International Journal of Cooperative Information Systems
© World Scientific Publishing Company

CONSTRUCTING A PERSONAL WEB MAP WITH ANYTIME-CONTROL OF WEB ROBOTS

SEIJI YAMADA
CISS, IGSSE, Tokyo Institute of Technology
4259 Nagatsutacho, Midori
Yokohama 226-8502, JAPAN

NORIKATSU NAGINO
CISS, IGSSE, Tokyo Institute of Technology
4259 Nagatsutacho, Midori
Yokohama 226-8502, JAPAN

In this paper, we propose a PWM (Personal Web Map) which is a personal and small database of interesting Web pages to a user and develop a method to construct it under the user's control of multiple Web robots. While general search engines with very large databases are valid for information retrieval in the WWW, it is still important that a user constructs a small, personal database of relevant Web pages to his/her interest. For such a Web page database, we propose a PWM and develop a PWM system. First a user gives keywords indicating his/her interest to a system, and it constructs a PWM concerned with the keywords. For building a useful PWM, it is necessary that a user can interrupt the construction of a PWM anytime and instruct a sub-field which should be explored more. For this function, we develop an anytime-control algorithm for multiple Web robots. A density blackboard is used for controlling Web robots, and an uniform distributed PWM is built. Whenever a system is interrupted by a user, it provides a valid PWM in terms of keeping search space wide, and indicates many alternatives on which he/she wants more information. From Web pages in a database, document vectors are generated and used to construct a 2D-map of a PWM by using self-organization maps. A user easily recognizes interim results through the 2D-map, and gives instruction by clicking a node about which he/she wants more detail information. We made experiments by subjects and found out that our method outperformed breadth-first search for constructing a useful PWM. As results, a PWM system is considered as a promising approach to assist a user in gathering relevant information in the WWW.

Keywords: Information gathering, the WWW, Web robots, anytime-control, user preference, SOM.

1. Introduction

The accessible information through the Internet is increasing explosively as the WWW becomes widespread. While the computer resource has become inexpensive rapidly. Thus a user is able to gather relevant Web pages to his/her interest and locally store them in a personal Web page database on a hard disk. Using such a personal Web page database, a user is able to retrieve information precisely because the database is filtered by user interest, and to investigate interesting Web pages locally without influence on network traffic. Hence constructing a personal Web page database is important and our research aim is concerned with it.

A practical and simple way to construct a personal Web page database is to use a search engine with the interesting information as a query. The search engine provides a list of relevant Web pages (called a *hit list*) to a user, and we can implement a software agent that gathers the Web pages through the links indicated in the hit list. However, since a database of a search engine is huge and adequate filtering is hard, many irrelevant Web pages may be indicated in a hit list and gathered by the software agent. Also though a database of a search engine is very large, it includes only a small part of all Web pages in the WWW[15]. Furthermore a search engine hardly have sufficient recency of gathered Web pages because the updating them by fetching a lot of modified Web pages is very expensive[18]. Consequently the gathered Web pages using a search engine include many useless ones and do not include many useful ones. This is why we do not use a search engine for constructing a personal Web page database.

For building a personal database of Web pages, it is important that a user can control the construction of a database to customize it. We propose a PWM(Personal Web Map) as a database which a user can control its construction. PWM is a layered database consisting of classified Web pages and constructed under user's control. A user can determine keywords as input to a PWM system and a PWM is gradually built by gathering relevant Web pages to the keywords and classifying the gathered Web pages. Also a user interrupt the gathering anytime and a PWM system show a 2D-map of a PWM through a Web browser. The 2D-map consists of nodes indicating classified subsets of gathered Web pages, and a user can give feedback to the system by selecting a node on which he/she wants more information. The construction of a PWM is interactive since a user can control the granularity on a PWM.

WebWatcher[10] and Letizia[16] are able to indicate the Web pages which a user wants to see next. Using browsing history, they learn to predict useful Web pages for a user. These systems are consider as customizing systems that learn user's preference in searching Web pages. Unfortunately the systems do not build personal Web page databases.

SPHINX[20] is a framework in which a user can achieve the personal crawling tasks. However the customization is on searching of Web pages, not building a personal Web page database. Thus the purpose is different from ours.

Fish search[2] is a distributed search algorithm for gathering relevance information. Agents have energy which is gained from relevant Web pages and lost from irrelevant pages. Agents having hight-energy can reproduce themselves and others having low-energy may die. ARACHNID[17] is a more excellent system that can gather information by learning information agents. In the similar way to fish search, agents reproduce themselves or die depending on their energy, and obtain energy from relevant Web pages. Furthermore ARACHNID is adaptive to the change of an environment. These systems do distributed control of agents, however the control method is inspired by an artificial life approach and different from our anytime-control.

Kohonen's SOM (Self-Organizing Map) have been used to classify documents in the WWW. WEBSOM[7,8] is able to classify articles in network news groups and display the clustered results. WEBSOM just does clustering documents when they are given, and has no contribution to information gathering. In this research, we use SOM for clustering gathered Web pages.

There are studies on planning to generate procedures for gathering information through computer networks. The *Softbot* [6] provides a framework and a interface for describing operators. A complete partial-ordering planner is used. *Occam* [14] is also a planner for gathering information. It is more efficient and able to reason about the capabilities of different information sources. *Sage* [11] was developed for integrating planning, execution, replanning, and sensing to gathering information in distributed resources. The aim of these studies is to generate a plan as a procedure of gathering information, and a plan consists of UNIX commands, database operations. In contrast with these studies in which a operational procedure of information gathering is generated automatically, a PWM system actually gathers relevant Web pages under user's preference.

Our research is concerned with the *Web Robot*[9] which is used to gather Web pages for a search engine database. However it is not controlled and traces to only linked pages with breadth-first search-like fashion. A PWM system utilizes more sophisticated search and the experimental comparison will be described later.

Some learning systems have been developed for information gathering and browsing in the WWW. *ShopBot* [4] learns the text pattern indicating the price of CD-ROMs, and searches for the cheapest one more efficiently than a human. The purpose of *ShopBot* is different from our research. *WebWatcher* [10] and *Letizia* [16] are able to indicate the Web pages which a user wants to see next. Using browsing history, they learn to predict useful Web pages for a user. These studies focused on how to extract knowledge from gathered Web pages. In contrast with that, a PWM system is concerned with how to gather Web pages.

Navigation planning[23] automatically generates a sequence of Web pages by which a user can understand a concept systematically. First a user inputs a query indicating his/her a target concept and a navigation planning does planning using operators which are generated automatically from Web pages. In this system, action corresponds to understanding a Web page, and a navigation planning system is able to generate such operators from Web pages by utilizing tag structure and a indexing method. Unfortunately the system cannot acquire user's interest interactively.

We summarize the contents of this paper. In section 2, we describe the overview of a PWM system and a PWM as a structured database. Next we explain the construction of WPS and a density blackboard. Anytime-control of multiple Web robots which is suitable to build a PWM is described. In section 3, we describe human computer interaction in a PWM system. Using SOM, a 2D-map is generated for indicating the interim results of Web page gathering, and a user can select a node on which he/she wants more information. In section 4, we make experiments for evaluating our approach. Web robot search like breadth-first search is experimentally compared with our approach. In section 5, limitations and open problems are discussed. Finally section 6 concludes this research.

2. Information gathering with multiple Web robots

2.1. *Overview of a* PWM *system*

Fig.1 shows the overview of a PWM system. It roughly consists of a PWM, a 2D-map, SOM and Web robots. A user can control to construct a PWM. The input of the system are *keywords* on which a user wants to build a PWM, and a system outputs a 2D-map of a PWM to him/her. A user can click a node on the 2D-map for gathering more information about it.

Multiple Web robots gather relevant Web pages to keywords. They monitor a density blackboard in a PWM, and try to gather pages in the most sparse area. This is called *anytime-control*, and we will mention its detail later. The anytime-control makes a database of Web pages uniform and provides a valid 2D-map with many alternatives a use can select. Web pages gathered by Web robots are stored in a Web page database. All Web robots work asynchronously.

A density blackboard in a PWM is updated with keyword vectors detected from Web pages in the database. A 2D-map is generated depending on document vectors by SOM.

After gathering, using PWM and a IR system, a user is able to retrieve information precisely because of filtering by user interest and investigate interesting Web pages locally without influence on network traffic. Also the tree structure of WPSs may be utilized for constructing a directory structure like YaHoo's interface.

2.2. PWM

As seeing from Fig.1, a PWM (Personal Web Map) is consisting of a Web pages database, tree-structured WPSs and a density blackboard. A WPS is a set of keyword vectors generated from Web pages in the Web page database. A density blackboard indicates the distribution of keyword frequency in the gathered Web pages.

Fig.2 shows more detailed WPSs and a density blackboard. The first layer includes a WPS_0^0 including keyword vectors of directly relevant Web pages to

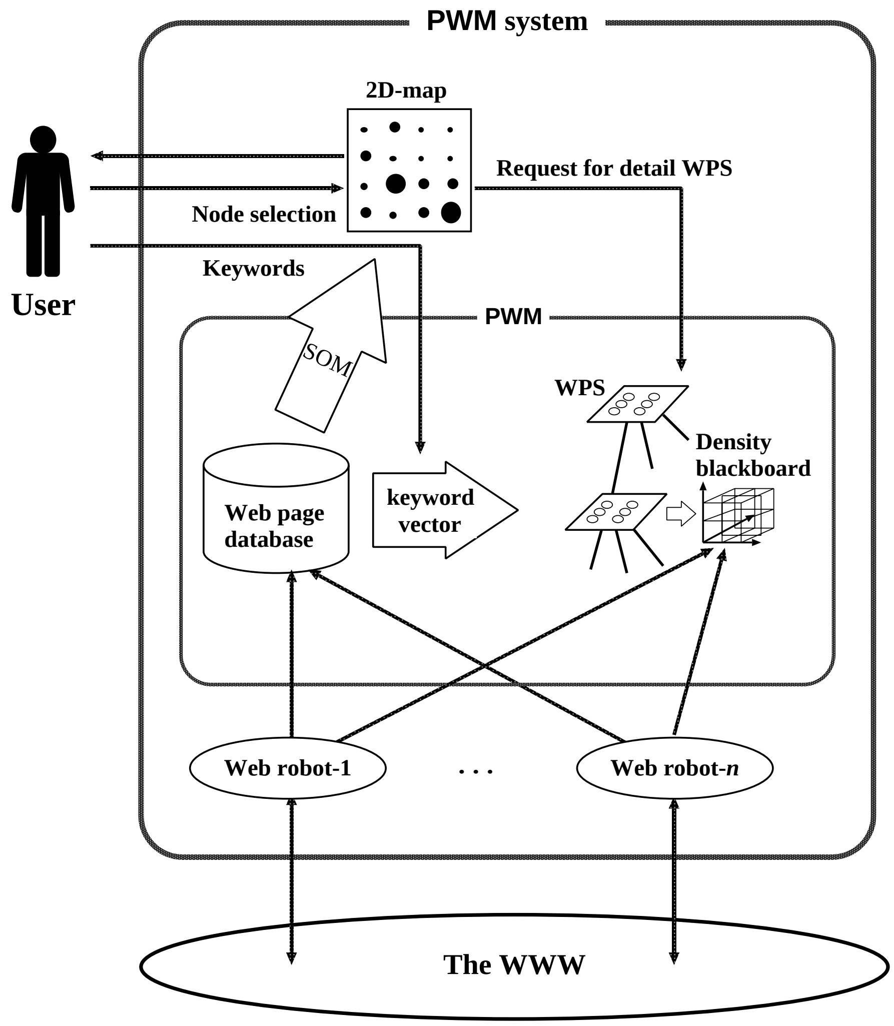


Fig. 1. A PWM system.

keywords given by a user as input. WPS_j^i means the jth WPS in the ith-layer, and a layer may include multiple WPSs. A single WPS is constructed by Web robots' information gathering at once and it is called a *current* WPS. In Fig.2, if the current WPS is WPS_0^1, the density blackboard is generated from a *current* WPS_0^1 and Web robots fetch Web pages by monitoring the density blackboard under anytime-control.

During Web page gathering, the Web pages are classified by SOM and clusters (white circles of WPS in Fig.2) are generated whenever a user requests it. Then the 2D-map, in which a node corresponds to a cluster in WPS, is indicated to a use. If he/she points a cluster in a WPS_j^n about which he/she wants

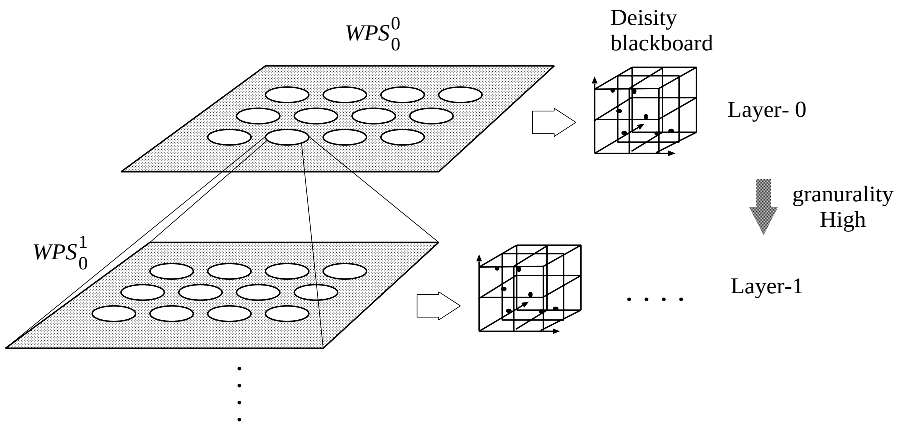


Figure 2 WPS and a density blackboard.

to know more, a system will generate the more detail WPS_k^{n+1} for the cluster k in a $(n+1)$th layer. With such a procedure, a user is able to control construction of a PWM for including detail information about which he/she wants to know more.

In a PWM, each WPS contains about hundreds of Web pages. You may claim a PWM must be a small subset of a large database for a large search engine. However we often face the facts that many interesting Web pages are not captured by such a search engine. We consider a PWM which is controlled by a user is promising for a fast personal search engine with a necessary and sufficient Web page database.

2.3. *Constructing WPS*

For constructing a WPS, a system generates *keyword vector*s using the occurrence frequency of keywords in a Web page. A keyword vector is similar to a term vector used for the vector space model[22] in information retrieval. A keyword vector V_p of a Web page p is described in the following.

$$\begin{aligned} V_p &= (v_{p1}, v_{p2}, \cdots, v_{pN}) \\ &= \left(\frac{f(p,t_1)}{m(D,t_1)}, \frac{f(p,t_2)}{m(D,t_2)}, \cdots, \frac{f(p,t_N)}{m(D,t_N)}\right) \\ &\quad \text{where} \quad m(D,t) = \max_{p \in D} f(p,t). \end{aligned}$$

The D is a set of Web pages and the $f(p, t_i)$ indicates the occurrence frequency of the input keyword t_i in a Web page p. The value v_{pi} stands for normalized $f(p, t_i)$ by the maximum value in Web pages of a Web page database

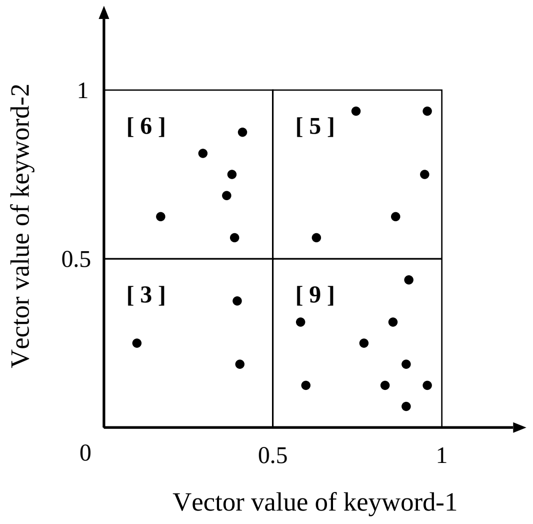


Fig. 3. A density blackboard.

D. The range of v_{pi} is [0, 1] and the v_{pi} shows the importance of t_i in the Web page. Also N is the number of keywords inputed by a user. Finally a set of these keyword vectors V_ps is a WPS.

2.4. *A density blackboard*

A blackboard system[5] provides a useful architecture to control a distributed multi-agent system. Agents read and write the information to layered shared memory, called a *blackboard*, and share them flexibly. Thus we use a blackboard for Web robots to share the information. The shared information is a *density distribution* which is the distribution of the number of keyword vectors in a Web page database in the keyword vector space.

The range [0, 1] of each axis in a n-keyword vector space is divided into m patches. Since the n-keyword vector space has n dimensions in the keyword vector space, m^n patches are generated in total. Next the densities for each patches are computed, and they are stored in a density blackboard. Fig.3 shows a two dimensional density blackboard for two keywords. Each axis is divided into two patches. The black points and the numbers within blankets indicate keyword vectors of Web pages and densities of patches respectively. As mentioned later, if a patch has high density, it has been explored well and Web robots do not need to investigate there. If a patch has low density, it was not explored sufficiently and Web robots should investigate there more.

2.5. *Anytime-control of Web robots using a blackboard model*

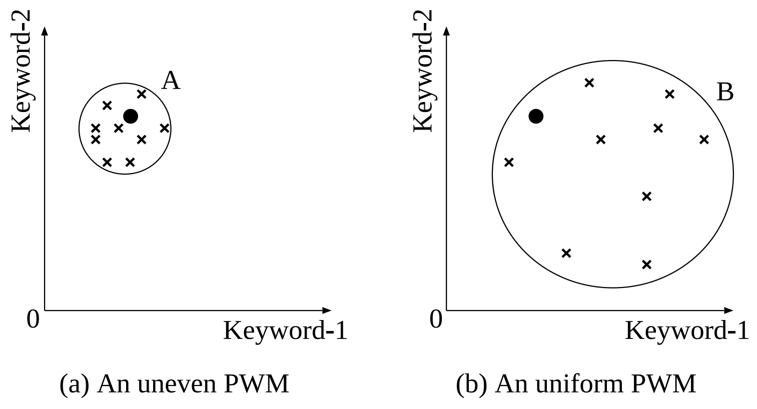


Fig. 4. An uniform PWM.

2.5.1. *Control policy*

Though many search engines use multiple Web robots for gathering Web pages, few ones control them effectively. Thus we propose *anytime-control*: a novel method for controlling Web robots to build a PWM.

An important point to build a PWM interactively is that the building process may be interrupted anytime by a user. A PWM system allows a user to interrupt information gathering to see intermediate results and control it. Hence it should be able to indicate *appropriate results* whenever a user requests. We consider the appropriate results are Web pages which were gathered *uniformly* on keywords. If it does not control Web robots, the obtained PWM will be uneven and its coverage will be far more narrow than an uniform one.

Fig.4 shows an uneven PWM and an uniform PWM when ten Web pages were gathered. The two keywords were given to a PWM system, and the X and Y axes indicate the relevance to keyword-1 and keyword-2 respectively. The black circles stand for start Web pages, and the crosses indicate keyword vectors of gathered Web pages. Though both of the PWM started from the same keyword vector, the results are different in the coverage (the circle A and B). Since a user selects a keyword vector about which he/she wants to gather more detail information, the coverage of subsequent layers is restricted within circle A and B. Thus the coverage of an uneven PWM becomes far more narrow than that of the uniform one. Obviously we prefer a PWM with a wide coverage.

2.5.2. *A procedure for a Web robot*

Thus we need an algorithm that returns an uniform PWM whenever a user interrupts. We developed such an algorithm which is called *anytime-control*[*] .

[*] This name is inspired by an anytime-algorithm [1] which, returns valid results anytime, has been studied in real-time problem solving in artificial intelligence.

The control is a simple distributed procedure shown in the following. Each Web robot acts using the algorithm asynchronously, and stops when a user interrupts it or the number of obtained Web pages becomes the given limit number or a limit time has come.

Anytime-control procedure for a Web robot

(1) See a density blackboard and determine the most sparse patch.
(2) Select a Web page α randomly in the patch.
(3) Select a linked Web page β randomly in the Web page α.
(4) Fetch a html file of the page β.
(5) Generate a keyword vector from the obtained file and write it into a density blackboard. Also store the file in a Web page database.
(6) Go to (1).

In the above procedure, Step (1) makes a PWM uniform. Each Web robot tends to explore more in an area with the least information, and this keeps the density of Web pages uniform. Additionally, URLs of the fetched Web pages are stored, and the Web pages are not selected in Step (2) and (3) again.

Note that in order to gather Web pages in the same patch to the Web page α for Step (3) and (4), a system tries to get Web pages linked from α. This is a valid way supported by some evidences. Menczer[17] pointed out $R > G$, where R is the conditional probability that a Web page is relevant given that it is linked by another Web page that is relevant with the same query, and G is the generality of the query, i.e., the fraction of Web pages that are relevant. Since $R > G$ means that is is more likely to hit a relevant Web page from another relevant Web page than from any random Web page, the linked Web pages in Step (3) and (4) tend to be within the same patch to α. Another evidence is a real-time search mode of WebCrawler[3]. The algorithm assumes that following links from Web pages that are similar to what the user wants is more likely to lead to relevant Web pages than following any link from any Web page. Under the assumption, WebCrawler works well.

2.6. *A procedure of a* PWM *system*

Eventually the procedure of a PWM system is described in the following.

(1) Initialize a WPS^0_0 as a empty set, and set a current WPS^i_j by WPS^0_0.
(2) A system inputs MetaCrawler[19] the keywords as a query, and obtains the most relevant 13 Web page. A PWM system initializes a density blackboard by giving it the document vectors of the 13 Web page as a start point.
(3) Start Web robots under anytime-control for gathering relevant Web pages.
(4) If interrupt of a user is given, work SOM to make a 2D-map of PWM and display it to a user. If a user clicks a node in the 2D-map, set a current WPS by WPS^{i+1}_k, and go to Step (2).

(5) If the number of gathered Web pages becomes the limit number, stop and wait a user's request.

3. Human computer interaction

3.1. *SOM-based display*

SOM (Self-Organizing Maps)[12,13] have been widely applied to build 2-dimensional visualization from a large text database. Because the learning algorithm is simple, and the input of high-dimensional vectors are automatically classified. For example, WEBSOM[7,8] has been developed for clustering a lot of articles for network news.

Since the keyword vector has the identical dimension to the number of keywords and it is usually more than three, a system needs to reduce the dimension to two in order to indicate a PWM to a user. Thus we can construct a 2D-map of a PWM by using SOM. We explain the SOM-based procedure in the following. The SOM learns with random input vectors in the document vector space, and the document vectors of gathered Web pages are given to the learned SOM and classified.

(1) Generate a set S of *document vector*s from Web pages of the current WPS in a Web page database. The document vector is slightly different from a keyword vector in normalization. It is normalized as a unit vector using the following formula. The E_p and $f(p,t)$ are the document vector of a Web page p and the occurrence frequencies of a word t in a Web page p.

$$\begin{aligned} E_p &= (e_{p1}, e_{p2}, \cdots, e_{pN}) \\ &= \left(\frac{f(p,t_1)}{l(p)}, \frac{f(p,t_2)}{l(p)}, \cdots, \frac{f(p,t_N)}{l(p)}\right) \\ &\quad \text{where} \quad l(p) = \sqrt{\sum_{i=1}^{N} f(p,t_i)^2} \end{aligned}$$

(2) Fig.5 shows the structure of SOM. Construct SOM using the number of keywords as the number of the input nodes and 25 (= 5×5) as the number of competitive nodes. The competitive layer is set to two dimension.

(3) Generate a set R of random document vectors with random values through [0, 1] in each dimension of a document vector. Give the R to SOM repeatedly, and update the weights between input nodes and competitive nodes using the following procedures[12].

(a) Let an input vector and weights of links from all input nodes to a competitive node u_i be $E_p = [e_1, e_2, \cdots, e_n]$ and $U_i = [u_{i1}, u_{i2}, \cdots, u_{in}]$ respec-

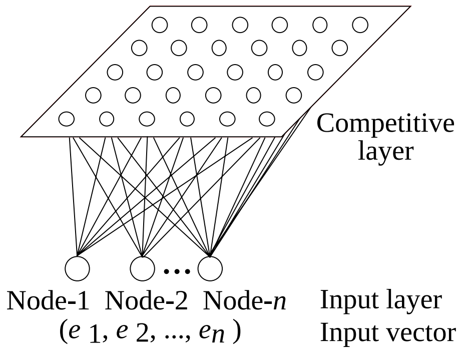


Fig. 5. Self-Organizing Network.

tively. In a PWM system, the n is set with the number of keywords N.

(b) SOM compute the similarity between the input vector and competitive nodes. The similarity is evaluated by Euclidean distance using the following formula.

$$\|E_p - U_i\| = \sqrt{\sum_j (e_{pj} - u_{ij})^2}$$

(c) Since we need a 2D-map, a square having the winner as the center is used as neighborhood. The formula for updating the weights of neighbors is shown in the following, where the α is a learning rate.

$$u_{ij}^{\text{new}} = u_{ij}^{\text{old}} + \Delta u_{ij}$$

$$\Delta u_{ij} = \begin{cases} \alpha(e_j - u_{ij}) & : i \text{ in the neighborhood} \\ 0 & : \text{otherwise} \end{cases}$$

With the following equations, the α and the size of neighborhood are scheduled to decrease as learning progresses.

$$\alpha = 0.5\left(1 - \frac{t}{10000}\right) \qquad d = 5\left(1 - \frac{t}{10000}\right)$$

(4) After learning, the set S of document vectors of gathered Web pages is inputted to a learned SOM, winner nodes for each vector are determined, and classification is done. The winner node of a Web page indicates the class. Also *unit keyword vector*s are inputed to a learned SOM and the winner nodes are labeled with corresponding keywords. The winner nodes are called *keyword node*s, and the unit keyword vector of the nth keyword

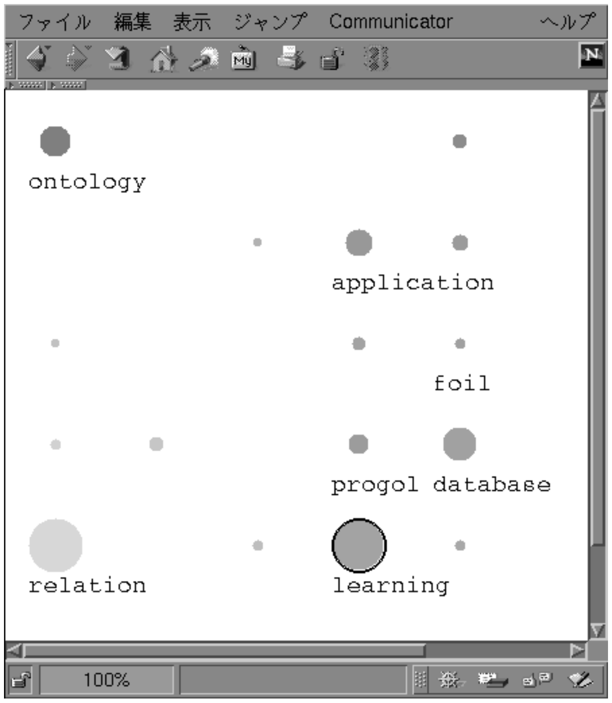


Fig. 6. A 2D-map of a PWM.

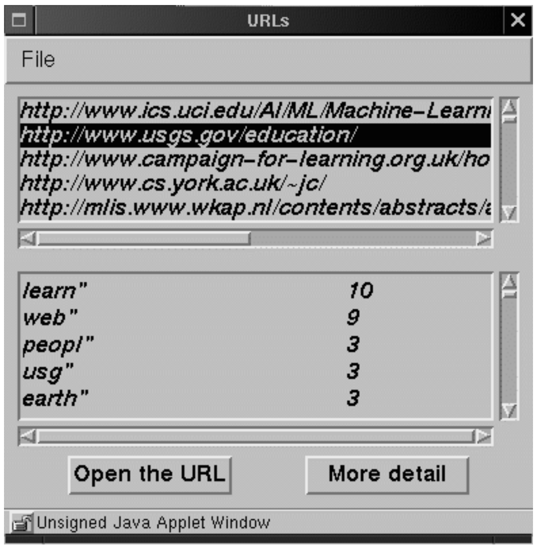
is (0, ···, 1, 0, ···, 0) where only the nth value is 1. The keyword nodes are colored differently, and the other nodes among them are colored in graduation. Furthermore the size of a node is in proportion to the number of keyword vectors included in the node.

Labeling and coloring the nodes enables a user to easily understand which a node include the relevant Web pages to a keyword. Also learning with random unit vectors makes the positions of keyword nodes widely distributed in a 2D-map. This helps a user to select a node on which he/she wants to know more.

An executed example of 2D-map for first layer PWMs is shown in Fig.6. The 2D-map was generated from keywords "application", "database", "foil", "learning", "ontology", "progol", "relation". You can see the keyword nodes, and the distance between them stands for their similarity. The size of a node stands for the number of Web pages classified in the node.

3.2. *User feedback*

Using a 2D-map, a user can easily point nodes about which he/she wants to know more. When a node is pointed by a user, a system opens a dialog window called a *node info window* like Fig.7. With a node info window, a user can see the URLs of the Web pages included in the pointed node and the occurrence



```
http://www.ics.uci.edu/AI/ML/Machine-Learn
http://www.usgs.gov/education/
http://www.campaign-for-learning.org.uk/ho
http://www.cs.york.ac.uk/~jc/
http://mlis.www.wkap.nl/contents/abstracts/
```

```
learn"          10
web"             9
peopl"           3
usg"             3
earth"           3
```

Fig. 7. A node info window.

frequencies of all keywords in the Web page. A user determines whether he/she wants to know more about the node, and clicks the more detail button for request. Note that a 2D-map provides the access to intermediate nodes except keyword nodes. This makes user's selection of intermediate concepts easy.

If the more detail button is clicked, a PWM system will gather Web pages for constructing the more detail WPS in the next deeper layer. The procedure is basically similar to one described in section 2.5.2, however the density blackboard is restricted. The occurrence frequencies of n keywords is restricted within the region R satisfying $v_1^{min} \leq v_1 \leq v_1^{max}, \cdots, v_N^{min} \leq v_N \leq v_N^{max}$, where $v_i^{min} = \min_{p \in S} v(p,i)$, $v_i^{max} = \max_{p \in S} v(p,i)$, N is the number of input keywords, $v_1, v_2, \cdots, v_N$ are variables in nth axis, S is a set of keyword vectors included in the pointed node, and $v(p,i)$ is a nth value of a keyword vector p. This is the minimum region including all the keyword vectors in the pointed node, and a system gathers only the Web pages having keywords included in the restricted region. This restricts the next WPS within the neighborhood of the pointed node. Fig.8 shows this process in 3-dimensional WPS_j^i. The node G is selected by a user and WPS_k^{i+1} is created in the restricted region R.

4. Experimental evaluation

In order to verify the effect of anytime-control in a PWM system, we made experiments with 10 subjects. The subjects were master course students in a computer science department, and the keywords which they used are shown in Table.1. We did not restrict the user's keyword selection.

To compare anytime-control used in a PWM system through all the ex-

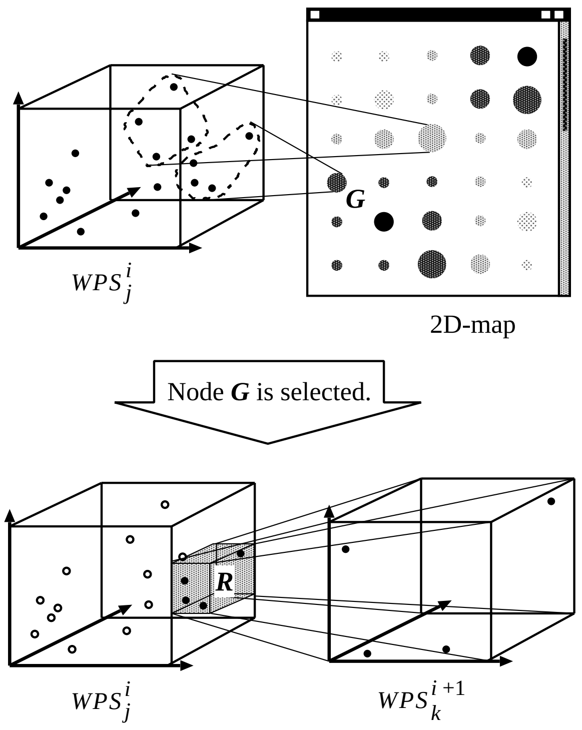


Fig. 8. Generating a WPS by user feedback.

periment, we used general *Web robot search* used in many search engines. In most search engine, Web robots are not control effectively and the strategy is a breadth first search. Thus we call it *Web robot search* described in the following.

Web robot search

(1) Initialize the Web page list $F = [\,]$.
(2) Set current Web pages α by starting Web pages P_0, and append all linked pages from α to F. The P_0 is generated in the same way to a PWM system described in Step (2) of section 2.6.
(3) Pick the head page β in F and remove it from F.
(4) Fetch the page β.
(5) Add all linked pages in β into the tail of F.
(6) Go to Step (1).

Two methods: a PWM system using anytime-control and a Web page gathering system using Web robot search were compared through the following identical procedures in same settings. The 2D-map consisted of 25 nodes (5×5), and three Web robots worked as different processes. We implemented the systems using Perl, Java, SQL and a parser program on Linux in a Celeron (433M Hz) PC-AT machine with 128M RAM.

Table 1. Keywords given by 10 subjects.

Subject	Keywords
p1	director, mystery, actress, movie, actor
p2	php3, postgreSQL, JDBC, JAVA
p3	genetic, programming, GP, intron, tree
p4	cribbage, crib, card, board, game, run, skunk, double
p5	linux, kernel, install, distribution, debian, dpkg
p6	agent, www, navigator, personalize, profile, AI
p7	pet, zoo, food, cat, fishing
p8	relation, learning, ontology, database, progol, foil, application
p9	multiple, robot, agent, cooperate, strategy
p10	slither, link, puzzle, faq, tips

4.1. *Evaluating uniformity of gathered Web pages*

For verifying that anytime-control can construct a uniform PWM, we investigate the standard deviation SD of the number of Web pages in each patch, which is described like the following formula. The x_i and $\overline{x}$ are the number of Web pages in a patch i and the average of them for all the patches. The n is the total of Web pages. The SD indicates the scattering of Web pages, thus a PWM is more uniform as the SD is smaller.

$$SD = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \overline{x})^2}$$

Fig.9 shows the experimental results. In the graph, the x-axis and y-axis stand for the number of gathered Web pages and SD. As seeing from the graph, anytime-control is able to keep a PWM more uniform than Web robot search as gathered Web pages increases.

The above uniformity was evaluated in a density blackboard. However we need an uniform 2D-map because a user can actually operate only it. Thus we also investigated the uniformity in a 2D-map. For each subject, Web page gathering using both of anytime-control and Web robot search were done for three hours, and 2D-maps were generated by SOM. Fig.10 shows the experimental results. The x-axis and y-axis stand for subjects and the standard deviation for the number of Web pages classified in nodes of a 2D-map. Seeing from this figure, the standard deviation of a 2D-map by anytime-control is less than that by Web robot search for all subjects. Hence we verified the uniformity of anytime-control in a 2D-map was also larger than that of Web robot search.

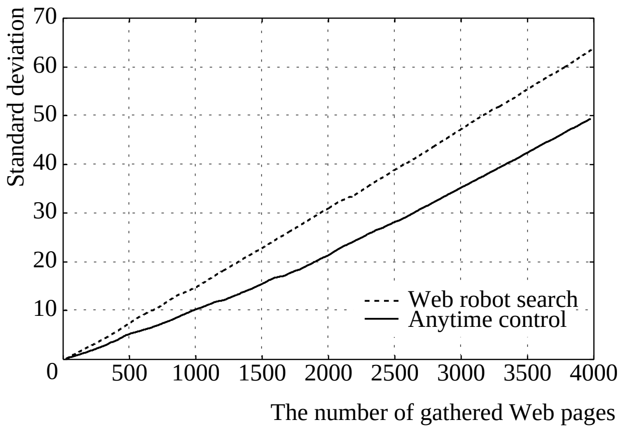


Fig. 9. Uniformity of gathered Web pages.

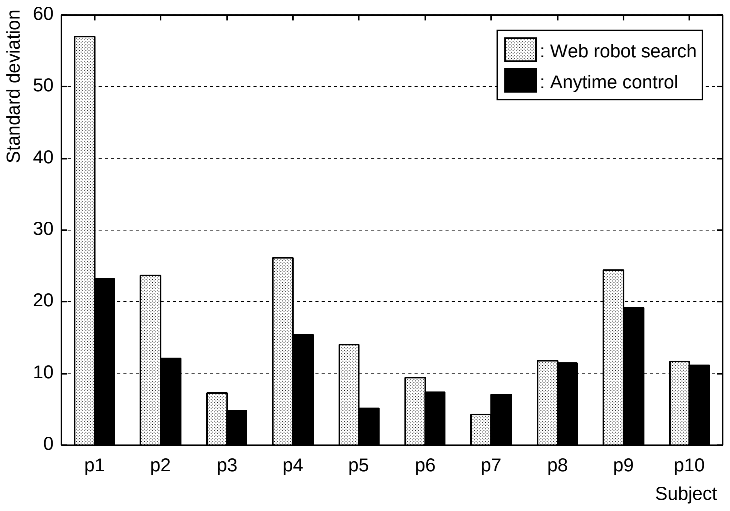


Fig. 10. Uniformity in a 2D-map.

4.2. *Evaluating the gathered relevant Web pages*

The purpose of a PWM system is to gather relevant Web pages to a user's interest effectively. Thus we made experiments for evaluating Web pages gathered by a PWM system. Then we investigate the number of relevant Web pages in gathered Web pages in the 1st and 2nd layer WPSs.

First Web page gathering was done in the first layer for three hours, and a 2D-map was generated. Then a subject selected a node on which he/she wants to know more in the 2D-map and next gathering was done in the second layer for three hours again. After gathering, a subject scores Web pages using the following scores.

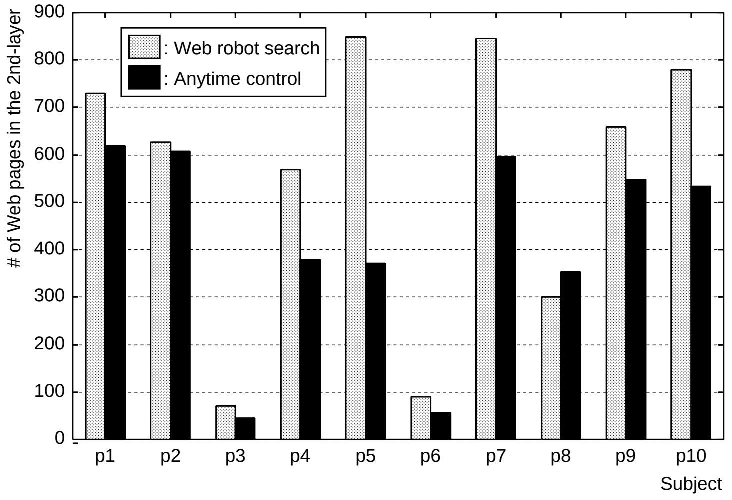


Fig. 11. The number of Web pages gathered in the 2nd-layer.

- 2 for a Web page which is *significantly* relevant to user's interest.
- 1 for a Web pages which is *slightly* relevant to user's interest.
- 0 for a Web pages which is not relevant to user's interest.

Fig.11 shows the number of Web pages gathered in the 2nd-layer. Seeing from the graph, the numbers of Web pages have large variance. We consider this variance is caused by the network traffic which is significantly dependent on Web sites. Another interesting results is that for almost subjects, the number of Web pages gathered by anytime-control is less than that by Web robot search. The reason for this results will be discussed in section 5.1.

Next we investigated the total scores for each subject and normalized them by dividing by the maximum score. The results are shown in Fig.12, where the x-axis and y-axis stand for subjects and normalized total scores. Seeing from the figure, anytime-control outperformed Web robot search for almost subjects. Thus we found out a PWM system is promising for gathering relevant information to user's interest in the WWW.

5. Discussion

We evaluated a PWM system through several experiments and verified the utility. In this section, we discuss limitation and open problems in our approach.

5.1. *Overhead for constructing a* PWM

In comparison with Web robot search, a PWM system needs additional processing like parsing Web pages, computing keyword frequency and updating a density blackboard. This costs significantly because the text processing is

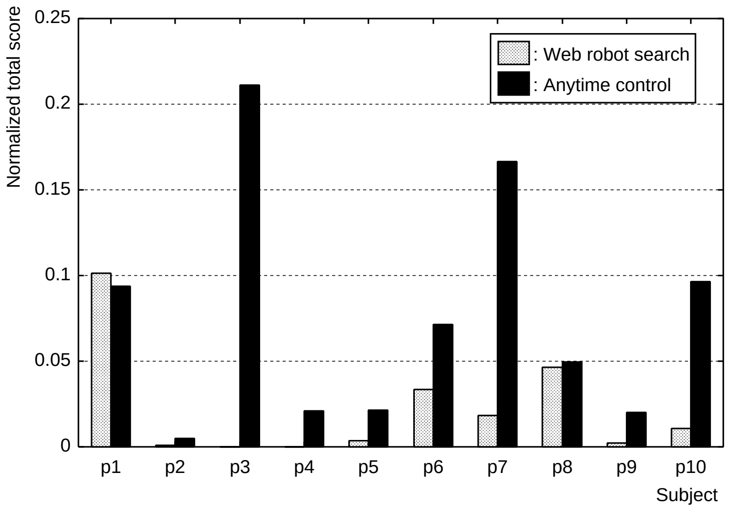


Fig. 12. The relevance of gathered Web pages.

generally expensive. This cost is why the number of Web pages gathered by anytime-control was less than that by Web robot search in Fig.11. It is our open problem to decrease the overhead.

5.2. *The timing to stop Web page gathering in the 1st-layer*

In a PWM system, a user can stop Web page gathering whenever he/she wants to do so. However, when should a user interrupt Web page gathering in the 1st-layer for obtaining the maximum relevant Web pages to his/her interest? Since this problem seems to be solved analytically, we made additional experiments by varying the ratio of the time for 1st-layer gathering and the time 2nd-layer one. Fig.13 shows the experimental results, where the x-axis and y-axis stand for the ratio of the 1st-layer gathering time to the 2nd-layer gathering time and normalized total scores. As seeing from this graph, we did not obtain any tendency. Thus we conclude the relevance of gathered Web pages to user's interest is hardly dependent on the timing to stop Web page gathering in the 1st-layer.

6. Conclusion

We proposed a PWM (Personal Web Map) for a user to gather interesting Web pages, and developed a PWM system consisting of PWM, multiple Web robots and a 2D-map. The PWM includes a Web pages database, tree-structured WPSs and a density blackboard. The system is able to do anytime-control for multiple Web robots to gather relevant Web pages effectively. For controlling Web robots, a density blackboard is used, and an uniform distributed PWM

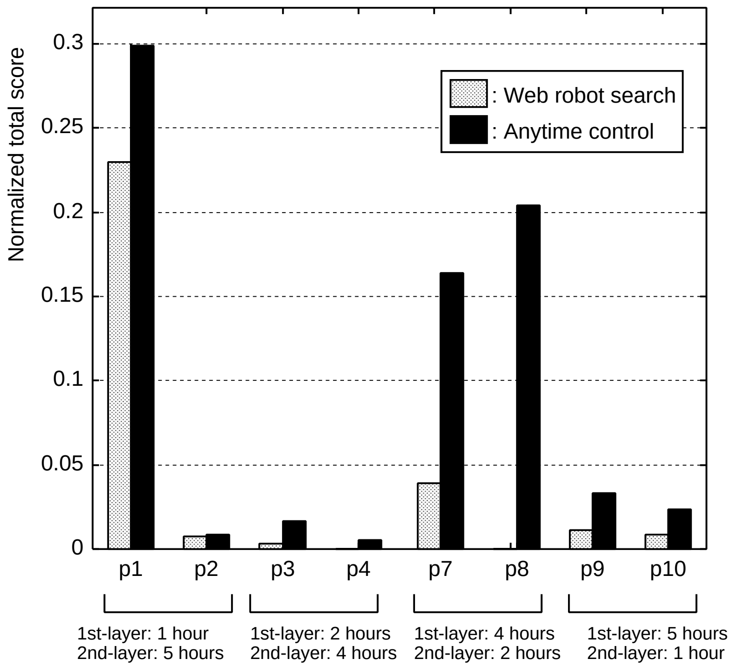


Fig. 13. The relevance of gathered Web pages as varying the timing to interrupt.

is built. Using anytime-control, the Web robots search the areas in which the least Web pages have been gathered in the keyword vector space. From Web pages in the database, document vectors are generated and used to SOM-based classification for constructing 2D-map. A user easily recognizes a PWM through the 2D-map, and gives feedback by selecting a node about which he/she wants more detail information. The selected node is expanded into a next-layer node including detail information.

We implemented a PWM system and made experiments by comparing with Web robot search. For evaluating the utility of anytime-control, the uniformity of gathered Web pages was investigate. Then the relevance of gathered Web pages to user's interest was also evaluated. As results, we found out that our PWM is a promising approach to assist a user in gathering the relevant information to his/her interest in the WWW.

Acknowledgments

The authors would like to acknowledge the suggestions of Prof. K. Nitta, Prof. S. Kobayashi, Prof. T. Kobayashi, Prof. M. Watanabe and Prof. H. Kinukawa.

This work has been supported by "Biologically Inspired Adaptive Systems" of Research for the Future Program in Japan Society for the Promotion of Science. We would like to express our gratitude to it.

References

1. M. Boddy and T. Dean, Solving time-dependent planning problems, in *Proc. of the 11th Int. Joint Conf. on Artificial Intell.* (1989) 979–984.
2. P. De Bra and R. Post, Information retrieval in the world-wide web: Making client-based searching feasible, in *the 1st Int. WWW Conference* (1994).
3. F.C. Cheong, *Internet Agents: Spiders, Wanderers, Brokers, and Bots* (New Riders, 1996).
4. R. B. Doorenbos, O. Etzioni, and D. S. Weld, A scalable comparison-shopping agent for the World-Wide Web, in *Proc. of the 1st Int. Conf. on Autonomous Agent* (1997) 39–48.
5. L. D. Erman, F. Hayes-Roth, V. R. Lesser, and D. R. Reddy, The Hearsay-II speech-understanding system: Integrated knowledge to resolve uncertainty, *Computer Surveys* **12** (1980) 213–253.
6. O. Etzioni and D. Weld, A SoftBot-based interface to the Internet, *Communication of the ACM* **37**, 7 (1994) 72–76.
7. T. Honkela, S. Kaski, K. Lagus, and T. Kohonen, Newsgroup exploration with WEBSOM method and browsing interface, TR-A32, Helsinki University of Technology of Computer and Information Science (1996).
8. T. Honkela, S. Kaski, K. Lagus, and T. Kohonen, WEBSOM—self-organizing maps of document collections, in *Proc. of WSOM'97, Workshop on Self-Organizing Maps* (1997) 310–315.
9. K. Jamsa, S. Lalani, and S. Weakley, *Web Programming*, (Jamsa Press, 1996).
10. T. Joachims, D. Freitag, and T. Mitchell, Webwatcher: A tour guide for the World Wide Web, in *Proc. of the 15th Int. Joint Conf. on Artificial Intell.* (1997) 770–775.
11. C. A. Knoblock, Planning, executing, sensing, and replanning for information gathering, in *Proc. of the 14th Int. Joint Conf. on Artificial Intell.* (1995) 1686–1693.
12. T. Kohonen, The self-organizing map, in *Proc. of the IEEE* (1990) 1464–1480.
13. T. Kohonen, *Self-Organization Maps* (Springer-Verlag, 1995).
14. C. T. Kwok and D. S. Weld, Planning to gather information, in *Proc. of the 13th National Conf. on Artificial Intell.* (1996) 32–39.
15. S. Lawrence and L. Giles, Accessibility and distribution of information on the Web, *Nature* **400** (1999) 107–109.
16. H. Lieberman, Letizia: A agent that assists Web browsing, in *Proc. of the 14th Int. Joint Conf. on Artificial Intell.* (1995) 924–929.
17. F. Menczer, ARACHNID: Adaptive retrieval agents choosing heuristic neighborhoods for information discovery, in *Proc. of the 14th Int. Conf. on Machine Learning* (1997) 227–235.
18. F. Menczer and A. E. Monge, Scalable Web search by adaptive online agents: an InforSpiders case study, in *Intelligent Information Agents*, eds. M. Klusch (Springer, 1999) 323–347.
19. MetaCrawler, **http://www.metacrawler.com/**.
20. R. C. Miller and K. Bharatb, Sphinx: a framework for creating personal, site-specific Web crawlers, *Computer Networks and ISDN Systems* **30** (1998) 1–7: FP12.
21. J. J. Rocchio, Relevance feedback in information retrieval, *The Smart system – experiments in automatic document processing* (Prentice Hall Inc, 1971) 313–323.
22. G. Salton, *Automatic Text Processing*, (Addison-Wesley, 1989).
23. S. Yamada and Y. Osawa, Planning to guide concept understanding in the WWW, in *AAAI 1998 Workshop on AI and Information Integration* (1998) 121–126.