Using Software Agents to Index Data in an E-Travel System

Patrick Harrington
Department of Mathematics and Computer Science, Northeastern State University
Tahlequah, OK 74464, USA
harringp@nsuok.edu

Minor Gordon, Andy Nauli, Marcin Paprzycki, Steve Williams, Jimmy Wright
Computer Science Department, Oklahoma State University
Tulsa, OK 74145, USA
{minorg, nauli, marcin, stw, jimmyww}@cs.okstate.edu

ABSTRACT

In this note we discuss the problem of indexing information from the Internet, with the goal of delivering personalized content to users of an Internet-based travel support system. We introduce the form of index tokens that will be stored in the system and describe an agent-based subsystem designed to support the indexing function.

Keywords: Content Management, Internet, Software Agents, Data Indexing

1. INTRODUCTION

The development of a system for delivering information gathered from the Internet to a user is one of the more mundane applications of the agent paradigm, and yet, when designed with sufficient complexity to allow for information filtering, content personalization and management (e.g. updates performed in timely fashion) etc., such a system may test the power of any development model. One particular problem in the content-driven application that may be especially adapted to an agent-based approach is that of content gathering – the storage and retrieval of information available on the Internet. In this note we analyze this problem in the context of an agent-based travel support system, one of the paradigm-defining scenarios of software agent development.

The Problem of Content Management

In order for an e-travel support system to accurately reflect available travel options and information, a robust strategy for obtaining this content from sources on the Internet is required. Existing content provision systems typically approach this problem in one of two ways:

1) *by aggregation*: retrieving beforehand **all** information that the system will possibly need in the future, and organizing it in databases in a predefined (by humans) format for future retrieval,
2) *by selection*: indexing information to maintain a "map" as to what information (and where) is available on the Internet, and retrieving the actual content only as it becomes necessary to satisfy user's queries.

Most online travel content gateways (e.g. Expedia, Travelersadvantage, etc.) employ the first method, storing the majority of browseable content locally and calling out to the primary source systems on the Internet (e.g. those run by travel providers such airlines) for verification of locally-cached information (e.g. flight schedules, seat availability and ticket prices). The main advantage of this approach is the immediate local availability of content; this is also a disadvantage, in that it leads to problems of data coherency. In addition, the amount of data and continuous local processing necessary for aggregation systems to work makes them extremely resource intensive.

The majority of search engines (e.g. Yahoo, Interia, Lycos, etc.) take a hybrid approach, aggregating only a limited store of data (such as page headers and few selected pages) necessary to support the search function. This approach attempts at striking a balance between the amount of content stored locally, frequency of local information updates and the precision of the search function. Rudimentary content organization and differentiation available in browsers combined with relative freshness of data,

while relatively satisfactory for typical searches (of content that changes infrequently) is not enough to support travel-oriented services (where the freshness of content is paramount importance).

Our e-travel system fully embraces the second approach (content management *by selection*) by attempting at developing a well-organized and highly cross-referenced index of Internet-based content (for a description of a number of similar systems see [1]). The proposed system dynamically utilizes remote content by referencing local indices – pointers. It focuses on the classification of content instead of the content itself, as in a library catalog (or yellow pages), only storing enough information in indices to satisfy user queries. This approach eliminates the problems of data coherency, prominent in aggregation systems, and is expected to assure that the system will not waste resources on extraneous data. The downside of this approach is that content must always be retrieved from a remote site. If a content provider becomes unreachable, the e-travel system is unable to retrieve the information and thus to fulfill the user's request. More generally, any slowdown in reaching the content provider is reflected in the performance of the system. Nevertheless, in designing the e-travel system we felt that the advantages of accurate indexing and the avoidance of data coherency issues outweighed the disadvantages of remotely-stored content. We also expect that approach based on indexing will improve the limited queries options and result displays caused by traditional database logic and principles [1, 15].

2. CONTEXT OF THE TRAVEL SUPPORT SYSTEM

History

The initial design of the travel support system was presented in [2, 5, 12, 13] and while it is being constantly modified (and this note discusses some of these modifications), the general idea of dividing the functionality into two coordinated subsystems, one handling content management and the other content delivery [4], remains unchanged. In this note we concentrate on the content management aspects of the system.

Sources of Content Indices

The travel options and information that is presented by the e-travel system originates from two types of sources on the Internet: verified and unverified. Verified sources are referred to as *Verified Content Providers* (*VCP*), which designation implies a degree of conformance to expected standards of accuracy, format and availability of described travel options. Content from *VCP*s can be either fed directly to the system or gathered by search agents, as described in [2]. In the first case we assume that incoming indices (pointers to available information) are both in the required format and complete, and thus can be immediately stored in the system without further processing. In the second case, the acquired content indices may be incomplete and/or require further processing. When dealing with unverified sources the situation is similar to the latter case with an added component of necessary verification and deconfliction of remote information. (at this stage of system design we will omit these last two issues and assume that they have been successfully resolved). Let us note, that this approach allows us to address one of the important research issues raised by Nwana and Ndumu in [11]; how to deal with dynamically changing content and form of the Internet-based information. Here, we assume that the *VCP*s are in contractual agreement with the travel agency and any changes in their site design will be communicated to our system, allowing it to be adjusted accordingly. Since the *VCP*s are the primary sources of the information, changes occurring in the remaining sites do not threaten the functioning of our system. Regardless of source, the acquired indices are stored in the central *registry* for later access by the content delivery functions of the system. When the user requests information, a relevant content pointer is either found in the *registry* and the process of content extraction from the provider(s) is initiated (while additional search agents may be released to the Internet seeking additional content), or a new index search and acquisition is forced, in order discover relevant content (from both *VCP*s and unverified sources).

3. STRUCTURE OF INDEX TOKENS

The e-travel system relies heavily on the accuracy and completeness of local content indices. They must be succinct enough to be easily acquired and stored, yet verbose enough to satisfy all of the requirements of both content management and content delivery subsystems. Consider the following scenario: a user wishes to make travel arrangements to visit Mt. Rushmore, a historical monument. The user must first travel to South Dakota (requiring a

means of transportation), and perhaps find a place to stay (hotels in the area). She may also wish to know about local restaurants or other places of interest. In order to satisfy the user's request for travel arrangements, the system must initially make two major distinctions based upon the query alone: location (South Dakota) and desired destination/attraction (Mt. Rushmore). In addition, the e-travel system must also be able to resolve multiple providers of content relating to Mt. Rushmore, in order to find those indices, which will eventually yield the most desirable response for the user (for the purpose of this paper we skip the question of content provider ranking, which is one of the possible ways of dealing with potential information overload).

Our current design of indices evolved from our early attempts to develop a classification system of the world of travel content [2], and was adapted to satisfy the above requirements. We now describe an index as a tuple consisting of:

(<*provider*>,<*type*>,<*location*>,<*?notes?*>)

The *provider* field describes the content provider and the access protocol in the URI form

access_protocol://server/

e.g.

ota://www.jims_house_of_beer.com/.

The possible access protocols are: HTTP, SOAP, EDI, OTA, etc. The *type* field is the class of content referred to by the index, e.g. flights, accommodations, entertainment, historical monuments etc. (here, we utilize a modified Yahoo! categorization of the world of travel; for more details see [17]). The *location* field contains the geolocation information in the form of a pair (latitude, longitude). Finally, the *notes* field is natively available in the *ebXML Registry/Repository* (our selected data storage technology) in the form of slots [17] and here the meta-descriptions of tuple completeness and other administrative data will be stored. We refer to the entire tuple as an index *token*. The following is an example of a complete index token that is ready to be stored in the system (the *notes* filed is omitted, but in this case it would contain information that the token is complete and no further processing is required):

(edi://www.drp_sushi_palace.com/, restaurant, (25'45'', 34'67''))

Once a complete index token is successfully inserted into the *Registry*, it is ready for processing by the content delivery subsystem (as described in [2, 5, 12]) and can be utilized to prepare responses to user queries.

4. INDEX ACQUISITION

We now consider the actual process of index acquisition. As indicated above, there are two sources of index tokens: *VCP*s that feed complete indices directly to the system (this relationship is pre-defined by agreements between providers and the e-travel system); and search agents, which explore both the *VCP*s and other repositories on the Internet. Tokens acquired by search agents may or may not be complete, and if their source is unverified, the content referred to may also need to be validated and deconflicted. Within the system all incoming tokens (from *VCP*s and search agents) are received and handled by an indexing agent, which inserts them into the stable index; the *registry*. Incomplete or not yet validated tokens are marked as such in the *notes* field of the index tuple. Furthermore, incoming tokens may have various priority levels, also indicated in the *notes* field: tokens acquired for a user currently interacting with the system will have to be made ready (completed, validated and deconflicted) for use as quickly as possible, while other tokens may be processed when the system is idle. More information about the index *Registry* and its implementation is available in [17]. Figure 1 summarizes the above-described general functionality.

5. CONTENT GATHERING

The problems of content indexing and retrieval are part of one of the crucial issues confronted by Internet-related research: how to introduce "understanding" to machine-web interaction. One of the reasons that most online content gateways choose the aggregation approach to content management is because it is easier to implement, despite its resource-intensiveness. The more "intelligent", selective approach of indexing content for later utility requires an in-depth, machine "understanding" of the content in order to reliably utilize it.

Interpreting Sources

In recent years there has been a resurgence of interest in ontologies as a way of dealing with the

problem of machines "understanding" the semantic of information on the web. Many claim that agents with ontologies will be the next breakthrough technologies for web applications [6]. This has been the thrust of the Semantic Web project [14] – the development of an ontology-described content infrastructure that will allow machines to interpret semantics as opposed to mere syntax. This capability has been realized in web pages hosted by several organizations. According to the DAML Crawler [3], as of the time of our writing, there are semantically 21,025 annotated web pages. Unfortunately, this number is negligible compared to the total of 7 billion web pages on the Internet. Therefore, today, it is not realistic to assume that agents can simply understand the web-content.

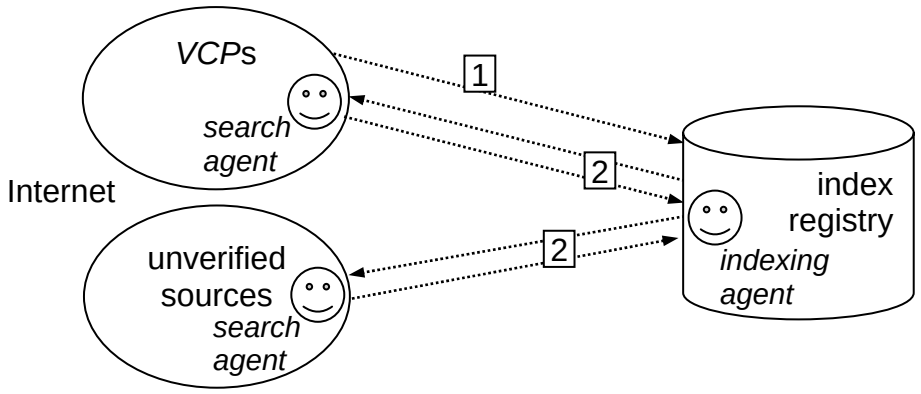

Figure 1. Information gathering and indexing; 1 – flow of index tokens originating from the *VCP*s, ready for insertion to the *registry*, 2 – flow of index tokens resulting from the Internet searches

The design of our e-travel system takes into account the eventual existence of a semantically-described web; and, in particular, development of a complete and generally accepted ontology of travel, but it does not rely on it. Rather, we plan to implement an intermediate solution that allows us to depend on agent "understanding" only within the e-travel system, which working assumption is supported by adapting the perimeter of the system (ie. the index acquisition system) to simulate semantic gathering [6, 14].

One of the typical approaches to developing agents with the necessary functionalities is through topical web crawlers [9]. Topical web crawlers take advantage of knowing the context of the query to differentiate between the relevant and irrelevant web pages. Web pages are considered to be relevant if their similarity value satisfy a given threshold. Similarity value is calculated based on lexical analysis of the web page.

Another approach to semantic understanding of the web is through application of wrappers. For example, information agents in Heracles [15] are trained to locate meaningful information in the web pages by being shown examples consisting of web pages labeled with markers to indicate where the information is located. These examples are then used to develop a set of wrappers that are subsequently utilized in intelligent searches.

Data Completeness

One of the major problems in indexing data surveyed on the Internet is the possibility of incomplete index tokens being returned by search agents. No agent can acquire information that is simply not available. For instance, the majority of content providers (e.g. web sites) do not support complete geospatial information desired by our system (the (latitude, longitude) pair), but rather provide an address (complete or partial).

All incomplete tokens (as well as tokens gathered from unverified sources) are flagged, assigned priority and inserted into the *registry*. They are then processed by token completion, validation, deconfliction (CVD) agents. These agents traverse the *registry* and process the incomplete / unverified tokens. As an example let us consider the case of a token that is missing the location data. It is known who is the *provider* of the data, that it is a *hotel*, while the *location* field contains no data. The CVD agent will therefore create an instance of a query agent. This agent will communicate with the content provider (using the specified protocol) and establish

that the *hotel* in question is the Boardwalk Casino in Las Vegas and recover its street address (from the same location, or from a separate content provider uncovered during a separate web-search). This information will be returned as an ACL message to the CVD agent responsible for managing this particular token. The CVD agent will then query the GIS subsystem (see [2, 12, 13] for more details) where reverse geocoding will result in the (latitude, longitude) pair. This information will be inserted it into the token and the incompleteness flag removed, thus making it a full member of the *registry*. In similar way other incompleteness in index tokens will be dealt with.

6. CONCLUDING REMARKS

In this note we have reported on our progress in developing and agent-based travel support system. Our principal motivation is an attempt at implementation of a realistic agent system that can be used to establish potential and limitations of a more general class of agent-based systems. In this we follow the methodological lead of Nwana and Ndumu [11] who have stressed the importance of the implementation and experimentation phases of agent system development. We are also challenged by the fact that all the past projects have been limited in scope [10, 16] or abandoned in early stages of development. At present, we are in the process of implementing the above-described functionalities. For the agent platform we have selected JADE [8], which is fully FIPA compliant and supports the ACL communication language. The ebXML Registry/Repository has been selected as the technology for implementing the *registry* of index tokens [17]. To insert the index tokens into the *Registry*, we have decided to use the Java API for XML Registry (*JAXR*), which provides a uniform API for interfacing with XML registries. To extract content from web pages, we use the HTML parser package from HTMLParser project [7].

At the time of writing of this note we have implemented the classification scheme and the manual functionalities of the *registry* [17] as well as rudimentary non-automated versions of search agents. Currently, we are designing and implementing web filtering functionalities of our search agents (to be based on techniques described in [2, 9]) as well as integrating them with the *registry* (which involves the implementation of the indexing agent). We rely on third-party GIS subsystem as the primary source for latitude and longitude information (limited to US and Canadian address only). We acknowledge the drawbacks (stability and inconsistency) caused by such dependency, but we consider such drawbacks to be insignificant for the system during the development time. Finally, our categorization of the world of travel will be primarily based on the Yahoo! catalog and the work of the Open Travel Alliance (OTA) – a non-profit organization that tries to introduce next generation of standards into the travel industry. OTA has published several specifications that provide a universal format for collecting and exchanging information (including common communication infrastructure) for large part of the "world of travel" (e.g. airline and hotel reservations, car rentals, golf course reservations, travel insurance etc.). We plan to utilize their specifications in our system (see [17] for more details).

There exist a large number of research and/or practical issues that need to be addressed; e.g. how much intelligence do we want to include in our search agents for them to be effective in filtering web content and supplying our system with complete index tokens while being relatively lightweight, how many agents of various types (indexing, token completion, search, GIS etc.) are required to prevent processing bottlenecks, is the proposed indexing schema robust enough to support the content delivery functions, how does the proposed indexing schema match with the personalization oriented functions that the system is to support (in particular user behavior data storing and mining [5]) etc. We believe that they, as well as many others, will have to be answered through experiments performed with the system we are developing. We will report on our progress in subsequent publications.

7. REFERENCES

[1] Abramowicz, W., Kalczynski, P., Wecel, K. (2002) "Filtering the Web to Feed Data Warehouses." Springer Verlag Publishing, New York.

[2] Angryk, R., Galant, G, Gordon, M., Paprzycki M. (2002) "Travel Support System – an Agent-Based Framework," Proceedings of the International Conference on Internet

Computing (IC'02), CSREA Press, Las Vegas, pp. 719-725

[3] DAML Crawler (n.d.). Retrieved Feb 11, 2003 from http://www.daml.org/crawler/

[4] V. Galant, J. Jakubczyc and M. Paprzycki. "Infrastructure for E-Commerce." In Nycz M., Owoc M. L. (eds.), Proceedings of the 10th Conference on Knowledge Extraction from Databases, Wrocław University of Economics Press, 2002, 32-47.

[5] Galant V. and Paprzycki M. (2002) "Information Personalization in an Internet Based Travel Support System." Proceedings of the BIS'2002 Conference, Poznań, Poland, April, 2002, pp. 191-202

[6] Hendler, J. (2001) "Agents and semantic web," IEEE Intelligent Systems Journal, 16(2), pp. 30-37

[7] HTMLParser (n.d.). Retrieved Feb 11, 2003 from http://htmlparser.sourceforge.net

[8] JADE (n.d.). Retrieved Feb 11, 2003 from http://jade.cselt.it

[9] Menczer, F., Pant, G., and Srinivasan, P. "Topical Web Crawlers: Evaluating Adaptive Algorithms," ACM Transaction on Internet Technology, 5(N), pp. 1-38

[10] Ndumu, D., Collins, J., Nwana, H. (1998) "Towards Desktop Personal Travel Agents," BT Technological Journal, 16 (3), pp. 69-78

[11] H. Nwana, D. Ndumu, A Perspective on Software Agents Research, The Knowledge Engineering Review, 14 (2), 1999, 1-18

[12] Paprzycki M., Angryk R., Kołodziej K., Fiedorowicz I., Cobb M., Ali D. and Rahimi S. (2001) "Development of a Travel Support System Based on Intelligent Agent Technology," in: S. Niwiński (ed.), Proceedings of the PIONIER 2001 Conference, Technical University of Poznań Press, Poznań, Poland, pp. 243-255

[13] Paprzycki M., Kalczyński P. J., Fiedorowicz I., Abramowicz W. and Cobb M. (2001) "Personalized Traveler Information System," in: Kubiak B. F. and Korowicki A. (eds.), Proceedings of the 5th International Conference Human-Computer Interaction, Akwila Press, Gdańsk, Poland, pp. 445-456

[14] Semantic Web (n.d.). Retrieved Feb 11, 2003 from http://www.semanticweb.org

[15] Staab, S and Werthner H., "Intelligent Systems for Tourism", in IEEE Intelligent Systems, November/December 2002, pp. 53-55

[16] Suarez J. N., O'Sullivan D., Brouchoud H., Cros P. (1999) "Personal Travel Market: Real-Life Application of the FIPA Standards." Technical Report, BT, Project AC317

[17] Wright, J., Williams, S., Paprzycki, M., Harrington, P., Using ebXML Registry/Repository to Manage Information in an Internet Travel Support System, Proceedings of the 6th BIS Conference, Colorado Springs, June 2003, to appear