Multiagent Simulation of Learning Environments

Elizabeth Sklar and Mathew Davies
Dept of Computer Science
Columbia University
New York, NY 10027 USA
sklar,mdavies@cs.columbia.edu

ABSTRACT

One of the key issues in designing appropriate and effective learning environments is understanding how learners advance and what factors contribute to their progress. This holds true for both human and machine learning environments. In Artificial Intelligence, there is a long tradition of studying human skill acquisition in order to design intelligent agents that learn. Using insight gained from analyzing co-evolutionary machine learners, we have been experimenting with human learning environments by simulating the interactions in a classroom. Here we detail our classroom model, formulated as an *electronic institution*. We describe the types of interactions that can occur between agents bearing one of two *roles* – TEACHER or STUDENT – and define a *dialogic framework* and a *performative structure* for these agents. We share the results of simulation experiments, demonstrating how particular sets of interaction rules can correspond to certain styles of human teaching and learning.

Categories and Subject Descriptors

I.2.11 [**Distributed Artificial Intelligence**]: Multiagent systems

General Terms

Experimentation

Keywords

Multiagent simulation, learning environments

1. INTRODUCTION

Throughout the history of Artificial Intelligence, researchers have studied humans in an attempt to encapsulate behavioral characteristics and modes of learning in computational models. The result within the niche of machine learning is a broad range of techniques, from genetic algorithms [13] to support vector machines (SVM) [23]. While these methods have traditionally been used to build artificial agents that can adapt in dynamic environments or learn to recognize faces, they can also be used to model humans and demonstrate environmental effects on learners. One of the key goals of the work described here is to bring lessons learned from building and studying machine learners back into the realm of human learning and education. We view the human education system as a complex social environment and expand on the simEd model of education as a multiagent simulation [21]. Multiagent simulation is applicable because the education system is a complex, noisy and dynamic environment, consisting of independent, self-interested entities that conform to roles within an organized hierarchy while exhibiting individual and sometimes conflicting behaviors.

The general formalism underlying simEd is based on an *electronic* or *e-institution*, a multiagent framework that consists of [2, 7, 8]:

- *roles* that characterize different types of agents (i.e., who the agents are);
- *norms* that define allowable behaviors for each type of agent (i.e., what the agents can do);
- a *dialogic framework* that describes the ways in which agents can interact (i.e., what the agents can say); and
- a *performative structure* that depicts *scenes* of interactions between agents and *transitions* for linking one scene to the next (i.e., when the agents can do and/or say what).

simEd is constructed as a set of hierarchical "sub-institutions": *classrooms*, *schoolhouses* and *schooldistricts*. Each of these sub-institutions exhibits the characteristics of an *e-institution*, as listed above. The underlying goal within each sub-institution, and across the system as a whole, is *student learning*, which is measured in the classroom.

The work presented here expands on the simEd classroom with the goal of understanding the interplay between various aspects of learning environments in order to be able to design them more effectively. We begin by outlining features of both human and machine learning environments, identifying the *roles* within the classroom. Then we highlight an interaction model that was initially developed as an analysis of co-evolutionary machine learners and then applied to human learning environments, and is used here as the basis for the *dialogic framework* in the classroom. Next we describe a simple agent architecture governing the behavior

AAMAS'05, July 25-29, 2005, Utrecht, Netherlands.

of each type of classroom agent, and we specify *norms* to emulate different styles of teaching and learning. Finally we demonstrate a *performative structure* by showing experimental results of stringing *scenes* of interactions together and measuring various outcomes.

2. TAXONOMY OF LEARNING ENVIRONMENTS AND ROLES AND NORMS

We start with a basic characterization of human and machine learning environments, emphasizing that we are considering both humans and machines (agents), as learners and will discuss three types of settings: ones in which there are only humans (as in a traditional human classroom), ones where there are only agents (as in a traditional machine learning environment) and ones in which humans and agents interact and learn together (as in intelligent tutoring systems or interactive learning systems).

A human classroom typically involves one (or a small number of) human teacher(s) and a larger number of human students and consists of the following components [24]:

- *classroom environment* — the physical arrangement of the room and the structure for interactions between students and teachers and amongst students;
- *classroom management* — prescribed strategies for scheduling activities for students and for handling discipline issues, focused on the development of students' social skills;
- *curriculum* — techniques for structuring, organizing and presenting the domain that is being taught/learned, focused on the development of students' academic skills;
- *teaching methodology* — pedagogical philosophies and educational theories that underpin a teacher's instruction model;
- *instruction model* — particular ways of incorporating and delivering the above components, such as "constructivism" [16] or "multiple intelligences" [11]; and
- *evaluation and assessment* — the way in which a teacher determines if a student is learning.

We use the general term "interactive learning system" (ILS) which not only includes the more specific (and perhaps more familiar) term "intelligent tutoring system" (ITS), but also provides a broader definition encompassing environments that are designed for more exploration on the part of the student than ITS's (which are typically more structured and scripted according to a highly engineered, domain-dependent model). Henceforth, we will refer to ILS's when discussing environments that involve human STUDENTS and agents as TEACHERS. Note that up to now, we have only considered this pairing when examining mixed human-machine learning environments, but in future will examine other combinations — in particular human TEACHERS and agent STUDENTS, for example interactive systems that adapt to human users by training on input from user interactions [5, 10]. A typical ILS consists of the following components [14, 19]:

- *domain knowledge* — a representation of the topic that the student is learning;
- *teaching component* — an instructional model that is used to guide the student through the knowledge domain;
- *user interface* — the interaction mechanism that lies between the human student and the computerized system;
- *student knowledge* — a "user model" of the student in relation to the domain knowledge, indicating how much of and how well the student knows the domain;
- *system adaptivity* — the means by which the system adapts automatically to the student's behavior, backtracking when the student makes mistakes and moving ahead when the student demonstrates proficiency with portions of the domain; and a
- *control component* — the central part of the system architecture which holds all the pieces together.

Contrast these with the typical components of a machine learning system, in which both STUDENTs and TEACHERs are agents [6]:

- *learning element* — the means by which agents in the system adapt;
- *performance element* — the behavior of the student, as it performs tasks within the domain being learned;
- *critic* — a means for evaluating the performance of the student; and a
- *problem generator* — a representation of the domain space being learned and a mechanism for generating questions pertaining to that domain.

Although the terminology is different and the boundaries between components also vary somewhat, we can draw parallels between the three sets of components. The specification of *domain knowledge*, encompassed by *curriculum*, is obviously necessary for all three settings, as are the shared issues of how to *represent* the domain computationally and how to *generate (test) problems* within that domain. We liken the aspect of *problem generation* in a machine learning system to the *teaching component* in a human interactive learning system. The *system adaptivity* and *learning element* components are essentially the same. The *user interface* and *performance element* provide the same function — a means for the learner to "act" in the environment, using the domain knowledge being learned. Finally, the *evaluation and assessment* aspects are folded into the *student model* and the *critic* components.

We are interested in abstracting away the details of a specific domain and focusing on the mechanical aspects of learning environments, in order to run high-level experiments whose results could contribute to future design of learning environments. Based on the above characterizations, we define the following components of our simulation:

- representation of knowledge domain;
- model of what the student knows, i.e., their "mastery" of the subject;
- teacher behavior model — similar to the *instruction* aspect of the human classroom;

- student behavior model — a simulation of the student acting in the domain environment, demonstrating what s/he does and doesn't know; and
- an assessment piece that measures how much individual students know as well as statistics across a group of students (i.e., a classroom).

Returning to the electronic institution formalism, we identify two roles in a simEd classroom: TEACHER and STUDENT. Since the actions of these agents focus on student learning, their *norm*s center on facilitating that end. The details of the norms can vary, depending on the teaching style of the TEACHER and the learning style of the STUDENT. Variations in norms is one of the aspects we experiment with in section 5. The performative structure (detailed in section 4) allows for any number of both TEACHER and STUDENT agents, although our experimental results (presented in section 5) involved one TEACHER and 33 STUDENTS[1].

3. DIALOGIC FRAMEWORK

At the heart of the simEd classroom is the *dialogic framework*, which describes what the TEACHER and STUDENT can say. The inspiration for the mode we have adapted is taken from a game theoretic model based on the Iterated Prisonner's Dilemma [3, 4] and provides an encoding of STUDENT and TEACHER interactions. The model is called the *Meta-Game of Learning*, or MGL, and was originally developed as an analysis of co-evolutionary machine learners [17]. It was subsequently applied to human learning environments [20].

In the MGL, we consider the TEACHER and STUDENT to be players in a game in which each player can make one of two moves at each iteration of the game (see figure 1). The TEACHER goes first and presents to the STUDENT either a *hard* or an *easy* question. The STUDENT responds with either a *right* or a *wrong* answer.

STUDENT:	*right*	*wrong*
TEACHER: *hard*	*learning*	*frustration*
easy	*verification*	*boredom*

Figure 1: Meta-Game of Learning.

Since we assume that the overarching goal of any learning system is sustained progress on the part of the STUDENT, we say that any time the TEACHER or STUDENT makes a move that drives towards the *hard* question and *right* answer state, then that agent is "cooperating" (in IPD terms); otherwise the agent is "defecting". It will be seen (below) that it is convenient to use these labels, although the word choice is rather arbitrary and should not be taken to indicate any malintent if an agent chooses to "defect".

The original work of the MGL was used to explain the phenomenon of *mediocre stable states* in co-evolutionary machine learning environments [17]. In co-evolutionary learning, populations of agents interact and develop through these interactions. The hope is that agents from one population inspire agents from another population to learn, evolving into a type of "arms race" spiral in which the populations take turns being each other's TEACHER and STUDENT [1, 12]. However, since the populations are responsible for evaluating each other, the agents can essentially form tacit agreements to jointly pursue routes that are mutually beneficial and are optimal locally but not globally.

The parallel is obvious in the human learning environment. Teachers evaluate students by giving them grades, but students at the same time submit teaching evaluations of their instructors. Here, again, they can reach a tacit understanding in which teachers agree to make their classes easy and give students high grades; and in return, students will give teachers strongly positive teaching evaluations. Such scenarios would result in lazy students who would not advance very far, and the system would settle into a stable, yet mediocre state.

4. PERFORMATIVE STRUCTURE

Having defined roles, established norms and created a dialogic framework for the simEd classroom, we now explain the operation of these components within the sub-institution by detailing its *performative structure*. We label the classroom Ω and specify it using the triple:

$$\Omega = (\{C\}, \{T\}, \{S\})$$

where C represents the domain knowledge; T represents a set of TEACHER agents, describing the behavior of the teacher, incorporating teaching methodology and instructional model; and S represents a set of STUDENT agents, incorporating both behavior and knowledge models (i.e., how the student acts, how much s/he knows and how s/he progresses). The simulation proceeds in *scenes*; as described in section 4.6, one scene involves 6 steps on the part of the agents.

4.1 Representation of knowledge domain

The domain knowledge is represented as a set of concepts $\{C\}$, illustrated using a graph (see figure 2). A node stands for a single concept (C_k) — a bit of information in a domain, such as the spelling and meaning of a vocabulary word or an arithmetic equation. Links between nodes denote relationships between bits of information; weights on the links indicate the strength of relationships between the nodes. For example, if the domain were vocabulary, then the link between the nodes SKI and SLALOM would be weighted more heavily than the link between SKI and BOOT, because "slalom" is particularly related to "skiing", whereas "boot" can be used in many contexts other than "skiing". This is similar to WordNet [9].

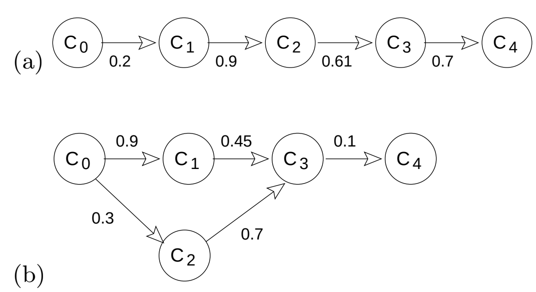


Figure 2: Sample graph of concepts $\{C\}$

[1]This is the typical teacher-student ratio in our local inner-city schools.

Each concept has a difficulty associated with it, a real value between 0 and 1, where lower values indicate easier questions, represented by $C_k.difficulty$. These values, and the ordering of concepts, can be engineered by simEd users, based on their own experience and intuition, or on statistical analysis of test results. For experimental purposes, we set the difficulty values randomly. As described in section 5, we run experiments with the concept order chosen either at random or in ascending order of difficulty.

4.2 Teacher behavior model

In scene i, a teacher presents concept C_k to one (or more) students. The teacher controls the order and timing for which each C_k is presented to each student. Thus, different teacher behavior models and student outcomes can be simulated. For example, the "lecture model", such as in a typical university classroom, can be demonstrated with a simple linear algorithm: $k = i$, implying that one new concept is presented in each scene and that the teacher does not respond to feedback, but simply proceeds blindly by presenting C_{k+1} in scene $i+1$ regardless of whether the students have learned C_k or not. A more flexible teacher ("lecture-feedback model") would make adjustments, not moving on to C_{k+1} until most of the students in the class have learned C_k. A personalized teacher ("tutorial model") would adjust to individual students and present possibly different C_k's to each student in the same scene, depending on what each student is ready to learn.

Figure 3 illustrates the architecture of the behavioral model for the TEACHER, showing the underlying agent-based control mechanism. Computationally, the behavior is effected by two numbers that approximate a TEACHER's emotional state and level of motivation. The underlying assumptions here are: (1) if a teacher perceives that a student is lazy, then the teacher's motivation to perform well goes down, and vice versa (teachers are enthused to teach to eager students); (2) if a student answers a question correctly, the teacher's emotional state will improve (i.e., the teacher will be happier than if the student answers incorrectly). Note that this model is based purely on empirical observation and informal conversation with many of those engaged in the teaching profession at various grade levels.

We experiment with three different norms for teachers, corresponding to those outlined above. The teacher's behavior is ultimately reflected in a value we refer to as *teaching_rate*, indicating the rate at which a teacher moves from one concept to the next. The formulae are listed below, where q (as in "question") indicates which concept the teacher is presenting in scene i. We describe an individual teacher as $t_j \in \{T\}$, where each t_j is dedicated to each student $s_j \in \{S\}$; we can treat this set of teachers as one teacher by giving the same behavior to all t_j's.

1. **lecture model**, which ignores students' progress and "blindly" proceeds, introducing one new concept at each scene without checking to see if the students are actually ready to proceed:

$$\forall t_j \in \{T\}, t_j.teaching\_rate \leftarrow 1 \qquad (1)$$

2. **lecture-feedback model**, which moves ahead only if some percentage of the students are ready:

$$\begin{aligned}&\forall t_j \in \{T\},\\ &\quad \text{if } (\sum_{j=1}^{n} s_j.\Sigma_q) \geq ready\_threshold \\ &\quad \text{then } t_j.teaching\_rate \leftarrow 1\\ &\quad \text{else } t_j.teaching\_rate \leftarrow 0\end{aligned} \qquad (2)$$

where (as described below) $s_j.\Sigma_q = 1$, if student s_j has learned concept C_q, and 0 otherwise; and *ready_threshold* indicates the threshold number of students who need to be ready in order to move on to the next concept (for the experimental results presented here, this value is set to half the class size)

3. **tutorial model**, which adjusts according to the progress of individual students, as if each student s_j had his own private teacher t_j:

$$\begin{aligned}&\forall t_j \in \{T\},\\ &\quad \text{if } (s_j.\Sigma_q)\\ &\quad \text{then } t_j.teaching\_rate \leftarrow 1\\ &\quad \text{else } t_j.teaching\_rate \leftarrow 0\end{aligned} \qquad (3)$$

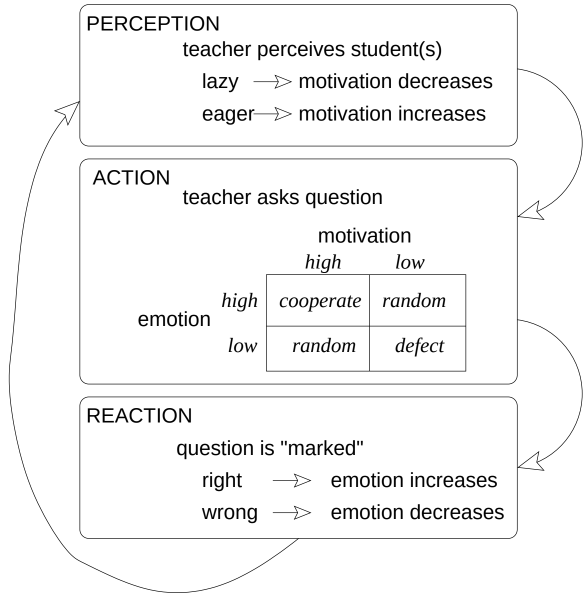


Figure 3: Behavioral model for teacher

Currently, the teacher can only progress in increments of whole concepts:

$$q \leftarrow q + t_j.teaching\_rate \qquad (4)$$

where $teaching\_rate \in \{0, 1\}$. Future work will explore real values, $teaching\_rate = [0, 1]$, in particular for the models that adjust based on feedback regarding students' progress as a real number (instead of $[0, 1]$, see below). This would be akin to a teacher planning to cover a new topic each class period, but finding that she needed to re-visit aspects of a topic in a subsequent class because students indicated that they did not completely understand the lesson the first time.

4.3 Student knowledge model

In order to represent computationally a student's knowledge of the domain, we define $\Sigma \subseteq \{C\}$ to be the set of concepts that the student has been exposed to, or in standard MAS terminology, the agent's set of beliefs about the domain. If student s_j has been exposed to concept C_k, then $s_j.\Sigma$ updates according to:

$$s_j.\Sigma \leftarrow s_j.\Sigma \cup (C_k, \gamma) \tag{5}$$

where $\gamma = \{0, 1\}$ represents whether the student has "learned" (or "grasped") the concept ($\gamma = 1$) or not ($\gamma = 0$). Current work is exploring the use of a real-valued γ between 0 and 1 to indicate the amount of concept C_k that the student has actually learned (i.e., "grasp"), but for the work presented here, we stick to a boolean-valued γ. Note that we use the phrase "exposed to" to mean that the teacher has presented the concept and the student attended the class in which the teacher made the presentation. This allows us to simulate the effects of phenomena such as "absenteeism" (when students miss classes due to illness or other less excusable reasons).

4.4 Student behavior model

Figure 4 shows the architecture of the behavioral model for the STUDENT. Three numbers effect the student's behavior: *aptitude*, *emotion* and *motivation*. As above, this model is based on empirical observation, but incorporates factors which are commonly evaluated in psychological assessments of students. Additional factors, such as physical impairments like hearing loss or learning disabilities like dyslexia, could also be included, but are omitted here for simplicity.

A student's aptitude is a real value between 0 and 1 and relates to concept *difficulty*. Student s_j considers concept C_k to be "hard" if $s_j.aptitude < C_k.difficulty$ and "easy" otherwise. A student's motivational level is effected by the teacher's choice of question. If the student perceives that the question is "hard", we say that the student feels challenged and is motivated to try and answer the question correctly; otherwise, he is less motivated. And, if a student answers a question correctly, he will be happy (his emotional level will increase); otherwise, his emotional level will decrease.

We set a real value *learning_rate* to indicate how much of a concept a student can learn, based on his aptitude:

$$s_j.learning\_rate \leftarrow (s_j.aptitude / C_q.difficulty) \tag{6}$$

Thus, if a student finds a concept "easy", then *learning_rate* ≥ 1. Currently, we cap *learning_rate* at 1, indicating that a student cannot proceed faster than the teacher (i.e., no more than one concept per scene). Future work will explore the notion of students "reading ahead" of the teacher.

4.5 Assessment component

We use several measures to determine how well a student (or teacher) is performing:

- *progress* — which concept number a student is learning in a given scene
- *question* — which concept number a teacher is teaching in a given scene
- *done* — indicates whether a student has learned all the concepts or not (1 or 0)

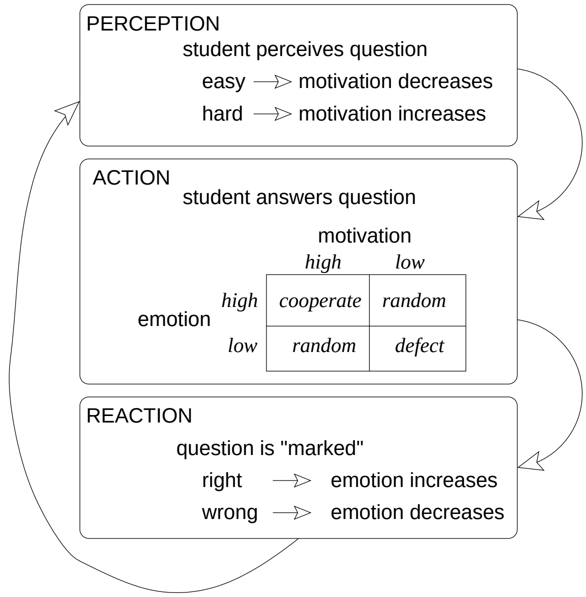


Figure 4: Behavioral model for student

One would assume that *progress* = *question*, however that is not necessarily the case. If *progress* > *question*, then this is like the student is jumping ahead in the textbook. It is not clear how realistic it is to model this, and we have tried capping the value of *progress* so that it cannot exceed *question*. If *progress* < *question*, then the student is moving more slowly than the teacher. This happens frequently in the "real world" (i.e., human classroom), and teachers need effective ways of bringing up students who fall behind. Thus it makes sense to look at the correlation between *progress* and *question* for each student.

To examine how the class as a whole is corresponding to a particular teaching style, we can look at the mean *progress* value for the class. We can also look at the total number of *done* values for the class.

4.6 Scene description

A *scene* contains six steps, as follows:

1. teacher perceives: sets challenge (value between 0 and 1, indicating how much to move ahead in each scene), based on teacher's motivational level (we make the assumption that teachers are motivated to cooperate with motivated students, but not with lazy students);
2. teacher acts: moves ahead according to value of challenge and sets the value of question (q), then presents C_q;
3. student perceives: examines the question put forth by the teacher — is concept hard or easy? (we make the assumption that students are more motivated to answer hard questions correctly and that they become frustrated or lazy if they are continually asked question that are hard);
4. student acts: if a student's emotional level is high and so is her motivation, then the student tries to get the right answer; but if emotion and motivation are both

low, then the student does not try to answer the question;

5. student reacts: emotional level goes up if student's answer is right; otherwise, emotional level goes down;

6. teacher reacts – emotional level is updated based on student's performance (right answers make teachers happy).

This procedure can be repeated for a fixed number of scenes, for example, once per concept in $\{C\}$, or until all students are *done*.

5. SIMULATION EXPERIMENTS

We have run a series of experiments comparing the different rules governing teacher and student behavior, in an attempt first to verify that our models look sensible and second to examine the outcomes. We use a strictly linearly dependent model for C, and assign to each member of C a *difficulty* value between $[0, 1]$. The student's aptitude values are also set between $[0, 1]$, so that if a student's aptitude is greater than or equal to a concept's difficulty, then we say that the concept is considered easy for that student.

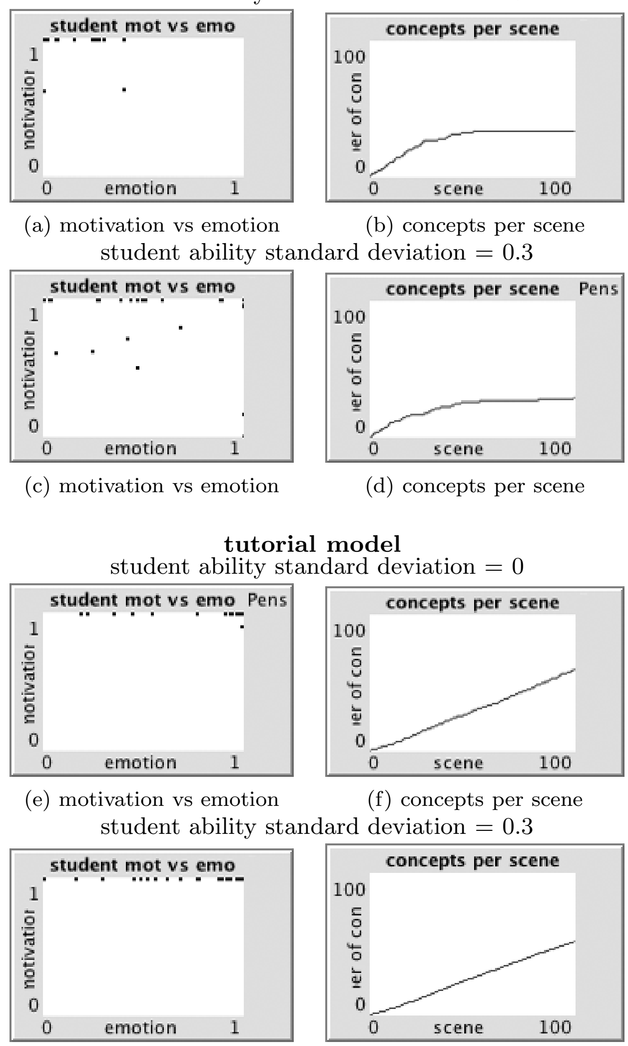
In the experiments, we also varied the standard deviation of the students in the class (0.0, 0.1, 0.2 and 0.3), as well as the ordering of concepts (sorted ordered from least to most difficult versus random order). In figure 5, we show sample results from the **lecture feedback model** and the **tutorial model**, respectively, for standard deviations of student *aptitude* set to 0 and 0.3, with concepts ordered randomly. As expected, the progress of a class with a low standard deviation is greater than that of a class with a high standard deviation. Also, as expected, the progress of a class when using the **tutorial model** is higher than when using the **lecture feedback model**. As well, students' emotion and motivation levels are higher at the end of runs using the tutorial model than they are with the lecture feedback (or the lecture model, not shown).

We have been using both the RePast [18] and NetLogo [15] multiagent simulation environments. The images in figure 5 were produced using the NetLogo simulator.

6. SUMMARY

We have presented recent work on the simEd classroom, a multiagent simulation of a learning environment that models behaviors and interactions of "student" and "teacher" agents. We demonstrated the viability of the simulation with sample experimental results, showing how different models of teacher behavior can effect the outcomes in terms of student performance. Current work involves refining the behavioral models for both types of agents and considering further aspects of learning environments to produce more realistic and robust models. For example, we are considering peer relationships within a classroom when students learn together in groups. As well, we are examining interactions between agents and using argumentation-style dialogues to model conversations between student and teacher agents [22]. We are also exploring the use of statistical results from psychological and achievement assessment data collected from actual classrooms.

lecture feedback model
student ability standard deviation = 0



(a) motivation vs emotion (b) concepts per scene
student ability standard deviation = 0.3
(c) motivation vs emotion (d) concepts per scene

tutorial model
student ability standard deviation = 0
(e) motivation vs emotion (f) concepts per scene
student ability standard deviation = 0.3
(g) motivation vs emotion (h) concepts per scene

Figure 5: Simulation results

For graphs (a), (c), (e) and (g), the vertical axis represents *motivation* as a value between 0 (not motivated) and 1 (highly motivated). The horizontal axis represents *emotion* as a value between 0 (sad) and 1 (happy). The figures contain a snapshot at the end of the run. Motivation and emotion were both initialized using a normal distribution (across all students) centered on 0.5. For graphs (b), (d), (f) and (h), the vertical axis represents the average percentage of concepts completed by the population of students and the horizontal axis indicates the progression of time, representing each scene chronologically.

7. ACKNOWLEDGMENTS

We would like to thank Simon Parsons for his collaboration on the SimEd project. This work was made possible by funding from NSF #REC-02-19347.

8. REFERENCES

[1] P. J. Angeline and J. B. Pollack. Competitive environments evolve better solutions for complex tasks. In S. Forrest, editor, *Genetic Algorithms: Proceedings of the Fifth International Conference (GA93)*, 1993.

[2] J. L. Arcos, M. Esteva, P. Noriega, J. A. Rodriguez-Aguilar, and C. Sierra. Environment engineering for multiagent systems. Preprint submitted to Elsevier Science, 2004.

[3] R. Axelrod. *The Evolution of Cooperation*. Basic Books, 1984.

[4] K. Binmore. *Fun and games: A text on game theory*. D. C. Heath and Company, Lexington, MA, USA, 1992.

[5] A. Cypher. Eager: Programming repetitive tasks by example. In *Proceedings of CHI'91*, 1991.

[6] C. R. Dyer. http://www.cs.wisc.edu/∼dyer/cs540/notes/learning.html, 2003.

[7] M. Esteva, D. de la Cruz, and C. Sierra. ISLANDER: an electronic institutions editor. In *First International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2002)*, pages 1045–1052, Bologna, Italy, 2002.

[8] M. Esteva, J. A. Rodriguez, C. Sierra, P. Garcia, and J. L. Arcos. On the formal specifications of electronic institutions. *Agent-mediated Electronic commerce (The European AgentLink Perspective), LNAI 1991*, pages 126–147, 2001.

[9] C. Fellbaum, editor. *WordNet: An Electronic Lexical Database*. Bradford Books, MIT Press, 1998.

[10] P. Funes, E. Sklar, H. Juillé, and J. B. Pollack. Animal-animat coevolution: Using the animal population as fitness function. In *From Animals to Animats 5: Proceedings of the Fifth International Conference on Simulation of Adaptive Behavior*, 1998.

[11] H. Gardner. *Frames of Mind: The Theory of Multiple Intelligences*. BasicBooks, 1983.

[12] W. D. Hillis. Co-evolving parasites improve simulated evolution as an optimization procedure. In L. et al., editor, *Proceedings of ALIFE-2*, pages 313–324. Addison Wesley, 1992.

[13] J. H. Holland. *Adaption in Natural and Artificial Systems*. University of Michigan Press, 1975.

[14] M. A. Mark and J. E. Greer. Evaluation methodologies for intelligent tutoring systems. *Journal of Artificial Intelligence and Education*, 4:129–153, 1993.

[15] http://ccl.northwestern.edu/netlogo/.

[16] S. Papert. Situating constructionism. *Constructionism*, 1991.

[17] J. B. Pollack and A. D. Blair. Co-evolution in the successful learning of backgammon strategy. *Machine Learning*, 32:225–240, 1998.

[18] http://repast.sourceforge.net.

[19] E. Sklar. *CEL: A Framework for Enabling an Internet Learning Community*. PhD thesis, Department of Computer Science, Brandeis University, 2000.

[20] E. Sklar, A. D. Blair, and J. B. Pollack. Co-evolutionary learning: Machines and humans schooling together. In *Workshop on Current Trends and Applications of Artificial Intelligence in Education: 4th World Congress on Expert Systems*, 1998.

[21] E. Sklar, M. Davies, and M. S. T. Co. SimEd: Simulating Education as a MultiAgent System. In *Proceedings of the Third International Conference on Autonomous Agents and MultiAgent Systems (AAMAS-2004)*, pages 998–1005, 2004.

[22] E. Sklar and S. Parsons. Towards the Application of Argumentation-based Dialogues for Education. In *Proceedings of the Third International Conference of Autonomous Agents and Multi Agent Systems (AAMAS)*, pages 1420–1421, 2004.

[23] V. Vapnik. *Statistical Learning Theory*. Wiley, New York, 1998.

[24] L. R. Weaver and J. A. Carroll. Merrill education's link to foundations website. http://cwx.prenhall.com/bookbind/pubbooks/foundations-cluster/, 2000–2001.