社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2003－ＩＣＳ－133（20）
2003／9／15

Future View: Web browsing assistant based on classifier systems

Norikatsu NAGINO† and Seiji YAMADA††

† CISS, IGSSE, Tokyo Institute of Technology 4259 Nagatsuta, Midori, Yokohama, 226–8502 Japan
†† National Institute of Informatics 2–1–2 Hitotsubashi, Chiyoda, Tokyo, 101–8430, Japan

Abstract In this paper, we propose a Future View system that assists user's usual Web browsing. The Future View will prefetch Web pages based on user's browsing strategies and present them to a user in order to assist Web browsing. To learn user's browsing patterns, the Future View uses two types of learning classifier systems: a content-based classifier system for contents change patterns and an action-based classifier system for user's action patterns. The results of learning is applied to crawling by Web robot, and gathered Web pages are presented to a user through a Web browser. We experimentally show effectiveness of navigation using the Future View.

Key words web navigation, classifier system, browsing pattern

Future View: クラシファイアシステムに基づくWebブラウジング支援

棚野 憲克† 山田 誠二††

† 東京工業大学 〒226–8502 横浜市緑区長津田町 4259
†† 国立情報学研究所 〒101–8430 東京都千代田区一ツ橋 2–1–2
E-mail: †{nagino,seiji}@nii.ac.jp

あらまし 本論文では，ユーザの日常的なブラウジングタスクを支援するシステム，Future View を提案する．Future View は，ユーザのブラウジング戦略に基づいた Web ページの先読みを行ない，ユーザに提示し，ブラウジングを支援する．Future View では，Web ページのコンテンツ情報に基づくブラウジングパターンと，ユーザの行動に基づくブラウジングパターンを２種類のクラシファイアシステムを用いて学習する．本論文では，Future View の設計を説明し，実験によりその効果と学習されるユーザのブラウジングパターンについて考察する．

キーワード Web ナビゲーション，クラシファイアシステム，ブラウジングパターン

1. Introduction

Over the last decade, we have withnessed an explosive growth in the information available on the World Wide Web(WWW). Now days we can find Web pages in almost all fields. However, finding objective Web pages is very hard for a user because of the width of the Web, therefore empirical browsing strategies are very important for user's efficient browsing. Search engines are often used as an available tool for the purpose of information gathering. However, the results returned from search engines may include useless web pages for a user. Especially, if user's objective Web pages is not indexed in the search engine's database, a user have to do browsing start with the results of search engines. The Web is also available for user's browsing tasks that a user crawl according to his/her interests. Users may crawl to discover interesting Web pages without specific goal. User's browsing strategies are also important in such a case. For example, a user may often start browsing with Web pages including links for various topics such as Directory Services. A user may also go the round of "What's new" Web pages which is updated frequently or some digest news pages.

Many techniques to assist users on their browsing tasks have been developed. For example, there are many methods on gathering relative Web pages on some keywords [3], [8], and recommending next links or relative Web pages for a user from the current Web page [1], [5], [7]. Information on Web page contents is mainly used in those techniques. However, it is not enough for crawlers to narrow their search spaces. They do not consider strategic search patterns. For example, a user don't select hyperlinks soon which he selected recently. Moreover, though those techniques have an effect on assisting user's browsing tasks for searching with a specific goal, it is difficult to assist a user when his/her interest changes rapidly. There are also some techniques to assist users on their browsing tasks by learning accessed Web pages sequences [6], [10], [11]. However they assist a user in a closed space on a same Web site because they use logs on a Web server. They do not assist users on user's usual browsing tasks in which their interest frequently change and visit various Web sites.

It is difficult to assist user's Web browsing in a open space because of a wide search space. It is also difficult to learning user's

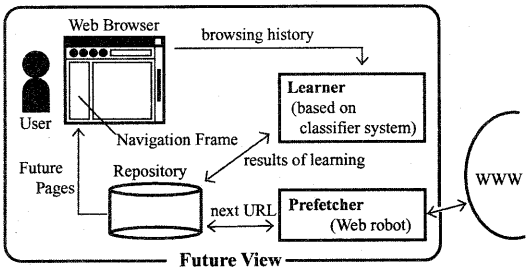

Figure 1 System overview

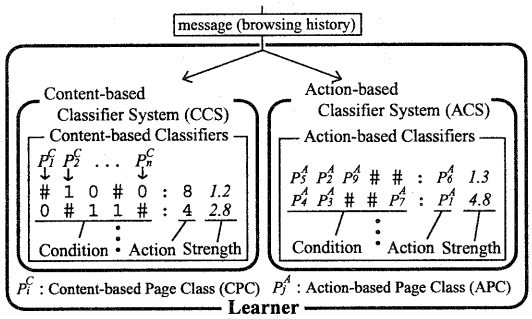

Figure 2 Architecture of a Learner

browsing patterns. Each accessed Web pages' sequences are dissimilar as input data for a learning component. Therefore, no useful techniques have been developed for Web navigation to assist user's usual browsing. We can make the search space sufficiently narrow, if we consider not only similarity of the Web pages but also user's strategic search patterns.

In this paper, we propose a Future View system to assists user's browsing tasks. The Future View learns user's browsing patterns, and prefetches Web pages according to user's browsing strategies by applying learned browsing patterns for crawling Web pages. Web robot will identify Web pages that will be reached by a user in the near future, and the Web pages are presented for a user through a Web browser in order to assist user's browsing tasks. A Content-based learning technique is used to learn user's browsing pattern. Moreover, the Future View uses a learning technique based on user's actions in order to take precedence possible Web pages accessed by a user in the future. These two types of learning are developed with evolutionary learning method, e.g. classifier systems. Gathered Web pages are ranked according to the possibility of access and listed. If the results correspond to user's interest and include objective Web pages, he/she can access to the Web pages directly through a user interface. On the other hand, if the results do not correspond to interest of a user, he/she must change his/her browsing strategy.

2. Future View Architecture

Figure 1 shows an overview of the Future View. The Future View consists of two main components. The first component Learner is a learning component based on classifier systems(CS) [4]. It receives information about accessed Web pages from an altered Web browser "Mozilla" provided as open source. The Learner is constructed with two types of classifier systems: a Content-based CS(CCS) and an Action-based CS(ACS) (Fig. 2). A CCS learns user's browsing patterns based on contents of accessed Web pages. On the other hand, a ACS learns user's browsing patterns based on user's actions. Accessed Web page information is transformed to a value whether it is included in a "Page Class". A Page Class is an abstract representation of Web pages set based on various features of its. A condition part and an action part of a classifier consist of values for Page Classes. We call page classes for a CCS "Content-based Page Class(CPC)" and page classes for an ACS "Action-based Page Class(APC)". We show the detail of its in the section 3.1 and 3.3. The second component is a Prefetcher. Our crawler prefetches Web pages referring the results of learning classifiers. A crawler stores the searching states at the point in time with information of Web pages. And in order to prefetch more deep Web pages, prefetched Web pages are considered as a part of browsing history. Gathered Web pages are always displayed as *Future pages* in the *Navigation Frame* with a function of a Web browser "Sidebar", while browsing.

3. Learning User's Browsing Patterns

3.1 Design of CPCs

In this section, we show the way of designing CPCs. When a user browses on a topic, the topic can be characterized with appeared words in Web pages on the topic. We can define CPCs based on some viewpoints below using *Browsing Session* in order to distinguish the type of words aimed by a user. The "browsing session" means a browsing sequence on a single topic, whether a user access to related Web pages from many links pages or using search engines.

- **topic continuity**
  - Web pages which include top n words of high TFIDF values in the title or body or anchor text.
    - The TFIDF [9] values here for all words in the current document are calculated with the other documents accessed before the current session.
  - Web pages accessed by following links of which anchor text include keywords input for search engines in the current session.

- **content difference on a topic**
  - Web pages which include more words with high TFIDF value and not included the other Web pages in the same session.
    - The TFIDF values here for each word in the current session are calculated with only documents in the current session.
  - Web pages which the very similar Web pages didn't appear in the current session.

3.2 Learning with a CCS

Each classifier consists of CPCs for a CCS are represented as below.

$$\langle \text{classifier} \rangle = \langle \text{condition-part} \rangle : \langle \text{action-part} \rangle$$

$$\langle \text{condition-part} \rangle = C_1^C, \ldots, C_n^C = \{1, 0, \#\}^n$$

$$\langle \text{action-part} \rangle = A^C = \{P_1^C, \ldots, P_n^C\}$$

A C_i^C means a i-th component of a condition-part. When a CPC corresponding to a i-th component is represented as a P_i^C and the current Web page is represented as p_{curr}, each component C_i^C means: if the C_i^C equals 1, then a p_{curr} is included in a page class P_i^C, and if the C_i^C equals 0, then a p_{curr} is not included in a page class P_i^C, and if the C_i^C equals "#", then whether a p_{curr} is included in a page class P_i^C or not. Here, A^C means a CPC which the next Web page p_{next} will be included in . Whenever a user visit a new Web page, a set of values for matching is constructed with a message (browsing history) and it is compared with a condition part of a classifier for matching. A CCS performs updating strength of each classifier using the *Bucket-Brigade* algorithm in the reinforcement component and GA in the discovery component like standard classifier systems.

3.3 Design of APCs

In this section, we show the way of designing APCs. An APC represents a set of Web pages based on user's actions used in an ACS, and the classifiers consists of the APCs represent a user's characteristic browsing pattern. We provide a guideline for defining APCs and samples of APCs below.

- **interests to news**
  - Web pages accessed by following new added links.
- **strategic search**
  - Search engine's top page accessed by directly inputting the URL to a browser.
  - Web pages which accessed by following a link near by the previous followed link and were not followed today yet.

3.4 Learning with an ACS

The Future View uses an ACS to learn user's browsing patterns based on user's actions. Each classifier consist of APCs for a ACS are represented as below.

$$\langle \text{classifier} \rangle = \langle \text{condition-part} \rangle : \langle \text{action-part} \rangle$$

$$\langle \text{condition-part} \rangle = C_1^A, \ldots, C_n^A = \{P_1^A, \ldots, P_m^A, \#\}^n$$

$$\langle \text{action-part} \rangle = A^A = \{P_1^A, \ldots, P_m^A\}$$

Here, i means the order of access to a Web page, and C_i^A means a APC in which the i-th Web page should be included. The current Web page is included in an APC of C_n^A, and a Web page accessed at the previous step is included in an APC of C_{n-1}^A. When the current Web page is represented as p_{curr}, each component C_i^A means: if the C_i^A equals P_j^A, then a p_{curr} is included in a page class P_j^A, and the C_i^A equals "#", then whether a p_{curr} is included in any page class or not. Here, A^A means a APC which the next Web page p_{next} will be included in . Whenever a user visits a new Web page, a set of values for matching is constructed with a message (browsing history) and it is compared with a condition part of a classifier for matching. An ACS updates strength of each classifier based on the *Bucket-Brigade* algorithm in the reinforcement component and GA in the discovery component like standard classifier systems in the same way to an CCS. In addition, we use the following techniques for learning efficiency.

- *Partial Matching* [2]: The Future View performs partial matching sequence instead of exact matching.
- Using page classes and *instances* for classifiers: The Future View uses not only page classes but also URLs of Web pages as instances.
- *Cover (detector) operator* [12]: In a discovery component of an ACS, new classifiers are created a matching classifiers out of the current message when there is no classifiers match the current message.

4. Prefetching and Presenting Future Pages

A prefetcher gathers Web pages as *Future Pages* start with the user's current Web page applying learned browsing patterns, and stores them to a repository. A prefetcher search *Future Pages* with the standard *best-first search* algorithm with a evaluation function using values of the last applied classifier's strength. The table 1 shows a procedure of gathering *Future Pages*.

Table 1 Procedure of gathering Future Pages

(1) **Initializing:** Let p_0 is a current Web page, s_0 is a browsing sequence to the current Web page, the initial value for the current trail $g_0 = 0$, the estimation value $h_0 = \theta_{strength} + \alpha$ and initialize a search queue list as $L = [(p_0, s_0, g_0, h_0)]$.
(2) **Selecting the best page:** Pick up an element with the highest heuristic value $f_i = g_i + h_i$ in a search queue list L, and delete it from a L. We represent the picked up element (p_i, s_i, g_i, h_i). And the element into an opened list OL.
(3) **Detecting a set of applicable classifiers:** For CCS and ACS, a mesage from a browsing sequence s_i match classifiers, and a match set is made. A prefetcher detects all classifiers from each match set as a set of *applicable classifiers* which have a strength value more than threshold $\theta_{strength}$ and are applicable to the current state.
(4) **Prefetching Web pages:** A prefetcher obtains all Web pages as the results of applying each action of *applicable classifiers*. Each element $e = (p_{i+1}, s_{i+1}, g_{i+1}, h_{i+1})$ is made for an obtained Web page. Here, p_{i+1} is a obtained Web page, $g_{i+1} = f_i = g_i + h_i$ is a value for each trail, h_{i+1} is the larger of the strength values of last applied CPC and APC, and a s_{i+1} is a new browsing sequence that a page p_{i+1} is added into a previous browsing sequence s_i. All elements are added into a search queue list L.
(5) **The termination condition:** If a search queue become empty ($L = [\,]$) or the number of opened elements in OL reach the max search numbers, this procedure is terminated, otherwise return to 2.

Future Pages are presented to a user through a browser. Fig.3

Table 2 Settings for CCS and ACS

settings of CCS and ACS
Population size of a set of classifiers: 500
Length of a message: 10
The number of CPCs: 10
The number of APCs: 12
Frequency of running GA: 1 time per 30 step
Selection algorithm: Elite
Modifying strengths of classifiers: Backet Brigade
Initial strength values: 1.0
Limited maximum strength values: 4.0
Max. depth of searching(prefetching): 7
Max. number of gathering distinct *Future Pages*: 10

show a user interface. A left upper frame display URLs as a browsing history and the "New Session" button to split browsing sessions. A left lower frame indicates a list of *Future Pages* with a title, a URL string, a thumbnail image of the Web page, a button to display a trail from the current Web page and a part of contents of the Web page. A user can access the Web page by clicking a URL link.

5. Experiments

In order to investigate effect of a Future View, we evaluate the results of using a Future view by four users. They are not researchers, and don't have knowledge of evolutionary computation and any other areas. After they used a Future View for three days, they evaluated many *Future Pages*. We show *Future Pages* which are presented by a Future View and the results of evaluating qualities of presented *Future Pages* by a user.

We use the settings showed in Table.2 for CCS and ACS.

5.1 Example of Future Pages

We show a few typical learned patterns and presented *Future Pages*.

- "*come and go*" pattern
  Each user have used a search engine sometimes. After they have inputed query keywords for a search engine, they have come and gone the list page including the results of search query and the next pages followed any links.

  Now a user visited Web pages P_1, P_2, P_3 in Fig. 4, and now reading the Web page P_3. Then Web robot visited Web pages P_2, P_4, P_2, P_5, P_2, and Web Pages P_4, P_5 are presented as *Future Pages*.

  In this pattern, CCS tended to have brought the good result. For example, Web pages P_4, P_5 actually related to search keywords. Some advertisement links were excluded.

- "*daily fixed*" pattern
  A user usually browsing start with his bookmarks page as a hub page. The page include some old links, and he don't select the links recently. If he visit his bookmarks page at first of the day, Web robot follows valid links excepting old links.

  In this pattern, ACS tended to have brought the good result. Because fixed urls of Web pages and the orders of visiting have been learned by ACS.

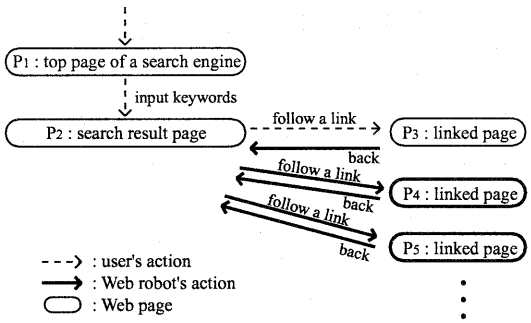

Figure 4 "*come and go*" pattern

Figure 5 Example of browsing history

We also show an interesting pattern. A user has performed the following browsing occasionally.

- He has used a search engine sometimes.
- After he has inputed words to a search engine, he has selected some results from the search engine one by one from the top. (So far, he has perform browsing same as "*come and go*" pattern.)
- After he has selected some results from the search engine, he has has accessed to a News site which have been often used.
- He has selected some new articles.

A user visited a Web page about "Linux OS", and he broke off the current session by pushing "New Session" button, and he accessed to the top page of search engine "google". Now he have just selected some results from search engine and back to the results pages of search engine(P_2 in Fig. 4 and Fig.3). Figure 5 shows the accessed list of him. Then, Web robot visited Web pages P_j, P_2, P_k, P_l, P_k(Fig. 4), and Web Pages P_k, P_2, P_l, P_j are presented as *Future Pages*(Fig. 7).

5.2 Qualities of Future Pages

We investigated qualities of *Future Pages* in order to confirm effectiveness compared with the other systems; there are not appropriately comparable systems considered the open search space, however. Therefore we compare *Future Pages* with Web pages searched by using "random walk". And we also investigate the difference of

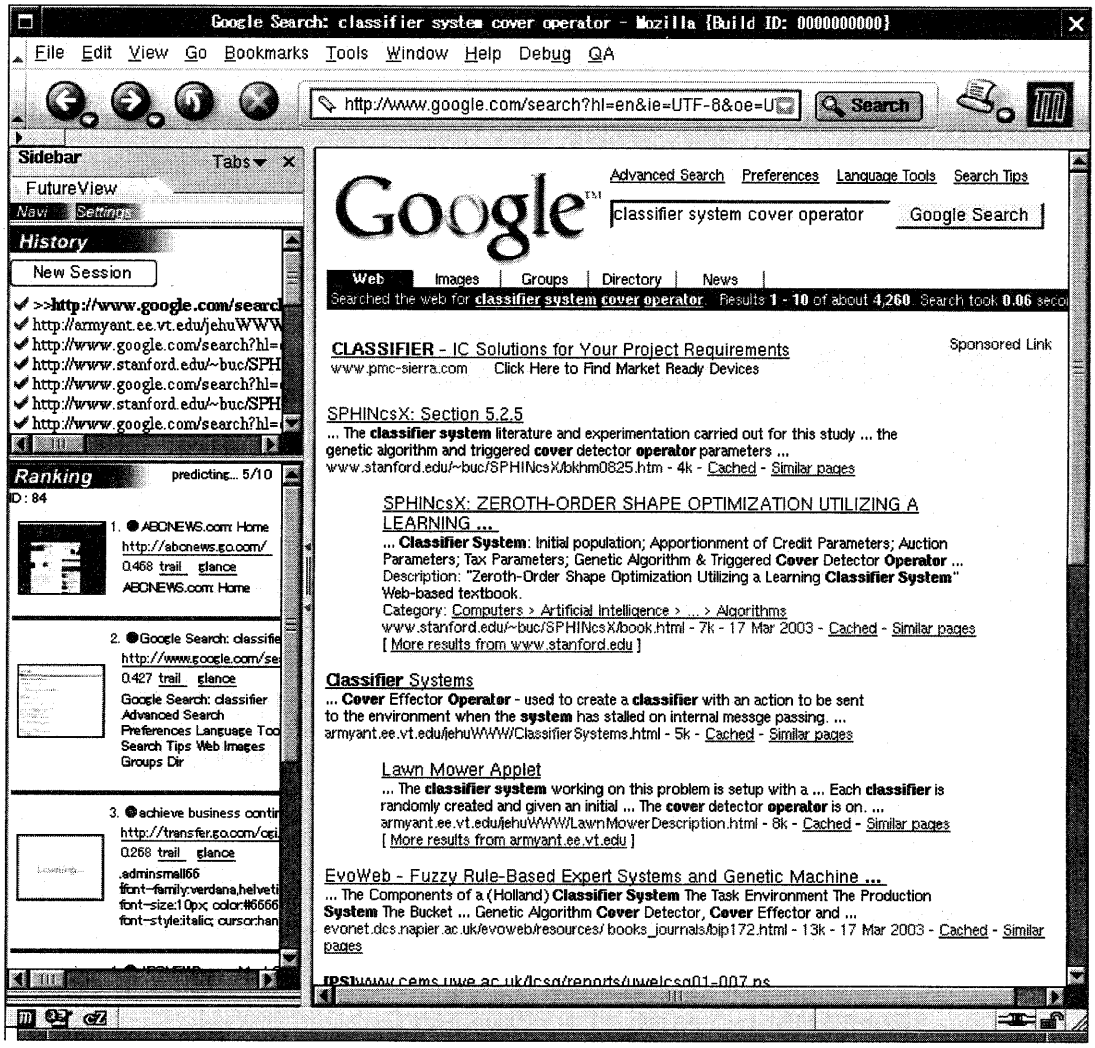

Figure 3 User Interface

effects between an ACS and a CCS.

In order to investigate the different effects between an ACS,a CCS and "random walk", four subjects evaluated up to 10 *Future Pages*(if there is a few Web pages for presenting the user, the number of *Future Pages* may less than 10 pages) for 5 current Web pages at each case that a prefetcher is applied the results of learning of "ACS and CCS", "only ACS", "only CCS" and "random walk" for the comparison(150 or less *Future Pages* for a subject in total). Here, "random walk" is that Web robot select a link from all of links in followed Web pages at random. In addition, "random walk" exclude some useless links for pretreatment.

Each *Future Pages* is evaluated with 3–1 points: 3 for interesting Web pages, 1 for indifferent Web pages and 2 for intermediate value. The Table5. 2 shows the average values of the total values of evaluations for four subjects. We also compare the results for two types of user's browsing, with the specific goal(*searching*) or without the specific goal(*browsing*).

Table 3 Evaluation of Future Pages

	browsing	searching
CCS and ACS	**1.0** (8.1)	**0.98** (7.6)
CCS only	**0.38** (2.7)	0.19 (1.4)
ACS only	0.64 (6.3)	**0.76** (7.0)
random walk	0.65 (6.6)	0.73 (7.3)

$m(n)$: m is the average values of the total values of evaluations for four subjects, and it is normalized by the value for $e_{c,a}$ for *browsing*. n is the average value of the number of presented pages.
We compare the evaluation value with gathering *Future Pages* with both CCS and ACS, only CCS, only ACS and "random walk" for *browsing* and *searching*.

In the results, CCS tended to have brought the high evaluation value for *browsing* than for *searching*. On the other hand, ACS

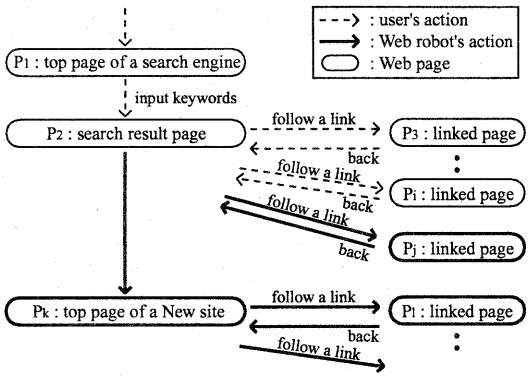

Figure 6 Example of interesting pattern

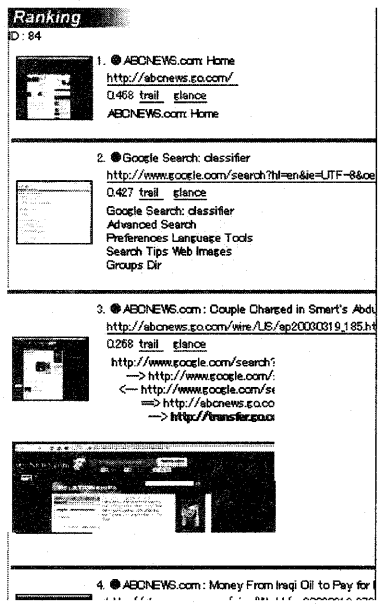

Figure 7 Presented Future Pages

tended to have brought the high evlauation value for *searching* than for *browsing*. Moreover, when we have used both of CCS and ACS for a Future View, it has brought more high evaluation values for *browsing* and *searching* than using only CCS, only ACS or performing "random walk". If its attention is paid to the number of presented pages, it for "CCS and ACS" is larger than for "CCS only", "ACS only" or "random walk". It is based on the combination of "CCS" and "ACS". It means that a Future View prefetch and present more interesting *Future Pages* for a user with CCS and ACS than the other cases. The reason the numbers of presented pages for "CCS only" and "ACS only" are small because a Web robot have not applyed classifiers with a low strength than a threshold value.

The results of this experiments shows that a CCS and an ACS can work suitably for some kind of user's browsing. We confirmed that a Future View seemed to all users to working effectually for all browsing, and it worked seamlessly even when he/she change his/her browsing strategies according to the situation.

6. Conclusions

We proposed a Future View system that assists user's Web browsing on the Web as an open space. We also show a concept of "*Future Pages*" gathered with user's browsing patterns. A Future View gathers relative Web pages to the current Web pages or keywords. A CCS and an ACS are components for learning user's browsing patterns. We provide policies for designing them. We also investigated learned browsing patterns and its results, and verified the effectiveness of presented *Future Pages* by a Future View.

References

[1] M. Balabanovic and Y. Shoham. Learning information retrieval agents: Experiments with automated web browsing. In *Proceedings of the AAAI Spring Symposium on Information Gathering from Heterogenous, Distributed Resources*, pages 13–18, 1995.

[2] L. B. Booker. Improving the Performance of Genetic Algorithms in Classifier Systems. In *Proceedings of the International Conference on Genetic Algorithms and Their Applications*, pages 80–92, Pittsburgh, PA, 1985.

[3] P. De Bra and R. Post. Information Retrieval in the World-Wide Web: Making Client-based Searching Feasible. *Computer Networks and ISDN Systems*, 27(2):183–192, 1994.

[4] J. H. Holland and J. S. Reitman. Cognitive Systems Based on Adaptive Algorithms. In *Pattern-Directed Inference Systems*, pages 313–329. Academic Press, 1978.

[5] T. Joachims, D. Freitag, and T. Mitchell. WebWatcher: A Tour Guide for the World Wide Web. In *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence*, pages 770–775, 1997.

[6] R. Kosala and H. Blockeel. Web Mining Research: A Survey. *SIGKDD: SIGKDD Explorations: Newsletter of the Special Interest Group (SIG) on Knowledge Discovery & Data Mining, ACM*, 2, 2000.

[7] H. Lieberman. Letizia: An Agent That Assists Web Browsing. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, pages 924–929, 1995.

[8] F. Menczer, R. K. Belew, and W. Willuhn. Artificial Life Applied to Adaptive Information Agents. In *Working Notes of the AAAI Symposium on Information Gathering from Distributed, Heterogeneous Databases*. AI Press, 1995.

[9] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison Wesley, 1999.

[10] M. Spiliopoulou. Web usage mining for Web site evaluation. *Communications of the ACM*, 43(8):127–134, August 2000.

[11] J. Srivastava, R. Cooley, M. Deshpande, and P. N. Tan. Web usage mining. *SIGKDD: SIGKDD Explorations: Newsletter of the Special Interest Group (SIG) on Knowledge Discovery & Data Mining, ACM*, 1, 2000.

[12] S. W. Wilson. Knowledge growth in an artificial animal. *Proceedings Genetic Algorithms and their Applications*, pages 16–23, 1985.