Multi-level Navigation for Curriculum Planning in Intelligent Tutoring Systems

Yaowei Liu
School of Information Technology
University of Queensland
Brisbane, Queensland 4072, Australia
Wei Lai
Dept. of Maths and Computing
University of Southern Queensland
Toowoomba, QLD 4350, Australia
Wong Kin Keong
School of Applied Science
Nanyang Technological University
Nanyang Ave., Singapore 639798

Abstract - This paper presents and investigates the application of multi-level planning techniques, together with meta and micro level knowledge architecture model, in building the domain knowledge, planning and navigate the curriculum of an intelligent tutoring system(ITS). Curriculum issues have been important to ITS. In order to truly individualize instruction, ITS must be able to reason about the curriculum, understand its implications, and be able to dynamically redesign the curriculum. This leads to the dramatic increase in the information an planner handles. The multi-level planning, made possible by the meta and micro level knowledge model, efficiently manages large domain knowledge base, in that it largely eases the planning task both in complexity and in storage.

1. INTRODUCTION

The research of Intelligent Tutoring System (ITS) has a rich history, especially in the seventies when ITSs were used as testbed and source of inspiration for new AI conceptions and techniques. One of its goal is the provision of individualized instruction of equivalent quality to that of a human tutor [1, 2]. This suggests that the system should possess many of the characteristics of an effective human tutor. Among these characteristics, the ability to make adequate curriculum is the central issue. A standard definition given by Halff [3] says: curriculum is the selection and sequencing of material for purposes of instruction. In actual classroom practice, the curriculum is the stellar object around which most teaching activities orbit. Teachers are encouraged to organize their classes according to a curriculum which guides their decision making throughout.

Curriculum in CAI courses is preprogrammed. We call this hardwired. For a student, the curriculum is a control path through the frames, with the particular sequence of frames depending on how he performs on each frame's test. For a practical system, explicitly predicating all control paths, evaluating all the student understanding at each frame is expensive, and often impossible. Thus, normally, only main control paths are wired, and many dimension of student understanding are ignored by the test. These facts means that the curriculum in CAI is expensive, inflexible, non-individualized, and rigid.

ITS systems is intended to provide new and effective ways of teaching and learning, and will help making them more individualized than is possible with CAI. It differs from CAI systems in that it has the knowledge about its subject domain, and it dynamically creates curriculum for an individual student and can modify it according to the change of environment.

A typical ITS system consists of a domain knowledge component, a student knowledge component, an instructional planning component (or planner), and a plan executor (Executor). See Fig 1

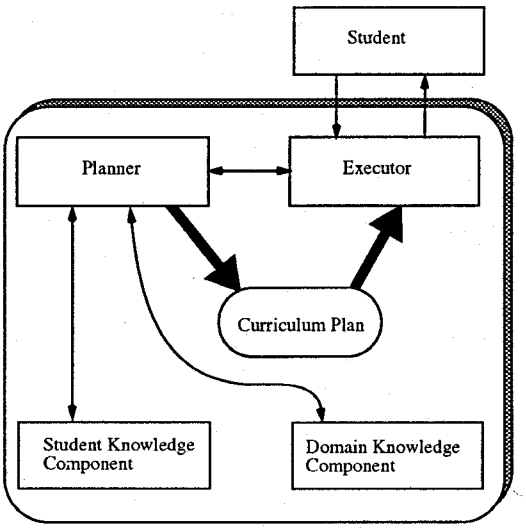

Fig. 1 Structure of an ITS

2. CURRICULUM IN EXISTING ITS

ITS system operates based on three types of knowledge: knowledge about the subject, knowledge about the student and knowledge about teaching. Knowledge about the subject is found primarily in the domain knowledge base component, which includes domain concepts and how they relate to each other, and can be used to generate teaching tasks, to evaluate the student's performance. Knowledge about the student is stored in the student model, which contains the system's belief on the student's state of knowledge and the student's attitudes: what he knows and what he does not know, also his

0-7803-4253-4/97/$10.00 © 1997 IEEE

strengths, weaknesses and learning preferences. Knowledge about teaching is both implicit in the design of the Planner and Executor, and explicit in the form of teaching operators.

The planner develops a curriculum, i.e. a teaching plan, tailored to the particular student being taught, based on the domain knowledge and student model. The executor teaches the student according to the curriculum. The generation of a planner is analogous to conducting a thorough search for good paths through a maze, and recording them in the form of a map. It is responsible for determining what to do next at each point in the instructional interaction, and help the student learn the domain knowledge in an instructionally sound manner. It uses knowledge about the leaner, about the domain and about the pedagogical process, to make informed pedagogical decisions.

Research reveals the role of a teacher in learning course [4, 5, 6]. The effective teacher has knowledge of subject matter, has a pedagogical view of the subject matter, has knowledge of student errors and misconceptions, has knowledge of plans and strategies that students use to solve problems, has insight into how a student's knowledge has evolved, is aware of learning theories and theories of knowing, has knowledge of instructional techniques and when to use them, has knowledge of how to interact with student and knows when to control the teacher-student interaction and when to relinquish control.

An ITS is charged with the same responsibility, though how to achieve this is a big challenge. Many researcher have worked on it for long time, in a variety of areas: the representation of the knowledge to student, the diagnosis and representation of how and what the student is achieving, etc. This paper focuses on the curriculum planning issue.

In existing Intelligent Tutoring Systems (ITS), the curriculum is induced from the knowledge base and student model [8]. The ITS deliver component (the real teaching operator) executes according the curriculum, and may invokes the planner to re-plan when the circumstance changes. All these are done on knowledge level: each knowledge unit in knowledge base is treated as the base unit to be planed and executed. But, this approach faces the problem in current trend. The objective of ITS research is a more adaptable, flexible, and individualized ITS, where the planner is supposed to navigate the ITS to suit each individual, to intelligently navigate the ITS around the failure of a teaching operation for more adaptability. To achieve this goal, more detailed domain knowledge, and a more individualized curriculum are required. To plan the curriculum at first, and to dynamically redesign the curriculum to meet the changing and individualised learning environment could become impractically costing.

This lead to finer granularity in both the domain knowledge base and the student model. Same as a human teacher, the more configuration of the domain knowledge, the better targeted curriculum could be developed for students; and better understanding about the student's knowledge, the easier the system to evaluate student's achievement and learning difficulty, then to guide him to achieve the learning aim. However, the finer granularity also cause the problem of an dramatically increased domain knowledge base, student model, the plan, and hence the costing in planning and replanning the curriculum.

3. MULTI-LEVEL CURRICULUM PLANNING

The multi-level Curriculum Navigation presents a new approach to accommodate adaptability, flexibility, and individualization in ITS. The aim of this approach is to eases the load in planning the curriculum in the large domain knowledge, and reduce the size of stored plan, meanwhile, maintain the effect of a comprehensive curriculum. When the learning environment changes forcing the current curriculum abort, this also reduce the work in redesign the curriculum.

The overall strategy is to plan from the top level, to get a coarse level curriculum, each teaching node representing a group of connected finer grained domain knowledge. To teach this node, a sub curriculum is generated, each teaching node among it representing a group finer grained domain knowledge. At the bottom level, each teaching node will be an actual operator: representing to deliver a domain knowledge unit to student. Any lower level curriculum is created only when the corresponding higher level knowledge node is to be presented to students, i.e. be executed. Accordingly, the domain knowledge is divided into fine grained unit[7]. These primary knowledge units are then grouped together to form a higher level knowledge unit. In this way, domain knowledge units are grouped until the top level. Group of lower level knowledge units are depend on their conceptual association and logical relation. For highly logical knowledge, this could be grouped at separate level in the same way the knowledge are classified in subjects, sub-subjects, etc. In practical use, the number of level depends on the size of the domain knowledge.

At each level of the domain knowledge (same in curriculum), a group of finer level knowledge nodes are grouped in one node, if there is no more than one link between the group and any other group.

Take the Prolog language knowledge base for an example. To see the forest, we'd better see a tree first. For four levels

from the top level, we have the domain knowledge as in Fig. 2.

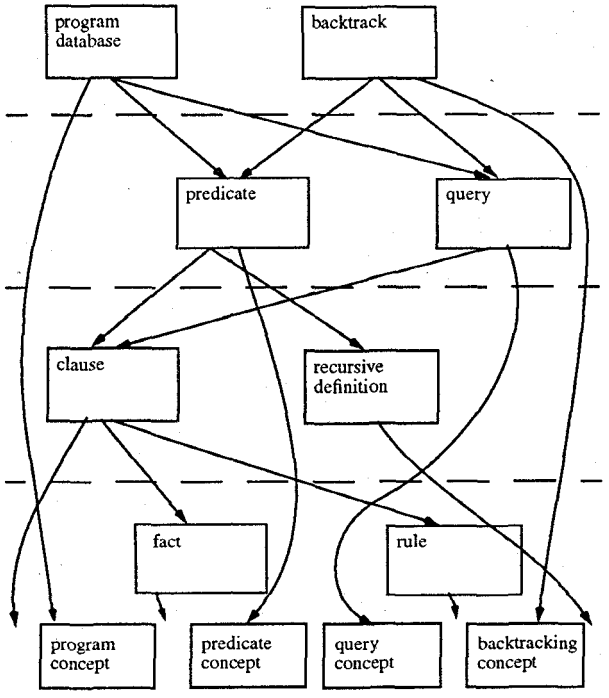

Fig. 2 Domain knowledge multi-level composition

We can see that at the knowledge program database is composed of the knowledge of predicate and query, a finer level knowledge, in addition to the general concept of program. The backtrack is composed of the knowledge of predicate and query, a finer level knowledge, in addition to the general concept of backtracking, though it is much complicated for teaching. Same situation applies to other levels of knowledge.

Fine grained knowledge for domain knowledge also bring other advantages to the whole ITS. Taking a patient going to see doctor for example. The doctor can see the symposium of the illness, but he cannot simply decide the real problem from that superficial symposium, as the problem may come from some internal organ. Same for the failure cause diagnosis in ITS. The failure of a teaching operation about an knowledge could be caused by any one of the following reasons:

- the student's failure in understanding this knowledge,
- the student's misunderstanding of its immediate prerequisite knowledge, or
- the student's the misunderstanding of its earlier prerequisite knowledge, or
- the misunderstanding of either a whole knowledge or a component of a knowledge.

In the last case, the analysis on the bases of knowledge may not be enough. That is why breaking down into the fine grained knowledge becomes a necessary for a intelligent failure detecting system. This is particularly useful for misconcept detection. In current available ITS systems, misconcept are treated separately, and stored in the student model as a independent concept.

The multi-level planning centres around instructional goals. That could be goals that the system has for the student, goals the student has for the system, goals the system has for itself. In the whole, the planning procedure goes through the procedure:

1) goal generation;

2) plan generation;

3) plan execution monitoring; if there is successful finish or failure, go back to 1)

In the first step, a goal is generated, either by the feedback from student, or the system monitoring.

In step 2, with the goal generated in step 1. If there is no plan existed for this goal, generate a new plan. First look at the top level, search what route should take to take achieve this goal (note, this goal may be on sub knowledge of a knowledge node), record this route. If this is the atom knowledge, stop and go to step three. Otherwise, pick the final knowledge with the lower level knowledge nodes of this node as the goal, and lower the current level to this level, start plan generation again.

In step 2, with the goal generated in step 1. If there is a plan existed for this goal, the this plan should be checked again if it is still achievable according to the current domain knowledge base and student model (if a student repeatedly could not achieve a goal, the path from the current know to this goal will be disconnected from the domain knowledge). If it could, go ahead to the next level. If it could not, modify the plan by search a new route from the most recently achieved knowledge.

In step 3, during the execution of the plan, the system monitors the student's learning process. If a knowledge unit achieved, it is marked in the student model and in plan. If a student fails to learn a knowledge, it try it a few times more before resort to other ways of teaching the same knowledge. Say from teaching theoretical first, example later, to teaching example first to theoretical later. If all methods failed, it will mark this in the knowledge, then go back to step one.

4. CONCLUSIONS

Multi-level Navigation is a consequence of individualised ITS. It heavily relies on the proper construction of domain knowledge base and student model, as well as the proper application of pedagogical knowledge in teaching assessment. Research into any of these issues may well suggest interesting results to knowledge engineering, student modelling. They are out the reach of this paper.

Have a further look at multi-level navigation, we draw several conclusions below:

- it could greatly improve the performance of ITS, by reducing the generation time of a plan;
- it also reduce the size of a current plan by always maintain only part of a detailed plan;
- it also reduce the re-plan time, as re-plan would normally result few re-plan action at the top level. As most of the detailed plan (lower level plan) would not be created unless they are needed, it only involves much fewer work than other planning method;
- it makes fine grained domain knowledge practical in use, that in turn, could bring remove many misconcept knowledge from knowledge base [7].

During our work a number of other issues arose, particularly with respect to definition and application of multi-level navigation.

The case study here considers the general simulation of apply the method to Prolog language teaching case. A more precisely specification of this method and ITS system in the whole with Z specification [9, 10] would give a better definition.

Granularity is an important consideration in student modelling an in pedagogy. Explicit representations of granularity are central to being able to handle the uncertainties inherent both in diagnosis and in pedagogical decision making. Multi-level navigation touches one side of the knowledge granularity. On the other side, organising knowledge at differently levels of granularity, and teach them forwards and backwards could enhances the teaching performance.

5. REFERENCES

[1] G. I. McCalla, The Search for Adaptability, Flexibility, and Individualization: Approaches to Curriculum in Intelligent Tutoring Systems, *NATO ASI Series*, Vol. F 85, Adaptive Learning Environments, 1985, pp. 91—121.

[2] Radbud Winkels, *Explorations in Intelligent Tutoring and Help*, IOS Press, 1992.

[3] H.M. Halff, Curriculum and instruction in automated tutors, In: *Foundations of intelligent tutoring systems* (M.C. Polson, & J.J. Richardson, eds.) Hillsdale, NJ: Lawrence Erlbaum Associates 1988

[4] P.L. Grossman, S.M. Wilson & L.S. Shulman, Teachers of substance: Subject matter knowledge for teaching, *In M.C. Reynolds (Ed) Knowledge Base for the Beginning Teacher*, Toronto: Pergamon Press, 1989, pp. 23—26.

[5] P.L. Peterson, E. Fennema, T.P. Carpenter and M. Loef, *Teachers' pedagogical content beliefs in mathematics, Cognition and Instruction*, 6(1), 1989, pp. 1—40.

[6] F. Elbaz, *Teacher thinking: A study of practical knowledge*, New York: Nichols, 1983.

[7] G. I. MaCalla and J. E. Creer, Granularity-Based Reasoning and Belief Revision in Student Models, Laboratory for Advanced Research in Intelligent Educational systems, *Research Report 9302.*

[8] D. R. Peachy and G. I. McCalla, Using Planning techniques in intelligent tutoring systems, *International Journal of Mam-Machine Studies*, Vol 24, No. 1-6, 1986, pp. 77—88.

[9] J.M. Spivey. *The Z Notation: a Reference Manual.* Prentice-Hall, New York, 1989.

[10] R. Duke, P. King, G. Ross and G. Smith, The Object-Z specification language, *Technical Report* TR-91-1, Software Research Centre, University, of Queensland, 1991.