Generating a Course Domain for Several Planner Types: a First Approach

Lluvia Morales

Univerisity of Granada
llmoral@correo.ugr.es

Abstract. This paper describes the importance of automatic generation of domains in planning and describes the feasibility to produce them with the generation of a course domain for the HTN planner SIADEX, that is just one of the planners which we try to produce an usable and coherent planner domain.

1 Introduction

A planning domain is the core of any planning application that is usually written in the Planning Domain Description Language (PDDL)[10] and with a different structure depending to the planner type used.

Actually, the knowledge engineering process to acquire a complete and functional planning domain, is made by a planning expert who takes lot of time to be able to deeply learn the problem domain and takes a domain expert role. This time imply big costs for the enterprise who requires an intelligent solution for their planning problem. Those reasons have been the obstacle for the practical application of planning techniques in real life.

How can we simplify the work of domain experts in order to construct a planner domain?

This is the question that motivate our investigation and in next sections we show how to solve and prove that is possible not to use human work to acquire a planning domain for an specific real-world domain and its particular application cases. First, we will describe the problem we are working on, the real world domain, and then we will explain our actually designed algorithm to generate a planning domain used by the Hierarchical Task Network(HTN)[6] planner SIADEX[1]. We will finish giving our conclusions and future work related to our thesis objectives.

2 The real-world problem domain

The real world problem domain we are working on deals with the necessity to automatically obtain a learning design[7] (a personalized educational plan) for each learner who is registered in a specific course.

In this case we have created a course over the ILIAS web-based management system platform[9] which allows us to completely label every learning object(which is called primitive action or compound task in a planner environment depending of its hierarchy level) using the IMS-MD standard or, at least, using following metadatas to label la objects in order to take advantage of them for the construction of a planner domain.

Object metadatas which are described next are classified according to:
Hierarchy relations: (1)*Is-Part-Of*. It describes a hierarchical structure between learning objects through the course. (2)*Is-Based-On*, provide ordering relations between primitive actions. (3)*Requires*. Report content dependencies.
Object Attributes: (1)*Coverage*. If this metadatum takes the 'OPTIONAL' value it means that the learning object could be omitted in the plan if necessary (if there the student have no time or if the object is an extra-exercise). If it takes a 'NULL' value then it is mandatory. (2)*Language*. Defines if the object language will be spanish, english or another. (3)*Typical Learning Time*. Temporal constraints imposed by instructor. (4)*Learning Resource Type*. Is an educational metadatum that we use to give an order at primitive actions. (5)*Other Platform Requirements*. Describe if there are multimedia special requirements to use the learning object. (6)*Difficulty*. Metadatum that defines the level that is required by the student for this object to be in his plan.

The student must answer several questions in order to obtain following mandatory metadatas too:
Student Metadatas(1)*English Level*. To determine if he could take an high level english object. (2)*Multimedia*. If the student has multimedia or not, and to use or not objects with multimedia requirements. (3)*Previous Courses Level*. Note in a related course. According to this metadatum a required task could be taken or not. (4)*Honey-Alonso Style*. It help us to order actions to the student learning style defined by this psicological test.

3 A course planning domain for HTN-SIADEX

Previously studied metadatas are now used in an algorithm process in order to encode them into the required planning domain in PDDL for SIADEX.

Firstly we obtain the hierarchy set of learning objects which are part of the course using is-part-of and required metadatas in order to create a tree as we can see in the algorithm piece showed in Figure 1.

After that we analyze this primitive actions (objects of which no other objects are dependent of) in order to obtain new necessary tasks depending on its particular property metadatas coverage, difficulty and name as we can see in Primitive Actions processing and new task generation part of the algorithm in Figure 2, rearranging by this way the hierarchy between the objects tree.

Next, we examine order relations given by is-part-of metadatum or special properties like the HONEY-ALONSO style between tasks for defining last level tasks methods, given by primitive actions, and the ordered sequence of them as is shown in the tasks processing part of the algorithm in Figure 2.

O: objects set
LG: learning objectives (defined by teacher)
is-part-of(o): objects set which are parents from o.
requires(o): objects set which are dependents from o.
$A = LG$
While A is changing:
  Select A not explored yet
  $Next(a) = \{o \in O | a \in is - part - of(o) \vee o \in requires(a)\}$
  $A = A \cup Next(a)$

Fig. 1. Tree extraction from the hierarchy of learning objects in a course

Primitive Actions Processing and New Tasks Generation
With $P_A = \{a \in A | \neg \exists a' \in A, a \in is - part - of(a')\}$
While $\exists a \in P_A$ not processed :
  Select $a \in P_A$ and mark it how processed
  $E_a = \{a' \in P_A | name(a) = name(a')\}$
  If $E_a \neq \emptyset$ then:
    Create a task TA with:
      $name(TA) =$ "$Multiple$" + $name(a)$
      $learning - resource - type(TA) = learning - resource - type(a)$
      This task will not have preconditions and only one method
    Copy every relation of $a' \in E_a \cap a$ to TA
    TA must be is-part-of(a') where $a' \in E_a \cap a$
    Mark every $a' \in E_a$ to TA how processed
  If $coverage(a) =$ "$optional$" then
    Create a task TA with:
      $name(TA) =$ "$Optional$" + $name(a)$
      $learning - resource - type(TA) = learning - resource - type(a)$
      This task will not have preconditions and two methods, yes or not optional.
    Copy relations from a to TA
    TA must be is-part-of(a)
  If $difficulty(a) \in \{$"$high$","$very - high$"$\}$ then
    Create a task TA with:
      $name(TA) =$ "$Difficult$" + $name(a)$
      $learning - resource - type(TA) = learning - resource - type(a)$
      This task will have the *If* $difficulty(a) \in \{$"$high$","$very - high$"$\}$ *then* precondition and two methods, yes or not can use its associated primitive.
    Copy relations from a to TA
    TA must be is-part-of(a)

Tasks Processing
With $T_A = \{a \in A | \exists a' \in A, a \in is - part - of(a')\}$
While $\exists a \in T_A$ not processed
  Select next $a \in T_A$ which are not processed
  $Succ(a) = a' \in A | a \in is - part - of(a')$
  If $Succ(a)$ is totally ordered according to is-based-on relation that is to say, $a_1 \in is - based - on(a_2), a_2 \in is - based - on(a_3), ..., a_{n-1} \in is - based - on(a_n)$
    Y*ou must order the actions in a list according to this order*
  Else
    You must order the actions according to two honey-alonso style and their learning-resource-type
    If $honey - alonso - style = pragmatic$ then
      Objects must follow the next order in a methods list according to its learning-resource-type: simulation objects, experiment objects, problem statement objects, exercise objects, and lecture objects.
    If $honey - alonso - style = theoretical$ then
      Objects must follow the next order in the methods list according to its learning-resource-type: problem statement objects, simulation objects, experiment objects, exercise objects, and lecture objects.
  If $requires(a) \neq \emptyset$
    $\forall t \in requires(a)$ create a new task TD_t with:
      If student did pass t (according to his level on it) precondition.
      Two methods pass or not-pass.
      $name(TD_t) =$ "$Dependency$" + $name(a)$
      Put new TD_t in the first place of the list.

Fig. 2. Tree rearrangement and task generation

This algorithm is used to generate a planning domain that SIADEX uses to obtain plans called IMS-Learning Designs adapted to the features and needs of every student form the course analyzed [1].

4 Concluding remarks and future work

Has may be seen at previous sections we have been able to create an algorithm that generate a well structured planning domain for the HTN planner SIADEX.

This algorithm can be improved, but first we work on define an algorithm to generate a planning domain for other kind of planners [3][4] and then we are going to describe our results in each case proving that those algorithms generate an usable and adaptable domain according to different courses and students, using experimental results.

References

1. L.Castillo, L. Morales, A. Gonzalez-Ferrer, J. Fdez-Olivares and O. García-Pérez. Blind Review. *Knowledge Engineering and Planning for the Automated Synthesis of Costumized Learning Designs*, 2007
2. J. Fdez-Olivares, L. Castillo, O. Garcia-Perez and F. Palao. *Bringing Users and Planning Technology Together: Experiences in SIADEX* ICAPS 2006
3. A. Gerevini, A. Saetti, I. Sirina and P. Tonielli: *Planning in PDDL 2.2 domains with LPG-TD.* International Planning Competition booklet (ICAPS-4), 2004.
4. V. Vidal and H. Geffner. *CPT: an optimal temporal POCL planner based on constraint programming* International Planning Competition booklet (ICAPS), 2004
5. D. S. Nau, S. J. J. Smith, and K. Erol (1998). *Control strategies in HTN planning: Theory versus practice.* In AAAI-98/IAAI-98 Proceedings, pp.1127-1133.
6. Kutluhan Erol, Dana Nau, and James Hendler. *HTN planning: complexity and expressivity.* In Proc. AAAI-94, 1994. Seattle.
7. *IMS-GLC IMS Global consortium* http://www.imsglobal.org/.
8. ANSI-IEEE. *IEEE Standard for Learning Objects Metadata* http://ltsc.ieee.org/wg12/.
9. *ILIAS Learning Management System* http://www.ilias.de/ios/index-e.html
10. D. Long and M. Fox. *PDDL 2.1: An Extension to PDDL for Expressing Temporal Planning Domains.* Journal of Artificial Intelligence Research, 20:61-24, 2003