Using Explicit, A Priori Contextual Knowledge in an Intelligent Web Search Agent

Roy Turner[1], Elise Turner[1], Thomas A. Wagner[1], Thomas J. Wheeler[1], and Nancy E. Ogle[2]

[1] Department of Computer Science, University of Maine, Orono, Maine 04469 USA
{rmt,eht,wagner,wheeler}@umcs.maine.edu
[2] School of Performing Arts, University of Maine, Orono, Maine 04469 USA
Nancy_Ogle@umit.maine.edu

Abstract. The development of intelligent Web search agents will become increasingly important as the amount of information on the Web continues to increase. Since intelligently searching the Web depends on the searcher understanding the context not only of the query and user, but also of the information sources and the information retrieved, intelligent Web search agents will need to have mechanisms for representing and using contextual knowledge. In this paper, we discuss the kinds of contexts and contextual knowledge such an agent will encounter. We use as an example a Web search agent we are beginning to develop, FERRET, that will search for scholarly information about music. We then propose some ways in which explicitly represented, *a priori* contextual knowledge can be used by the search agent, and we discuss directions for future research.

1 Introduction

The World Wide Web is increasingly being relied upon as a source of scholarly information for serious users. However, searching the Web for useful information can be frustrating and time-consuming. The user must devise a search strategy that takes into account the locations of search engines and their characteristics, including the reliability of their results, what form the queries must take, and which keywords are most likely to garner the desired results. The user must then evaluate the potentially large set of results with respect to relevance and usefulness. Often the process is one of trial and error, with the user submitting a search, then refining the keywords based on the results obtained and submitting a new search, and so forth. Searching the Web thus effectively requires not only knowledge of the domain of the search, but also a great deal of knowledge about properties of the search engines and content sites that are available. To search efficiently, the user is forced to become an expert on the process of searching. As noted in a recent review of intelligent information agents: "Far from being the answer to everyone's information dreams, distributed sources of online information [...] often turn into an information nightmare" [1].

The object of the FERRET (Facilitated Elaboration and Retrieval for Researchers' and Experts' Tasks) project is to create an intelligent search agent that will ease the task of finding scholarly information on the Web. FERRET's domain will be 20th-century music, and the target users will be musicians and musicologists. The agent itself will be an expert at searching the Web, freeing its users from the need to become such experts themselves.

Figure 1 shows the structure of FERRET. The user will enter a query in English. For example, a singer who is interested in finding material for a recital might submit the query: "musical settings of Emily Dickinson's poems". The agent will then use its knowledge of the user, the domain, and how to search the Web to carry out the search. The *query elaborator* module will examine the query and create a more useful query, based on its knowledge. A *task structure generator* will then take the elaborated query and create a TAEMS [2] task structure, a plan-like representation of different ways the appropriate results might be obtained. This will be passed to the design-to-criteria *scheduler* [3], which will examine the task structure and select the best way of carrying out the query, given the constraints given by the user and those imposed by the agent itself, based on its knowledge. An *execution subsystem* will then actually carry out the search by interacting with search engines and known knowledge sources (e.g., Web sites). A *report generator* will filter out extraneous results and summarize the remainder for the user. A *search manager* will coordinate and control the entire process, interacting with the user as necessary via the user interface to restrict or expand the search and to assess the quality of the results.

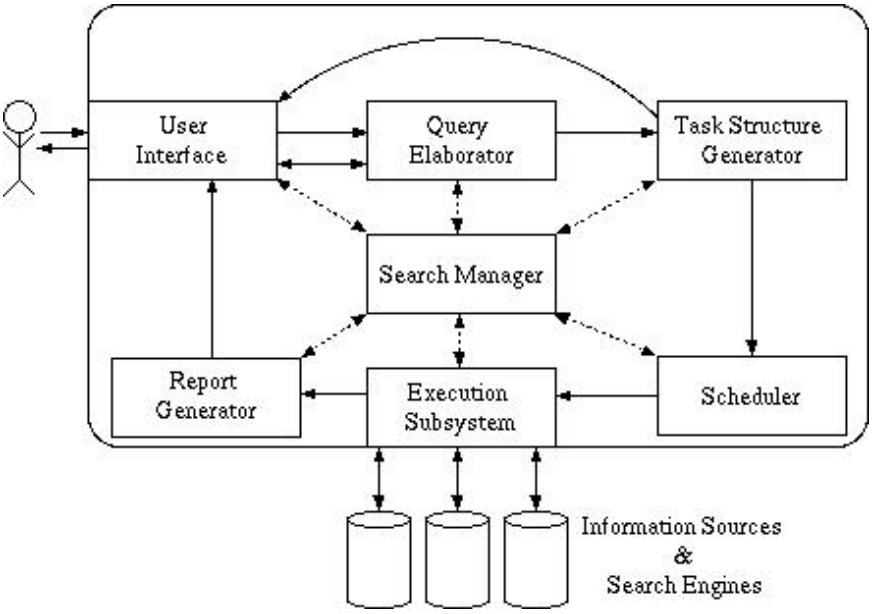

Fig. 1. Planned structure of FERRET.

An intelligent agent such as FERRET needs contextual knowledge to behave appropriately for situations in which it finds itself. In general, an agent can use its knowledge of kinds of situations to handle ambiguity of input, to determine situation-specific meaning, to handle unanticipated events, and to determine how best to achieve its goals [4]. By recognizing the context it is in, the agent can use *a priori* knowledge about the context to guide its perception, reasoning, and action. In the particular case of a Web search agent, paying attention to the context can help the agent determine what its user really means by his or her query, how to elaborate the query effectively, how best to plan for and schedule search actions, how to behave to effectively interact with various Web search engines and content sites, how to construct a coherent set of results, and how to control the overall search process.

In the remainder of this paper, we will examine the role of context in intelligent Web searching, then discuss how we intend to make use of explicitly-represented contextual knowledge in the FERRET search agent. We briefly examine some related work, then conclude with a discussion of future work.

2 Context in Intelligent Web Searching

Before examining the role of context in Web searching, we must first define what a context is. By "context", we mean the *kind* of situation in which an agent finds itself. That is, it is the class of known situations the current situation has been recognized as being a member of. It is the agent's knowledge about the class of situations—the context—it is in that aids it in behaving appropriately. Thus context recognition is a key process for a context-sensitive agent.

In general, recognizing the context enables an agent to quickly bring to bear a wealth of knowledge about how to behave in that context. Recognizing the kind of user, for example, can tell the agent much about what the user's query really means, what he or she is likely to want for results, and constraints on the search.

Context has many components or dimensions, what we have called elsewhere *contextual aspects* [5]. For a Web search agent, an important component of the context is the user. Different kinds of users behave differently when using the system, and they require different results. For example, a singer and a composer of electronic music may each ask the system to find "music based on Emily Dickinson's poems", but the agent should use its knowledge of the user to determine what kind of music to search for. It is also likely that different kinds of users will use different vocabularies, or that they will phrase search queries differently. For example, one would expect an expert in the area of late 20th-century music to use a more technical vocabulary than a beginning music student. By recognizing the category of user, the agent may be able to make the distinction between a novice user using non-technical terms because he or she has no other option and an expert user using such terms to indicate the level of formality he or she wants in the results.

In addition to obvious categories of users such as novice or expert, singer or composer, there may be more subtle distinctions that the agent should make. For example, some users will be more impatient than others. If the agent can identify the user as being usually impatient, then it can generate appropriate time constraints for the search. Particular users will have idiosyncrasies. For example, a scholar may consider some types of music "serious"; a query such as "find serious music related to the work of Edna St. Vincent Milay" should thus, for this user, restrict the results to the kind of music he or she considers serious, even if the information found is not in any way described as "serious" by its author.

The user may provide information to the agent that the agent can use to recognize other aspects of the context. For example, if the user tells the agent that he or she is in a hurry, then the agent should recognize the context as "time constraints present". Knowing this, the agent might decide to give preference to Web sites and search engines that are fast. The user could similarly tell the agent that he or she does not wish to pay more than x for the search, or provide other task-related information.

The search query itself can also provide an indication of the context which can provide predictive information to guide the agent. For example, some kinds of queries may lead to more hits than others; if the agent expects, based on this aspect of the context (i.e., the kind of query), there to be more hits than there are, then it should suspect that something has gone wrong in the search. The semantics of portions of the query might also establish a context. For example, in the query, "musical settings of Emily Dickinson poems", "musical settings" establishes a context in which the agent should know that it should look for the poems themselves (among other things), as opposed to criticism of the poems, discussions of musical imagery used in the poems, etc.

Features of the results found may also establish a context that can guide the agent. For example, the situation of finding a very large number of hits relative to what is expected might be recognized itself as a kind of context, namely one in which the agent should do further query elaboration. Recognizing this would allow the agent quickly to determine how to handle the problem. As another example, it may be possible to recognize a context analogous to the situation of a piece swap in chess, in which a game-playing program's static evaluation function applied halfway through the swap would give a drastically different value than when the swap is completed. An example of this in the Web search domain might be the case where the hit rate for the search was initially low, but ramped up exponentially as the time allotted for the search expired. This kind of situation may occur frequently enough that it makes sense recognize it as a distinct context. Knowledge about this context would allow the agent to predict that if it devoted just a little more time to the search, it would drastically improve its chances of obtaining a good answer for the user.

There may also be features of the search engines and content sites that define a context with predictions useful to the agent. For example, searching some sites may require particular vocabularies or ontologies. Recognizing this would

facilitate query elaboration and search, as well as the summarization process. In addition, some sites may be more or less scholarly than the language in which the query was expressed by the user. In this case, the agent may need to do some translation of terms between the user's query and the site. It would know to do this based on recognizing the aspects of the context corresponding to the user and the site or search engine being used.

Contextual knowledge can also help a Web search agent handle unanticipated events. This includes such things as the user interrupting the agent to give it additional information or to change the search constraints, unexpected responses from search engines or content sites, or failure of some portion of the agent's search plan. Contextual knowledge would help here by providing information about how important the event is likely to be in the current context, what may have caused it, and how to handle it. For example, if the agent has decided to use a particular search engine to find examples of Dickinson's poems, but there is a time-out while contacting the agent, it may decide to respond differently based on whether its context includes a time constraint. In that case, it may opt to try a different search engine rather than to retry this one.

Summarization of results can also benefit from contextual knowledge. The agent's knowledge of its context can help it determine what to present to the user, based on what it predicts the user already knows. It can help it select the level of detail to give to the user and the kind of language to use (e.g., expert vocabulary versus novice vocabulary). It can help the agent decide what to rate highly or filter out based on knowledge about the kind of Web site or author that produced the result.

3 Using Contextual Knowledge in the Web Search Agent

Our approach to using contextual knowledge is *context-mediated behavior* (CMB) [4, 6]. In this approach, contexts with some predictive worth for the agent's behavior are identified, either by the agent's designers or by the agent itself (e.g., via learning), and these contexts are explicitly represented as knowledge structures called *contextual schemas* (c-schemas). Each c-schema represents a context or significant aspect of a context that, if recognized, provides important information about how to behave in the context. C-schemas can be thought of as generalizations of cases of problem solving, and consequently, they are similar to the internal knowledge structures generated by some case-based reasoners (e.g., [7, 8]).

Explicitly representing contextual knowledge is important. It allows the agent to compare the representation to the current situation to determine the degree of fit. It allows the agent to recognize a context, then have immediate access to all the knowledge it has about that context. It also facilitates knowledge acquisition, learning, and maintenance by making all knowledge about a context readily available to the agent and the maintainers.

A first step toward representing contextual knowledge is to determine what kinds of contexts, or contextual aspects, that there are. Basically, contextual

aspects are the components or dimensions of the overall context that can be individually identified, then whose knowledge can be blended to generate a coherent picture of the overall context. In this domain, these aspects would include:

- user context: what kind of user is using the system?
- query context: what is subject of the query? what is the form of the query (e.g., scholarly, colloquial, etc.)?
- task context: what constraints are there on the task (e.g., time, cost)?
- search plan context: are there properties of the search plan selected that can be used to make predictions about the results?
- execution context: what is the status of the execution of the plan? have queries to search engines failed?
- result context: are the results what was expected (quality and quantity)?

There will be many c-schemas representing each kind of context. For example, there will be c-schemas for various user contexts corresponding to particular users, kinds of users, and so forth. Although in general the agent's current situation will be represented by finding and combining different kinds of c-schemas, there will be some some c-schemas that directly represent combinations of kinds of contexts, for example, a particular kind of user making a particular kind of query. The general rule will be that unless such a combination makes predictions for behavior that cannot be generated by combining the components, then only the components will be represented. This will allow a large number of context representations to be generated from a relatively few c-schemas. It also allows novel contexts to be represented by composing existing c-schemas.

The agent will need to recognize its initial context and to monitor the situation to detect changes in the context. We will use a separate context manager module for this purpose. This will be a version of ECHO, the Embedded Context-Handling Object, which is discussed elsewhere [4]. Briefly, ECHO will watch the agent's evolving problem-solving situation and use features of that situation to recognize the aspects of the current context by a diagnostic reasoning process. The c-schemas "diagnosed" as fitting the current situation will be merged, and knowledge from them will be given to the agent's other modules. When the situation changes sufficiently, ECHO will adjust its notion of what the context is to correspond to the new situation.

In order to get the contextual knowledge to the modules that need it, we plan to treat the agent as itself a kind of multiagent system, with each of its modules corresponding to an agent. Each module/agent will register when it starts with ECHO. It will tell ECHO how it can get information from the module about what it is currently doing and how the module would like to access the contextual knowledge. For example, the module might request ECHO to simply notify it when the context changes; the module could then ask for information as it needs it. Alternatively, it could request that ECHO send it all relevant information when the context changes.

When it registers, a module would also tell ECHO what kinds of information it is interested in receiving. The user interface, for example, would request knowledge related to understanding the user's commands and constraints and building

an initial query. The query elaborator would need knowledge about how to elaborate or refine the query in the current context; this information could come from the user context as well as the query context. The task structure generator would need information to allow it to create an appropriate set of plans for the context; this would include constraint information from the user context and hints about search engines to use from the query context. The design-to-criteria scheduler might request information about the user's constraints from the user context as well information about likely success from the search plan context. The execution subsystem might request information related to detecting and handling unanticipated events as well as information about search engines and Web sites it needs to interact with. The report generator might request information from the user context so that it can tailor the report to the user. The search manager would need a variety of information from all the different aspects of the context in order to insure that the overall agent is behaving appropriately for the context. For example, it might request knowledge, perhaps from a c-schema representing the result context aspect, that allows it to decide to devote a little more time than it intended to the search in order to get better results.

The agent's context manager will watch the evolving problem-solving situation to decide when the context has changed. For ECHO, a context change occurs when the set of c-schemas matching the current situation changes. When the context changes, ECHO will notify the other modules and, depending on how they registered with it, send them knowledge about the new context.

In the short term, the agent's knowledge will be provided by humans. In the longer term, however, we would expect the agent to be augmented with the ability to modify its contextual knowledge based on its own experience, including learning about new contexts. Although their discussion is beyond the scope of this paper, context-mediated behavior has some features that should make it relatively easy to acquire knowledge from the sort of similarity-based generalization done in some case-based reasoning systems (e.g., [7]). This sort of learning might be augmented with explanation-based learning mechanisms to allow the agent to modify its contextual knowledge base over time.

4 Related Work

There are many existing intelligent information retrieval (IR) systems for the World Wide Web and other distributed information sources, for example, BIG [9], WebACE [10], InfoSpiders [11], OySTER [12], and WebMate [13]. Of these, BIG is most similar to FERRET. We use some of the same components and technologies, in particular TAEMS task structures and the Design-to-Criteria scheduler.

Although context is taken into account by some of these systems, it is usually narrowly defined and plays a relatively minor role. The most common contextual knowledge that is represented and used by intelligent IR systems is user profiles. Generally, the profiles are learned by the system based on observing the user's behavior as he or she uses the system or from explicit user feedback. For example,

various clustering and other machine learning techniques are used by WebACE, OySTER, and WebMate. In some cases (e.g., WebMate), the user model is as simple as a set of vectors describing "domains of interest". In these approaches, the vectors hold weighting information for words in documents based on how often a word occurs in a document matching that domain of interest and how often such documents contain the word.

Some systems do attempt to model and use other aspects of context. For example, the SINGLESOURCE system [14] has a simple representation of the task context that various tools use to aid selection of documents and for other purposes. Another system, InfoSpiders [11], uses neural net agents in an evolutionary model to create connectionist representations of the local context of individual documents. Some information filtering systems, such as Ringo [15], use a kind of extended user context by incorporating information from other users' preferences to suggest documents (or music, in Ringo's case) the user might find interesting.

Our approach to representing and using contextual knowledge, context-mediated behavior, was first developed in the MEDIC and Orca projects [16], and its development continues in Orca. MEDIC was a medical diagnostic system, and Orca is an intelligent mission controller for autonomous underwater vehicles (AUVs). CMB is an active focus of research in our laboratory. In addition to the AUV and Web search agent domains, it is being considered for use in intelligent multiagent system control [17].

5 Conclusion and Future Work

Context is important in all facets of an intelligent Web search agent's activities. Its context includes aspects relating to user or the kind of user, the search query itself, other elements of the task, the information sources it is considering using, and the results so far obtained. By explicitly representing knowledge about the contexts in which the agent may find itself, the agent can compare its current situation to its contextual knowledge to determine what its current context is, then use its contextual knowledge to determine how behave appropriately for the situation. This should help the agent search more efficiently, as well as provide better results for the user.

We are just beginning the FERRET project, which aims to build a context-sensitive, intelligent Web search agent. The project is just beginning, although it builds on current work by some of the authors on intelligent Web searching and context-sensitive reasoning. FERRET will be designed along the lines described above to create a tool for serious musicians and musicologists to use to search the Web.

The project provides a rich testbed for our work on context-sensitive reasoning. During the project, we will examine the question of which aspects of context are important to identify and represent in this domain, which will complement earlier work [5]. Another research topic will be the contextual knowledge in these aspects of context: what knowledge does it make sense to represent given the

Web search task, and how should it be represented? A version of the context manager, ECHO, is currently being developed in the Orca project. This will be be extended and tested in FERRET's domain. Context diagnosis and merging contextual knowledge from different aspects will be important topics in this work. Contextual knowledge maintenance will also be very important. Ultimately, techniques from case-based reasoning and similarity-based and explanation-based learning will need to be brought to bear to allow the search agent to update and maintain its own knowledge about contexts.

References

1. D. S. Haverkamp and S. Gauch. Intelligent information agents: Review and challenges for distributed information sources. *Journal of the American Society for Information Science*, 49(4):304–311, 1998.
2. K. Decker. Task environment centered simulation. In M. Prietula, K. Carley, and L. Gasser, editors, *Simulating Organizations: Computational Models of Institutions and Groups*. AAAI Press/MIT Press, 1996.
3. T. Wagner, A. Garvey, and V. Lesser. Complex goal criteria and its application in design-to-criteria scheduling. In *Proceedings of the Fourteenth National Conference on Artificial Intelligence*, pages 294–301, July 1997.
4. R. M. Turner. Context-mediated behavior for intelligent agents. *International Journal of Human–Computer Studies*, 48(3):307–330, March 1998.
5. E. H. Turner, R. M. Turner, J. Phelps, C. Grunden, M. Neale, and J. Mailman. Aspects of context for understanding multi-modal communication. In *Proceedings of the 1999 International and Interdisciplinary Conference on Modeling and Using Context (CONTEXT-99)*, Trento, Italy, 1999.
6. R. M. Turner. A model of explicit context representation and use for intelligent agents. In *Proceedings of the 1999 International and Interdisciplinary Conference on Modeling and Using Context (CONTEXT-99)*, Trento, Italy, September 1999. (Available as a volume in *Lecture Notes in Artificial Intelligence*, Springer-Verlag.).
7. H. S. Shinn. The role of mapping in analogical transfer. In *Proceedings of the Tenth Annual Conference of the Cognitive Science Society*, pages 738–744, Montreal, Canada, 1988.
8. K. Sycara. Arguments of persuasion in labour mediation. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI-85*, pages 294–296, Los Angeles, 1995.
9. V. Lesser, B. Horling, F. Klassner, A. Raja, T. Wagner, and S. X. Zhang. BIG: A resource-bounded information gathering agent. *Artificial Intelligence*, 118(1–2):197–244, 2000.
10. D. Boley, M. Gini, R. Gross, E. Hong Han, K. Hasting, G. Karypis, V. Kumar, B. Mobasher, and J. Moore. Document categorization and query generation on the World Wide Web using WebACE. *Artificial Intelligence Review*, 13(5):365–391, 1999.
11. F. Menczer and R. K. Belew. Adaptive retrieval agents: Internalizing local context and scaling up to the Web. *Machine Learning*, 39(2/3):203–242, 2000.
12. M. E. Müller. Machine learning based user modeling for WWW search. In Workshop on Machine Learning for User Modeling, 7th International Conference on User Modeling, 1999.

13. L. Chen and K. Sycara. WebMate: A personal agent for browsing and searching. In *Proceedings of the 2nd International Conference on Autonomous Agents and Multi Agent Systems*, Minneapolis, MN, May 1998.
14. K. D. Fenstermacher and C. A. Marlow. Supporting consultants with task-specific information retrieval. (Presented at the AAAI-99 (National Conference on Artificial Intelligence) Workshop on Intelligent Information Agents. On the WWW as `http://people.cs.uchicago.edu/~fensterm/writing/AAAI-workshop99.pdf`, accessed January 18, 2001.), 1999.
15. U. Shardanand. Social information filtering for music recommendations. Master's thesis, Massachusetts Institute of Technology, 1994.
16. R. M. Turner. *Adaptive Reasoning for Real-World Problems: A Schema-Based Approach*. Lawrence Erlbaum Associates, Hillsdale, NJ, 1994.
17. R. M. Turner and E. H. Turner. Organization and reorganization of autonomous oceanographic sampling networks. In *Proceedings of the 1998 IEEE International Conference on Robotics and Automation (ICRA'98)*, pages 2060–2067, Leuven, Belgium, May 1998.