Awareness Mechanisms for an Intelligent Tutoring System

Roberto Pirrone and Vincenzo Cannella and Giuseppe Russo
DINFO - Dept. of Computer Science and Engineering, University of Palermo
Viale delle Scienze, Building 6 - 90128 Palermo, Italy
pirrone@unipa.it, {cannella, russo}@csai.unipa.it

Abstract
In this work an Intelligent Tutoring System (ITS) is presented that is able to enrich its knowledge base, merging both partially structured and unstructured information in a unique internal representation framework. The system gathers new information about the domain in response to both a curiosity mechanism and sensory stimuli coming from the interaction with the user. The architecture is an evolution of TutorJ, an actual ITS already presented by some of the authors. The cognitive framework is a very promising one to overcome the limitations of classical ITSs, and to achieve the stated goal. The use of cognitive architectures is intended for a double result: a lower effort for a knowledge-intensive activity like the construction of an ITS and better results in terms of a rich interaction with students. In this framework the system is aware of its internal status, and a curiosity mechanism has been modeled to gather new knowledge from web sources that is integrated in the original knowledge base. The whole architecture is explained in detail, and the information fusion mechanisms are presented along with a simple case study.

Introduction
A learning agent embodied in an interactive system has to improve effectively the student’s skills using different learning modalities to achieve major learning objectives such as knowledge acquisition, comprehension, application, analysis, synthesis, and evaluation. The effectiveness increases if the student has a good perception of the gaps she is bridging and a clear explanation of how the learning strategies are performed. Another important aspect is related to the student’s capabilities to mix the efforts useful to increase her knowledge. In the context of Educational Psychology a student able to define autonomously the way to increase her knowledge performs a self-regulated learning process (Zimmermann and Schunk 1998). Students that are able to self-regulate themselves have some characteristics (Zimmermann 2001) (Montalvo and Gonzales-Torres 2004):

- They are able to perform cognitive strategies to deal with the information size growth.
- They are able to adopt time and effort strategies.
- They presents an high level of curiosity and motivation for novelties.
- They are committed to result’s achievement.
- They are able to perform metacognition processes (strategies about cognition process).

The design of an intelligent agent has to face the challenge to interact with this kind of students. It has to be proactive and based on heterogeneous and dynamic sources of knowledge. Integration is a key but is still far from being achieved in the AI. An autonomous system should be able to adapt itself and to increase its cognitive skills with different kind of memorization (working, long-term, episodic, deep). The emotional aspect is another important issue as a possible key to approach social interaction.

In this work we present a cognitive architecture for an ITS that extends TutorJ an actual system already presented by some of the authors (Pirrone, Rizzo, and Pilato 2008). We increased the awareness of the system that is more autonomous than the previous one in the task of providing the user with the best reply to her requests. The architecture we present has a curiosity mechanism leading the system to enrich its knowledge base with new information gathered both from partially structured (wiki) and unstructured (folksonomies) sources on the web. The new knowledge and the related documents are represented in a unique internal framework both as symbolic and sub-symbolic information. When stimulated the system acts as a learner developing both perceptual curiosity when the systems perceives the student, which wants to deepen a topic that is not well presented, and epistemic curiosity when the systems wants to explore autonomously its environment to gather new knowledge.

The rest of the paper is arranged as follows: next section presents the state of the art about Intelligent Tutoring Systems that are regarded as learning enabled cognitive architectures. The third section describes the architecture of TutorJ, while the fourth one points out its limitations as a cognitive architecture. The fifth section describes the presented architecture, and the sixth one explains the information fusion mechanism we implemented along with a simple case study. Finally, the last section reports some conclusions and future work.
State of the Art

A traditional ITS architecture is defined in terms of some cooperating modules (Wolff et al. 1998), (Rickel 1989). The main components are the expert model, the student model, the instructional model, and the learning environment. Traditional approaches to e-learning are based on the definition of the most appropriate environment for the student’s individual needs (Piramuthu 2005). Moreover, the design of the student model, the expert model and the instructional model are necessary. The expert module is able to explain the knowledge of the system and to provide the student with corrections and hints. The student model is used to compare student’s abilities and skills to the expert model. The instructional model is used to present new knowledge in a suitable way for student’s skills acquisition. Many theoretical models to build an ITS have been proposed while the most used are Model-Tracing Tutors (MTTs) (Gertner and VanLehn 2000) (Butz, Duarte, and Miller 2006) and Constraint-Based Model Tutors (CBMTs) (Mitrovic, Martin, and Suraweera 2006) (Mayo, Mitrovic, and McKenzie 2000) (Gonzalez, Burguillo, and Llamas 2007). Other used approaches are the bayesian one (Butz, Hua, and Maguire 2006) while another important effort is in the direction to define them explicitly as cognitive architectures (Goh and Quek 2007) (Samsonovich et al. 2008).

Cognitive Architectures

Probably, the Human Information Processor Model (HIRM) is one of the main models developed to describe human cognitive processes. It has inspired the Model Human Processor (MHP). According to this model a cognitive architecture is composed by a series of modules. Each module is devoted to a specific function involved in the cognitive processes. There are three main kinds of modules: perceptual module, cognitive module and motor module. EPIC (Executive Process-Interactive Control) (Kieras, Wood, and Meyer 1997) is inspired to the MHP, and is focused both on the human cognition and on the performance. EPIC is a generative model. The production-rule cognitive processor generates a proper action in response to a certain stimulus. The main perceptual processors are the visual and the auditory ones. EPIC has no autonomous learning mechanism. ACT-R (Anderson et al. 2004) (Newell 1994) is a hybrid system based on chunking. The ACT-R architecture consists of a module to retrieve information from a declarative memory, a procedural memory, a goal stack to manage the current goal, and a visual module to identify objects. The declarative memory containing the chunks, and the procedural memory containing rules are symbolic structures. They’re accessed by a neural activation mechanism. Access to declarative chunks is determined by an activation representing the probability the chunk has to be retrieved. It is a fixed-attention architecture. ACT-R is goal-oriented system. At any time, it is focused on a single goal and a single rule fires. Pattern matching processes select the production rules to fire, while conflict resolution select applicable rules. ACT-R is also able to learn automatically chunks and rules. Soar (Laird 1993) (Newell and Laird 1983) (Laird and Rosenbloom 1993) (Laird et al. 1993) is a theory about cognitive architectures defining a computational model to describe the way a system can reach a goal. Soar defines a problem space for a task domain that consists of a set of states and a set of operators. Each state represents possible situations in the domain. Operators transform one state into another one. Soar selects between available problem spaces, states, and operators, during the process of reaching a goal. A problem is solved through a sequence of decisions that are selected by a knowledge-driven mechanism. Problem spaces are specialized for external interaction, natural language processing, design, and so on. There are also general problem spaces supporting the computations required in the specialized problem spaces. Problem spaces are dynamic, as new operators and even new problem spaces can be learned through experience. When it lacks the knowledge to solve a problem Soar can move it from the native problem space to a more complete one. CAPS (Just, Carpenter, and Varma 1999) is a production system based cognitive architecture that has been inspired by the functional aspects of the human brain. It is focused on short-term memory, reading and language. Its procedural knowledge is made up by a collection of if-then rules. When a rule fires, it is executed, and changes the activation of the other ones. Apex (Freed 1998) is a software tool developed to simulate human agents. An agent has limited resources and acts in complex, time-pressured, and uncertain task environment. Apex provides the agent with the ability to manage its resources while accomplishing multiple tasks in parallel.

TutorJ

TutorJ (Pirrone, Rizzo, and Pilato 2008) is an integrated Intelligent Tutoring System able to understand a specific request made by a student about a particular subject, and to suggest a suitable learning proposal. It was developed by some of the authors also on the basis of some previous studies about conversational agents (Agostaro et al. 2005). The system assesses the level of training of the student. Our approach follows the MTTs. Fig. 1 illustrates the architecture.

Expert Model

Our expert model is codified into a Cyc-based (Matuszek et al. 2006) commonsense ontology. This ontology describes both a collection of facts about the domain and the attitude of the teacher when she reviews all the concepts needed to present a specific topic. Several information sources may be integrated in the ontology. The integration process is one of the novel topics of this work and will be discussed later. The way to give information to students is inspired to face-to-face lessons. Student-system interaction makes use of natural language, and TutorJ provides lessons and suggestions according to its knowledge about the domain and to a set of prerequisites between the concepts that are coded inside the ontology. The ontology has a hierarchical representation relying on two levels. The first level represents explicitly the structure of the domain using typical relations like generalization, composition and so on. The second level tries
The TutorJ provides an environment to assess the student and to supply the learning materials. The student is assessed by a question/answer procedure through a natural language interaction. The dialogue is managed by a chatbot that is an artificial conversational agent extending the A. L. I. C. E. architecture (A.L.I.C.E. 2008). The chatbot communicates with the ontology to provide a learning path between the topics to be learned by the user. Moreover it communicates with the LSA module that support assessment projecting the user sentences in the LSA space to evaluate their closeness to existing topics in the KB. A 2D visual map representing the LSA space is used to browse documents and to draw the learning paths. The map itself is our instructional model. It is implemented using AJAX and SVG. Documents related to the same topic are clustered using a 2D SOM network (Kohonen 1995) as constellations in a starred sky.

**System limitations**

The presented architecture is general and must be targeted for a specific subject. Each new instance of the architecture implies the development of both levels of the domain ontology and the LSA processing of the learning materials to create their space. Moreover, these instances need to be frequently updated. This is a typical problem in ITS literature. The main drawback of our architecture is that it is not able to modify itself to learn form its experience. TutorJ is a “rational” system, not a cognitive one. It is able to interact with the student to plan her studies, but it is not able to adapt the planning procedure according to the changes in the domain because it’s not able to learn from experience. Its knowledge can be modified only thanks to manual intervention. The web is a very important and dynamic source of information. An example are wikis and folksonomies. A cognitive system could learn from these sources that provide both new symbolic knowledge and new documents. Such a cognitive system should be “aware” to perceive the modifications of its internal status and add new knowledge through a spider that browses continuously several web sources. This particular behavior of the system is similar to the behavior of a human teacher that must study to remain up-to-date. In particular it can be modeled as a curiosity behavior. To achieve this goal several mechanisms have to be integrated in the present architecture. At first, heterogeneous information sources have to be integrated: wikis posses their own structure that can be imported in the ontology, but folksonomies do not. Moreover, concepts synonymy has to be resolved through a thesaurus. Finally, new knowledge has to be verified as regards both truth and interpretation.

**The proposed cognitive architecture**

We extended the TutorJ architecture in the sense of the HIPM paradigm to overcome the drawbacks explained in the previous section. The new architecture is outlined in fig. 2. Like in ACR-T, memory modules manage both symbolic and sub-symbolic structures. The ontology module manages a symbolic memory regarding the learned domain. The LSA module manages a sub-symbolic memory that merges documents, concepts and terms in a common Euclidean space thus providing an integrated representation of all the domain. It codes the elements with similar meaning as points that are close in the space. WordNet stores a similar knowledge represented in a symbolic form, and it’s used also for
disambiguation purposes. Finally, the memory module for the procedural knowledge inside the chatbot describes natural language interaction structures as suitable rules that are written in Artificial Intelligence Markup Language (AIML). All the previous memories are long-term ones. The system owns also a working memory that stores stimuli from the environment. In turn, stimuli trigger the cognitive module, which evaluates the actions to be executed. The working memory stores also the status of conversation and of the whole system (conversation mode, material supplying, gathering new knowledge). The sensor/motor modules interact with the elements of the environment. The chatbot interface and the map interact with the student using natural language and graphics. The other sensor/motor modules are devoted to the learning material distributed over the web.

Figure 2: The proposed cognitive architecture.

Each module deals with a particular task. Sometimes a complex task is accomplished by several modules together. Possible tasks are the natural language interaction, management of learning contents, planning a learning path. The Chatbot engine is the true cognitive module coordinating the whole system. It interacts with the KB to plan the learning path according to the domain concepts and to the learning strategies that are coded in this memory module. The chatbot engine manages the conversation through its interface using the AIML sentences recorded in the procedural memory. Finally, it communicates with the LSA engine (see fig. 4) to support assessment of the user sentences.

Figure 3: Chatbot module in detail.

The last crucial component of our architecture is the LSA module (see fig. 4). It is composed by the LSA engine that performs similarity measures between concepts and documents in the LSA space. This is the core memory for the domain, and it is produced automatically by the system when loading a collection of documents, wikis, and folksonomies to create a common space. The domain expert supplies the starting collection that is enriched browsing the web.

Figure 4: The LSA module in detail.

As ACR-T, our system is goal-oriented. The system must answer to the student’s questions. The goal induces the construction of two specific sub-goals: the assessment of the training level of the student, and the planning of a learning path customized to the user needs. As already stated, there are two possible interaction modes with the student. In the first one, the system’s attention is focused on the conversation, represented by the log stored in the working memory, the subject under investigation, and the learning path planned at the current time. In the second interaction mode, the attention is turned to what documents the student is reading, or she just read, and what stages of the learning path has been visited. During the conversation, the system detects recursively a series of sub-goals to assess the training level for each specific topic. The detection of these sub-goals is driven by the ontology and by the sub-symbolic memory. When the main module has planned the path, the system switches the interaction mode allowing the graphical browsing of the documents. During this phase the student consults the learning material. The system could also fail when trying to reach the sub-goals, a lack of knowledge. Moreover, the system updates periodically its knowledge base, and consults autonomously the web using specialized spiders to find new or more updated information. In this respect the system can be regarded as a human teacher updating her knowledge. We modeled the goal of increasing the system’s knowledge as a curiosity mechanism. Computational models are so far from a consistent definition of curiosity as it is expressed by psychologists. According to Loewenstein (Loewenstein 1994) curiosity “occurs when an individual’s informational reference point becomes elevated in a certain domain, drawing attention to an information gap”. He proposes an information theory about curiosity like a feeling of deprivation that is more urgent since the awareness of what is known and what is unknown becomes more clear. The two most important points in this theory are the awareness of what is known (and what is unknown) as a trigger for curiosity (perceptual curiosity), and the feeling of deprivation as a result of a lack of information in a particular subject (epistemic curiosity). In our framework both perceptual and epistemic curiosity are present. Perceptual curiosity arises when the system perceives from the conversation that its knowledge is not sufficient to satisfy the student, and starts to browse web sources. Epistemic curiosity arises when the system...
attains what is called optimal discrepancy in Education Psychology. In this case the system measures the difference between the structure (concepts and relations) of the knowledge gathered from the web and the one stored in its KB. If this difference is above a suitable threshold, the new knowledge is integrated in the long term memory.

Information fusion

We analyzed two possible web sources: wikis as a source of partially structured knowledge, and folksonomies as a source of unstructured knowledge. Both of them are integrated in a structured knowledge base that is an OWL ontology where the rules to browse the prerequisites of a concept are coded using JESS. When the spider surfs a wiki, the topics treated by each article and the link structure are written as an OWL document. Links in a wiki represent a logical relation that hasn’t an explicit name or value, so we defined the conceptuallyRelated OWL property whose strength attribute ranges in $[0,1]$. At first, strength is set to 1. The OWL document is processed using WordNet to discover which concepts are synonyms for the ones already present in the KB. Synonyms are used as reference concepts to merge the OWL document created from the wiki with the existing ontology. Two more processing steps are performed. At first, we use a semantic similarity measure already presented by some of the authors (Pirrone et al. 2008) to devise if some sub-graph of the new knowledge structure is analogous to some other one in the existing KB. The same measure is used to assess if the system has attained an optimal discrepancy between its internal knowledge structures and the knowledge gathered from the web. If the similarity measure between two sub-graphs in the merged ontology is above a suitable threshold, a new conceptuallyRelated property is instantiated between their root concepts. Finally, all the new concepts and their descriptions gathered from the wiki are projected in the LSA space, while their URIs are stored in the DB of the learning materials. In the LSA space the distance between each couple of concepts linked by a conceptuallyRelated property is measured to provide a value to the strength attribute. Folksonomies are treated in a similar way. They’re disambiguated using WordNet, and synonyms are removed from the initial bag of concepts. The remaining ones are projected in the LSA space and new conceptuallyRelated properties are instantiated with respect to the concepts in the original KB that are closer than a suitable distance. We implemented our information fusion mechanism in a simple application scenario: the knowledge domain about the pizza. The system has its own ontology (Drummond et al. 2006) and integrates a set of related URIs from a wiki (Wiki 2008). The user wants to know something about the Margherita pizza. The expert model queries his memory and trough a sequence of reasoning steps it is able to reply that a Margherita is a type of pizza with some properties (see Fig. 5): it is a vegetarian pizza with only mozzarella and tomato as toppings. After this first explanation the system can provide the user with more information using the resources in Wikipedia. In the proposed example the system is able tell the user that the name Margherita is taken from an italian queen. This information was not present in the original pizza knowledge base.

Conclusions and Future Work

A cognitive architecture providing awareness mechanisms for an Intelligent Tutoring System has been presented that is able to increase its knowledge to improve the interaction with the user. The presented framework is inspired to the HIPM paradigm and extends TutorJ an actual ITS already presented by some of the authors. The new system is aware of its internal status as regards both the advances in the conversation and the actions to perform to reach its goals. In particular, the system exhibits both perceptual and epistemic curiosity as mechanisms to trigger information gathering from the web. Finally, a novel information fusion procedure has been presented to merge the knowledge from wikis and folksonomies with an existing OWL ontology. Sub-symbolic representation of the system’s knowledge is a key point in this framework because it’s used to provide a strength to the new relations that arise in the enriched KB. The LSA space is a useful tool, but it’s not sufficient for our purposes because it is a simple “bag of words”. Future work will be devoted to transform the LSA space in a true Conceptual Space in the sense of Gärdenfors where each dimension has a meaning, and distance measures between concepts have a straightforward interpretation as symbolic relations. Moreover we’ll refine the information fusion procedure including the analysis of all the information sources in a wiki (metadata, topic descriptions, and so on) to devise also named relations along with their attributes. Finally, the curiosity mechanism will be investigated in more detail in the direction a flexible framework for a system with a true ability to learn from experience.

References


