A Student's Assistant for Open e-Learning

Aparna Lalingkar
IIITB*
Bangalore, India
e-mail: aparna.l@iiitb.ac.in

Srinivasan Ramani
IIITB* and HP Labs India
Bangalore, India
e-mail: ramanisl@vsnl.com

***Abstract*- Students often need to go beyond the canned content provided by an e-Learning system's designer. A teacher usually suggests a supplementary corpus, traditionally in the form of books for optional reading and consultation. In an e-Learning context the supplementary corpus would consist of resources which are accessible over the Web, some specified by a teacher and some identified with the help of a search engine. The location of the resources is irrelevant as long as they are accessible over the Web. We offer a structured description of a student's assistant designed to identify relevant information from the supplementary corpus. The users are students who indicate a need for help in specific e-Learning situations. Student's learning styles and personalization issues are discussed in this context.**

Keywords*- *e-Learning; Student's Assistant; Agent; Web Search; Open Corpus; Personalization

I. INTRODUCTION

The World Wide Web (WWW) offers us a rich collection of reusable [1, 2, 3] educational resources such as electronic textbooks, tutorials, resources from educational digital libraries (DLs), various repositories of educational materials [4, 5]. The Open Educational Resources (OER) movement has led a large number of reputed organizations, for instance, members of the Open CourseWare Consortium (OCW Consortium) [6] and contributors to the National Programme on Technology Enhanced Learning (NPTEL) [7], to make valuable content freely available over the Web. Some of the content in OER is valuable for re-use in different e-learning systems; some of it is also interoperable.

In regard to the quality and reliability of content, the reputation of its source, such as a respected institutional website or a widely used public collection offer significant evidence. Given the availability of such resources, the student need not be confined to the content and questions planned and canned in an e-Learning system. Even in the middle of an e-Learning session, she should be free to access supplementary reading material.

We offer a structured description of a system – a student's assistant named Helpsys, which is meant to be prototypical student's assistant. In the next section we discuss open corpus adaptive e-learning systems by defining supplementary corpus, discussing personalization and role of learning styles in student's learning followed by a discussion of problems faced by a student to search relevant information on the Web.

II. OPEN CORPUS ADAPTIVE E-LEARNING SYSTEMS

A. *Supplementary Corpus*

Brusilovsky & Henze, [4] define a closed corpus of documents as one in which documents and relationships between the documents are known to the system at design time. They define an open corpus as a set of documents that is not known at design time and, moreover, can constantly change and expand. They have pointed the need to go beyond the closed world of canned content.

The designer of an e-Learning course may insert a number of hyperlinks in his material to point to supplementary material he recommends. We define a supplementary corpus as a collection of resources which are specified by teacher as possibly relevant educational materials, including the open set of relevant resources available on the Web. Some of the supplementary material may be hosted locally on the school's LAN, as the location of a resource does not matter in a networked environment.

B. *Personalization*

Using a cognitive model of a student covering her needs, knowledge, skills, preferences and performance is the essence of personalization in an e-Learning context. It leads to more efficient learning. Many researchers have discussed it [4, 2, 3, 8] and have provided for some personalization [5, 1] in the systems developed by them. Adaptive educational systems make some provision for taking into account the preferred learning style of the student. Dagger et al. [9] address challenges faced while creating personalized e-Learning content such as complexity and time involved in composing the adaptation component. Henze et al. [10] have proposed and discussed the use of personalized ontologies for creating personalized e-Learning.

C. *Learning Styles*

Different students have different learning needs as per their knowledge level, learning styles, and preferences [4, 8, 11, 12]. Felder [13] has mentioned several research efforts that show students' characterization as per their different learning styles: students focus on different types of information and have inclinations to deal with, and understand, information provided. Further, the understanding achieved could be different with different students. Recognizing the learning style of a student contributes to her interest in learning [13].

*International Institute of Information Technology Bangalore

Reategui and Zattera [14] describe an interface agent as one which communicates with users in natural language and promotes collaboration by encouraging students to help each other. They have taken into account student learning styles to make suggestions for collaboration between students who are likely to be helpful to each other. They have reported the positive impact of interface agents in students' learning, both in the students' perception of their learning experiences and in their actual performance doing a particular assignment.

D. The Need

Consider a student who needs to go beyond the resources identified by the teacher, to find relevant material on the Web. This may include those resources that have appeared after the course material was authored. One option is for the student to use a search engine to locate documents on the net and to look through a number of these documents to get the information she needs. However, this poses a few problems:

- Students, can, in general do casual searches for information; but very few of them manage to acquire an adequate skill set to be efficient searchers.
- Search engines usually report a large number of hits and the student would need to examine many of them.
- Some of these may be long documents and the student would need to search inside the documents to find what she is looking for.
- Some of the documents found may be unsuitable to the student because they are written for persons with a higher level of education.
- Surfing the Web for information often results in frequent distractions and slows down the student's progress.
- Searches often point to the home page with no directly accessible relevant information, even though the snippet may show a few relevant lines. In some cases, the student might even have to navigate from the home page down to a text downloadable from that site and to search many pages to locate what is relevant.

If an e-Learning system is to recommend relevant content to the student, the search has to be easy to use and quite precise. In ease of use and precision, the tool has to be superior to a search engine.

Nature and requirements of Helpsys followed by detailed description of its structure are presented in the next section.

III. Student's Assistant – Helpsys

A. An Overview

Helpsys (Fig. 1) is a system we propose as a solution to many of the problems mentioned above. The issues involved in the design of this system are discussed here.

Helpsys needs to give timely and appropriate help to a student facing some difficulty in a learning situation, taking into account the information available on that individual student, including some description of what the student knows. Helpsys should display documents selected from selected parts of the Web to help the student understand the Learning Situation (LS) better. It is important that only selected extracts, referred to here as segments, from a few relevant documents are displayed to her. How can a computer application search the Web and locate information relevant to a given LS in an e-Learning situation? We visualize LS as a set of related "screens" (the focus being on the text content) presented by an e-Learning System.

We define LS as a sequence of 0 or more Information Screens (IS) presented to the student optionally ending with a Question[1] Screen (QS), which may contain a test question. The student should be able to ask for help when she is reading any screen of LS. Helpsys should have access to all the screens that have been presented to the student, and to the success or otherwise of a student in answering each question attempted by her. One way of ensuring that Helpsys gets information on what the student is reading is to have it interwork with the browser used. Then Helpsys can retrieve a copy of every web page being displayed by the browser. Many e-Learning systems store student performance on a database. In the case of the Learning Management System Moodle, the database used is MySQL. Helpsys could get information on the student's performance on each item from the database.

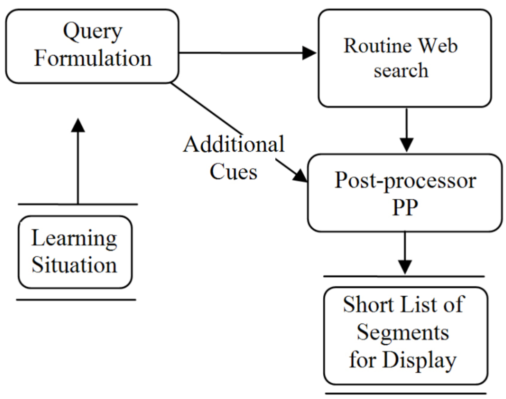

Figure 1. Structure of Helpsys

Helpsys (Fig. 1) starts with LS as the input and returns a set of recommendations of what is to be displayed to the student who has invoked it. It does this by first creating a query for a Web search and uses a search engine to get a list of possibly relevant material. Then a Post-Processor evaluates the results given by the search engine to select what needs to be presented to the student. A structured description of the system is presented in Subsection C. of this Section.

[1] The word "question" is used to refer to the natural language question given in an LS. In contrast, the word "query" is used to refer to a system-generated sequence of query words given to a search engine.

Helpsys derives necessary information from these screens to dynamically maintain a personalized vocabulary for each student. It starts initially with a default vocabulary common to all students, such as the vocabulary involved in the previous grade of study. We refer to this default vocabulary as the vocabulary of the base corpus (VBC). As the student progresses in her e-Learning sessions, her initial vocabulary is built up by adding words from the new material that she has learnt.

An associated problem is one of what to present to the student [15]. It is better to save the student a lot of time by giving a few relevant segments instead of giving him whole web pages or other documents. Techniques for identifying snippets from documents usually focus on getting two or three lines in which query terms occur. However, what are needed in this application are (the longer) segments far more likely to contain the information searched for. The challenge of comprehending and assimilating what is given is not reduced by making it easier to locate the relevant segment. It is essentially a useful feature of the user interface, which makes learning easier. Further, the student does not need to be denied access to the whole document. A good solution is to present the most relevant paragraph(s) on a single screen, and allow the student to access the whole document by scrolling if necessary.

Helpsys allows the student choose the type of information – text, images, video clips, demos etc. - she prefers to learn from. As soon as the student indicates a need for help by clicking on a tab, Helpsys displays a menu of such options for the student to choose from (Fig. 2). The search could yield a view graph, a table, a demo, a video clip or a short segment of a video lecture.

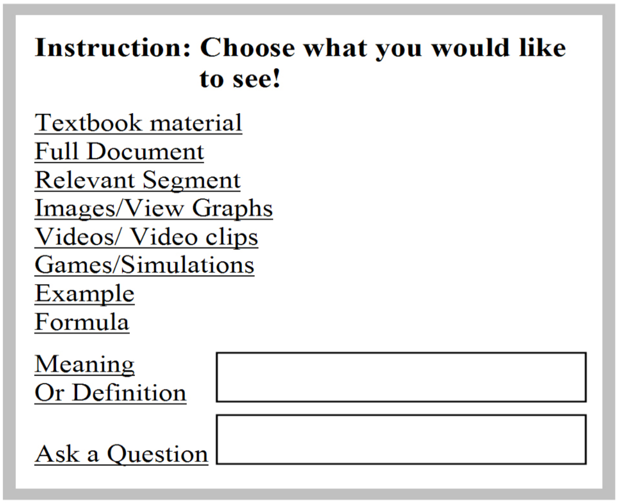

Figure 2. Choices offered by Helpsys

B. *Helpsys Requirements*

We list specific observations motivated by the foregoing:

a) Helpsys should provide assistance to students to locate relevant reading material, images, tools such as games, or other similar resources.

b) Helpsys should take into account the material a student has been exposed to, and any particular question she might be facing when she seeks help.

c) Helpsys should be usable with e-Learning systems in general, and not be confined to use with systems designed to inter-work with it.

d) Helpsys should do significantly better than a search engine used by a student who is not a trained information searcher.

e) Helpsys should have access to information on its user, covering educational level (or grade), language preference, level of academic performance, subject being studied etc.

f) Helpsys should have access to the supplementary corpus specified by the teacher, which may include the whole of the Web, or parts such as specified sites, domains, etc.

g) Helpsys should present one or more relevant segments of resources it identifies, rather than presenting whole documents.

h) Search results should be reused. If help is found for one student in a particular learning situation, the relevant URL should be stored for giving help to other students in the same situation.

i) The stored list of URLs which provide help should be editable to enable a teacher to add or delete items appropriately.

j) Helpsys should select material to be presented to the student, ensuring that words outside the base corpus vocabulary are not too many.

C. *A Structured Description of Helpsys*

We adopt with some modification the notation used by Henze and Nejdl [16] and extend it where necessary. This helps us to describe the structure of Helpsys in an accurate manner. This description is compatible with the manner in which a relational database application is specified. Since Moodle uses MySQL to store and manage its information, Helpsys, as described here, can be easily implemented as a Moodle module.

Definitions

SC: Supplementary corpus, a set of additional books/documents suggested by the teacher
D: The text from the information screen displayed to the student after the immediately previous test item.
TI: Test Item associated with D
LS: Learning Situation: (D, TI), Please see assumptions 1, 2, 3 & 4
Q(D, TI) is a function defined procedurally as follows:

Q(D, TI): the Web search query created from the document D and the test item TI as follows
Please note Q(D, TI) = w_4
Sets w, w_1, w_2 *and* w_3 *are now computed.*

$$w \leftarrow wordlist(D, TI)$$

$$w_1 \leftarrow unique(w - \{stopwordlist\}$$
$$w_2 \leftarrow stem(w_1)$$
$$w_3 \leftarrow sortbyisf(w_2)$$

Where isf is the number of sentences in LS in which a word occurs in the base corpus.

This sort creates w_3, a sorted list of words from w_2 in decreasing order of isf

$w_4 \leftarrow select(w_3, n)$ *We select the first n terms in w_3 to define w_4*

return w_4
end

The procedure defined above formulates search queries using (D, TI) as input.
SCS: Supplementary Corpus Search results returned by the search engine for the query Q(D, TI)
L_k : The student with id equal to k
V_k is L_k's personal vocabulary
Please see assumption 5
PP: Post processor which takes SCS(D, TI), D, TI and V_k as inputs and yields a ranked sequence of segments
PP(SCS(Q(D, TI)), D, TI, V_k) = a ranked sequence of relevant segments, please see assumption 6

Assumptions:

1) Two LSs might have a common D
2) Similarly two LSs might have a common TI.
3) We consider LS as a pair (D, TI). However, we permit cases in which TI is absent. This covers a situation in which a student reads a screen and does not understand it, so asks for help.
4) Similarly, in a LS we permit D to be absent. This is a situation in which the student faces a question without accompanying instructional text. For instance, this case is relevant to a student who is taking an online test. This would also cover the case in which a student types a question into the query box of Helpsys to seek help, stating explicitly what he needs help on.
5) Stop words are not included in V_k; Further, V_k consists of only stemmed, unique words.
6) PP will consider V_k while processing the documents SCS(Q(D, TI)) and will give higher scores to segments which do not demand a vocabulary significantly beyond V_k .

What Helpsys does on a request for help is:

$$\mathrm{Re}\,commend(PP(SCS(Q(D,TI)),D,TI,V_k)$$

In other words, if a student asks for help in any LS then Helpsys recommends the ranked sequence of relevant segments or other unit of information to the student as indicated by her (see Fig. 2).

Now, V_k is the personal vocabulary of student L_k. Let us assume that L_k has moved from Standard(c-1) to Standard c. If the student's grading of L_k at the end of Standard(c-1) has been satisfactory, we may assume that she has understood a large part of the vocabulary, $V_{(c-1)}$ used in the books used in Standard(c-1); therefore, initially in Standard c,

$$\forall k(V_k = V_{c-1})$$

If a student L_k successfully answers the question associated with LS then we can increment his vocabulary as follows, adding to it the list of words found for the learning situation LS, by carrying out the following operation:

$$V_k \longleftarrow V_k \cup V_{LS}$$

Where V_{LS}, is the vocabulary of LS, which is the set of stemmed, unique words from LS.

The next section compares Helpsys with a number of other related systems reported in literature.

IV. DISCUSSION

Several authors have reported agents for helping users in Web search. Letizia [17] is a user-interfaced agent which assists in Web-browsing. It watches pages visited by a user and her browsing behavior, to recommend other material on the web expected to be of interest to her. Letizia works in tandem with a Web Browser. In comparison, we visualize Helpsys working in tandem with an e-learning system, sharing a browser with it. Helpsys depends on a search engine as a tool to collect possibly relevant information, for further processing and selection for presenting to the student.

Systems like Web-Watcher [18] and Lira [19] take keywords from users and suggest hyperlinks. They consider user's evaluation of the information searched to improve future searches. For a given query Musag [20] generates a kind of thesaurus of semantically related concepts for each keyword and uses this thesaurus for further document retrieval.

The challenges faced by Helpsys are significantly different from the ones faced by Recommender Systems [17, 18, 19, 20] in general. Some of the differences are:

- Recommender systems base their decisions on several web pages the user has accessed. In contrast, Helpsys places considerable emphasis on the immediate LS the user is in.
- The search relevant to a given LS would cover the base corpus in addition to the supplementary corpus if the teacher wishes to adopt an open-book testing style during e-Learning sessions.
- The vocabulary of the base corpus is used by Helpsys to recommend reading mostly covered by this vocabulary.

The concept of Question Answering Systems [21, 22] is relevant here. Such systems go beyond document retrieval and present specific answers. These systems need to mine the documents they locate for information, and reason out the answer. This is a difficult task except in restricted contexts. It involves natural language understanding, which is an AI complete problem [23].

After giving a query to a search engine and getting the results, AnswerBus [22] carries out answer-extraction from the retrieved documents by categorizing words in a document as matching or not matching the original query words. It ranks answers, or the documents containing the answers, by using various techniques such as use of Question-Type, named-entities extraction, co-reference resolution, hit ranking and search engine confidence, and detection of sentence redundancy. In contrast, our focus is on helping a student in an e-learning situation, in which it is not necessary to do information mining and provide the answer. It is pedagogically more attractive, and sufficient to present relevant content to the student and let her do her own interpretation and comprehension. Finding relevant content with perfect precision may also be an AI complete problem, but approximate solutions are more acceptable in this context than in question answering.

The issue improving precision in search is discussed in the next section, referring to a sub-project.

V. COMPANION WORK

A major direction for future investigation involves identifying progressively better natural language processing techniques to improve post-processing of search results. Ideally the LS should be analyzed to identify concepts and relations[2] to search for during post-processing. A step towards this has been taken by a sub-project [24] which has used word-pairs for document search. LS is analyzed and a set of word-pairs is identified. Two words can constitute a pair only if they co-occur in one sentence. This set of word-pairs is used in post-processing to find relevant documents. Documents in which these word-pairs occur are given a higher score.

Vocabulary and word-associations within the base corpus are not the same as a regular ontology in terms of concepts and relations. However, vocabulary and word-associations do constitute an easy-to-use language model which enables better search for relevant material on the Web.

VI. CONCLUSION

Students facing a difficult learning situation during an e-Learning session often need to find relevant reading material from an open corpus. Helpsys, a Student's Assistant for identifying such information has been proposed, and structured description of the system has been offered. The work on Helpsys has taken us in the direction of mapping a learning situation into a search engine query and post-processing the search results given by the search engine. A technique has been described for maintaining the presumed vocabulary of the individual learner on the basis of demonstrated learning outcomes. This vocabulary enables a form of personalization in identifying information meant to help the student facing a difficulty during an e-Learning session.

[2] In the sense of description logic

Two sub-projects were spawned, one investigating the use of word-pairs taken from a learning situation for carrying out a search for intra-sentential word associations [24] to locate relevant documents or segments of documents. This sub-project implemented a re-ranking algorithm using two ideas: association search and vocabulary comparison.

Another sub-project [15] focused on locating multiple segments from multiple documents and ranking all the segments obtained on the basis of matching with query words.

Both sub-projects have carried out experiments and have presented early results.

ACKNOWLEDGMENT

We thank our project colleagues Venkatagiri Sirigiri, Sirisha Borusu, Sivaramakrishna Garimella and Madhav Sharma for sharing with us the tools developed by them. We acknowledge with thanks the full support given to this project by HP Labs India and the International Institute of Information Technology, Bangalore. This research is funded by HP Labs India

REFERENCES

[1] Sampson, D., Karagiannidis, C. and Cardinali, F. "An Architecture for Web-based e-Learning Promoting Re-usable Adaptive Educational e-Content", Educational Technology & Society, Vol. 5, No. 4, 2002, pp. 27-37.

[2] Brusilovsky, P. and Nijhawan, H. "A Framework for Adaptive E-Learning Based on Distributed Re-usable Learning Activities", In In: M. Driscoll and T.C. Reeves (eds.) Proceddings of World Conference on E-Learning, E-Learn 2002 (Montreal, Canada).

[3] Dolog, P and Sintek, M. "Personalization in Distributed e-Learning Environments", WWW2004, May 17-22, 2004, New York, USA, pp.170-179.

[4] Brusilovsky, P. and Henze, N. "Open Corpus Adaptive Educational Hypermedia", in The Adaptive Web, LNCS 4321, P. Brusilovsky, A. Kobsa, and W. Nejdl Eds. Springer-Verlag Berlin Heidelberg, 2007, pp. 671-696.

[5] Henze, N. and Nejdl, W. "Adaptation in Open Corpus Hypermedia", International Journal of Artificial Intelligence in Education, 12, 2001, pp. 325-350.

[6] Iiyoshi, T; Vijay Kumar, M. S., Eds. "Opening Up Education: The Collective Advancement of Education through Open Technology, Open Content and Open Knowledge", Cambridge, MA: MIT Press, 2008.

[7] Bhattacharya, B. "Distance education through technology mediated learning: The engineering education scenario in India", Paper presented at the Third Pan-Commonwealth Forum on Open Learning, Dunedin, New Zealand, 2004.

[8] Muntean, H. C. and Muntean, G. M. "Open corpus architecture for personalized ubiquitous e-learning", Pers Ubiquit Comput, 13, 2009, Springer-Verlag London, pp. 197-205.

[9] Dagger, D., Wade, V., & Conlan, O. "Personalisation for All: Making Adaptive Course Composition Easy", Educational Technology & Society, Vol. 8, No. 3, 2005, pp. 9-25.

[10] Henz N.; Dolog P. and Nejdl W. "Reasoning and Ontologies for personalized E-Learning in the Semantic Web", Educational Technology and Society, Vol. 7, No. 4, 2004 pp. 82-97.

[11] Parvez, S. M. and Blank, G. D. "A Pedagogical framework to Integrate Learning Style into Intelligent Tutoring Systems", JCSC, Vol. 22, No. 3, 2007, pp. 183-189.

[12] Felder, R. M. and Spurlin, J. "Applications, Reliability and Validity of the Index of Learning Styles", International Journal of Eng. Education, Vol.21, No. 1, 2005, pp. 103-112.

[13] Felder, Richard, "Reaching the Second Tier: Learning and Teaching Styles in College Science Education", J. College Science Teaching, Vol. 23, No. 5, 1993, pp. 286-290.

[14] Reategui, E. and Zattera, C. "Do Learning Styles Influence the Way Students Perceive Interface Agents?", IHC2008 – VIII Proceedings of the VIII Brazilian Symposium on Human Factors in Computing Systems, October 21-24, 2008, Porto Alegre, RS, Brazil, pp. 108-116.

[15] Sirisha Borusu,"Extending Search Inside Document", Unpublished M. Tech Thesis, June 2009, IIITB, Bangalore, India.

[16] Henze, N. and Nejdl, W. "Logically Characterizing Adaptive Educational Hypermedia Systems", Paper presented at the International Workshop on Adaptive Hypermedia and Adaptive We-based Systems (AH 2003), 20-24 May, Budapest, Hungary, 2003.

[17] Lieberman, H. "Letizia: An Agent That Assists Web Browsing", Proc. 14th International Joint Conference AI (IJCAI95), AAAI Press, Menlo Park, Calif., 1995, pp. 924-929.

[18] Armstrong, R., Freitag, D., Joachims, T. and Mitchell, T. "WebWatcher: A Learning Apprentice for the World Wide Web", AAAI 1995 Spring Symp. Information Gathering from Heterogeneous, Distributed Environments, AAAI Press, Menlo Park, Calif., 1995.

[19] Balabanovic, M. and Shoham, Y. "Learning Information Retrieval Agents: Experiments with Automated Web Browsing", AAAI 1995 Spring Symp. Information Gathering from Heterogeneous, Distributed Environments, AAAI Press, Menlo Park, Calif., 1995.

[20] Goldman, C. V., Langer, A. and Rosenschein, J. S. "Musag: An Agent That Learns What You Mean", Applied AI, Vol. 11, No. 5, 1997, pp. 413-435.

[21] Ravichandran, D. and Hovy, E. "Learning Surface Text Patterns for a Question Answering System", Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL), Philadelphia, July 2002, pp. 41-47.

[22] Zheng, Z. "AnswerBus question answering system" In E. M. Voorhees and Lori P. Buckland, editors, Proceedings of HLT Human Language Technology Conference (HLT 2002), San Diego, CA, March 2002, pp. 24–27.

[23] A. G. Hauptmann, "Speech Recognition in the Informedia Digital Video Library: Uses and Limitations", Proc. of ICTAI-95 7th IEEE Int. Conf. on Tools with AI, Washington, DC

[24] Sirigiri, V. S., Sivaramakrishna, G., Sharma, M., Lalingkar, A. and Ramani, S. "Finding documents on the web relevant to an e-Learning situation", submitted for International Workshop on Technology for Education T4E, Aug 2009, Bangalore, India (Paper Id: 156228627)