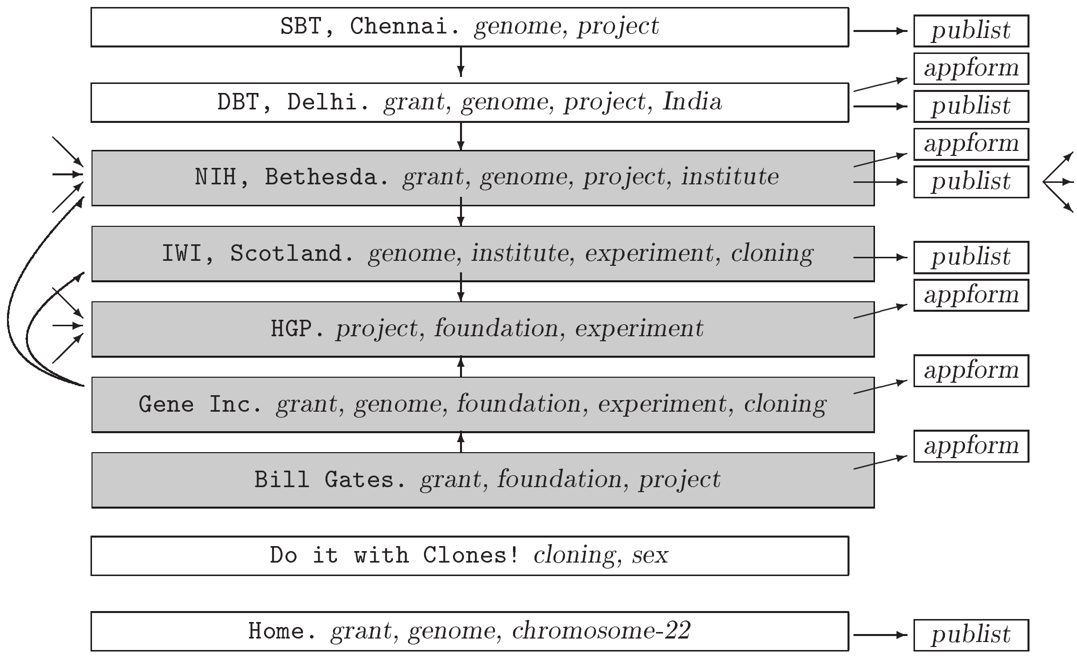

Figure 1: Grants and genomes on the web

The two propositions listed last decide parameters of interest to our researcher: whether the URL represents a place which commands status and whether it has money. Sites satisfying *highstatus* are shown shaded in the figure: the proposition is true in those sites which represent a foundation, or which do research in cloning, or which have a large list of publications (that is, a link to a site with keyword *publist* which satisfies *hub*). *Money* is true in the two government and commercial sites, as well as in all the sites having the keyword *foundation*.

This completes the (indirect) definition of the quality function. We now come to the navigation part. The navigation constraint ϕ is defined as the conjunction of formulas $\phi_i, i \in \{1, \ldots, 5\}$ given below.

$\phi_1 = \Box\neg spam$ eliminates junk sites.

$\phi_2 = \Box(\neg highstatus \supset \triangledown highstatus)$ ensures that a *highstatus* site is visited first.

$\phi_3 = \Box(authority \supset \triangledown\neg authority)$ similarly prioritizes sites which are less well known (and presumably have fewer contenders for grants).

$\phi_4 = \Box(\langle to\rangle submit \supset \triangleright submit)$ visits a link containing an application form.

$\phi_5 = interesting \wedge money?\Diamond(interesting \wedge money?\Diamond(interesting \wedge money))$ visits three sites, according to the user's rating and priorities, which are *interesting* and have *money*.

Having constructed the automaton, the user can now ask questions of interest: Is `Gene Inc` on a particular trajectory? Is there a trajectory visiting all interesting sites with money? and so on.

5 Discussion

The framework presented here is intended only as preliminary. As mentioned in Section 1, the design of the logical language should be guided more by application requirements than theoretical considerations. However, some simple extensions can be made without drastically altering the current framework.

- $\rho, k \models \phi_1 \mathbf{B} \phi_2$ iff if there exists $m \geq k$ such that $\rho, m \models \phi_2$, then there exists $l : k \leq l < m$ such that $\rho, l \models \phi_1$. (This allows priority in visiting locations.)
- $\rho, k \models \lhd\phi$ iff $0 < k < K$, $u_{k-1} = u_{k+1}$ and $\rho, k+1 \models \phi$. (Back button.)
- $\rho, k \models \unrhd\phi$ iff there exists $m \geq 0$ such that $k + m \leq K$, $\{u \mid (u_k, u) \in L\} = \{u_{k+1}, \ldots, u_{k+m}\}$ and and $\rho, k+m \models \phi$. (Visit all local edges.)
- $\rho, k \models \mathbf{E}\alpha$, iff $\{u \mid W', u \models_l \alpha\} \subseteq \{u_0, u_1, \ldots u_k\}$. (Exhaustive search.)
- $\rho, k \models \alpha??\phi$, iff $k < K$, $W', u_{k+1} \models_l \alpha$, $u_{k+1} \notin \{u_0, \ldots, u_k\}$ and $\rho, k+1 \models \phi$. (New location.)

The first two specifications can be easily incorporated into the automaton construction without increasing the complexity. The third case is rather tricky: the surprise is that it can be carried out without increasing the complexity. For the last two modalities, the automaton construction is easy to modify, but the cost increases dramatically. In the states of $\mathcal{A}_N$, we need to carry the set of u-equivalence classes visited, so that the check for exhaustive visit or new visit can be made. Because of this, the construction becomes doubly exponential in the size of the navigation constraint in N_W. However, if we were interested only in the construction of $\mathcal{A}_N$ with the property that $Beh(N_W) \neq \emptyset$ iff $Lang(\mathcal{A}_N) \neq \emptyset$, this could be done in singly exponential time.

There are other interesting and desirable extensions which are not minor and require reworking the theory. For instance, there is a good reason to want quantification over locations, or over information objects. Extending the framework to include page structure within URLs is easy though rather messy; a different formulation may be more convenient.

Another direction is explicit representation of *actions* in the framework, like bookmarking locations, downloading specific information objects, etc. An even more challenging step is to study dynamic nets, where links are made and broken during the course of navigation. This would lead to the consideration of several navigators cruising the web simultaneously, perhaps exchanging information among themselves.

Such a picture raises the obvious question of **information security** which is not modelled here at all. The style sheets envisioned here will be realistic only if security considerations become part of navigation specifications.

References

[Hen99] M. Henzinger: Tutorial on Web Algorithms, joint session of *FST & TCS* and *ISAAC*, Chennai (1999).

[KUK93] V.B. Kudryavstsev, Sh. Ushchumlich and G. Kilibarda: "On behaviour of automata in labyrinths", *Discrete Math. Appl.* **3**,1 (1993) 1–28.

[MMM96] A.O. Mendelzon, G.A. Mihaila and T. Milo: "Querying the world wide web", in *Symposium on Parallel & Distributed Information Systems* (1996).

[MB98] R.C. Miller and K. Bharat: "Sphinx: a framework for creating personal, site-specific web crawlers", in *Proceedings of the 7th International World Wide Web Conference*, Brisbane (1998).

[PMB96] M. Pazzani, J. Muramatsu and D. Billsus: "Syskill & Webert: identifying interesting web sites", in *Proceedings of the 13th National Conference on AI*, Portland (1996). Revised version in *Machine Learning* **27** (1997).

[SHMM98] C. Silverstein, M. Henzinger, H. Marais and M. Moricz: "Analysis of a very large Alta Vista query log", *SRC Technical Note* **1998-014**, `http://www.research.digital.com/SRC/`.

[Tho97] W. Thomas: "Automata theory on trees and partial orders", in *Proceedings of TAPSOFT*, *LNCS* **1214** (1997) 20–38.

[VW86] M.Y. Vardi and P. Wolper: "An automata-theoretic approach to automatic program verification", *Proceedings of the 1st LICS Conference*, Boston (1986) 332–334.