ELSEVIER

Decision Support Systems 28 (2000) 269–277

Decision Support Systems

www.elsevier.com/locate/dsw

Intelligent internet searching agent based on hybrid simulated annealing

Christopher C. Yang [a,*], Jerome Yen [a], Hsinchun Chen [b]

[a] *Department of Systems Engineering and Engineering Management, The Chinese University of Hong Kong, Hong Kong, People's Republic of China*

[b] *Department of Management Information Systems, University of Arizona, Tucson, AZ, USA*

Abstract

The World-Wide Web (WWW) based Internet services have become a major channel for information delivery. For the same reason, information overload also has become a serious problem to the users of such services. It has been estimated that the amount of information stored on the Internet doubled every 18 months. The speed of increase of homepages can be even faster, some people estimated that it doubled every 6 months. Therefore, a scalable approach to support Internet searching is critical to the success of Internet services and other current or future National Information Infrastructure (NII) applications. In this paper, we discuss a modified version of simulated annealing algorithm to develop an intelligent personal spider (agent), which is based on automatic textual analysis of the Internet documents and hybrid simulated annealing.

Keywords: Information retrieval; Intelligent agent; Searching agent; Simulated annealing; World-wide Web

1. Introduction

Information searching over the cyberspace has become more and more important. It has been estimated that the amount of information stored on the Internet doubled every 18 months. However, the speed of increase of home pages can be even faster and it is doubled every 6 months or even shorter. In some areas, such as, Hong Kong and Taiwan, the increasing speeds can be even faster. Therefore, searching for the needed homepages or information has become a challenge to the users of Internet.

To develop searching engines or spiders, which are ''intelligent'', or to reach high recall and high precision is always the dream to the researchers in this area. In order to qualified as an agent or intelligent agent, such searching agent or spider must be able to make adjustments according to progress of searching or be personalized to adjust its behavior according to the users' preferences or behavior.

The major problem with the current searching engines or spiders is that only few spiders do have the communication capabilities between the spiders and the users who dispatched the spiders. Since there is no communication, the users are difficult to trace or to understand the progress of searching and have to tie themselves to the terminals. This paper reports

* Corresponding author.
E-mail address: yang@se.cuhk.edu.hk (C.C. Yang).

0167-9236/00/$ - see front matter
PII: S0167-9236(99)00091-3

a searching engine, which uses the combination of CGI and Java to develop the user interface. It allows the users to keep track of the progress of searching. The users can also make changes on the searching parameters, such as, number of homepages to retrieve. There were several algorithms have been used to develop spiders, for example, best-first searching and genetic algorithms. In this paper, we will discuss the spider that we developed with the hybrid simulated annealing algorithm. We have made a comparison to compare the performance of spiders that developed with best-first search. The results will be reported in this paper.

Although network protocols and Internet applications, such as, HTTP, Netscape and Mosaic, have significantly improved the efficiency and effectiveness of searching and retrieving of online information, their usage is still accompanied by the problems that users cannot explore and find what they want in the cyberspace. While Internet services become popular to the users, difficulties with searching on the Internet is expected to get worse as the amount of on-line information increases, the number of Internet users increases (traffic increases), and more and more multimedia are used to develop the home pages. This is the problem of information overload or information explosion.

Development of searching engines has become easier. For example, it is possible to download executable spider programs or even source codes. However, it is difficult to develop spiders which do have satisfactory performance or unique features, such as, learning and collaboration.

There are two major approaches to develop searching engines, either based on keywords and huge index tables, for example, Alta Vista and Yahoo, or based on hypertext linkages, for example, Microsoft Explorer and Netscape browser. It is difficult for the keyword search to reach high precision and high recall. Slow response due to the limitations on indexing methodology and network traffics, and the inability for the users to use the appropriate terms to articulate their need always become frustrate the users.

Our approach is based on automatic textual analysis of Internet documents, such as, HTML files, aims to address the Internet searching problem by creating intelligent personal spider (agent) based on the hybrid simulated annealing algorithm. Best-first search has been developed and reported in our earlier publications [3,4]. In this paper, we propose an approach based on automatic textual analysis of Internet documents and hybrid simulated annealing based searching engine.

In Section 2, a short literature review will be provided. Section 3 will discuss the architecture and algorithms for building our searching spider. Section 4 will report the experiments that we have conducted to compare its performance with the other searching spiders. This paper is concluded with some comments about our spider and other factors that will affect its performance.

2. Literature review: machine learning, intelligent agents

2.1. Machine learning

Research on intelligent Internet/Intranet searching relies significantly on machine learning. Neural network, symbolic learning algorithms and genetic algorithms are three major approaches in machine learning.

Neural network model computation in terms of complex topologies and statistics-based error correction algorithms, which fits well conventional information retrieval models such as vector space model and probabilistic model. Doszkocs et al. [7] gives an overview of connectionist models for information retrieval. Belew [1] developed a three-layer neural network of authors, index terms, and documents using relevance feedback from users to change its representation over time. Kwok developed a similar three-layer network using a modified Hebbian learning rule. Lin et al. [11] adopted the Kohonen's feature map to produce a two-dimensional grid representation for *N*-dimensional features.

Symbolic learning algorithms are based on production rule and decision tree knowledge representations. Fuhr et al. [8] adopted regression methods and ID3 for feature-based automatic indexing technique. Chen and She adopted ID3 and the incremental ID5R algorithm for constructing decision trees of important keywords which represent users' queries.

Genetic algorithms are based on evolution and heredity. Gordon [9] presented a genetic algorithms based approach for document indexing. Chen and Kim [5] presented a GA-neural-network hybrid system for concept optimization.

Our hybrid simulated annealing approach is similar to genetic algorithm in producing new generation. However, the selection is stochastic based instead of probability base.

2.2. Intelligent internet searching agent

There are two major approaches of Internet searching: (1) client-based searching agent, and (2) online database indexing and searching. There are also some systems contain both approaches.

A client-based searching agent on the Internet serves as a program that operates autonomously to search for relevant information without direct human supervision. Several software programs have been developed.

TueMosaic and the WebCrawler are two prominent examples. Both of them are using the Best First Search techniques. DeBra and Post [6] reported tueMosaic v2.42, modified at the Eindhoven University of Technology (TUE) using the Fish Search algorithm, at the First WWW Conference in Geneva. Using tueMosaic, users can enter keywords, specify the depth and width of search for links contained in the current homepage displayed, and request the spider agent to fetch homepages connected to the current homepage. The Fish Search algorithm is a modified Best First Search. Each URL corresponds to a fish. After the document is retrieved, the fish spawns a number of children (URLs). These URLs are produced depending on whether they are relevant and how many URLs are embedded. The URLs will be removed if no relevant documents are found after following several links. The searches are conducted by keywords, regular expressions, or by relevancy ranking with external filters. However, potentially relevant homepages that do not connect with the current homepage cannot be retrieved and the search space becomes enormous when the depth and breadth of search become large (an exponential search). The inefficiency and local search characteristics of the BFS/DFS-based spiders and the communication bandwidth bottleneck on Internet severely constrained the usefulness of such an agent approach.

At the Second WWW Conference, Pinkerton [14] reported a more efficient spider (crawler). The WebCrawler extends the tueMosaic's concept to initiate the search using its index and to follow links in an intelligent order. It first appeared in April of 1994 and was purchased by America Online in January of 1995. The WebCrawler extended the concept of the Fish Search Algorithm to initiate the search using index, and to follow links in an intelligent order. However, the WebCrawler evaluates the relevance of the link based on the similarity of the anchor text to the user's query. The anchor texts are the words that describe a link to another document. These anchor texts are usually short and do not provide relevance information as much as the full document texts. Moreover, problems with the local search and communication bottleneck persist. A more efficient and global Internet search algorithm is needed to improve client-based searching agents.

The TkWWW robot was developed by Scott Spetka and was funded by the Air Force Rome Laboratory [15] The TkWWW robots are dispatched from the TkWWW browser. The robot is designed to search Web neighborhoods to find logically related homepages and returns a list of links that look like a hot list. However, the search is limited to one or two hops, or links, from the original homepages. The TkWWW robots can also be run in the background to build HTML indexes, compile WWW statistics, collect a portfolio of pictures, or perform any other function that can be described by the TkWWW Tcl extensions. The major advantage of TkWWW robots over other existing spiders is its flexibility in adapting to virtually an criteria possible to guide its search path and control selection of data for retrieval.

The WebAnts, developed by Leavitt at Carnegie Mellon University, investigates the distribution of information collection tasks to a number of cooperating processors. The goal of WebAnts is to create cooperating explorers (ants) that share the searching results and the indexing load without repeating each other's effort. When an ant finds a document that satisfies the search criteria, it shares with other ants so that they will not examine the same document. For indexing, cooperation among ants allows each indexer to conserve resources by distribution the

indexing load between different explorers. During query, user could restrict the query to the local ant or allow it to propagate to the entry colony.

The RBSE (Respository Based Software Engineering) spider was developed by David Eichmann and was funded by NASA. RBSE spider was the first spider to index documents by content. It uses the Mite program to fetch documents and uses four searching mechanisms, (i) Breadth First Search from a given URL, (ii) Limited Depth First Search from a given URL, (iii) Breadth First Search from unvisited URLs in the database, and (iv) Limited Depth First Search from unvisited URLs in the database. The database of RBSE spider consists of three tables, Spider Table, Leaf Table, and Avoid Table. The Spider Table keeps track of URLs that have been visited, the Leaf Table contains node documents, which do not have links, and the Avoid Table contains a permanent list of patterns to match documents against for visit suppression.

An alternative approach to Internet resource discovery is based on the database concept of indexing and keyword searching. They retrieve entire Web documents or parts of these documents and store them on the host server. This information is then indexed on the host server to provide a server-based replica of all the information on the Web. This index is used to search for web documents that contain information relevant to a user's query and point the user to those documents.

The World Wide Web Worm (WWWW) [12], developed by McBryan, uses crawlers to collect Internet homepages and was made available in March of 1994. It was an early ancestor to the newer spices of spiders on the Web today. After homepages are collected, the system indexes their titles, subject headings and anchor texts and provides a UNIX grep-like routine for database keyword searching. However, it does not index the content of the documents. The title, subject headings and anchor texts do not always accurate description of the document's information. Moreover, WWWW does not create a large collection of relevant documents in the database due to the limitation of its crawlers and its grep-like keyword searching capability is very limited.

AliWeb [10], developed by Koster, adopts an owner-registration approach to collecting Internet homepages. Homepage owners write descriptions of their services in a standard format to register with AliWeb. AliWeb regularly retrieves all homepages according to its registration database. AliWeb alleviates the spider traversal overhead. However, the manual registration approach places a special burden on homepage owners and thus is unpopular. The AliWeb database is small and incomplete for realistic searches.

The Harvest information discovery and access system [2] developed by Bowman, et al., presents a big leap forward in Internet searching technology. The Harvest architecture consists of five subsystems, Gatherer, Broker, Index/Search subsystem, Object Cache, and Replicator. Gatherer collects indexing information and it is designed to run at the service provider's site, which saves a great deal of server load and network traffic. Gatherer also feeds information to many Brokers, which saves repeated gathering cost.

3. Intelligent agent for internet searching

Our intelligent Internet agent is based on automatic indexing and hybrid simulated annealing.

3.1. Indexing

The goal of indexing is identifying the content of the Internet document and the method includes three procedures: (a) word identification, (b) stop-wording, and (c) term-pharse formation.

Words are retrieved by word identification and stop-wording. Words are first identified by ignoring punctuation and case. A list of ''stop word'', which includes about 1000 common function words and ''pure'' verbs, is then deleted. The common function words are non-semantic bearing words, such as on, in, at, this, there, etc. The ''pure'' verbs are words, which are verbs only, such as calculate, articulate, teach, listen, etc. High frequency words that are too general to be useful in representing document content are also deleted.

After identifying words and removing stop words, adjacent words remaining in the document are then used to form pharses. Pharses are limited to three words. The resulting pharses and words are referred as the keywords of the Internet documents. The

occurrence pattern of indexes, which appears in all documents, is identified.

We use Jaccard's score to measure the similarity of Internet documents. The score is computed in terms of the documents' common links and indexing. A document with a higher Jaccard's score has a higher fitness with the input document.

The Jaccard's score based on links are computed by dividing the number of common links by the number of total links of two documents. Given two documents, x and y, and their links, $X = x_1, x_2, \ldots, x_m$, and $Y = y_1, y_2, \ldots, y_n$, the Jaccard's score between x and y based on links is:

$$\mathrm{JS}_{\mathrm{links}}(x,y) = \frac{\#(X \cap Y)}{\#(X \cup Y)} \quad (1)$$

The Jaccard's score based on indexing is computed in terms of the term frequency and document frequency. Given a set of Internet documents, the term frequency and the homepage frequency for each term in a document. Term frequency, tf_{xj}, is the number of occurrences of term j in document x. Document frequency, df_j, is the number of document in a collection of N documents in which term j occurs. The combined weight of term j in document x, d_{xj}, is:

$$d_{xj} = \mathrm{tf}_{xj} \times \log\left(\frac{N}{\mathrm{df}_j} \times w_j\right) \quad (2)$$

where w_j is the number of words in term.

The Jaccard's score between document x and y based on indexing is then computed as:

$$\mathrm{JS}_{\mathrm{index}}(x,y) = \frac{\sum_{j=1}^{L} d_{xj} d_{yj}}{\sum_{j=1}^{L} d_{xj}^2 + \sum_{j=1}^{L} d_{yj}^2 + \sum_{j=1}^{L} d_{xj} d_{yj}} \quad (3)$$

where L is the total number of terms.

3.2. Search engine

In this section, we present two searching algorithms, best first search and our proposed algorithm hybrid simulated annealing.

3.2.1. Best first search

Best first search is a state space search algorithm [13]. We explore the best homepage at each iteration. The best homepage is the homepage with the highest Jaccard's score. The iteration is terminated when the number of homepages required by the user is obtained. The algorithm is summarized as follows and illustrated in Fig. 1.

- *Initialization*: Initialize the counter k to 0. Obtain a set of anchor homepages, $A = (A_1, A_2, \ldots)$, and the number of desired homepages from the users, D. Fetch the anchor homepages and save the links of the anchor homepages to the unexplored set of homepages, $H = (H_1, H_2, H_3, \ldots)$.
- *Find the best homepage*: Fetch the unexplored homepages and compute their Jaccard's scores. The homepage that has the highest Jaccard's score is the best homepage. The best homepage is then save in the output set of homepages.
- *Explore the best homepage*: Fetch the best homepage that is determined in the last step and insert its links to the unexplored set of homepages. Increment the counter by 1.
- *Repeat step 2 and 3 until $k = D$.*

3.2.2. Hybrid Simulated annealing

Simulated annealing algorithm is based on analogy between the simulation of the annealing of solids and the problem of solving large combinatorial optimization problems. A high temperature is initialized at the first iteration and a set of documents (configuration) is generated at each iteration.

We use Jaccard's score as the cost function to evaluate the current configuration. If the score is higher than that of previous configuration, the new configuration is accepted, otherwise, it will be ac-

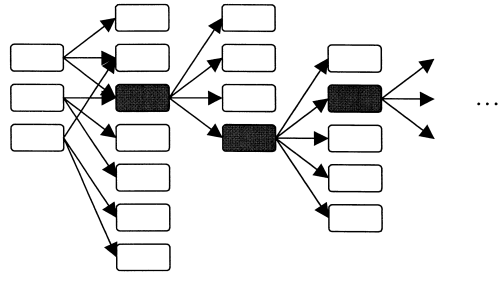

Fig. 1. Best first search.

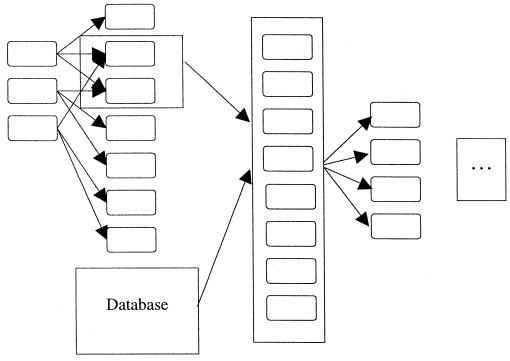

Fig. 2. Hybrid simulated annealing.

cepted based on the probability computed in terms of the current temperature and the Jaccard's score. We decrease the temperature at the beginning of each iteration. The process will terminate when the change of Jaccard's score between two consecutive iterations is less than a threshold or the temperature reaches a certain value.

A sketch of hybrid simulated annealing is presented as below.

• *Initialization*: The search engine takes a set of documents (homepages with their URLs), input_1, input_2, ..., from users and save into a set of CurrentConfiguration, $\text{CC} = \{\text{cc}_1, \text{cc}_2, \ldots\}$. Also Temperature, T, is initialized to a large value.

• *Generation of new configuration*: Fetch the documents that linked by the documents in the CurrentConfiguration. Compare the linked documents of these fetched documents and the linked documents of the documents in CurrentConfiguration. The fetched documents, which have the highest number of overlapping linked documents with the documents in CurrentConfiguration, will be saved in a set, $A = \{a_1, a_2, \ldots,\}$.

Table 1
The statistics of the average Jaccard's scores obtained from 40 cases of searching by hybrid simulated annealing and best-first search

	Hybrid simulated annealing	Best first search
Mean	0.08703	0.04138
Standard deviation	0.08519	0.04996

Table 2
Precision statistics of user evaluation

Number of input homepages		Hybrid simulated annealing	Best first search
1	mean	0.75	0.70
	standard deviation	0.061	0.073
2	mean	0.75	0.69
	standard deviation	0.057	0.070
3	mean	0.77	0.72
	standard deviation	0.055	0.071
4	mean	0.80	0.73
	standard deviation	0.060	0.065

Documents are also obtained from the ranked documents using SWISH in the user selected category in our database and saved in a set, $B = \{b_1, b_2, \ldots\}$.

The sizes of A and B are both half of the size of CurrentConfiguration and the size of CurrentConfiguration is double of the number of requested output documents.

Compute the Jaccard's score for each document in CurrentConfiguration as follows:

$$\text{JS}_{\text{links}}(\text{cc}_i) = \frac{1}{N}\sum_{j=1}^{N} \text{JS}_{\text{links}}(\text{input}_j, \text{cc}_i)$$

$$\text{JS}_{\text{index}}(\text{cc}_i) = \frac{1}{N}\sum_{j=1}^{N} \text{JS}_{\text{index}}(\text{input}_j, \text{cc}_i)$$

$$\text{JS}(\text{cc}_i) = \frac{1}{2}\left(\text{JS}_{\text{links}}(\text{cc}_i) + \text{JS}_{\text{index}}(\text{cc}_i)\right)$$

Table 3
Recall statistics of user evaluation

Number of input homepages		Hybrid simulated annealing	Best first search
1	mean	0.52	0.50
	standard deviation	0.024	0.029
2	mean	0.54	0.51
	standard deviation	0.022	0.032
3	mean	0.57	0.54
	standard deviation	0.025	0.025
4	mean	0.61	0.57
	standard deviation	0.030	0.029

Compute the average Jaccard's score, Avg, of the documents in CurrentConfiguration.

$$\text{Avg} = \frac{1}{M}\sum_{i=1}^{M} \text{JS}(\text{cc}_i) \quad (4)$$

For each document in *A* and *B*, if the Jaccard's score of the document is higher than Avg, save the current document in CurrentConfiguration. Otherwise, if

$$e^{-\frac{\text{JS}(x)-\text{Avg}}{T}} \geq \text{rand}$$

where *x* is an element of *A* or *B*, and rand is a random value between 0 and 1. Save the current document in CurrentConfiguration.

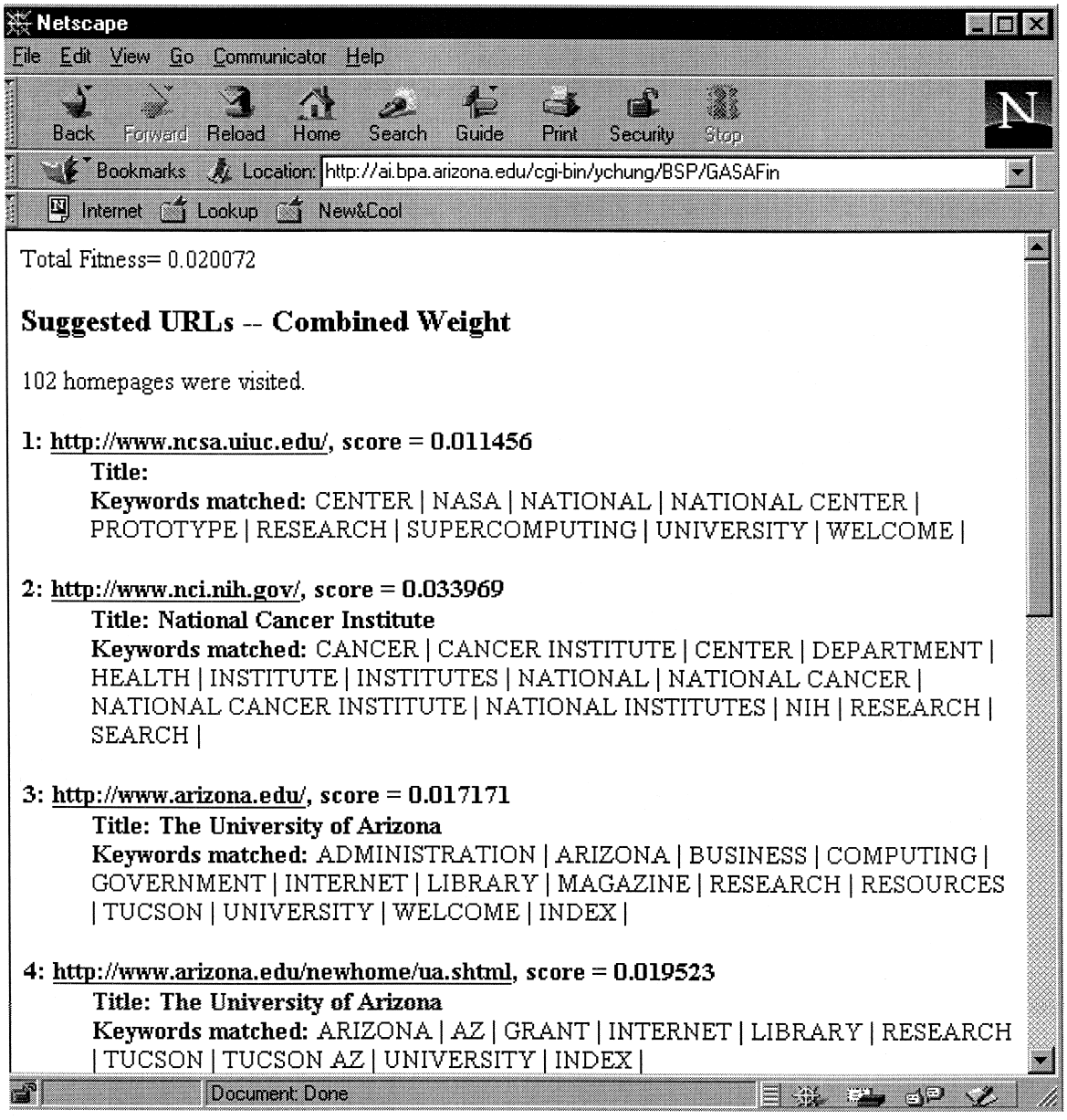

Fig. 3. The homepage of a searching result.

Decrease the temperature, T, and repeat the same procedure until T reaches a certain value or the change of Avg is less than a threshold (Fig. 2).

4. Experimental result

In order to examine the quality of results obtained by hybrid simulated annealing algorithm, we have conducted an experiment to compare the performance of the simulated annealing spider with the spiders that built with best-first search.

In our experiment, 40 cases are set up. For each case, 1 to 4 input homepages are submitted to the spiders based on hybrid simulated annealing and best-first search. Ten output homepages obtained as the result of searching. Homepages are chosen in the entertainment domain. The average of the output homepages' Jaccard's score for each case is recorded for comparing their fitness. In the experiment, we also recorded whether the output URL is originated from database. This will give us an idea on the percentage of the output URLs that are contributed by the database in the global search.

Table 1 shows the statistics of the fitness on the results obtained by the hybrid simulated annealing and best-first search performed on the 40 cases. The results show that the output homepages obtained by hybrid simulated annealing has a slightly higher fitness score than those obtained by best-first search, but the difference is not significant. The averages of 40 average Jaccard's scores for hybrid simulated annealing and best-first search are 0.08703 and 0.08519, respectively.

A user evaluation is also conducted. Each case has 8 subjects evaluating its relevance. It shows that higher precision and recall is obtained comparing with the best-first search (Tables 2 and 3).

Although the Jaccard's score does not show any significant difference among the performances of hybrid simulated annealing and best-first search, 50% of the homepages that obtained from hybrid simulated annealing are not originated from the input homepages. These homepages that obtained from our database (homepages in the set B) are most probably not linked to the input homepages, therefore, most of them will never be obtained by best-first search. In particular, when the number of links of the input URLs is restricted, the result of best-first search is poor and the best-first search may not be able to provide as many URLs as requested by the users for result. However, the hybrid simulated annealing does not have this problem because it is a global search. In the situation of restricted links of the input URLs, the hybrid simulated annealing is still performing very well, a significant difference between their performance is observed.

The result also shows that the Jaccard's score increase when the number of input homepages increase. The information provided to the agent increase as the number of input homepages increases. Given more information, the searching agent is able to understand more about the users such as their interests and expectation. It also the same reason why the intelligent searching agent has a better performance than the traditional search engine, which only takes keywords input.

Fig. 3 shows the homepage which display the final result of the search. It displays the total average fitness (based on Jaccard's score) and the number of homepages has been visited in the search. For each homepage, its address, score, title, and matched keywords are also displayed.

5. Conclusion

We present a hybrid simulated annealing based intelligent Internet agent for searching relevant homepage on the world-wide web. It supports a global search on the web. The spider obtains a set of homepages from users and search for most relevant homepages. It operates autonomously without any human supervision. We have conducted an experiment and it shows that a higher fitness score is obtained by the hybrid simulated annealing comparing with the best-first search. From user evaluation, we also find that higher precision and recall is obtained.

References

[1] R.K. Belew, Adaptive information retrieval, in: Proceedings of the Twelfth Annual International ACM/SIGIR Confer-

ence on Research and Development in Information Retrieval, Cambridge, MA, June 22–28, 1989.
[2] C.M. Bowman, P.B. Danzig, U. Manber, F. Schwartz, Scalable internet resource discovery: research problems and approaches, Communications of the ACM 37 (8) (1994) 98–107, August.
[3] H. Chen, Y. Chung, M. Ramsey, C.C. Yang, A Smart Itsy Bitsy Spider for the Web, Journal of the American Society for Information Science 49 (7) (1998) 604–618.
[4] H. Chen, Y. Chung, M. Ramsey, C. C. Yang, An Intelligent Personal Spider (Agent) for Dynamic Internet/Intranet Searching, Decision Support Systems 23 (1) (1998) 41–58.
[5] H. Chen, J. Kim, GANNET: a machine learning approach to document retrieval, Journal of Management Information Systems 11 (3) (1995) 7–14, Winter.
[6] P. DeBra, R. Post, Information retrieval in the World-Wide Web: making client-based searching feasible, in: Proceedings of the First International World Wide Web Conference, Geneva, Switzerland, 1994.
[7] T.E. Doszkocs, J. Reggia, X. Lin, Connectionist models and information retrieval, Annual Review of Information Science and Technology (ARIST) 25 (1990) 209–260.
[8] N. Fuhr, S. Hartmann, G. Knorz, G. Lustig, M. Schwantner, K. Tzeras, AIR/X — a rule-based multistage indexing system for large subject fields, in: Proceedings of the Eighth National Conference on Artificial Intelligence, Boston, MA, 1990.
[9] M. Gordon, Probabilistic and genetic algorithms for document retrieval, Communications of the ACM 31 (10) (1988) 1208–1218, October.
[10] M. Koster, ALIWEB: Archie-like indexing in the Web, in: Proceedings of the First International World Wide Web Conference '94, Geneva, Switzerland, 1994.
[11] X. Lin, D. Soergel, G. Marchionini, A self-organizing semantic map for information retrieval, in: Proceedings of the Fourteenth Annual International ACM/SIGIR Conference on Research and Development in Information Retrieval, Chicago, IL, October, 13–16, 1991.
[12] O. McBryan, GENVL and WWW: tools for taming the web, in: Proceedings of the First International World Wide Web Conference '94, Geneva, Switzerland, 1994.
[13] J. Pearl, Heuristics: Intelligent Search Stategies for Computer Problem Solving, Addison-Wesley Publishing, Reading, MA, 1984.
[14] B. Pinkerton, Finding what people want: experiences with the webcrawler, in: Proceedings of the Second International World Wide Web conference, Chicago, IL, October 17–20, 1994.
[15] S. Spetka, The TkWWW robot: beyond browsing, in: Proceedings of the Second World Wide Web Conference '94: Mosaic and the Web, October 17–20, 1994.

Christopher C. Yang is an assistant professor in the Department of Systems Engineering and Engineering Management at the Chinese University of Hong Kong starting from 2000. He was an assistant professor in the Department of Computer Science and Information Systems and associate director of the Authorized Academic Java Campus at the University of Hong Kong from 1997 to 1999. He received his BS, MS, and PhD in Electrical Engineering from the University of Arizona, Tucson, AZ, in 1990, 1992, and 1997, respectively. From 1995 to 1997, he was a research scientist in the Artificial Intelligence Laboratory in the Department of Management Information Systems, where he was active in the Illinois Digital Library project during the Digital Library Initiative I. His current research interests are digital library, cross-lingual information retrieval, Internet agent, information visualization, and color image retrieval. He was the program co-chair of the First Asia Digital Library Workshop. In 1998 and 1999, he was an invited panelist of the NSF Digital Library Initiative II Review Panel.

Jerome Yen is an associate professor of the Department of Systems Engineering and Engineering Management at the Chinese University of Hong Kong and also an adjunct associate professor of the School of Computer Science at the Carnegie Mellon University. He received his PhD degree in Systems Engineering and Management Information Systems from the University of Arizona in 1992. His current research interests include digital library, information economics, information retrieval, intelligent agents and financial engineering. He has published more than twenty journal articles in international journals, such as, Journal of Management Information Systems, IEEE Computer, JASIS, Journal of Mathematical Economics, and International Review of Economics and Finance. He is the recipient of the best paper award at the 28th Hawaii International Conference on System Sciences. He is the Program Chair of the First Asia Digital Library Workshop.

Hsinchun Chen is a Professor of Management Systems at the University of Arizona and head of the UA/MIS Artificial Intelligence Group. He is also a Visiting Senior Research Scientist at National Center for Supercomputing Applications (NCSA). He is a PI of the Illinois Digital Library Initiative project, funded by NSF/ARPA/NASA, 1994–1998, and has received several grants from NSF, DARPA, NASA, NIH, and NCSA. He is the guest editor of IEEE Computer special issue on ''Building Large-Scale Digital Libraries'' and the Journal of the American Society for Information Science special issue on ''Artificial Intelligence Techniques for Emerging Information Systems Applications''.