Development of an Agent System to Collect Schedule Information on the Web for Intermodal Transportation Network Planning

HYUNG RIM CHOI[1], HYUN SOO KIM[1], NAM KYU PARK[2],
BYUNG JOO PARK[1], MOO HONG KANG[1], JAE UN JEUNG[1]

[1]Departement of Management Information Systems, Dong-a University
840 Hadan-dong, Saha-gu, Busan 604-714, SOUTH KOREA

[2]Department of Distribution Management, Tongmyong University
535 Yongdang-dong, Nam-gu, Busan 608-711, SOUTH KOREA

Abstract: Logistics cost-cutting using a new transportation network design has recently emerged as a big issue for logistics-related firms. However, many difficulties exist in designing an intermodal transportation network, because the operation schedule information necessary for the transportation network design has been scattered. To solve this problem, a technology that can collect and provide the schedule data of all the transportation modes is needed. In this research, an agent system that can identify and collect schedule information provided through the Internet was developed. The schedule-collecting agent system developed in this research is divided into the following: Schedule Crawler, which extracts URLs (Uniform Resource Locators) from HTML (Hyper-Text Markup Language) documents, finds HTML pages, and identifies a schedule-providing page. Web Robot, which senses data change from the identified schedule providing page, extracts schedule data, and saves the extracted data in the schedule database. In addition, algorithm and heuristics that can perceive only schedule information from the concerned Web page were developed. To compare the performance of the system, an experiment was carried out using four shipping companies, and 135 shipping schedule information-providing HTML pages. As a result, a 99.8% schedule information page identification rate was demonstrated, and a 92.3% schedule data extraction success rate was exhibited in the concerned pages. The extracted schedules can be used as a schedule information system for an intermodal transportation network design by establishing a database of those schedules.

Key-Words: - Autonomous Agents, Intelligent Systems

1 Introduction

The third-party logistics industry mainly delivers goods from a starting place to an arrival place on behalf of the freight owner. To handle the work, an optimal transportation network is designed, to select transportation equipment between a starting place and an arrival place, schedule for departure/arrival, and compare freight. Then, a selection of the transportation service provider (shipping company, airline, etc.) and a reservation are made. The system to support the optimal transportation network design, however, was mostly simple search systems, such as Schednet(http://schednet.com), Shipping Gazette(http://www.ksg.co.kr) and Schedule Bank(http://www.schedulebank.co.kr), which can simply search the schedule information of ships, airplanes, and railways. Actually, a system to search an optimal transportation network, which considered intermodal transportation depending on the arrival points, did not exist.

The reason why a selection of the optimal transportation network is difficult can be divided into the following two points. First, difficulty in developing an algorithm that can choose an optimal logistics network among many possible logistics networks can arise. Second, difficulty in securing schedule data to select a suitable network can occur. In the algorithm field of an optimal transportation network plan, many studies applying various methodologies, such as Genetic Algorithm, Dynamic Programming, and Shortest Path, are currently being conducted [3]. However, basic information for the network design, that is, schedule information, harbor information, and airport information were not unified

and secured; therefore, an optimal transportation network design in view of intermodal transportation was difficult. At first we started to research the measures used to collect schedule information that is most necessary for the network design among such basic information. We can find out that shipping companies, airlines, and railway authorities provide schedule information in a variety of modes, and that schedule information is generally provided through the companies' own Web sites. This research, thus, helped to develop an agent system that can actively identify schedule information provided through the Internet and collect the information in real time.

Table 1. Schedule Providing Modes of Each Information Source

Company Type	Company Name	Providing Modes			
		Web	Excel	EDI	DB
Shipping Company	Hanjin Shipping	O	O		O
	HMM	O	O		
	Maersk (P&O Nedlloyd)	O			
	COSCON	O			
	ANL	O			
Airlines	Korean Air	O	O		
	Asiana Airlines	O	O		
	Other Airlines	O			
Railroad	Korea Railroad	O	O		O
	AMTRAK	O			
Other Informati-on Source	Shipping Gazette	O			
	Schedule Bank	O	O		
	Traxon	O		O	

2 Relevant Research

Many studies have been carried out about how to collect and manage data on the Web, which has been increasing substantially since the mid and late 1990s, when the Web appeared and enjoyed its boom. The search engine is typical among the research wave. A search engine is a system that helps users to find their desired Web sites by category or search words through the collection and categorization of the information on the Web [9]. The search engines search and save the Web documents by category, which are scattered worldwide, via programs called search agents, spiders, Web wanderers, or Web worms [9]. Many search agents, such as WebCrawler (Pinkerton, 1994), World Wide Web Worm (McBryan, 1994), Google Crawler (Brin and Page, 1998), CobWeb (da Silva et al., 1999), WebRACE (Zeinalipour-Yazti and Dikaiakos, 2002), and Ubicrawler (Boldi et al., 2004), appeared starting with RBSE (Eichmann, 1994) [1, 2, 4, 5, 10, 11, 14, 15]. Google Crawler, in particular, a search agent of Google.com, was developed using C++ and Python; more precise URL extraction was made through an advanced full-text indexing technique.

The main purpose of these search agents is to search Web sites by extracting URLs. To analyze the content of a Web page and set its category, techniques such as data mining or text mining has been applied. Kim and Lee developed an algorithm that can identify suitable tables with data by using such a mining technique [8]. The algorithm has several pre-processing rules, and identifies tables meeting the rules. Data tables can be found through analyses of the consistency of the cells within the tables.

In this research, an agent system identifying tables that provide schedule information and automatically collecting schedule information was developed, while extracting URLs and searching the Web pages in a way similar to the existing search agents.

3 Schedule Information Collecting Agent System Design

As mentioned in Section 1, many information sources provide their own schedule information through the Internet Web sites. In this section, the agent system to collect schedule data provided through the Web as mentioned above is described.

The schedule collecting agent system developed in this research is categorized as a Schedule Crawler, which finds the concerned HTML pages and identifies schedule-providing pages by extracting URLs in HTML documents, and a Web Robot, which senses data change from the identified schedule-providing pages and collects schedule information.

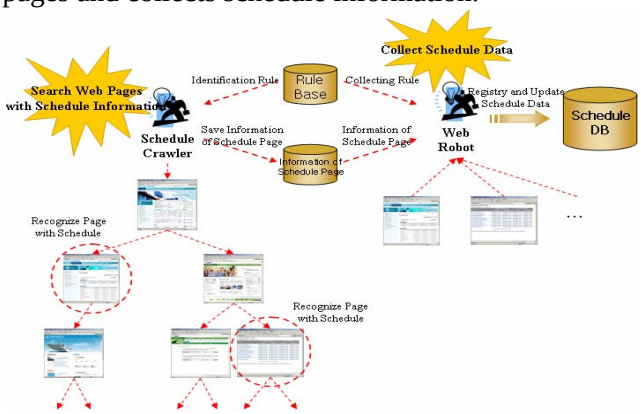

Fig. 1. Structure of the Schedule Collecting Agent System

3.1 Schedule Crawler

The process of the Schedule Crawler, which extracts URLs from HTML pages and finds schedule-providing pages by searching different HTML pages, is demonstrated in Fig. 2.

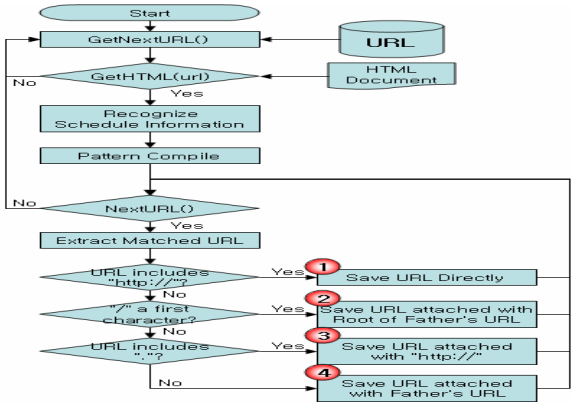

Fig. 2. The Process of the Schedule Crawler

As shown in Fig. 2, the Schedule Crawler obtains a URL from the database, brings the HTML documents from the URL, and analyzes whether schedule information exists, and then extracts the appropriate URL within the HTML. The process expressed in Pseudocode is as follows:

```
READ url from url_db
READ html of url
SET identifying_result as Boolean
FOR each identifying_rule in rule_db
IF matched area with identifying_rule in html
  SET identifying_result to true
ENDIF
ENDFOR
IF identifying_result is true
SET html_information in schedule_db
ENDIF
SET pattern to
'\\shref\\s=\\s*\"?[^\\s]+\"?[\\s>]'
SET pattern_compile to pattern in html
FOR each matched_url in html
IF matched_url includes 'http://'
  SET save_url to matched_url
ELSEIF '//' is a first character
  SET save_url to root_url and matched_url
ELSEIF matched_url includes '.'
  SET save_url to 'http://' and matched_url
ELSE
  SET save_url to url and matched_url
ENDIF
SET url_db to save_url
ENDFOR
```

3.1.1 Schedule Information Identification

Schedule information is mostly provided in the form of a table. Therefore, data tables are identified through pre-processing rules, and they are judged if the tables are data tables through a pattern analysis of whether the tables have schedule information. Kim and Lee [8] identified data tables using pre-processing rules. In this research, however, three pre-processing rules were used to identify whether the identified tables are not data tables as follows:

```
Rule 1: A table is included within a table.
    IF: A table exists within a table.
    THEN: That is not a data table.
Rule 2: There is a table with cell size 1 × 1.
    IF: The table's cell size is not 1 × 1.
    THEN: That is not a data table.
Rule 3: There are no data in a table.
    IF: There are no data within the table cell.
    THEN: That is not a data table.
```

The tables identified as data tables through the abovementioned pre-processing rules are analyzed again to find out whether or not the tables have schedule information. The schedule information is generally demonstrated in certain repeated forms of region names or codes and arrival and departure times, as shown in the following table.

Table 2. An Example of Schedule Table

Port	Arrival Date	Departure Date
LOS ANGELES	Nov 25, 2006	Nov 26, 2006
BUSAN	Dec 10, 2006	Dec 12, 2006
SINGAPORE	Dec 15, 2006	Dec 20, 2006

As demonstrated in the above table, in the first row, second row, and third row, the region name, arrival time, and departure time are indicated, respectively. The Schedule Crawler saves all the cell values in the abovementioned table, and analyzes the value forms by each column or row. As for the region name, the Schedule Crawler identifies the name's form by comparison with the region names and code tables that the Schedule Crawler holds. Concerning the arrival and departure times, the time is identified using regular expressions [6, 12, 13].

Looking at Table 2, the form of time has a pattern of "mmm dd, yyyy." The regular expression can express it in the following way, as shown in (1):

$$L_a = L([a\text{-}zA\text{-}Z]\{3\}\backslash\backslash s?[0\text{-}9]\{1,2\},\backslash\backslash s?[0\text{-}9]\{4\}) \qquad (1)$$

The cell values meeting (1) are identified as having a pattern that indicates time. The following Table 3 demonstrates a regular expression method according to the form of time.

Table 3. Regular Expression by Time Pattern

Time Pattern	Regular Expression
mmm dd, yyyy	[a-zA-Z]{3}\\s?[0-9]{1,2},\\s?[0-9]{4}
yy/mm/dd	[0-9]{2}/[0-9]{1,2}/[0-9]{1,2}
yyyy/mm/dd	[0-9]{4}/[0-9]{1,2}/[0-9]{1,2}
yyyy-mm-dd	[0-9]{4}-[0-9]{1,2}-[0-9]{1,2}
yy mmm dd	[0-9]{2}[\\s\\.-]?[a-zA-Z]{3}[\\s\\.-]?[0-9]{2}
mm/dd or mm-dd	[0-9]{2}[\\s-]?[0-9]{2}

The result identified in the above manner is expressed as a four-digit number by each cell, and the number indicates the date, time, region code, and region name in order. For example, when the cell values have region codes, they are expressed as "9919." The date and time have various patterns, and therefore, the pattern numbers of the dates and times matching the cell values are indicated. The result identified in this manner identifies whether the schedule table is the correct one by analyzing the consistency of each cell via (2).

$$\text{Consistency of data type} = \frac{\text{Number of the cells having major data type}}{\text{Total number of cells of a column (or a row)}} \quad \textbf{(2)}$$

Table 4. Consistency analysis by row

9999	9999	9999
9991	0999	0999
9991	0999	0999
9991	0999	0999
3/4=***0.75***	3/4=***0.75***	3/4=***0.75***

Table 5. Consistency analysis by column

9999	9999	9999	0 (Not a data cell)
~~9991~~	~~0999~~	~~0999~~	2/3=***0.67***
~~9991~~	~~0999~~	~~0999~~	2/3=***0.67***
~~9991~~	~~0999~~	~~0999~~	2/3=***0.67***

As shown in Table 4 and Table 5, in this table, we can observe that the average consistency is 0.75 and 0.50, respectively, and schedule information is provided lengthwise. We also see that schedule information is provided in the HTML page.

The following information is saved in the DB so that the Web Robot can collect the HTML pages identified as having schedule information:

1. Table's location: Saving the beginning and ending classifiers that can locate the location of the table possessing schedule information.
2. Schedule information's direction: Information about to which direction schedule is provided (Right hand side or down below).
3. The location and type of data: Region name (or code), arrival time, departure time, and type within the table.

3.1.2 URL Extraction

The Schedule Crawler carries out a function to identify if schedule information exists in the HTML documents of the URL, which is saved in the local DB. URLs, the subject of the search, should be continuously collected, and the Schedule Crawler also conducts URL extraction in each HTML page. The regular expression that was used for time pattern perception was used for URL extraction, and the regular expression to perceive URL character string is shown in (3).

$$L_a = L(\backslash\backslash\text{shref}\backslash\backslash\text{s}=\backslash\backslash\text{s}*\backslash"?[\hat{}\backslash\backslash\text{s}]+\backslash"?[\backslash\backslash\text{s}>]) \quad \textbf{(3)}$$

All URLs within the HTML pages are collected by the aforementioned regular expression. Due to a problem in not recognizing line changing (\n) in view of the regular expression's characteristic, the line changed parts must be all deleted in the HTML documents.

An example of a saved URL address is shown in Table 6, when the URL and the current HTML page address, http://www.def.com/abc/def/fgi.jsp, was collected through the regular expression.

Table 6. An Example of URL Address Collected and Saving

Collected URL	Saving URL
http://www.abc.com	***http://www.abc.com***
bcd.org	http://***bcd.org***
/lec/main.jsp	http://www.def.com***/lec/main.jsp***
intro/intro1.jsp	http://www.def.com/abc/def/***intro/intro1.jsp***
../../list.jsp	http://www.def.com/***list.jsp***
mailto:mongy@dau.ac.kr	N/A

The URL address collected from the example in Table 6 is changed to complete the URL according to the rules of ①, ②, ③, and ④ in Fig. 2.

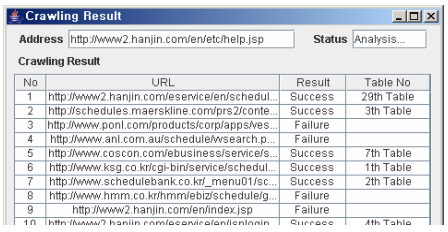

Fig. 3. Schedule Crawler Screenshot

3.2 Web Robot

The Web Robot extracts schedule information from the schedule information-providing pages collected from the Schedule Crawler, and performs a function to immediately apply a change in or deletion of schedule, when the schedule is changed or deleted.

The Web Robot extracts schedule information, based on the table location, information direction, data

positioning information, and the URL in the HTML page collected by the Schedule Crawler.

For example, when the Schedule Crawler analyzed the following HTML document and identified the information as shown in Table 7, the Web Robot collected the schedule information in the order shown in Figure 4.

```
<h1>Schedule for Vessel I</h1><br>
<table>
<tr><td>Port</td><td>Arrival</td><td>Depar
ture</td></tr>
<tr><td>LA</td><td>Nov 25, 2006</td><td>Nov
26, 2006</td></tr>
<tr><td>BUSAN</td><td>Dec 10,
2006</td><td>Dec 12, 2006</td></tr>
<tr><td>SINGAPORE</td><td>Dec 15,
2006</td><td>Dec 20, 2006</td></tr>
</table>
<br>View other schedules
```

Table 7. Information for Collecting Schedule Data in HTML Document

Information		Value		
Location of Data Table	Start Point	Schedule for Vessel I</h1> \n<table>		
	End Point	</table>\n View other schedules		
Location of Data	Region	0 (1st Column)	Type	91 (Name)
	Arrival Date	1 (2nd Column)	Type	09 (Only Date)
	Departure Date	2 (3rd Column)	Type	09 (Only Date)
Direction of Schedule Data		1 (Vertical)		

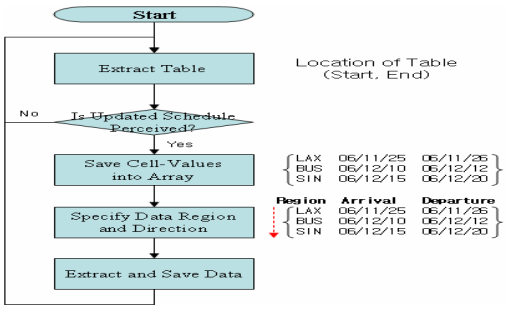

Fig. 4. The Schedule Collecting Process of the Web Robot

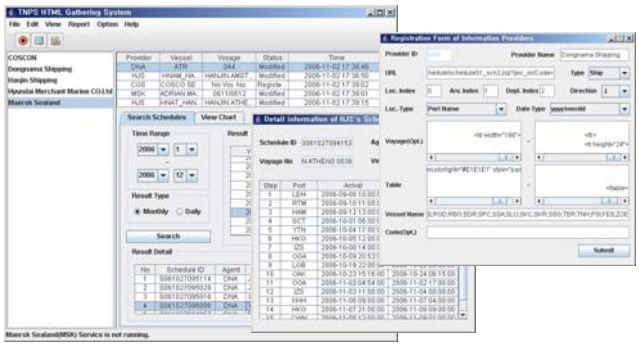

Fig. 5. Web Robot Screenshot

Web Robot extracts a table through parsers that indicate start and end of the schedule table. When ship code and the voyage number match, it compares collected schedule information with existing schedule information. If the compared information is judged as "newly added" or "changed" it extracts data of each cell and stores in an array. According to position and direction value of the data collected by Schedule crawler, it extracts region, departure and arrival time information from the array.

4 Experiment

An experiment was conducted in the following environment so as to find out the efficiency of the Schedule Crawler and Web Robot developed through this research:

Table 8. Experiment Environment

O/S	MS Windows 2003 STD Edition
Platform	J2SE 1.5
DBMS	MS-SQL 2000
H/W	CPU : Intel Zeon 3.06GHz RAM : 1GB HDD : SATA 160GB

The experiment targeted the each Web page for 1,353 ships of the four shipping companies providing ship schedule information; the experiment's results are exhibited in Table 9.

Table 9. Schedule Page Identification Experiment Results of the Schedule Crawler

	Pages with Schedule	Pages without Schedule
No. of Sample	861	492
No. of Identified Sample (Identification Success Ratio)	845 (98.1%)	492 (100%)

In the case of the Schedule Crawler, the HTML page identification and URL extraction time took 2 seconds on average, excluding the Internet connection speed. The HTML page identification success rate was 98.8%, and the URL extraction success rate was 99.8% on average.

About 92.3% of the schedule data was extracted from data schedule extraction by the Web Robot in relation to schedule HTML collected by the Schedule Crawler.

The reason why the success rates of HTML identification by the Schedule Crawler and schedule data extraction by the Web Robot are not 100% is that the agent could not judge atypical or irregular HTML tags, and the perception of region names failed due to

different region names. Currently, a study to solve this problem is underway.

5 Conclusion

In this research, an agent system was developed that searches HTML documents existing on the Internet, identifies them, and collects schedule information. The system's perception success rate was measured through a relevant experiment.

Through the agent system, transportation schedule information was collected. And, through the information, the establishment of a schedule database became possible for the design of an optimal transportation network in consideration of intermodal transportation.

This system is expected to be used as a supporting tool in establishing an integrated database, because this system can be applied to other information collection, in addition to schedule information.

However, a decreasing perception rate was demonstrated, due to atypical attributes of the HTML documents, and a result of a non-collecting schedule was shown, because of some cases in which the perception was not made, deriving from diverse indication methods when identifying region names. To solve these problems, the research team is currently studying a method to save atypical HTML attributes as a pattern, and identify data in reference to it. As for the problem in perceiving region names, a study to solve the problem by introducing techniques, such as ontology, is underway.

Acknowledgments:
This work was supported by the Regional Research Centers Program(Research Center for Logistics Information Technology), granted by the Korean Ministry of Education & Human Resources Development.

References:

[1] Boldi, P., Codenotti, B., Santini, M., and Vigna, S. UbiCrawler: a scalable fully distributed Web crawler, *Software, Practice and Experience*, Vol.34, No.8, 2004a, pp. 711–726.

[2] Cho, J., Garcia-Molina, H., and Page, L. Efficient crawling through URL ordering, *In Proceedings of the 7th conference on World Wide Web*, 1998

[3] Cho, J.H., Kim, H.S., Choi, H.R., Park, N.K., Kim, S.Y. Optimal Intermodal Transport Algorithm using Dynamic Programming. Korea Society Industrial Information Systems, *2006 Autumn Academic Conference*, 2006

[4] Da Silva, A. S., Veloso, E. A., Golgher, P. B. Ribeiro-Neto, B. A., Laender, A. H. F. and Ziviani, N.: Cobweb – a crawler for the Brazilian web, *In Proceedings of String Processing and Information Retrieval (SPIRE), Cancun, Mexico, IEEE CS Press*, 1999, pp. 184–191

[5] Eichmann, D. The RBSE spider: balancing effective search against Web load, *In Proceedings of the First World Wide Web Conference, Geneva, Switzerland*, 1994

[6] Jeffrey, F. *Mastering Regular Expressions*, O'Reilly. ISBN 0-596-00289-0, 2002

[7] John, L.C., Gary, R.A., Mark, J.C. Third-Party Logistics Study Results and Findings of the 2003 Eighth Annual Study. *Technical Report, Georgia Institute of Technology*, 2003

[8] Kim, Y.S., Lee, G.H. Efficient Algorithm for Table Identification of HTML Documents, *Journal of Korea Multimedia Society*, Vol.7, No.10, 2004, pp. 1339-1353

[9] Koster, M. Robots in the Web: threat or treat? *ConneXions*, Vol. 9, No. 4, 1995

[10] Pinkerton, B. Finding what people want: Experiences with the WebCrawler, *In Proceedings of the First World Wide Web Conference, Geneva, Switzerland*, 1994

[11] Risvik, K. M. and Michelsen, R. Search Engines and Web Dynamics, *Computer Networks*, Vol.39, 2002, pp. 289–302

[12] Searls, D.B. Representing Genetic Information with Formal Grammars, *In Proceedings of the 7th National Conference on Artificial Intelligence, American Association for Artificial Intelligence*, 1998, pp. 386-391

[13] Searls, D.B. Signal Processing with Logic Grammars, *Intelligent Systems Rev.*, Vol.1, No.4, 1989, pp. 67-88

[14] Shkapenyuk, V. and Suel, T. Design and implementation of a high performance distributed web crawler, *In Proceedings of the 18th International Conference on Data Engineering (ICDE), San Jose, California. IEEE CS Press*, 2002, pp. 357-368

[15] Tadhg, O, Ahmed, P. A Topic-Specific Web Robot Model Based on Restless Bandits, *IEEE INTERNET COMPUTING 1089-7801*, 2001, pp. 27-35