Automatic Generation of User Adapted Learning Designs: An AI-Planning Proposal

Lluvia Morales, Luis Castillo, Juan Fernandez-Olivares, and Arturo Gonzalez-Ferrer

Univerisity of Granada, Spain
lluviamorales@ugr.es, {L.Castillo,faro}@decsai.ugr.es, arturogf@ugr.es

Abstract. A Learning Design(LD) definition under the IMS-LD standard is a complex task for the instructor because it requires a lot of time, effort and previous knowledge of the students group over which will be defined the knowledge objectives. That is why, taking advantage from diffusion of learning objects(LO) labeling using IMS-MD standard, we have proposed to realize a knowledge engineering process, represented as an algorithm, over LO labels and user profiles to automaticaly define a domain that will be used by an intelligent planner to build a LD. This LD will be finally implemented in the ILIAS Learning Management System(LMS).

Keywords: Planning and Scheduling, e-learning, IMS standars, Automatic Generation of Planning Domains.

1 Introduction and Previous Work

Since the appearance in 2003 of the IMS-LD v.1 endorsed by IMS Global Consortium[6], lot of educators have tried to implement it within on-line LMS's they use. However, this implementation is not an easy work because LO's have to be completely labeled[1] and it is necessary to detail the process to use them in order to achieve each student objectives, that is, specify a LD. For this reason, researchers have actually being looking for techniques to facilitate and even skip the LD construction step that commonly is assigned to the tutor.

To date reaserchers have been working with ontologies and knowledge databases[7] to do the knowledge extraction process over LO's, as it is a semi-automatic procedure to obtain the knowledge since the beginning of the course in order to create a LD. Also, other researchers have attempted to obtain information about the user through the course and, taking advantage of his interaction with the Intelligent Tutoring System(ITS), have tried to design several plans by using an intelligent planner[2] but, this process constantly showing to the student different learning routes to follow. Approaches such as these or similar had begun to be proposed since 1986[11] who is actually investigating about the advance we are addresing in this paper, working with IMS standards and its integration in LMS's[9].

2 The Adaptive LD Construction Problem

In IMS-LD standard three representation levels of a LD are described. *Level B* works over the definition of personalized learning units according to different pedagogies. This take into account the reusability of LO's, the previous knowledge of each student and his preferences.

The job of exhaustively analyze the student characteristics and, after that, to define the better personalized learning unit for each one was initialy assigned to tutors and, after the integration of intelligent planning in ITS's, to planning experts. But, in order to automate this process and to save time and costs, we have proposed a knowledge engeenering algorithm explained in next section besides its required environment and practical application.

3 LD Automatic Construction Using AI Planning

3.1 Required Information

In order to be able to start with the LD generation process is essential to extract mandatory information about LO's of the subject using its metadatas(MD) and the user models of each one of the students registered.

From LO's we can extract two kinds of metadatas.

Hierarchy Relations Metadatas. `Is-Part-Of`. It describes a hierarchical compositional structure between LO's through the course as is shown in figure 1. `Is-Based-On`, provides ordering relations between primitive objects(PO) or compound objects(CO). `Requires`. Reports content dependencies between CO's.

Objects Attributes Metadatas. `Language`. Object required language i.e. spanish, english, etc. `Learning Resource Type`. Describes what kind of learning resource we are working on, i.e. lecture, simulation, exercise, etc. `Other Platform Requirements`. Describes if there are special hardware or software requirements to use the LO. `Difficulty`. Defines the performance level required by the student for this object to be in his plan.

In order to personalize the LD, our algorithm use the next *student profile* options from our LMS: `English Level`. To determine if he could take a high

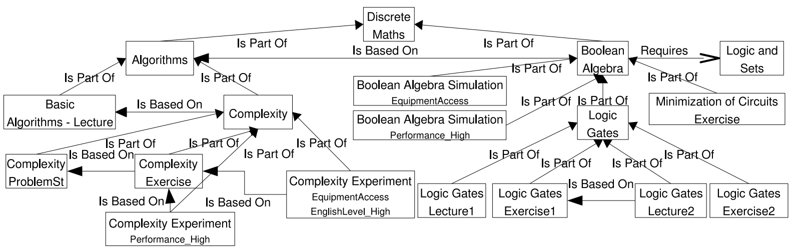

Fig. 1. Hierarchy Relations in a Subject

level english object. `Equipment`. Defines software and hardware availability of the student like java environment, bandwidth, etc. `Previous Courses Level`. Score in a related course. `Performance Level`. Performance level of the student. `Learning Style`. To offer each user a set of LO's with a temporal sequence that best fits his/her learning style. This style is defined by a psicological test, in this case the Honey-Alonso[4], that is answered by the student in his/her first visit to the LMS.

3.2 The Automatic Construction Process

In this section we show a brief description of the basic elements and steps followed by our algorithm to construct the planner domain and problem in the Planning Domain Definition Language[8], but a description of the basis of this documents is briefly explained in the next pharagraph, first.

LPG-td planner is a state-based intelligent planner based on local search and planning graphs methods that handles PDDL2.2 domains[3]; this is the planner that help us to create the LD. To date, we have to consider the following assumptions to define its problem and domain definition: *First*, the initial state of the planning problem is based on the contextual information extracted from LMS databases like user profiles and academic history. *Second*, the goal of the planning problem is translated from the learning objectives of a given course, in this case, the last LO's needed to complete the subject. *Third*, the set of the available actions in the domain is built from the LO's repository, so that every primitive object is translated into an action whose preconditions and effects are inherited from the information expressed in its MD.

The Domain Generation Algorithm. The planning domain generation algorithm is responsible of specifying required preconditions for every LO. These LO's are defined and labeled by the instructor in order to be used by the student in the better way according to the LO's attributes and his/her `Learning Style`.

The algorithm first analyze attributes and relations from each object; attributes give us state preconditions of PO's (actions in PDDL). Secondly, the algorithm defines order preconditions for PO's according to the requirements given by hierarchy relations. This second step is really arduous because it implies to check a subject graph from PO's, forming groups linearly arranged (even in a parallel way because of objects with the same name, but different attributes as *Complexity Experiment* in 1). These groups(called primitive groups) are ordered according to the `Is-Based-On` relation or the `Learning Style`, and subsequently we rise to each one of the hierarchical levels occupied by CO's(tasks in PDDL) and inheriting its relations to the first PO of the primitive group previously formed that is part of this CO. With this inheritance relations we can reorder those groups and form new ones.

The process described in the previous paragraph is done till cover every order relation from CO's which are part of a root compound object(*Discrete Maths* in figure 1 is and example) and finally we have to connect this root objects according to `Requires` relations.

Jhon	
LSTheoretical Performance_Low EquipmentAccess EnglishLevel_High PreviuosLogicAndSets	Jhon is a deductive student because his theoretical learning resource type. He has a low performance, but whit a high english level, the Logic and Sets subject finished, and whit the necessary equipment available.
Chris	
LSTheoretical Performance_High EquipmentAccess_No EnglishLevel_High PreviousLogicAndSets_No	Chris is a deductive student whit a high performance and english level, but he has not finished the Logic and Sets subject and he has not the necessary equipment available.

(a)

Chris	*Learning – Primitive Object*	*Requirements*
1	Basic Algoritms - Lecture	
2	Complexity - ProblemSt	
3	Complexity - Exercise	
4	Complexity - Experiment	Performance
5	LS - Lecture	
6	LS - Experiment	
7	LS - Exercise	
8	LS - ProblemSt	
9	Boolean Algebra - Simulation	Performance
10	Minimization Of Circuits - Exercise	
11	Logic Gates - Exercise2	
12	Logic Gates - Lecture1	
13	Logic Gates Exercise1	
14	Logic Gates - Lecture2	

Jhon	*Learning – Primitive Object*	*Requirements*
1	Basic Algoritms - Lecture	
2	Complexity - ProblemSt	
3	Complexity - Exercise	
4	Complexity - Experiment	Equipment, english
5	Boolean Algebra - Simulation	Equipment
6	Minimization Of Circuits - Exercise	
7	Logic Gates - Exercise2	
8	Logic Gates - Lecture1	
9	Logic Gates Exercise1	
10	Logic Gates - Lecture2	

(b)

Fig. 2. Two Studied Students - (a) Problem Definition (b) Obtained LD's

Using this process is possible to generate a planning domain in PDDL with the same expressive capacity than the IMS-MD repository.

Problem Generation. The problem file is extracted in an automated manner from each student profile mentioned in section 3.1, which are translated to predicates that the domain will be able to evaluate as in Figure 2-(a) where two expamles are showed.

LD Generation by LPG. Once domain and problem PDDL files are generated, the LPG planner creates the user profile adaptive LD's to the subject we are making the domain as in Figure 2-(b) examples of automatically generated LD's to the student models described in (a). Those LD's are adapted to each student, i.e. Chris has a `theoretical` LD, his LO's can be difficult resources, but can not take objects with `equipment` needs and he `requires` to take Logic and Sets PO's(Lecture, ProblemSt, Exercise and Experiment); while Jhon has a `theoretical` ordered plan too, but he needs easier LO's in his LD because of his low `performance`.

LD Integration in ILIAS. The procedure described along this paper has been fully integrated in the ILIAS LMS, which embeds a SOAP (Simple Object Access Protocol) server which recives the LD plan generated by LPG. ILIAS does not support IMS-LD specification yet, but we have translated the plan into a follow up guideline that appears over the student' ILIAS desktop.

4 Concluding Remarks and Future Work

It is important to point out three merits of this paper. First, that the labeling extraction from the IMS-MD can be carried out independently of the course over which our LD(in its abstraction level B) is going to be created. Second, that the knowledge extraction process is carried out by an algorithm (designed by our research team) that does not require the participation of any planning expert or tutor. However, the final product after this process, the LD, must be supervised by the instructor to make the considered modifications, in case it is needed. Finally, that this advance has been implemented on an IMS.

Althought, we thought that should be designed a domain representation that works out the common characteristics of a user group to be able to generate a collaborative LD. And, we must to identify the way to work with different kinds of LO's (i.e. optionals) and to represent global and partial deadlines that can be managed by a state-based planner.

References

1. IEEE Standard for Learning Objects Metadata, `http://ltsc.ieee.org/wg12/`
2. Camacho, D., R-Moreno, M.D., Obieta, U.: CAMOU: A Simple Integrated eLearning and Planning Techniques Tool. In: 4th International Workshop on Constraints and Language Processing (August 2007)
3. Gerevini, A., Serina, I.: LPG: A Planner Based on Local Search for Planning Graphs. In: Proceedings of AIPS 2002. AAAI Press, Toulouse (2002)
4. Honey Alonso Learning Style, `http://www.estilosdeaprendizaje.es/chaea/chaea.htm`
5. ILIAS Learning Management System, `http://www.ilias.de/ios/index-e.html`
6. IMS-GLC IMS Global consortium, `http://www.imsglobal.org/`
7. Kontopoulos, E., Vrakas, D., et al.: An Ontology-based Planning System for e-course Generation. In: Expert Systems with Applications, vol. 37 (1). Elsevier, Amsterdam
8. Long, D., Fox, M.: PDDL 2.1: An Extension to PDDL for Expressing Temporal Planning Domains. Journal of Artificial Intelligence Research 20, 24–61 (2003)
9. Mohan, P., Greer, J., McCalla, G.: Instructional Planning with Learning Objects. In: Baumgartner, K.M., Cairns (eds.) IJCAI 2003 Workshop Knowledge Representation and Automated Reasoning for E-Learning Systems (2003)
10. Myers, K.L.: Towards a Framework for Continuous Planning and Execution. In: Proceedings of the AAAI Fall Symposium on Distributed Continual Planning (1998)
11. Peachey, D.R., McCalla, G.I.: Using Planning Techniques in Intelligent Tutoring Systems. International Journal of Man-Machine Studies 24, 77–98 (1986)