제 18 회 한국정보처리학회 추계학술발표대회 논문집 제 9 권 제 2 호 (2002.11)

지능에이전트 기법을 이용한 검색엔진개발에 관한 연구

하남 응우옌*, 최규석*, 박종진**
*청운대학교 정보산업대학원
**청운대학교 인터넷컴퓨터학과
e-mail : namnhvn@chungwoon.ac.kr
namnhvn@yahoo.com

WebSearcher:A Study on Development of Information Retrieval system using Intelligent Agent Technology

Ha-Nam Nguyen*, Gyoo-Seok Choi*, Jong-Jin Park**
* Dept. of Information Technology, Graduate School of Chungwoon University
**Dept. of Internet Computer Engineering, Chungwoon University

Abstract

The dynamic nature of the World Wide Web challenges Information Retrieval System to find information relevant and recent. Intelligent agents can complement the power of search engines to deal with this challenge. In this paper, we explain in manner of building Information Retrieval System based on intelligent agent technology. We present a tool called WebSearcher. It was performed in Java environment. The object-oriented nature of Java and built-in facilities for multi-thread decreased our implementation effort. A modular software design makes it easy to configure the system for various experiments.

1. Introduction

In the past, most information in web pages was kept up to date in search engines' database. We were able to find any pages containing the information we were looking for by using these search engines. Nowadays, the dynamic, irregular nature and rapid proliferation of the WWW have made searching for useful information on Internet increasingly difficult. No search engine can perform a thorough search on the whole WWW. Challenges of current Information Retrieval Systems related with characteristics of the web data sources are:

- Huge and ubiquitous
- Mostly semi-structured or unstructured
- Diverse in quality
- Dynamic
- Distributed and autonomous

There are hundreds of Information Retrieval (**IR**) Systems available on Internet. But the best search engines can only index approximately 20% or 30% of web pages available on the WWW. Hence, it is very difficult to tell which of them is the best. If you can not receive any valid results from one search system, you should try others. It can cause difficulties for people, who interact with IR system frequently, because of different interfaces

Till now, there are many search systems using Intelligent Agent Technology. They are divided into two major types:

- *Web crawlers* use basic information retrieval techniques to automatically gather information from indexed Web pages (spiders, softbots, automatic indexes, crawlers), maintain and periodically update their own index database, and provide a rating-based one time query answering mechanism to the user.
- *Meta-search* execute a given query concurrently over a variety of search engines available on inter-net, then merge and present the results in a hom-ogeneous, ranking-based view to the user.

We do not focus on how to build an Information Retrieval Systems faster and stronger than those systems. We wish to

build a new system that is easier, more useful and has interfaces more friendly to users. WebSearcher focuses on three main aspects, which are combined into one system:

First, adopting the *meta-search* approach, it executes queries concurrently on many search engines. These queries were predefined in the system's *knowledge-base*. After that, it receives, combines and removes duplicate results from all web-accessible search engines. The result set will serve as the initial set of web addresses for following steps.

Second, WebSearcher uses *web-crawler* technique to perform deeper search on the World Wide Web. It opens and downloads each HTML document in order of initial set. WebSearcher then pulls out all of the links contained within the document. These links are added into initial set without duplicate value.

Third, a *ranking algorithm* is used to analyze content of the retrieved addresses, which helps users in selecting the best ones from set of results. Ranking algorithms are used to evaluate the degree of relevance of web pages. The higher score a page gets, the more relevant it could be for the user's requests. A good ranking algorithm is very helpful to crawl the Web more efficiently. Meta-search engines also need ranking algorithms to combine the results from other search engines.

2. Intelligent Agent – A New Technology For IR

What is an agent? "An agent is anything that can be viewed as perceiving its environment through sensors and acting upon that environment through effectors."[1]

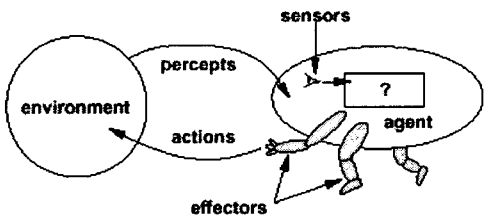


Figure 1: Agents interact with environments.

For Web searching agents, the environment is the World Wide Web for searching and the computer terminal for interacting with the user. The agent's percepts are the content of HTML documents retrieved from software sensors, which connect to the WWW (i.e., Internet) with HTTP Protocol. An agent's reasoning determines whether a web page contains targeted key words or phrases or not. If not, it goes to other pages and continues the search to complete the goal. It feedbacks to the environment by using output methods to return results to the user.

What exactly makes an agent "intelligent" is something that is hard to define. It has been the subject of many discussions in the field of Artificial Intelligence, and a clear answer has yet to be found. A workable definition of what makes an agent intelligent is given in [3]:

"Intelligence is the degree of reasoning and learned behaviour: the agent's ability to accept the user's statement of goals and carry out the task delegated to it.

At a minimum, there can be some statement of preferences, perhaps in the form of rules, with an inference engine or some other reasoning mechanism to act on these preferences.

Higher levels of intelligence include a user model or some other form of understanding and reasoning about what a user wants done, and planning the means to achieve this goal.

Further out on the intelligence scale are systems that learn and adapt to their environment, both in terms of the user's objectives, and in terms of the resources available to the agent. Such a system might, like a human assistant, discover new relationships, connections, or concepts independently from the human user, and exploit these in anticipating and satisfying user needs."

3. The architecture of WebSearcher

WebSearcher consists of five components: user interface, transformation, interface agent, display agent and knowledge base.

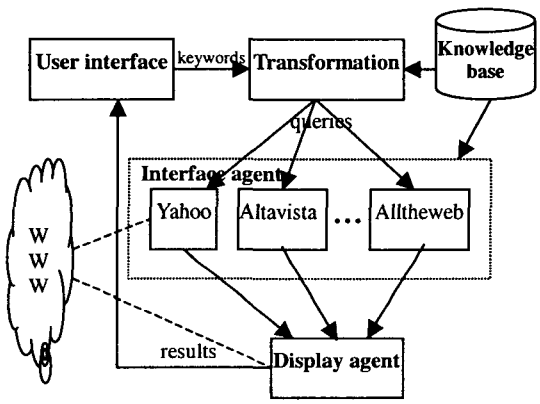


Figure 2: Architecture and interacting among components of WebSearcher

User interface interacts with users to receive keywords and displays results after the system finishes searching.

Before sending any queries to web accessible search engines, the system refines keywords entered by users. These words sometimes contain stop words. Three *transformations*[4] can be applied to keywords in order to improve the quality of searching and ranking results.

- Case Standardization. Words are represented using only in lower case letters. When this process is completed, these words "Internet", "internet", and "INTERNET" just have only single form "internet".
- Stop words (also called noise words) are eliminated. Small words, such as "the", "if" and "as", usually occur in high frequency throughout the collection, and thus would result in extremely long lists.
- Suffixes or variations in the form of a word are stripped. For example "consulting", "consulted" and "consultation" could all be represented in the index by "consult". Since the transformations are applied to queries, a query containing any one of theses terms will match documents that contain any one of the original forms. The suffix removal process,

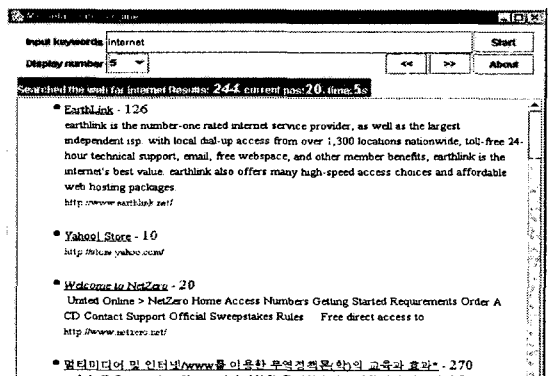
called *stemming*, is carried out by applying a set of heuristic rules that are specific to a particular language, like English.

Only the first two transformations were implemented in our system. We planned to include the third transformation in next version of the WebSearcher.

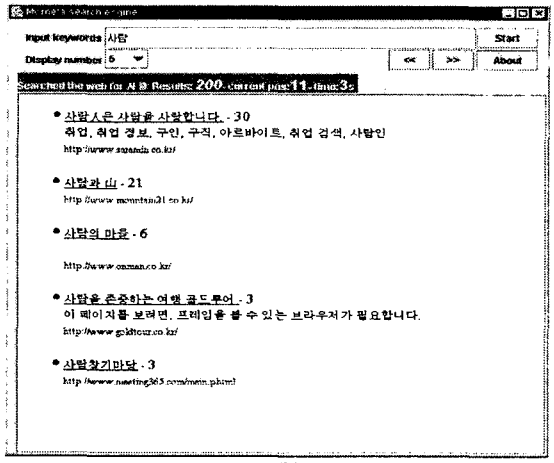
Each *Interface agent* translates the query to a form appropriate to its Internet accessible search engine, submits it and retrieves the results. Each search engine has a different interface and capabilities. We build one interface agent for each search engine that we can query. These agents are designed to gracefully respond to exceptional condition such as errors or unexpected results. Results are forwarded to the display agent in the same format as in the initial set.

The *Knowledge-base* contains useful data for *Transformation* and *Interface* parts. Specifically, it stores all *Stop words*, query structures of accessible web search engine and output's format of each search engine. We can add more entries to *Knowledge-base*.

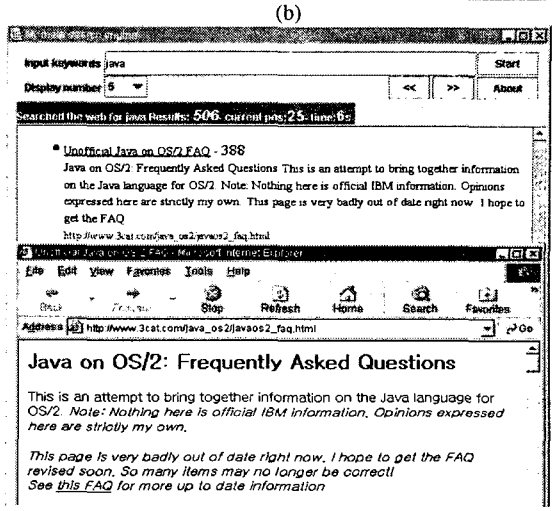
The *display agent* organizes results from interface agents and displays them to the user in an easy-to-understand form. There is a trade-off between response time and helpful order of results. Because it takes long time to rank on all links in initial set, we only perform ranking function on subset of it. By this way, we can not display in good order of results. The important tasks of display agent is ranking and meta-search on the web pages. The agent tries to open the web documents, which store in initial set. If that hyperlink is not exists, it tries another. Then, a display agent's reasoning (using ranking algorithm) determines whether a web page contains targeted key words or not. If not, it opens another hyperlink in order of initial set. It only accepts to return to users interface hyperlinks, which exist and contains keywords. It is easy to realize that display agent makes a rational decision, when it chooses the method to return results. And when agent scans content of documents, it simultaneously pulls out all of the links contained within the document. These links are added into initial set without duplicate value.



(a)



(b)



(c)

Figure 3: The user interface of WebSearcher during process. (a) Search on English keywords. (b) Search on Korean keywords. (c) View the top hit by clicking on underlines string (Title of URL) in the results window.

4. Searching Algorithm

Searching algorithm is a multi-agent system that performs online, dynamic Web search. Each agent checks information in List_of_Url (*initial set*), looking for new documents relevant to the user's keywords and interacting little with other agents. Searching algorithm performs tasks as follows (Figure 4):

1. It tries to open the hyperlink. If that hyperlink does not exist, it tries the next one in List_of_Url, i.e. the initial set of results.
2. It calculates keyword frequency(*ranking*) and collects more hyperlinks(*meta-search*) on that document.
3. Return step 1 until output buffer is full or it reaches to upper boundary of List_of_Url.

(ResNum: output buffer elements; DisNum: sum of returning results or max of output buffer elements; Pos: current position in List_of_Url; List_of_Url: sum of initial set elements)

One process only interacts with the others when updating new hyperlinks into List_of_Url. It updates List_of_Url only when new hyperlinks that are not yet in List_of_Url have been collected.

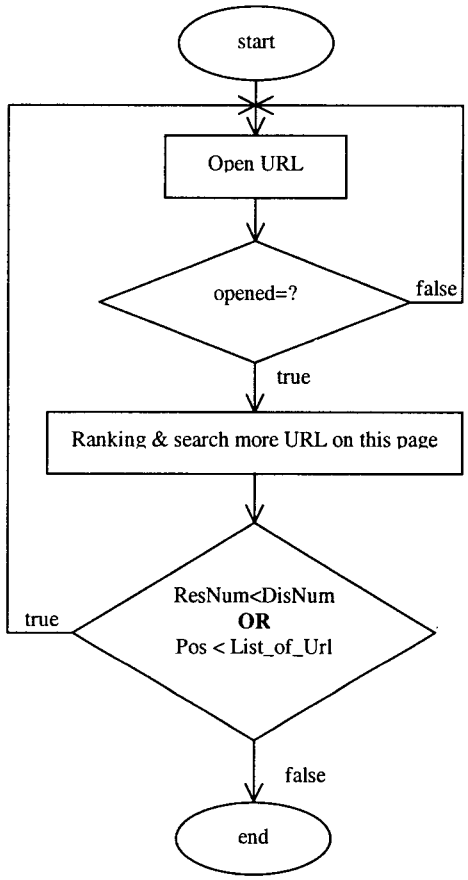


Figure 4: Block diagram of searching algorithm

Details of the ranking algorithm are given in Figure 5.

```
While not(EOF){
  word := <Read word by word>
  if (word ==<META tag>) {
    <Read next word>
    if (word ==keyword)
      score += META_MARKS
      if (<word contain keyword> OR
          <word is a part of keyword> )
        score += META_MARKS_NEARBY
    }
  if (word == <BODY tag>) {"
    <Read next word>
    if (word ==keyword)
      score += BODY_MARKS
      if (<word contain keyword> OR
          <word is a part of keyword> )
        score += BODY_MARKS_NEARBY
      if (word==<HYPERLINKS tag>)
        UpdateList_Of_Url()
  }
}
if (AllMatch==true)
  score += BONUS_MARKS
Update_Disp_Result(Title, Desc, links, score)
```

Figure 5: Pseudo-code of ranking algorithms

5. Future works

There are several directions for future work with WebSearcher. The first direction is to categorize web accessible Search engines according to their area of strength, then use the result as a criteria to select which search engines to contact for each particular queried topic. One possibility is to use learning-base approach similar to SavvySearch[5], where our system learns which search engines to be used based on previous experiences.

Other direction of research is to improve output methods; currently, we support ranking by words-match. Other options such as PageRank method [6] are being explored.

6. Conclusion

We have built the Information Retrieval System in Java environment base on intelligent agent technology. We have combined *meta-search, crawler* approachs and *ranking* algorithm into unique system. WebSearcher presents users with a single interface that receives requirement of users. After that, it transforms that requirement and built the queries for each web accessible Search engines. It then downloads those results and removes duplicate values into internal single list. At the end, WebSearcher built display results from this internal list. This task simultaneously implement ranking and crawling on each links in internal single list. The user need know only *what* he or she is looking for and our system takes care of *how* and *where*.

Reference

[1] Stuart Russell, and Peter Norvig. "Artificial Intelligent - A model approach". Prentice hall, Upper Saddle River, N.J., 1995

[2] Gerhard Weiss. "Multiagent Systems - A Modern Approach to Distributed Artificial Intelligence". The MIT Press Cambridge, Massachusetts, London, England, 2000

[3] Gilbert, Aparicio, et al. "The Role of Intelligent Agents in the Information Infrastructure". IBM, United States, 1995.

[4] Bronson Trevor, Edgar Weippl, and Werner Winiwarter. "A Modern Approach to Searching the World Wide Web: Ranking Pages by Inference over Content". 2001

[5] Daniel Dreilinger. "Thesis: Description and evaluation of a meta-search agent". Colorado State University, December 1996

[6] Brin, Motwani, Winograd. "The PageRank Citation Ranking: Bringing Order to the Web". January 29, 1998