On the Dynamics of Learner Models

A. Paiva [1] and **J. Self** [2] and **R. Hartley**[3]

Abstract.

In order for interactive learning environments to enrich and individualize the communication with learners, they use learner models, which can be seen as explicit representations in the system of some characteristics of that particular learner.

These explicit learner models are created during the interaction with the learner and based on the actions performed by him or her. The created content of the model will then influence the interaction to be established with that same learner. So, on the one hand, these models are created based on the knowledge interchange between the system and the learner; and, on the other hand, the knowledge interchange between the two may depend on the content of that model. This makes learner modeling a very dynamic process.

This dynamic feature of learner models is explored in this document by analyzing the mechanisms of change in learner models and its relation with the behavior of the learner. We define two types of changes: system and learner changes. Finally we describe some characteristics of a system that can cope with the described changes. This system uses a module that keeps the justifications of all the facts in the learner model, used to perform the defined changes.

1 Introduction

Learner models can be seen as an explicit representation in the system of some characteristics of a particular learner, which will enable the system to adapt the knowledge communication to that particular learner.

Although the work on learner models has had a considerable increase both in research and commercial terms, the dynamics of such models is still a problem to be tackled in this area. Although mentioned in several recent papers only a few have really addressed this problem. For instance, in [1] is said that:

"Keeping track of changes in the cognitive state of a student means in the first place to model skill and knowledge acquisition, i.e. changes of a relatively permanent and global nature."

The biggest problem in dealing with changes in learner models is that those changes are not monotonic in essence, because they rely on uncertain facts, and because learners also change their minds.

As Kono et al. stress in [6]:

[...] Student Modeling methods should be able to automatically manage the consistency of student answers in order to follow student's mind. However, there are very few attempts to formulate the nonmonotonicity of the student modeling process [11] [5].

But why is the treatment of nonmonotonicity in learner models so difficult? There are many reasons, such as:

- some facts in the learner model are inferred from the behavior of the learner, so they are not certain. If a contradiction occurs, they may or may not need to be removed or changed;
- some facts in the model were derived from other facts which may change. If that happens, the ones that depend on them may or may not need to be removed (propagation of change);
- learners change their minds during the knowledge interchange. The system has to detect when the learner changes his mind and represent the change, distinguishing it from a mere slip;
- learners may answer in an inconsistent way to the questions, and maintain that inconsistency. What should be maintained in the model in this case? How can we distinguish an inconsistency from a change of mind?

This document is organized as follows: a description of the changes in learner models will be made first; two types of changes will be distinguished and discussed. Finally a description of a system which copes with the changes described will be made.

2 Changes in the Learner Models

A learner modeling system is a system responsible for creating and maintaining the models of the learners. Normally, a learner modeling system is the component of an Intelligent Tutoring System, or of an Interactive Learning Environment, which is responsible for acquiring the model, making inferences based on the behavior of the agent, generate information for the model, diagnose the learner's mistakes, and keep the model consistent and adequate. As far as changes are concerned, a learner modeling system has to perform two types of changes in the models:

- System Changes - are changes that have to be performed because the acquisition process is uncertain and some of the acquired facts may be incorrect.
- Learner Changes - these changes have to be performed and explicitly represented to follow the process of changing of the learner.

The **system changes** (also treated by Kono et al [6] [7]) are modifications in the model caused by inconsistent reasoning of the system itself. For instance, if a learner says that "the Moon keeps the same face turned towards the Earth", the system may infer that the learner knows as well that "the period of revolution of the Moon is 27 days".

[1] Department of Computing, Lancaster University, Lancaster LA1 4YR, UK Telephone (0524) 65201, Fax (0524) 38170, email : amp@comp.lancs.ac.uk or amp@cbl.leeds.ac.uk, and INESC, IST - Technical University of Lisbon, Portugal.

[2] Department of Computing, Lancaster University, Lancaster LA1 4YR, UK Telephone (0524) 65201, Fax (0524) 38170, email : jas@comp.lancs.ac.uk

[3] Computer Based Learning Unit, University of Leeds, LS2 9JT Leeds, UK, Telephone (0532) 334626, Fax (0532) 334635

ECAI 94. *11th European Conference on Artificial Intelligence* Edited by A. Cohn
Published in 1994 by John Wiley & Sons, Ltd.

However, when that learner says that the period of revolution is 29 days, the system has to revise the contradiction created.

On the other hand, **learner changes** are changes in the learner behavior that need to be kept or even simulated in the model. For example the learner says that "the new Moon occurs when the Earth makes a shadow on the Moon", and after an intervention by the system and realising that "the lunar eclipse occurs when the Earth makes a shadow on the Moon", the learner revises his former belief.

In both cases, the changes (nonmonotonic ones) occur when a contradiction is detected in the model. If the contradiction is a system contradiction, then a system change has to be performed. If the contradiction is a learner contradiction, then the system has to act according to the contradiction, and its elimination will only be possible when the learner himself removes it. The actions of the system, when the learner has a contradiction, have to be driven towards the aim of eliminating that contradiction, and therefore changing the learner's behavior. To do so, the system itself has to be able to predict the changes that a learner may or may not make because of that contradiction. So, in Learner Modelling Systems, learner contradictions will be detected and learner changes will hopefully occur after that detection. System contradictions, on the other hand, will be detected and immediately resolved.

For instance, in the following dialogue [4] the learner believes that "a new Moon occurs when the Earth makes a shadow in the Moon", and he also believes that "a lunar eclipse occurs when the Earth makes a shadow in the Moon". When he realises that both the "lunar eclipse" and the "new Moon" have the same definition, he realises that he has a contradiction because he doesn't believe that "the new Moon and the lunar eclipse" are the same thing.

In order for the system to create a model that is effective in situations like this, the learner changes have to be known and predicted, so that the appropriate interventions can be made by the learning system during the interaction. In this case, if the system is able to predict that a change will occur if the question Q8-P2 is performed, it will be able to effectively promote conceptual change in the learners.

1. SkyLab(Q3)> The Moon keeps the same face turned towards the Earth?
2. P1> The same face? No.
3. P1> It's a rotating satellite, right? It rotates around its own axis, so it can't actually have the same face towards the Earth.
4. SkyLab(Q6)>The New Moon occurs when the Earth makes a shadow on the Moon?
5. P1> Well..
6. P2> The New Moon is when you see only a line, isn't it.
7. P1> Yes, that's right, that's right. Ah that's interesting.
8. P2> What causes a new Moon?
9. P1> That's a different phenomenon... But I don't know. Yes, you might be right, its YES. It's when it makes a shadow.
(later)...
10. SkyLab(Q8)> A Lunar eclipse occurs when the Earth makes a shadow on the Moon.
11. P1> Yes.
12. P2> So, what's the difference between the New Moon and the Lunar Eclipse?
13. P1> Ah, a Lunar Eclipse I understand it to be, is when the Earth comes between Moon and the Sun, totally. Whereas this one (the Q6)..is.. Basically, this (Q6) is wrong. The New Moon occurs when the Earth makes a shadow on the Moon: its incorrect. Now that we think about it.
14. P2> What is causing it....
15. P1> Because, the light is coming from one side and it is rotating.

Dialogue 1

In summary, for a system to be effective in creating models of the learners with the aim of promoting conceptual change, both types of changes have to be taken into account.

3 System Changes

The **System Changes** occur because the acquisition is not certain. If a learner uses the formula "f=ma" the system can infer that he knows it and that he also knows that the increase of force will increase the acceleration. If that is proven to be wrong, the system has to **change** the inferences drawn. In our theory we will propose the following system changes [5]:

Expansion A system expansion is an operation in the model of the learner where a new fact is added to the model. If the new fact is derived from other facts, the system will also keep the justification for the new fact. In the case of Dialogue 1 if the model of the learner is:

$$LM1 = \{B_l Satellite(Moon),\\ B_l(Satellite(X) \rightarrow \neg Planet(X)),\\ B_l \neg Planet(Moon)\}$$

when the learner P1 answers question 3 an expansion of the model is made with the following beliefs:

$$B_l(Rotating(Moon, Earth) \& Satellite(Moon)\\ \rightarrow \neg SameFace(Moon, Earth))$$
$$B_l(Rotating(Moon, Earth))$$
and $B_l \neg SameFace(Moon, Earth)$

When an expansion of the model is executed the "reasons" why that expansion occurred must also be kept so that the revision of such beliefs can be made later on.

Update When a fact has to be changed in the model without checking the consistency, it is said that the system performed an updating operation. The update is a simple change of one fact for another, such as a characteristic of the learner.

Revision The Revision process occurs when a system inconsistency has been detected. The revision process is made such that most of the model is kept.
Continuing the previous example (LM1), if the learner says that $Planet(Moon)$, the model has to be revised, because the system inferred $B_l \neg Planet(Moon)$ assuming that the learner knew that $B_l(Satellite(X) \rightarrow \neg Planet(X))$, which is not the case. So, $B_l \neg Planet(Moon)$ and $B_l(Satellite(X) \rightarrow \neg Planet(X))$ will have to be eliminated from the model and $B_l Planet(Moon)$ included in the model.

Expansions and updates do not create many computational difficulties to a learner modeling system, but revisions bring some extra

[4] The dialogue presented is established between two persons (P1 and P2) collaborating to answer the system's (SkyLab) questions.

[5] These changes are named accordingly to the Alchourron, Makinson and Gardenfors theory [2] [3].

tasks to be performed. In order for the system to revise the facts in the model, it has to keep track of all the justifications for all the facts included in such a model and consequently it needs an extra component to maintain those justifications. For instance, in our case, to perform the described changes, we used a system, AMMS, which keeps the justifications for each fact in the model of the learner. This system accepts justifications for the beliefs, which will be nodes in the dependency network kept by the AMMS.

In fact, all the systems that cope with revisions in learner models have some kind of Reason Maintenance system included (see [6] [5] [8]).

However, apart from keeping the justifications of the facts in the model, the learner modeling system has to decide how to revise the model as well. It has to decide which facts to keep in the model, and which ones to eliminate. This choice is based on rationality criterion which depend on the "confidence" the system has on the facts in the learner model. The revision process is made according to the most "trustable" justifications. This notion of trust is based on the confidence the modeling systems have in the process of generating the beliefs to be included in the model. In Murray's work, for instance, this confidence is based on endorsements [8].

4 Learner Changes

Whereas a system contradiction is automatically followed by a revision, a learner contradiction is reported in the model, which will change when the learner changes himself. The learner however can change his beliefs in many different ways.

What we propose here is a theory describing possible conceptual changes made by the learners. However, the learner changes have the following characteristics:

- they are not simple to detect;
- they can be misinterpreted as modeling contradictions;
- they must be represented in the model;
- if a learner change is included in the model, the model should change;
- there can be more than one change, associated with the same learner behavior;
- they represent the "learning" process of the learner.

So, the system has to treat a "learner change" as a special kind of fact to be included in the model. Its inclusion will generate the change of the model itself.

The simplest solution for describing learner changes would be to say that the learner can acquire knowledge (L-expansion) and that the learner can forget knowledge (L-contraction). These two learner changes can be enough to explain the behavior of learners in some systems. However, this is not always the case, and learners revise their beliefs, remember facts they knew or keep inconsistencies, etc. In our theory we identified three types of learner changes (which have to be dealt with differently in computational terms): the **basic changes** (L-expansion, L-revision, L-contraction); the **structure changes** (L-compartmentalisation and L-agglomeration); and the **usage changes** (L-activation and L-acceptance)[9].

The structure changes are used to describe situations where the learner keeps more than one set of beliefs (with contexts) to hold inconsistencies. L-compartmentalisation transforms a set of beliefs into two or more sets in order to keep the inconsistency. The idea is to represent situations where two viewpoints bring two inconsistent results. L-agglomeration transforms two or more sets of beliefs into one maximal consistent set.

The usage changes are used to explain time limited use of beliefs, and the distinction between long term and short term memory. L-activation and L-acceptance change the state of a belief from passive to active and from active to passive respectively. Whereas L-activation represents the situation where a belief is used and becomes active, L-acceptance represents the situation when an active belief is accepted and kept to be used later.

Finally one must stress the utility of representing learner changes in learner models by mentioning some processes that the learner modeling system may perform with the learner changes:

- Perform the learner changes;
  When a learner changes his behavior, the change will be stored in the model, and subsequent alterations will happen. This will allow the learner modeling system to describe in terms of the changes the "learning" process of each particular learner.
- Simulate the learner changes;
  The learning system may want to simulate the learning process of the learner, by simulating some changes;
- Diagnose learner changes;
  The system has to determine what changes may or may not have occurred based on the learner's behavior;
- Predict learner changes;
  The system has to predict a possible change by the learner, based on the learning system interventions.

5 The TAGUS and Changes in Learner Models

The TAGUS (Theory and Application of General User Modeling Systems) is a project developed in order to test some techniques of learner modeling. It provides a language for describing learner models and respective acquisition procedures. In TAGUS a learner model is defined by having the following sets: a set representing the learner beliefs, $\mathcal{BL}$; a set representing the learner's reasoning rules, $\mathcal{RL}$; a set representing some tactics followed by the learner while solving problems, $\mathcal{CL}$; a set representing some of the reflection activities, $\mathcal{REL}$; a set representing the actions performed by the learner, $\mathcal{ACT}$; a set representing some characteristics of the learner, $\mathcal{CHAR}$; a set representing the changes performed by the learner, $\mathcal{CHAG}$ and a set of contexts $\mathcal{C}$.

The beliefs $\mathcal{BL}$, reasoners $\mathcal{RL}$, monitors $\mathcal{CL}$, reflectors $\mathcal{REL}$ [10] and the changes $\mathcal{CHAG}$ represent the system's beliefs about the actual knowledge of the learner, his capability to solve problems in the appropriate domain and the learner's learning performance during the interaction. The observable behavior of the learner is kept by $\mathcal{ACT}$.

The TAGUS system (named after the project) is able to create, acquire and maintain this type of learner model. Its architecture is represented in Figure 1.

We can see TAGUS as an independent Learner Modeling System which interacts with an application (an Interactive Learning Environment or an Intelligent Tutoring System) providing a set of functionalities to create and maintain learner models. These functions provided by TAGUS were built as the core of the system (Core Maintenance System).

The learner engages in a dialogue (an interaction) with the application and the latter provides the details of that interaction to TAGUS.

TAGUS has three main parts: the model of the learner with the characteristics described (the $\mathcal{LM}$), the acquisition module and the maintenance module. The acquisition module is used to create hypotheses to explain the behavior of the learner. Within the acquisition

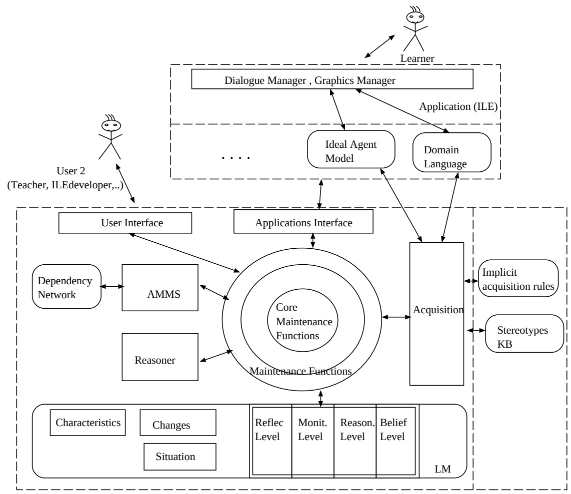

Figure 1. The TAGUS architecture

module, several components can be found in accordance with several strategies of acquisition: the stereotype based module; the diagnostic module; the implicit acquisition module and the explicit acquisition module. For each hypothesis generated into the learner model by the acquisition module, a justification for that generation is kept by the AMMS (the reason maintenance module). It is the AMMS that keeps the justifications for all facts stored on the model. When a contradiction (system or learner) is found, the AMMS gives the set of environments (set of causes) that support that contradiction. The modeling system will decide if it is a system inconsistency or a learner inconsistency. This decision is made by the use of a *trust* function on the facts in the model. The trust of a system in a fact in the model depends on how that fact was acquired, and on which rules were used to support that fact. If a fact in the model was a simple guess, then the system's trust on it is very low, which means that it will be more easily discarded.

If it is a system contradiction, the AMMS will find a new labeling for the network of justifications so that the learner model will become consistent. If it's a learner contradiction, its causes (environment) will be reported to the application, which will create an intervention according to the contradiction. Meanwhile that contradiction is kept in the model. Posterior actions by the learner which involve a change by the learner will trigger the elimination of the learner's inconsistency in the model (see example in Figure 2).

TAGUS also provides an interface to a user (user 2 in Figure 1) who can make direct manipulations in the model of the learner, perform changes in it and simulate the learner's possible behavior in terms of the components of the model.

In Figure 2 the previously presented dialogue is used to illustrate the changes in the learner model (L2) kept by the system.

In the example "expand" is a core maintenance function which performs a change operation (system change) in the learner model. The other operations (revision and update) are also used in the same way.

Finally, TAGUS is implemented using Prolog and Gödel [4] programming languages. Gödel was used to take advantage of the meta-programming facilities in the reasoning system. Prolog was used in conjunction with Gödel in order to provide an interface to a user.

6 Conclusions

Although until recently changes in learner models have been more or less ignored, many reasons can be given in favor of such an analysis: for example, the adequacy of feedback to the learner's behavior will be greatly changed and improved and the learner's learning process can be simulated and expressed in these changes.

The main goal of this work is to treat explicitly the changes in learner models in order to have information about the learner's learning performance. That information in the learner model will help to promote conceptual change during the interaction of a learner with a learning environment.

To do that, we first analyzed the problem of the dynamics of learner models. Two types of changes were identified: learner changes and system changes. These two types of changes were characterized, and a system described which copes with these changes in learner models.

ACKNOWLEDGEMENTS

The first author of this paper was partially supported by the JNICT (Junta Nacional de Investiga cão Cientifica) with the grant n. BD-1556/91-IA.

This work was partially supported by the Student Modelling in Intelligent Learning Environments project (SMILE project, UK SERC grant G9111130).

Many thanks to the Computer Based Learning Unit, at the University of Leeds, where this work was developed.

REFERENCES

[1] Bert Bredeweg and Joost Breuker, 'Device models for model-based diagnosis of student behaviour', in *Artificial Intelligence in Education, 1993*, eds., Stellan Ohlsson Paul Brna and Helen Pain. AACE Association for the Advancement of Computing i n Education, (1993).

[2] P. Gardenfors C. Alchourron and D. Makinson, 'On the logic of theory change: partial meet contraction and revision functions', *Journal of Symbolic Logic*, (50(2)), 510–530, (1985).

[3] P. Gardenfors, *Knowledge in flux*, MIT Press, 1988.

[4] Patricia Hill and John Lloyd, *The Goedel Programming Language*, The MIT Press, 1993.

[5] X. Huang, G. McCalla, J. Greer, and E. Neufeld, 'Revising deductive knowledge and stereotypical knowledge in a student model', *User Modeling and User-Adapted Interaction*, **1**(1), 87–115, (1991).

[6] Mitsuru Ikeda, Yasuyuky Kono, and Riichiro Mizoguchi, 'Nonmonotonic model inference -a formalization of student modeling-', in *IJ-CAI'93*, (1993).

[7] Yasuyuky Kono, Mitsuru Ikeda, and Riichiro Mizoguchi, 'A modeling method for students with contradictions', in *Artificial Intelligence in Education, 1993*, eds., Stellan Ohlsson Paul Brna and Helen Pain. AACE, Association for the Advancement of Computing in Education, (1993).

[8] William Murray, 'An endorsement-based approach to student modeling for planner-controlled tutors', in *12th International Joint Conference on Artificial Intelligence*, pp. 1100–1106. Morgan Kaufmann Publishers, (1991).

[9] A. Paiva, J. Self, and R. Hartley, 'Learner changes in learner models', Technical Report 79, AI-ED Report, Department of Computing, Lancaster University, (1993).

[10] John Self, 'Dormorbile: a vehicle for metacognition', in *Intelligent Computer Technologies in Education (to appear)*, eds., T. Chan and J. Self, (1993).

[11] D. Sleeman and J.S. Brown, *Intelligent Tutoring Systems*, Academic Press, 1982.

Dialogue	Learner Model	Changes (operations on the model)
	LM={Bl Satellite(Moon), Bl Satellite (X) -> ~ Planet(X), Bl ~ Planet(Moon)}	
1. The Moon... 2. The same face... 3. Its a rotating sat..	LM={Bl Satellite(Moon), Bl Satellite (X) -> ~ Planet(X), Bl ~ Planet(Moon), Bl ~SameFace(Moon,Earth), Bl Rotating(Moon,Earth) & Satellite(Moon) -> ~ SameFace(Moon,Earth), Bl Rotating(Moon,Earth)}	expand(~ SameFace(Moon,Earth)) ::Justification(2, AcquisitionRuleNegAns) expand(Rotating(Moon,Earth) & Satellite(Moon) -> ~SameFace(Moon,Earth)) :: Justification(e1, 2, AcquisitionRuleJ, 3) ..
4. The New ... 5. Well... 6... 7... 8... 9. That's a ...Yes.	....	expand(Causes(NewMoon,ShadowEarth)) :: Justification(4,5, 9, AcquisitionRule)
10. A Lunar Eclipse.. 11. So, what is the..	LM={Bl Satellite(Moon), Bl Satellite (X) -> ~ Planet(X), Bl ~ Planet(Moon), Bl ~SameFace(Moon,Earth), Bl Rotating(Moon,Earth) & Satellite(Moon) -> ~ SameFace(Moon,Earth), Bl Rotating(Moon,Earth), (a) Bl Causes(NewMoon,ShadowEarth), (b) Bl Causes(Eclipse,ShadowEarth), (c) Bl Different(NewMoon, Eclipse), Bl inconsistency((a), (b), (c))}	==Domain/LM inconsistency Motivates intervention, predicting possible change. expand(Bl Causes(Eclipse,ShadowEarth)) : Justification(10,11,AcquisitionRulePosAns) expand(Bl Different(NewMoon,Eclipse)) : Justification(13,AcquisitionRuleInterv) expand(Bl inconsistency(Causes(Eclipse,ShadowEarth), Causes(NewMoon,ShadowEarth)))
	LM={Bl Satellite(Moon), Bl Satellite (X) -> ~ Planet(X), Bl ~ Planet(Moon), Bl ~SameFace(Moon,Earth), Bl Rotating(Moon,Earth) & Satellite(Moon) -> ~ SameFace(Moon,Earth), Bl Rotating(Moon,Earth), (a) Bl ~Causes(NewMoon,ShadowEarth), (b) Bl Causes(Eclipse,ShadowEarth), (c) Bl Different(NewMoon, Eclipse), Bl inconsistency(~(a), (b), (c)), Revision(~Causes(NewMoon,ShadowEarth))}	expand(L-revision(~Causes(NewMoon,ShadowEarth))) Justification(13, AcqusitionRuleRev) revise(~Causes(NewMoon,ShadowEarth)) elmi: Causes(NewMoon,ShadowEarth)

Figure 2. An illustration of some learner model changes in TAGUS