Applying Logic Inference Techniques for Gaining Flexibility and Adaptivity in Tutoring Systems

Matteo Baldoni, Cristina Baroglio, Viviana Patti

Dipartimento di Informatica, Università degli Studi di Torino
C.so Svizzera 185 – I-10149 Torino (Italy)
E- mail {baldoni, baroglio, patti}@di.unito.it

Abstract

In this article we present the most recent advancements of a research aimed at applying reasoning techniques to the design and implementation of adaptive services in tutoring systems. In particular, we faced tutoring tasks necessary to study plan construction and validation, and proved the usefulness of reasoning about actions techniques such as planning, temporal projection and temporal explanation.

1 Introduction

In this paper we present the most recent results of a work that investigates the application of reasoning techniques, taken from the Artificial Intelligence field of "reasoning about actions and change", to the implementation of adaptive services in a web-based educational system. Our research was settled in an applicative framework close to our experience as university computer science teachers, due to the simplicity of finding and structuring the necessary information, however, the same approach could be used for supporting the definition of other kinds of university curricula.

The issues that we tackled can briefly be described as follows. In Italian universities, students list the courses that they mean to attend in their years as undergraduate students, producing so called *study plans*. Study plans are not definitive but can be changed along time, according to the student's most recent experiences. Study plans should respect the rules stated by given guidelines and their consistency is verified by university professors. Both the definition and the validation of study plans are time-consuming, difficult tasks, which require knowledge about all the available courses (prerequisites, topics that are taught) and about the general rules; in case of study plan validation professors also use information about the student's current situation (e.g. passed exams). Usually, inconsistency resolution can be accomplished only by discussing with the student his/her choices.

In our work we dealt with the above mentioned problems by building a *recommendation system*, named WLog, that can be accessed via the web. Such a system exploits *logic reasoning mechanisms* that work on a knowledge model (the domain), taking into account the student's preferences, goals, and current competences. Notice that our aim was not to build a tutoring system that guides the student in learning some instructional content but a decision support system devoted to the specific task of study plan definition. Therefore we do not monitor the student's progress.

All the reasoning techniques that we have used come from the research area of reasoning about actions and change. In this framework, the basic intuition is that each course can be naturally modelled as an *action*: the action of attending the course. Each *attend course* action has prerequisites to its execution and some effects, possibly conditioned to further requirements (Baldoni, Baroglio & Patti, 2001). Prerequisites and effects are described in terms of knowledge elements, called *competences*, whose vocabulary and relationships defines an ontology that is part of the knowledge model.

Study plan construction can, actually, be interpreted as a special case of *curriculum*, or page, *sequencing*, a well-known adaptive method in the field of Adaptive Educational Hypermedia and Intelligent Tutoring Systems (Weber & Brusilovsky, 2001; Henze & Nejdl, 2001; Stern & Woolf 1998, Baldoni, Baroglio, Henze & Patti, 2002), for suggesting personalized learning or study paths through a hyperspace of information sources. Study plan construction could, in fact, be described as a curriculum sequencing task where the atomic units of information that compose the hyperspace are *course descriptions*. Our claim is that by using formal reasoning techniques it is possible to improve the quality of the interaction, because such methods allow not only to construct personalized sequences but also to "justify" the proposed solutions and, as we will see, to supply other precious feedbacks to the users. In the next sections we will start by introducing the system architecture and, then, we will shortly describe the domain knowledge for the developed application together with the reasoning mechanisms

used to perform the adaptive tutoring services. Conclusions, comparisons with the literature and an outline of the future works end the paper.

2 The WLog system

The prototype system that we developed, *WLog*, has a multi-agent architecture. Agent technology allows complex systems to be easily assembled by creating distributed artifacts, that accomplish their tasks through cooperation and interaction. Such systems are modular and, therefore, flexible and scalable. Each module can, in fact, be developed by exploiting the best, specific technology for solving a given issue; moreover, new components can be added incrementally for supporting new services.

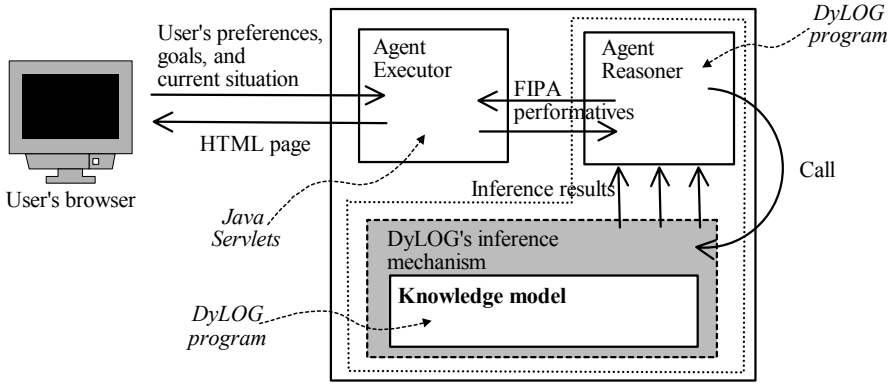

Figure 1: The system architecture

WLog consists mainly of two kinds of agents: *reasoners* and *executors*. *Reasoners* are written in DyLOG, an agent programming language based on a logical theory for reasoning about action and change in a modal-logic setting (Baldoni, Giordano, Martelli & Patti, 2001), executors are Java servlets embedded in a Tomcat web server. Executors are the interface between the rational agents and the users; they mainly produce HTML pages by following the directives sent by reasoners, and they forward the collected data to the reasoners themselves. Reasoners collect inputs from the users (preferences, goals, information about the current educational situation) and invoke the inference mechanism of the DyLOG language on the domain knowldege model in order to accomplish one of the possible reasoning tasks, i.e. building a study plan or validating a student-given study plan. Also the knowledge model is defined in the DyLOG language. While the use of DyLOG for representing the knowledge model and performing inferences bears advantages that will be introduced in the next sections, reasoners main loop is written in DyLOG for implementation convenience and it could be developed also in other programming languages. The communication among the agents has the form of a FIPA-like message exchange in a distributed system (FIPA97). Each agent is identified by its location, which can be obtained by other agents from a facilitator, and has a private mailbox where it receives messages from other agents. Users access the system by means of a normal web browser; until the end of the interaction, an executor will play the part of the interface between the user and a dedicated reasoner, whose location was obtained by consulting the facilitator. Supposing that a free reasoner is available, the interaction starts with the declaration of the user's goal (e.g. “I want to become an expert of web applications”). The user's goal is adopted by the reasoner, that will start a conversation aimed at collecting information about the user initial situation. For instance, the user will be asked about successfully passed exams. In the case of study plan validation, instead, the system will ask both the study plan to validate and the competences that the student wishes to acquire.

3 “Reasoning about actions” techniques in the tutoring domain

The system *domain knowledge* contains an *ontology* and the descriptions of all the available *courses*, given in terms of ontology elements; due to the particular application domain, ontology elements are

called *competences*. Competences can be either atomic or composed by smaller pieces of competence. They are used for defining the preconditions and the effects of courses and of the users' learning goals. Notice that a same competence may be acquired in different ways: it could be directly supplied by more than one course, or it could be obtained by learning the smaller pieces of competence it is made of by attending different courses. The decoupling of courses and competences is particularly useful in the perspective of building open systems, in which courses can be added or removed along time.
As mentioned in the introduction, each course X is interpreted as an "attend course X" action, which can be executed by a reasoner. During a study plan construction or validation task, action execution takes place in the reasoner's mental state: the agent works on hypothetical scenarios for satisfying the user's requests. Therefore, on a hand, action preconditions are evaluated against a set of competences, that the system supposes the user to have at a certain point of his/her career; on the other hand, action application increases the set of supposed competences of a student. It is a sort of projection of what will happen in the future. We can, then, think to the whole reasoning process as a sequence of *transitions*, produced by action execution, between *states*, each state being a set of fluents (properties whose truth value may change over the time). States can intuitively be seen as snapshots of the agent's mind. They contain the (possibly incomplete or uncertain) *information* that an agent has at a certain moment and, in particular, its beliefs about the user. More specifically, in the application at issue, states contain information about: (1) the set of courses that the student should have attended (either in a hypothetical scenario or, in the case of already attended courses, in reality), (2) the competences that the student should have acquired, (3) the learning goal of the student.
Returning to courses, in the DyLOG terminology they are defined by means of *atomic* actions. Atomic actions are used also for retrieving inputs from the world (sensing actions) and for interacting with the user (suggesting actions). Besides atomic actions, DyLOG allows one to define *complex* actions (or *procedures*). Procedures are defined in terms of other actions and can be recursive. In the application at issue, they are used for describing *curriculum schema*, in which specific professional expertise is described in terms of high-level competences.
With regard to the reasoning mechanisms, basically we exploit three different reasoning techniques, that have been deeply studied in Artificial Intelligence: *procedural planning*, *temporal projection*, and *temporal explanation*, that we are now going to introduce.

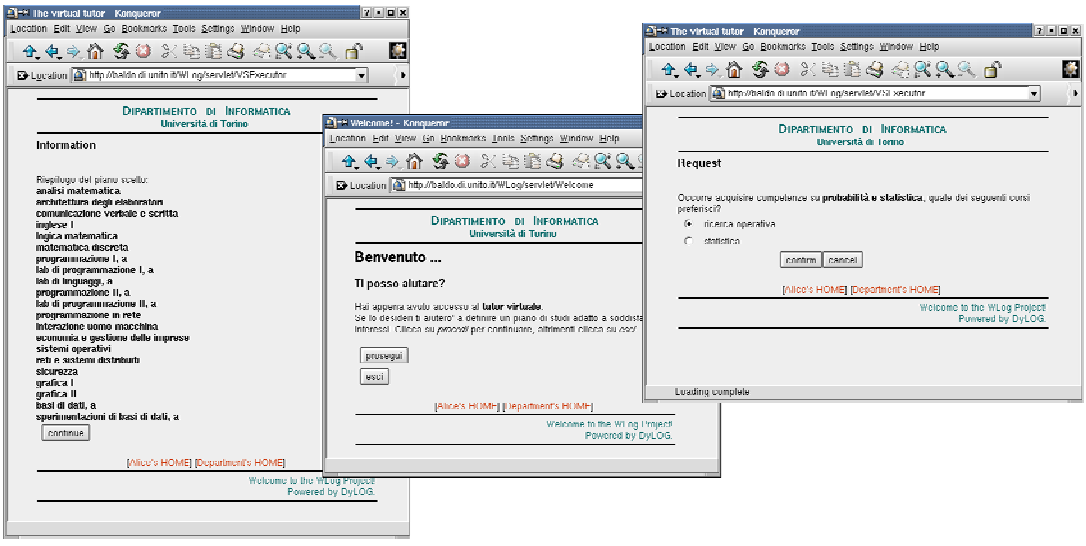

Figure 2: Interacting with WLog for building a study plan.

3.1 Study plan construction

Procedural planning allows our agents to build study plans that accomplish the user's learning goal. This reasoning mechanism consists in searching for those terminating executions of a procedure, after which a given condition holds. In our case, procedures define the guidelines for acquiring some professional expertise, the final condition is the user's learning goal, and the initial state is the student's situation in terms of passed exams. For instance, let us suppose that the student poses the question "I want to follow a 'web applications' curriculum [*professional expertise*] but I am also interested in video-games [*competence*]. Is there some study plan in the web applications curriculum that suits my

interests?". In this case the reasoner looks for those terminating execution sequences of the 'web applications' procedure that supply notions about video-games. Each of the execution traces that it finds are linear plans for achieving the learning goal. When a same competence is supplied by many courses, that can be successfully inserted in a study plan for achieving the learning goal of interest, the system offers the various alternatives to the user, leaving the choice up to him/her; in more technical words the system produces *conditional plans*, where each branching point is interpreted as an point of interaction with the user. Therefore, a procedure encodes the competences that a particular professional figure should have; the planning process uses the procedure, the ontology and the course descriptions for building course sequences that fit the user. Afterwards, the agent proposes the conditional plan to the user, interacting with him/her at each branching point, in order to select one of the alternative sequences. Observe that all of them are guaranteed to satisfy the learning goals due to the properties of the reasoning mechanism. The advantage of exploiting a procedure for constraining the search of a plan stands in the higher efficiency (w.r.t. an unbiased planner) of the whole process. As a last observation, the learning goal could also contain further conditions, such as a study plan maximum length.

3.2 Study plan validation

Temporal projection is a reasoning mechanism that checks if a given condition holds after the execution of a sequence of atomic actions. In our case, the sequence of atomic actions is a student-given study plan and the condition is a set competences he/she is interested in. Temporal projection returns a boolean answer (yes/no) but in order to support a student, a more informative feedback, that explains what went wrong, is necessary. To this aim, *temporal explanation* is applied.

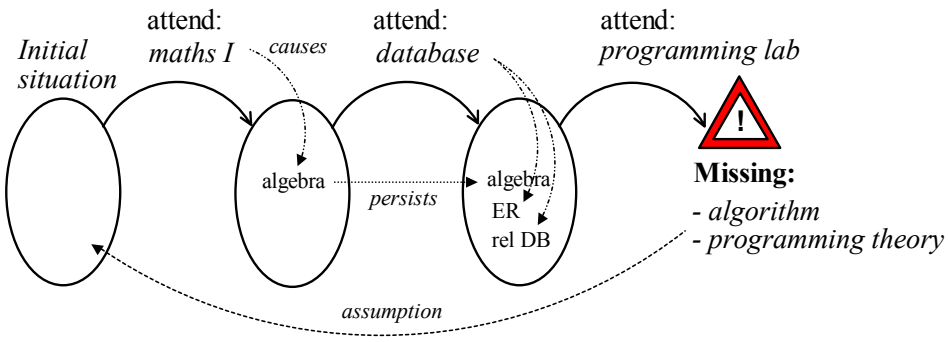

Figure 3: An example of study plan validation with explanation of failure.

Let us explain this mechanism with the example reported in Figure 3. Consider a student who produced a study plan that is a sequence of three courses: maths I, database, and programming lab. The student would like to acquire a final expertise about event programming, which is a consequence of attending the programming lab course. Let us suppose that the database course can be attended if the student has competence about algebra but that programming lab requires as preconditions algebra, programming theory and algorithms. The validation of this plan fails; in fact, although by attending maths I the student acquires competence about algebra, and therefore he could attend the database course, the programming lab course cannot be attended because the student does not have competence about algorithms and programming theory. Our system infers that the missing competences are the cause of failure by means of an *abductive* process that works on course preconditions and effects and completes the initial state with all the assumptions necessary to make the actions applicable. Intuitively this means that the plan is correct (i.e. it allows to achieve the learning goal) only if the student, before attending it, already has all the missing competences.
Notice that this approach works because educational applications of the kind that we tackle have a particular nature: they are *monotonic*, in the sense that each attend-course action increases the number of expected competences of a student.

4 Conclusion and Related Work

In this paper we presented an overview of the most recent enhancement of a work in which we applied reasoning techniques, that have been developed by the Artificial Intelligence community, to the development of adaptive services in a tutoring system. The tasks that we tackled are not aimed at producing course presentations that are tailored to particular characteristics of the user, as done for instance in PEGASUS (Macías & Castells 2001), but they are aimed at producing (validating) navigation paths in a hyperspace of knowledge items; such paths are adapted to the particular user. Adaptation is the result of formal reasoning on both a domain knowledge and knowledge about the student. The naturality of interpreting courses as actions makes the adoption of reasoning about actions mechanisms straightforward. Moreover, the kind of representation that we use makes the maintanance of a dynamical course corpus easy; courses can be added/removed dynamically without changing the inference system, which automatically will take them into account.
We proved the usefulness of three reasoning techniques in a curriculum sequencing applicative framework, finding some advantages w.r.t. other (e.g. stochastic-based) techniques used in the field: the possibility of performing multiple kinds of reasoning (not only sequencing) in a unified representative framework, the possibility of returning informative feedbacks, the ease of knowledge representation due to the symbolic, declarative approach. Moreover, differently than other approaches, such as ELM-ART (Weber et al. 2001) and MetaLinks (Murray, 2001), our reasoning approach allows to build multi-step trails for achieving the user's learning goal and not only to suggest the next best step.
We believe that this same approach could be useful in other educational applications; in particular, we are currently developing an adaptive system where a student practices with a tool in a learning-by-doing framework, properly assisted by an artificial tutor, that proposes exercises and silently monitors the student's actions until either he/she makes some mistake or succeeds in the goal accomplishment. The "reasoning about actions" paradigm seems to be very promising also for this kind of applications.

References

Baldoni, M., Baroglio, C., Henze, N., & Patti, V. (2002) . Setting up a framework for comparing adaptive educational hypermedia: First steps and application on curriculum sequencing. In *Proc. of the ABIS-Workshop 2002, Personalization for the Mobile World, Workshop on Adaptivity and User Modelling in Interactive Software Systems*, pp. 43-50, Hannover, Germany.

Baldoni, M., Baroglio, C., & Patti, V. (2001). Structereless, Intention-guided Web Sites: Planning-Based Adaptation. In *Proc. 1st Int. Conf. on Universal Access in Human-Computer Interaction*, a track of HCI International 2001, vol. 3, pp. 237-241, New Orleans, LA, USA. Lawrence Erlbaum Associates.

Baldoni, M., Giordano, L., Martelli, A., & Patti, V. (2001) Reasoning about complex actions with incomplete knowledge: a modal approach. In *Proc. of the 7th Italian Conference on Theoretical Computer Science, (ICTCS01)*, volume 2202 of LNCS, pp. 405-425, Torino, Italy.

Brusilovsky, P. (2000). Course sequencing for static courses? Applying ITS techniques in large-scale web-based education. In *Proc. of the 5th Int. Conf. on Intelligent Tutoring Systems, (ITS 2000)*, Montreal, Canada.

Henze, N., & Nejdl, W., (2001). Adaptation in open corpus hypermedia. *In IJAIED, Special Issue on Adaptive and Intelligent Web-Based Systems*, 12(4), pp. 325-350.

Macías, J. A., & Castells, P., (2001). Adaptive Hypermedia Presentation Modeling for Domain Ontologies. In *Proc. of 10th International Conference on Human-Computer Interaction (HCII'2001*), New Orleans (Louisiana), pp. 710-714.

Murray, T., (2001). MetaLinks: Authoring and Affordances for Conceptual and Narrative Flow in Adaptive Hyperbooks, In *International Journal of Artificial Intelligence in Education*, 2001.

Stern, M. K., & Woolf, B. P., (1998). *Curriculum Sequencing in a Web-Based Tutor*. In *Proc. of Intelligent Tutoring Systems*, LNCS 1452, 1998. Springer.

Weber, G. & Brusilovsky, P., (2001). ELM-ART: An Adaptive Versatile System for Web-based Instruction. In *IJAIED, Special Issue on Adaptive and Intelligent Web-based Systems*, 12(4), pp. 351-384.