An Intelligent Search Modeling using Avatar Agent

Dae Su Kim[1] Kwang-Baek Kim[2], and Chang-Suk Kim[3]

[1]Dept. of Computer Science, Hanshin University, Korea
[2]Dept. of Computer Engineering, Silla University
[3]Dept. of Computer Education, Kongju University
E-mail: daekim@hanshin.ac.kr

Abstract— **In this paper, we proposed an intelligent search modeling using avatar agent. Our system consists of some modules such as agent interface, agent management, preprocessor, interface machine. Core-Symbol Database and Spell Checker are related to the preprocessor module and Interface Machine is connected with Best Aggregate Designer. Our avatar agent system does the indexing work that converts user's natural language type sentence to the proper words that is suitable for the specific branch information retrieval. Indexing is one of the preprocessing steps that make it possible to guarantee the specialty of user's input and increases the reliability of the result. It references a database that consists of synonym and specific branch dictionary. The resulting symbol after indexing is used for draft search by the internet search engine. The retrieval page position and link information are stored in the database. We experimented our system with the stock market keyword SAMSUNG_SDI, IBM, and SONY and compared the result with that of Altavista and Goole search engine. It showed quite excellent results.**

Index Terms—**Avatar system, rank, Core-Symbol Database, Best Aggregate Designer, Preprocessor, Matching Module**

I. INTRODUCTION

Recently, lots of information retrieval system is widely used through World Wide Web[4,5,7]. Some web agent systems are now under developing that analyzes user's simple keyword or category retrieval and returns proper retrieval results that users want[3,6,8]

Now a web agent system like Goole has a rank structure using feedback among related documents and it upgrades accuracy of user's request. Some web agents employ the strategy that the importance of a document is decided by the number of links to that document[2,3]

Our avatar agent system does the indexing work that converts user's natural language type sentence to the proper words that is suitable for the specific branch information retrieval. Indexing is one of the preprocessing steps that make it possible to guarantee the specialty of user's input and increases the reliability of the result. It references a database that consists of synonym and specific branch dictionary.

The resulting symbol after indexing is used for draft search in the internet search engine. The retrieval page position and link information are stored in the database. But there is no general decision criteria which document is more authoritative data.

Inference module acknowledges a document as high authority that are more linked from other documents and we apply this point to the PageRank. We also studied and applied some hash tree based optimization and heuristic pruning for the inference rule base.

II. PRELIMINARY AND BACKGROUNDS

1. Core-Symbol Database

The most easiest information retrieval method for user is to search by the commonly used familiar words. But to search for some information in the specific branch fields, we must know core keys by which user can find that information with high priority. If common users know the specific branch fields that they want to search even if their input information consists of common words. They can find core keywords by the specific branch database.

In the case of agent based information retrieval, agent separates user's natural language type input with incomplete and noise to morpheme. And it references Core-Symbol Database to find keyword that we want to find.

Core-Symbol Database consists of a group of keywords that most frequently used and some synonyms as auxiliary words. It selects the most effective keywords to search.

If a user inputs a natural language type query, agent selects keywords and auxiliary keywords.

If some words are not efficient for retrieval, agent replaces that word with more frequently used or higher priority words.

2. Best Aggregate Designer

An agent inference is proposed[6]. This system set up PageRank by BackLink.

The page rank PR(A) for the page A is defined as follows.

$$PR(A) = (1-d) + d\left(\frac{PR(T1)}{C(T1)} + \frac{PR(T2)}{C(T2)} + \ldots + \frac{PR(Tn)}{C(Tn)}\right)$$

Where C(A) is the number of links from page A. d is an arbitrary constant, the value is given 0.85. Table 1 illustrates the algorithm for the strong authority.

Table 1. Algorithm for the strong authority

```
Subgraph(σ, ε, t, d)
 σ  : a query string
 ε  : a text-based search engine
t,d : natural numbers
Let Rσ denote the top t results of ε and σ

Set Sσ := Rσ
For each page p ∈ Rσ
 Let T+(p) denote the set of all pages p points to
 Let T-(p) denote the set of all pages pointing to p
 Add all pages T+(p) to Sσ
 If |T-(p)| ≤ d then
      Add all pages in T-(p) to Sσ
 Else
  Add an arbitrary set of d pages from T-(p) to Sσ
 End
Return Sσ
```

We define 'Hubs' as an important page that has many links to the authority for a specific subject. If a page is linked from many Hubs, it becomes good authority and if a page is liked to many authority, it becomes good Hubs. Table 2 illustrates an algorithm to find optimal Hubs and authority.

Table 2 Algorithm to find optimal Hubs and authority.

```
Let set of x(p), {x(p)} be a vector x, and let set of y(p), {y(p)}
be a vector y

Iterate(G,k)
 G : a collection of n linked pages
 k : a natural number
 Let z denote the vector (1,1,1,...,1) ∈ Rn.
 Set x0 := z.
 Set y0 := z.
 For i = 1,2,...,k
 Apply I operation to (xi-1, yi-1), obtaining new x-weights
  x'i.
 Apply O operation to (x'i, yi-1), obtaining new y-weights y'i.
 Normalize x'i, obtaining xi.
 Normalize y'i, obtaining yi.
End
Return (xk, yk).
```

III. DESIGN OF AVATAR AGENT SYSTEM

Agent system consists of Matching Module, Inference Module, and Agent Module. First, the user inputs natural language message via character type avatar interface that the user is familiar with. Input message goes to the preprocessor of Matching module. The message in the Matching Module is classified to the type through the automata and the attention of the user is decided through several extracting procedures. This input string is extracted in various ways depending on the type of Core-Symbol Database. If we want to apply this mechanism to other applications, the system can work very extensively. An intelligent search modeling using avatar agent is illustrated in fig. 1.

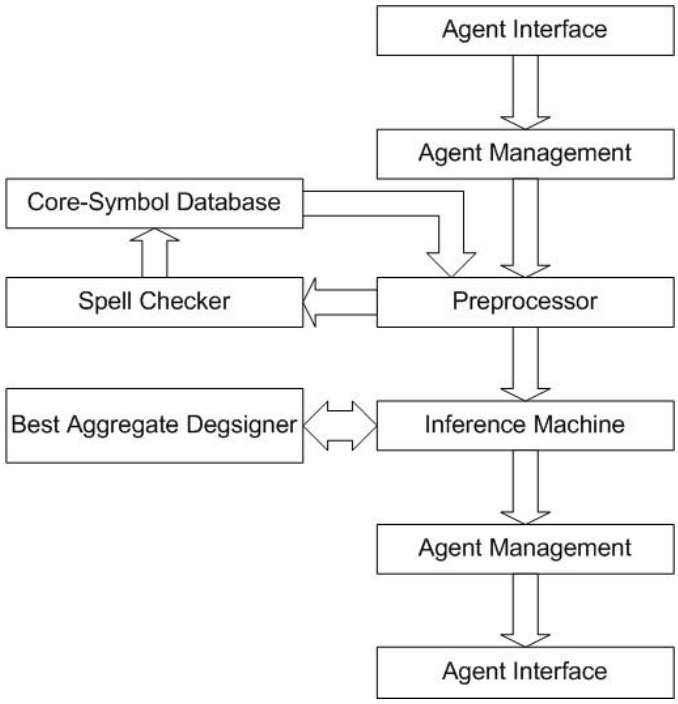

Fig. 1 An intelligent search modeling using avatar agent

The parsed input message is transferred to the Inference Machine in the Inference Module. Inference Machine calls the Best Aggregate Designer and find the most optimal result value by using PageRank, BackLinks, HITS technolgy. Best Aggregate Designer provides the best optimal information and proven data that is considering several retrieval failures. Therefore the user can get the wanted and natural results.

1. Matching Module

Matching Module is a module that converts user's character string input to several words that the engine can recognize. The Matching Module consists of 3 major parts, Preprocessor, Spell Checker, and Core-Symbol Database as illustrated in Fig. 2. The user's input from the agent management parts passes the Preprocessor and classified to several words. These words are verified in the Spell Checker Module and the results are converted to useful words in the Core-Symbol Database. This method can prevent lots of garbage pages that we usually can get the results in the web sites by simple matching retrieval

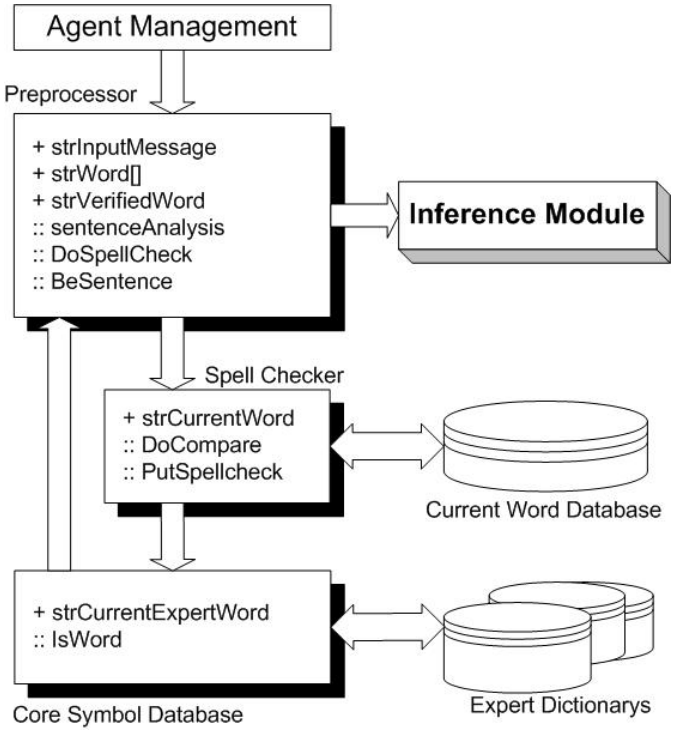

Fig. 2 Block diagram of Matching Module

2. Preprocessor

Preprocessor is activated when it receives user's input strings and those are analyzed by automata's syntax analysis. The sample sentence analysis is illustrated in Table 3.

Table 3. Sample sentence analysis

[Input sentence] Add Hangul to the Word Database
[Sentence Analysis] Add 'Hangul' to the Word Database
[Final results] Word Database = Word Database + 'Hangul'

We utilized HAM 5.0.0 Module (Hangul Analysis Module) for this work. Input sentence is stored in the membership variable strInputMessage in Preprocessor class and separated by words, and stored in the strWord array in the sequence. By utilizing membership function SentenceAnalysis, HAM5.0.0 is called, and the results are indexed in the strWord sequentially. Then the membership function DoSpellChecker is called, and the spell checking is processed. In the one of the internal membership variable strVerifiedWord, useful words after finishing the Core-Symbol Database work are stored in the array.

3. Spell Checker

Even though we successfully extracted correct words from the given input, we may make a big error when the user inputs an erroneous word. Therefore separated words are checked in the database based on the data in CurrentWordDatabase. It a word is not in the database we delete in the word not to make big mistake. CurrentWordDatabase system make it possible to look, update, add by query.

4. Core-Symbol Database

We can find the correct data by selecting the optimal word in a user wanted specific field by utilizing proper database. But if we retrieve words by simple comparison or simple pattern based on pattern matching, we can not find he optimal results. Therefore we try to find the synonym in the expert's point of view.

Core-Symbol Database has several special Core-Symbol Database depending on the Agent Avatar System. Core-Symbol Database is a text file format with some rules and it make it possible to look, update. It provides indexing format database to achieve multi-function Agent Avatar System and compatible among Agent Avatars.

Useful words sent to the membership variable PutSpellCheck in the Spell checker are stored in the strCurrentWord array and each indexed words are substituted to the optimal words for retrieval by membership function IsWordExpert, and those are stored in the Preprocessore's strVerifiedWord array. Table 4 illustrates a storing format of Core-Symbol Database file.

Table 4. A storing format of Core-Symbol Database file.

// [] An example of Stock market Core-Symbol Database file
// [Useful word] Word list that should be changed to useful word
[Debt] debt obligation, debt money, borrowed money
[Stock] stock, stock right
[Rise] Rise, up
[Down] down, decrease, fallen
[Invest Trustee] Trustee Invest

IV. EXPERIMENTS AND DISCUSSION

The retrieval results of 'SAMSUNG-SDI', 'IBM', and 'SONY' among upper 100 pages are illustrated in Table 5.

Table 5. Retrieval results of 'SAMSUNG-SDI', 'IBM', and 'SONY' among upper 100 pages.

Retrieval intention mismatch

Search Engine	SAMSUNG-SDI	IBM	SONY
http://kr.altavista.com	7%	3%	11%
http://www.google.co.kr	6%	2%	7%
Avatar Agent	1%	0%	1%

Overlapped retrieval result

Search Engine	SAMSUNG-SDI	IBM	SONY
http://kr.altavista.com	12%	67%	55%
http://www.google.co.kr	7%	58%	43%
Avatar Agent	0%	0%	0%

We define the user's intention when we try to get that company's stock related information by querying word 'SAMSUNG-SDI'. In the of Altavista and Google search engine, the mismatch ratio was 7% and 6% respectively. But in our avatar agent system, the mismatch ratio was only 1%. We also tried 'IBM' and the result was 3%, 2%, 0% respectively.. In case of 'Sony', the result was 11%, 7%, 1% respectively.

In our avatar agent system focuses more on the correctness in indexing function on page authority, reliability, and hit ratio even though we restrict the volume of recalled pages. Therefore in this experiment, we critically reduced the mismatch ratio by considering hundreds of recall that a user can search continuously. By using retrieval intention mismatch and well-related database, we reflected the user's search intention more precisely, and reduced rank weight if the topic is not so relevant.

One of the problems to be solved in our system is that it takes considerable amount of time to get the proper results.

V. CONCLUSION

In this paper, we proposed an intelligent search modeling using avatar agent. Our system consists of some modules such as agent interface, agent management, preprocessor, and interface machine.

Our avatar agent system does the indexing work that converts user's natural language type sentence to the proper words that is suitable for the specific branch information retrieval. Indexing is one of the preprocessing steps that make it possible to guarantee the specialty of user's input and increases the reliability of the result. It references a database that consists of synonym and specific branch dictionary. The resulting symbol after indexing is used for draft search by the internet search engine. The retrieval page position and link information are stored in the database.

We experimented our system with the stock related information by querying words SAMSUNG_SDI, IBM, and SONY and compared the result with that of Altavista and Goole search engine. It showed quite excellent results. Further researches are on progress to reduce the processing time to get the proper results.

REFERENCES

[1] Swarup Acharya, Phillip B. Gibbons, Viswanath Poosala, Sridhar Ramaswamy - Join Synopses for approximate query answring. Technical report, Bell Laboratories, Murray Hill, New Jersey, Full version of the paper appearing in SIGMOD'99, 1999.

[2] Fay Chang, Minwen Ji, Shun-Tak A. Leung, John MacCormick, Sharon E. Perl, Li Zhang - Myriad: cost-effective disaster tolerance. Conference on File and Storage Technologies, Monterey, CA, pp. 28-30, 2002.

[3] Monika R. Henzinger, Valerie King - Maintaining minimum spanning trees in dynamic graths. In Proc. 24th Internat. Colloq. Automata Lang. Prog, pp. 594-604. Springer-Verlag, 1997.

[4] Tomonari Kamba, Krishna Bharat - "An interactive, Personalized, Newspaper on the WWW," Proc. 4th Intl. World Wide Web Conference, 1995.

[5] Daphne Koller, Mehran Sahami : Hierarchically classifying documents using very few words, Proc. of the 14th International Conference on Machine Learning ICML97, pp. 170-178, 1997.

[6] Larry Page, Sergey Brin, R. Motwani, T. Winograd - "ThePageRank citation ranking: Bringing order to the Web," Standford Digital Library Technologies Project technical report, 1998.

[7] Jay M. Ponte, W. Bruce Croft - A Language modeling approach to information retrieval. Ph.. D thesis, University of Massachusetts at Amherst, 1998

[8] Amit Singhal, Marcin Kaszkiel - A Case Study in Web search using TREC algorithms. In Proceedings of the 10 th International World Wide Web Conference, pp. 708-716, Hong Kong, May 2001.

[9] Sean Quinlan, Sean Dorward - Venti: a new approach to archival storage. in First USENIX conference on File and Storage Technologies. Monterey, CA, U.S.A., 2002.

[10] Georges R. Harik et. al. - Learning Linkage. Foundations of Genetic Algorithms, 4, pp. 247-262., 1996.