Multiple Agents Acting in Parallel within an Intelligent Real-time Tutor

Chris Eliot
Beverly Park Woolf

Computer Science
University of Massachusetts

Abstract

This paper examines the problem of achieving interaction among multiple agents within an intellgient tutor. The tutor combines agent technology, medical protocals, planning, real-time simulation, and student and domain modeling to yield a tutor capable of monitoring a student's progress with out the need of a human teacher. This tutor provides one of the first complete integrations of real-time simulation and knowledge-based reasoning based upon sophisticated principles of communication and coordiation. We report on novel techniques for commonsense reasoning about plans, multi-agent reasoning and recovery from unexpected situqations.

Introduction

The Cardiac Tutor integrates a number of advanced technologies, including knowledge-based tutoring, simulation, plan recognition, student modeling, domain reasoning and multimedia. Integration of these diverse elements is one of the novel aspects of this system.

It provides one hour of training which appears to be equivalent in to an hour with a human instructor during traditional mega-code training. Students perceive the cardiac simulation as it was intended, i.e., clinical interventions applied to simulated patients undergoing cardiac arrest.The same cognitive skills are required to operate the simulation as required during mega-code training. We have yet to obtain data providing definitive support for each of these claims, as explained in [Eliot and Woolf, submitted], but evidence supports the claim that ongoing practice using the Cardiac Tutor is an effective way to maintain medical skills and that group practice using the Cardiac Tutor is particularly effective for creating a positive learning environment.

Simulation-based tutoring provides students with valuable practice, which is often impossible to obtain in the real world. A simulation coupled with an intelligent tutor is particularly effective because students can practice problem solving in a realistic setting while the tutor can recognize when situations arise that are well suited for learning specific aspects of the domain. Properly situated teaching is effective because the student is better able to comprehend the significance of knowledge if it can immediately be applied to solve a current problem.

The user model for the Cardiac Tutor has been described elsewhere in [1995] and the iterative development and evaluation process are described in [,,,,,,, submitted]. This paper briefly outlines these components and then focuses on the multiple

agents, and reviews, provides an overview of the Cardiac Tutor, Sdetails the .., and Section 5 describes ....

An Overview of The Cardiac Tutor

Instruction in the Cardiac Tutor is organized around a realistic simulation of a cardiac resuscitation situation where the student is able to practice and receive knowledge-based feedback. Domain topics include all of the cardiac rhythms and the interventions used in Advanced Cardiac Life Support (ACLS). The student's comprehension level for each topic is computed by comparing the student's performance with a model of an expert's behavior, represented using a planning formalism encoding knowledge from the standard ACLS protocols [AMA, 1992].

The tutor maintains a student model including a detailed critique of the student's actions during the previous simulation and an overall summary of the student's performance. The student model dynamically alters the simulation to increase the student's opportunities for learning and is represented so that domain topics relevant to each state can be determined. Before the simulation transitions from one state to another, the learning value of each possible outcome is predicted and used to bias the simulation context towards states which are expected to pertain to the student's learning needs, See Figure 12 <???? alsoEliot & Woolf, 1994, 1995].

The mechanisms for simulation-based tutoring are broken down into four fundamental components: • Goal selection or reasoning about the student's needs and the best context for meeting them.

- Plan formation or moving from one context to another based on goal satisfaction.
- Plan instantiation or simulating a new context/plan, even though student input might force the system to revise or abandon outstanding plans.
- Situated reasoning or applying situation-specific knowledge to help achieve the student's goals when the system reaches a desired context.

Proper training for ACLS requires approximately two years of closely supervised clinical experience in an emergency room, ambulance, or similar medical facility. The cost of this training period is high, both in patient care and for the health industry. The high level simulation corresponds closely with the protocols written by the American Heart Association [AHA, 1987] both in terms of level of detail and structure. The system was easily modified to reflect the changeover to the new ACLS protocols described in JAMA [AHA, 1994] despite having been designed and implemented before these protocols were published

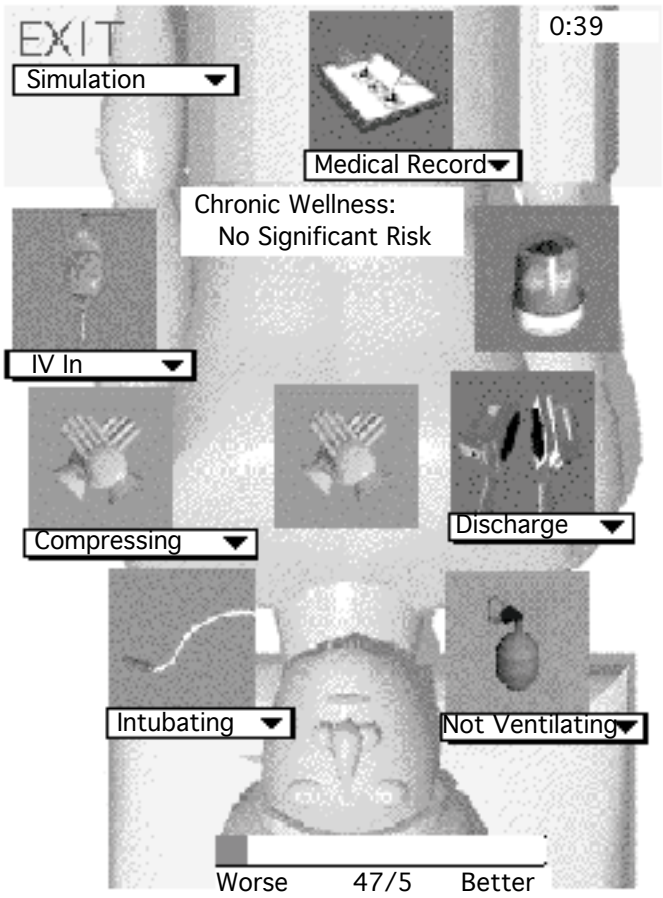
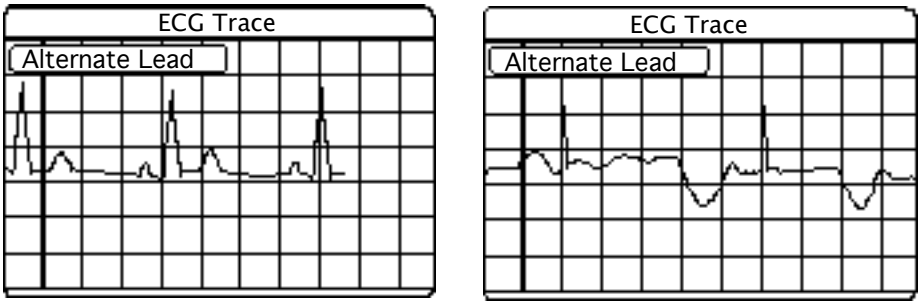


Figure 1. The Simulated Patient

Figures 1-2 show the screen images generated by a simulation of a patient undergoing a series of arrythmias. Figure 1 shows that the intravenous line has been installed and the patient is being intubated. The icons on the chest and mouth indicate that compressions are in progress and ventilation is not. Since the pacemaker (shown in Figure 2) was ineffective, the student tried a sequence of drugs, specifically Epinephrine and Atropine. Following the second dose of Atropine the ECG converts to ventricular fibrillation. Although a pacemaker would not be installed at this point, the previously installed pacemaker continues to generate spikes which are visible on the monitor, Figure 2.

The recommendation for ventricular fibrillation is to immediately apply up to three electrical shocks with the defibrillator initially set at 200 joules, increasing to 300 and then 360 joules. For each countershock the defibrillator is charged to the desired setting. After it is completely charged a few seconds later, the student pressed the *stand clear* icon and selected the *defibrillate* command. Cardioversion is also possible or the student may return the defibrillator to the uncharged state by selecting *dump charge*.



Sinus Rhythm

Ventricular Fibrillation
with pacemaker capture

Figure 2. Simulated ECG Traces.

During retrospective feedback, or post resuscitation conference, every student action can be reviewed At the end of the simulation history a performance review is shown, listing the incorrect actions taken by the student. Each action in the history and performance review is connected to the original simulation state and states connected to the knowledge base, so the student can request additional information about past actions

The Cardiac Tutor is an environment for the application of medical interventions; it is an intelligent interactive simulation of the patient and emergency room team. Generally, this simulation is a descriptive model rather than a causal model. The 'patient', Figure 1, undergoes simulated cardiac arrest, e.g., ventricular fibrillation, bradycardia, asystole and other arrhythmias, while the student employs a variety of interventions, including precordial thump, defibrillation, intubation, IV, or medications, (e.g., lidocaine, atropine, epinephrine, etc.) The simulation provides a realistic exposure to the task allowing medical personnel to practice treatments. A mathematical simulation of a real-time ECG trace, Figure 2 ,based upon the causal cardiac model is presented along with other computer generated graphics. Other pertinent information is provided, including background sounds (emergency room noises) and foreground sounds (shouted responses) generated by computer. The patient's ECG trace, blood gases, vital signs, and general health are constantly visible for monitoring, partly dictated by clinical reality and partly under control of the student (through intervention selection).

Several different simulation mechanisms were used (e.g., the rhythm model, the ECG model, the drug models and the agent models), providing a flexible way to implement system knowledge of processes at different levels of detail. Each simulation mechanism required an event queue. The multiple event queues were coordinated by a central priority based synchronization mechanism which ensured that events were processed as close to the intended real-time moment as possible. High priority events were closely synchronized with the real-time clock of the computer; low priority events could be delayed or deleted in some cases. Consequently the system was able to maintain adequate real-time performance during complex simulations despite limited computational resources. A process based model simulation language with some object oriented aspects was the primary foundation for implementation. Events within the simulation drive the sound and graphic routines and pass information up to the tutoring module. A

causal model of the conduction paths of the heart was implemented for the purpose of generating the simulated ECG trace. In a stochastic simulation the consequences depend upon randomness to some extent and most events have several outcomes. Student input is treated like other events within the simulation.

<Chris--do the last three sentences above sound out of place?

A special purpose language for this causal model was implemented based on the observation that the macro and lexical closure facilities of Common Lisp provide efficient means to encapsulate native code. A causal model of the electrical pathways in the heart that accurately models the aberrant behavior of a specific arrhythmia does not directly produce a clinically accurate representation of an ECG trace. The recorded ECG wave is altered during transmission through tissue and bone and it is a superposed image of the electrical activity of all components of the heart.

<Chris--How much of the next paragraph should be included?

The simulated ECG was generated starting from a causal model of the electrical pathways of the heart. A numeric array was maintained including the deflection of the ECG monitor at a resolution of 100 points per second. A double buffer scheme was employed so that valid data was always available for several seconds before and after the current simulated time. The visible ECG wave was drawn on the screen based directly upon the contents of these data buffers. When insufficient data was available in the data buffer, or when processor time was available, the causal model was updated.When critical events occurred within this model, predefined wave-forms were added to the combined ECG wave stored in the data buffers.

The Cardiac Tutor efficiently integrated several simulation models at different levels of detail. The simulations were coordinated through the operation of a central priority based queue mechanism synchronized to the real-time clock of the computer.

Multimedia techniques were used to improve the realism of the simulation including original digitized sound and high quality graphics. Animation technology was also integrated with the system, e.g., for showing movies of normal cardiac operation in comparison with abnormal behavior. The simulated patient was animated during the simulation, using overlays of graphics to depict ongoing interventions, such as CPR, ventilation and IV insertion.

Multiple Agents in a Tutoring System

A plan recognition system implemented inside the simulation-based tutor served as a model of the domain for comparison with the student's actions. In the simulation-based tutor, the student performs actions which are evaluated by the tutor to assess consistency with the expert's actions, encoded as a planning formalism augmented with information needed to interpret these protocols in a robust way during the simulation.

The protocol interpreter was implemented to model multiple medical personnel acting in parallel. Each action in a protocol is assigned to a role, by the domain expert, and each agent is dynamically assigned one or more roles by the system. Every agent interprets the protocol semi-independently, selecting actions appropriate to its assigned roles and skipping others. The domain requires complex representations and reasoning mechanisms. Many features of the system had to be generalized for multiple, rather than single, instances, such as:

- Multiple agents, including doctors, nurses, EMT's.
- Multiple roles for each agent, such as airways, medications.
- Multiple actions for each agent, such as start IV, intubate, defibrillate.
- Multiple orders, since actions can be done out of order due to parallel activity.

The Domain Model consists of a set of protocols integrated with a real-time simulation of cooperating agents which follow the student's orders. Medical domain experts often express their knowledge in a different form than knowledge engineers would like, typically describing examples or linear sequences of actions, rather than giving general principles, rules or plans. The plan recognition system implemented within the simulation-based tutor used a protocol formalism designed to closely resemble a form used by domain experts to communicate with each other. The protocol interpreter differs from most planning formalisms because it is based upon sequential protocols, not goals, and protocol selection is determined by a straightforward mapping from the current state of the simulation. Plans are expressed in a linear form and the system implicitly recognizes opportunities for parallel activity. The recognition system supports recovery from incorrect user actions and accounts for synchronization of multiple agents. We designed the system this way to closely resemble the form in which our domain experts discussed their knowledge.

The tutor provides students with the opportunity to lead simulated resuscitations. The tutoring system reasons about the student, both as an agent within the simulation and as a student learning a task. The student is responsible for directing the simulated agents and most student commands are simulated as orders for simulated assistants to perform medical tasks. In addition, the student is represented within the simulation and directly performs some actions, such as operating the simulated defibrillator. The simulated team leader shouts an order for another agent to perform when the student selects a command from a menu. Each order is simulated as realistically as possible, including an appropriate real-time delay as the task is performed or if the simulated agent must first complete some prior task. (Additional problems, such as communication failures and errors in the way secondary agents perform their assigned tasks were not modeled in this simulation).

The Protocol Interpreter

The system reasons about the student's ability, by comparing the student's actions with a model of an expert's behavior. Any action that the team leader should take, or any order that should be given will be accepted as a valid student action. Any other

action selected by the student is considered incorrect. Because multiple agents are represented and can act in parallel there is generally more than one correct action that the student can select. Any action that any agent is ready to perform is a valid action for the student to request, either a command for a secondary agent or an action to perform directly. Figure 3 shows a simplified version of the protocol interpreter. The correct protocol must be selected initially by analyzing the state of the simulation. In this example, Protocol-3 is selected so Protocol-1 and Protocol-2 remain inactive. If the simulation had been in some other state initially then Protocol-1 or Protocol-2 could have been selected instead of Protocol-3.

Once a protocol is selected, the interpreter normally follows it to the end, provided that the simulation remains within a specified range of states. If the simulation state ever satisfies a termination predicate associated with the current protocol then that protocol is terminated and the selection process is restarted. In this example, the student has already completed the first action, namely, Act-31 so the next action required is to perform Act-32; any other action is incorrect. The current recommendation is indicated by a pointer which marks the last completed action in the selected protocol.

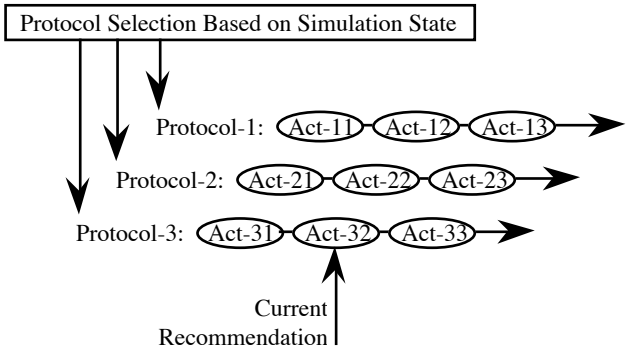


Figure 3. Protocol Recognition.

<< Chris Where are the symbolc for equivalent to and contained within--needed int he next paragraph??

Encoding the knowledge in this domain required a more sophisticated representation than depicted in Figure 3. Some actions may be optional and steps may be augmented with conditions indicating when it can be skipped. For instance, if Act-32 is optional in the current state of the simulation then the current recommendation is the set including actions: R ∫ {Act-32 Act-33}. A student action, A, is correct if A Œ R. Since several actions in sequence may be optional the set of current recommendations may include many actions. If the student's actions were always correct, updating the protocol pointer would be comparatively straightforward. However, incorrect student actions were allowed to affect the state of the simulation, possibly resulting in movement to a state where currently recommended actions are impossible or meaningless [Broverman, 1991]. The protocol interpreter required additional planning knowledge to detect and correct such problems.

The knowledge base allows preconditions and postconditions to be specified for protocol actions. When the protocol interpreter detects an incorrect student action it examines the preconditions of actions in the current set of recommendations and skips impossible actions by moving the pointer. More complex heuristics for replanning were considered but the simple expedient of skipping impossible actions was found to be effective during formative evaluations.

An Example

To illustrate the various steps in the protocal and planning mechanisms we provide diagrams showing the state of the plan recognition mechanism during a sequence of states involved in tracking a student's actions during a simulation.

The protocol interpreter modeled multiple agents acting in parallel. Each action in a protocol is assigned to a role, by the domain expert, and each agent is dynamically assigned one or more roles by the system. In the cardiac domain the roles are *leader*, *airways*, *circulation* and *medications*,. The *leader* is responsible for issuing orders and operating the defibrillator. The *airways* manager is responsible for ventilation and intubation. The *circulation* manager performs chest compressions and the agent in charge of *medications* inserts IV lines and administers all drugs.

The first state in this example of the plan recognition system is shown in Figure 4. The recommended action is to charge the defibrillator to 200 joules. (The student always performs the correct action during this example.) After protocol selection is complete and the plan recognition system is fully initialized, the nine step universal VF/VT protocol was selected and two subprotocols have been activated. The first step of the VF/VT protocol activates the subprotocol for a *sequential defibrillation* sequence at 200 joules, which in turn activates the subprotocol for a *stand clear* sequence, subprocedure A in Figure 5 The *sequential defibrillate* sequence is different from an isolated defibrillate sequence because several actions that normally follow electrical therapy are omitted for efficiency. In particular, CPR would normally be resumed following any defibrillation and vital signs should be determined; in sequential defibrillation electrical shocks follow in quick succession separated only long enough to determine if the ECG indicates a change in cardiac rhythm.

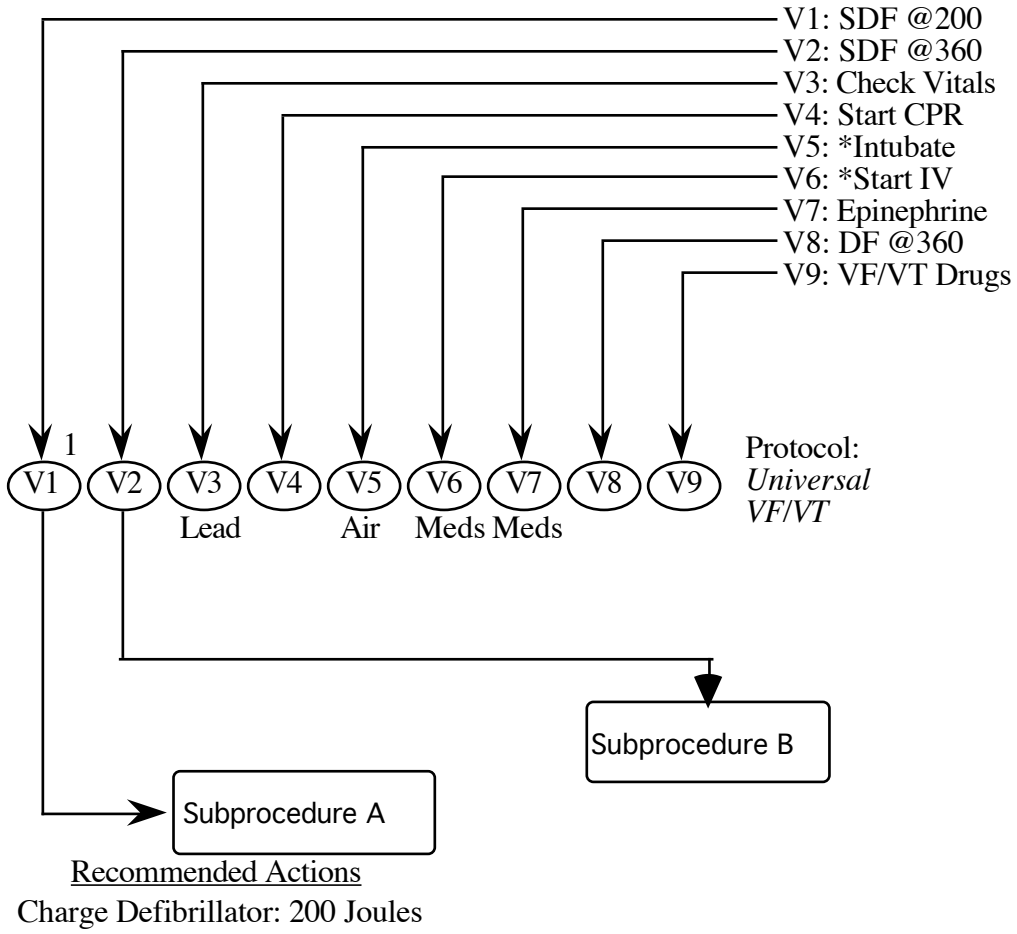


Figure 4. First Charge Defibrillator Step of VF/VT Example.

The simulated team leader agent in identified as 'Lead', the airways manager is identified as 'Air', the circulation manager is indicated by 'Circ' and the medications manager is identified as 'Meds' both in the plan steps and in the protocol pointers. All protocol pointers for the four agents initially point to the first step in the VF/VT plan, which is step V1.[1] Each protocol pointer is moved forward to the first plan

[1] The format used for a plan step Figures 4-6 is described here . Each plan step must be performed by an agent who has been assigned a specific role, except that synchronization steps apply to all agents. An asterisk (*) preceding the label of some plan steps indicates that the action is optional or conditional. Optional actions can be performed or not, at the discretion of the team leader. Conditional actions are required or prohibited, depending upon the state of the simulation.

During plan recognition each plan step is instantiated. The instantiated plan step contains information about its application during the current simulation. As each plan step is instantiated, a sequence number is assigned distinguishing it from all other plan steps instantiated since the simulation was initialized.

applicable to a role assigned to that agent. When subprotocol steps are encountered, the subprotocol is instantiated and the protocol pointer is made to point to the first step in the subprotocol and then moved forward linearly from there.

<<<<<Explain Subroutine A and B !!!!

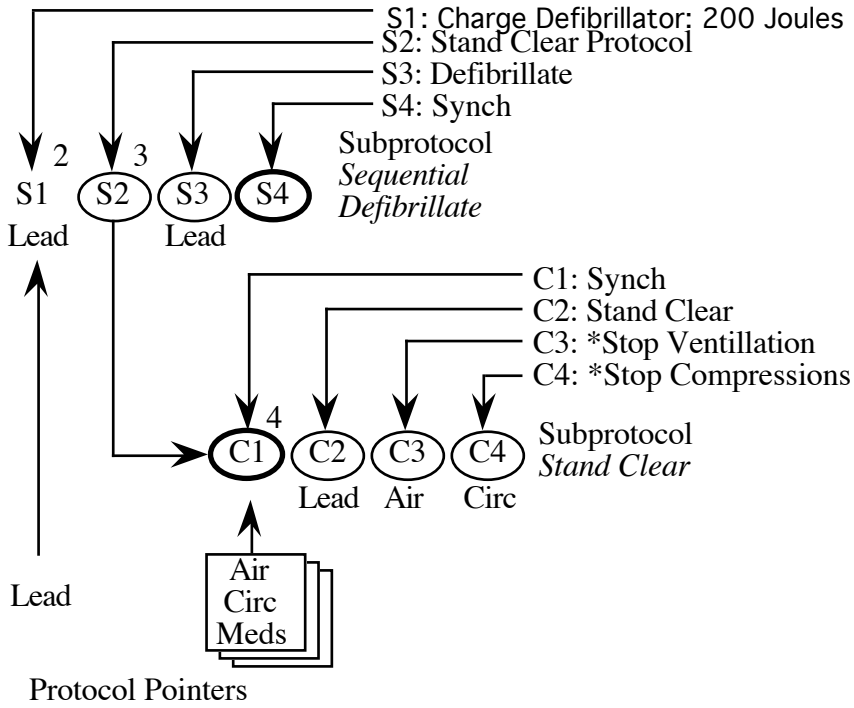
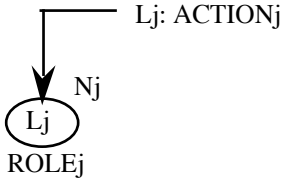


Figure 5 . Subprocedure A of VF/VT Example.

The protocol pointer for the team leader first points to step V1. This is a subprotocol step, so the sequential *defibrillate* subprotocol shown in Figure 6 is instantiated and the protocol pointer is made to point to the first step in that subprotocol, plan step S1. This step is assigned to the *team leader* role, so the protocol pointer for the team leader agent is not advanced further at the time. Hence the first (and in this case

---



Key for plan step j

Lj:	Label for naming this plan step.
ACTIONj:	Action to be performed at this plan step.
Nj:	Sequence number of this plan step.
ROLEj:	Role of the agent who must perform this step.

only) recommended action is to charge the defibrillator to 200 joules. The other protocol pointers are initialized and advanced similarly, except that none of the agents except the team leader can charge the defibrillator. Hence, those protocol pointers move past step S1 to step S2. This is another subprotocol step so the *stand clear* subprotocol of figure 6 is activated and the three agents assigned to medications, airways and circulation eventually point to the first step in this subprotocol, C1. This step is a synchronization step, with a sequence number 4. As each protocol pointer is moved to this step another check is performed to determine if the synchronization condition is satisfied. <<Chris the next few words need to be made into a sentence!>Determining if synchronization step C1 requires the plan recognition mechanism to inspect the protocol pointers for all agents requiring synchronization. (In this domain synchronization always involves all agents but the implemented plan recognition system provided for synchronization among a subset of the agents). In the state depicted in Figure 5. the protocol pointer for the team leader is located at step S1 which has sequence number 2 and sequence number 2 is less than 4 which is the sequence number for the synchronization step C1, so the synchronization condition is not satisfied. Since the synchronization condition is not satisfied the agents assigned to airways, circulation and medications must wait for the team leader to charge the defibrillator, Figure 5. . Those three agents are not recommended to perform any action at this time.

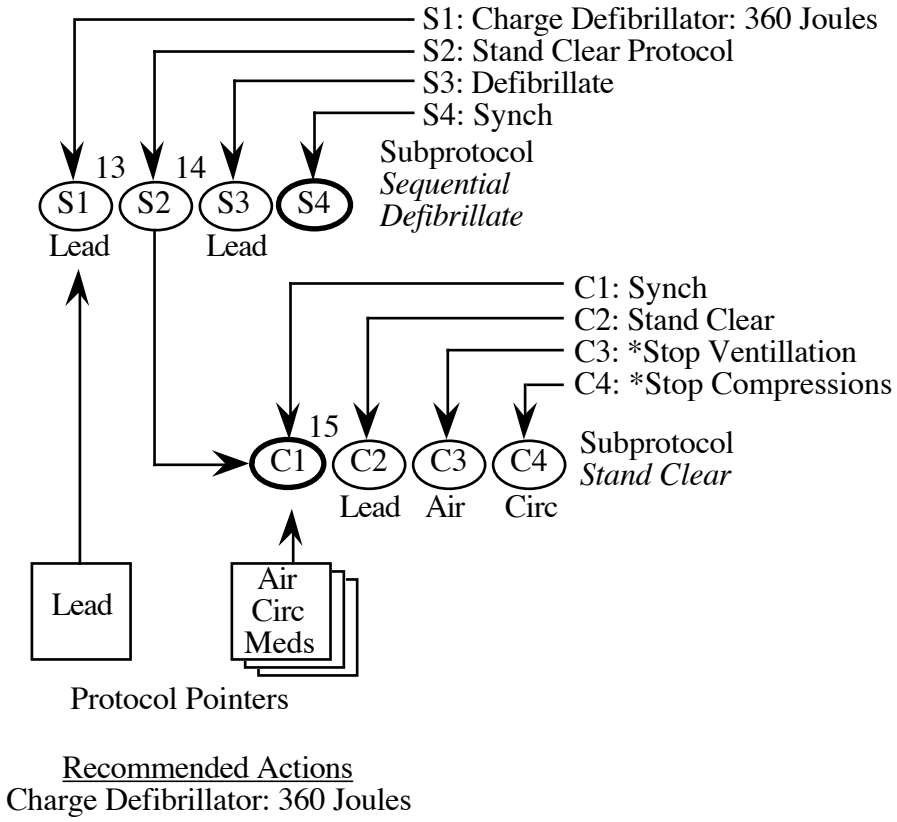


Figure 6 . Subprocedure B of VF/VT Example.

This example continues until all the action items V1- V9 are complete In this example, 38 plan steps are required

<<Chris add more here ?

Multiple Agents

The action items described in the example above, notably V1-V8, S1-S4 and C1-C4, are replaced in the following discussion wit generic actions, Act-nn

The role assigned to each action is part of the static knowledge about actions. The assignment of roles to agents is automatically defined at the beginning of each simulation and varied depending upon how many helper agents were available. The number of helpers and their role assignments does not change during a simulation.The simulated team leader is always assigned the single role of *leader*. The other roles are distributed among the available helper agents as equally as possible. The first agent is always assigned the first role.

The user is responsible for directing all the agents; hence user commands are interpreted as orders for the simulated agents to carry out. Each agent was assigned one or more roles and each action was assigned to a specific role. The key to plan recognition using protocols is the computation of the set of actions that are currently recommended, Figure 7. In this figure actions Act-31 and Act-35 are recommended; other actions are incorrect in this state. Each agent has a separate protocol pointer and interprets the protocol semi-independently, selecting actions belonging to his assigned roles while skipping other actions. Special synchronization actions required all agents to pause interpretation of the protocol until all other agents reached the same synchronization step.

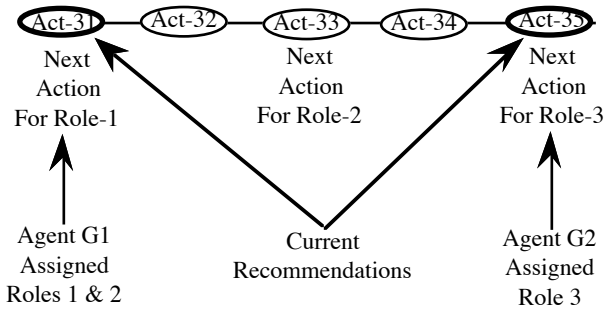


Figure 7. Multiple Agent Protocols.

Initially the protocol pointer for each agent was set to the start of the selected protocol. Then the protocol pointers were advanced past any actions that were irrelevant to the roles assigned to the associated agent. The set of actions recommended for any agent includes the current steps for each role assigned to that agent. If a recommended action is optional then the following step is also recommended. A sequence of actions may all be in the current set of recommended actions in this way. Synchronization steps may be optional at times, but determining this involves reasoning about all the agents simultaneously.

Special synchronization actions indicated which group of agents needed to coordinate at various stages in the protocol and these actions were inserted into the protocols as required. An agent reaching a synchronization action would check to determine if the other relevant agents had all reached the same action. If so, then all agents would be advanced past the synchronization action; otherwise the step could not be completed.

The effect of the synchronization actions had to be accounted for in determining the recommended actions. Recall that the set of recommended actions includes the action following the last action completed and any subsequent actions if that action is optional. Synchronization steps may be optional, but determining this involves reasoning about all the agents. Consequently, a copy of all of the protocol pointers was created as the first step in determining the recommended actions. These duplicate pointers were then advanced as far as possible past all optional steps to determine an upper bound. A second scan of the protocol for each agent was then done to collect the actual recommended actions using the previously determined upper bound to limit the search.

Figure 8 shows how a synchronization action is interpreted. Action steps A1 and A4 must be performed by the agent assigned role R1, which in this case is agent G1 and, similarly, action steps A2 and A3 must be performed by agent G2 who is assigned role R2. When action A1 is performed agent G1 is updated, but must wait at the synchronization step. When action A2 is complete then both agents G1 and G2 complete the synchronization step.

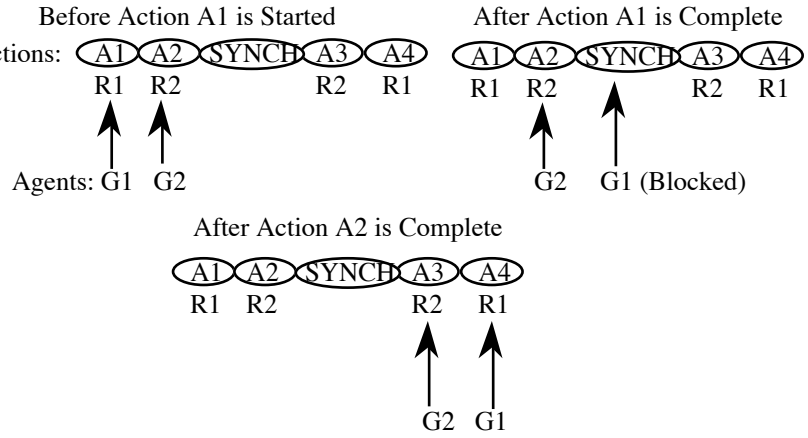


Figure 8. Synchronization of Agents.

Optional actions can interact with synchronization steps, Figure 8. The recommended actions, as specified by the protocols are indicated by the lines forming inverted trees. Several steps before the synchronization step are optional. The analysis to determine the recommended actions requires several steps:

- Agent G1 may skip actions A1 and A3, and is not responsible for action A2, so agent G1 can reach the synchronization step.

- Agent G2 may skip action A2 and is not responsible for actions A1 and A3, so agent G2 can also reach the synchronization step.
- Since both agents can reach the synchronization step it is also optional and both agents may perform later actions.
- Since A4 is optional, agent G2 may next perform any of the actions {A2, A4, A7}.
- Since A5 is required, agent G1 may next perform any of the actions {A1, A3, A5} but not A6.

In this state the recommended action set is {A1, A2, A3, A4, A5, A7}. Action A6 is not recommended because it must be performed by agent G1 who must complete action A5 first in every allowed sequence of actions.

Suppose that the student selects action A4 next. Because action A4 is recommended on the assumption that the synchronization step is satisfied, the student must have elected not to perform any of the steps A1, A2 and A3. Consequently, selecting action A4, which is performed by agent G2, affects the set of actions which agent G1 may next perform; actions A1 and A3 are no longer allowed because the sequences <A4, A1> and <A4, A3> do not satisfy the synchronization step before action A4 is taken. Hence, following action A4 the recommended action set would consist of {A5, A7}.

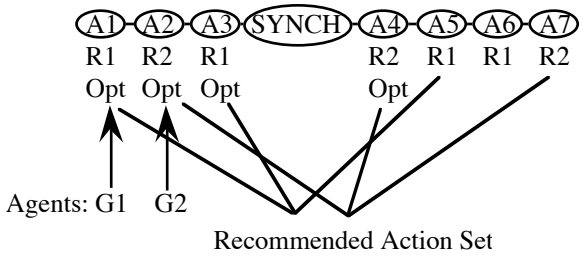


Figure 9. Optional Actions and Synchronization.

Medical experts presented domain knowledge in a format very similar to the linear protocol representation of Figures 3 and 8. We chose to directly implement knowledge in this form so that changes suggested by the domain experts could be easily incorporated into the system. However, the underlying semantics of the protocols may be more clearly understood by observing how they may be transformed into transition networks, Figure 10.

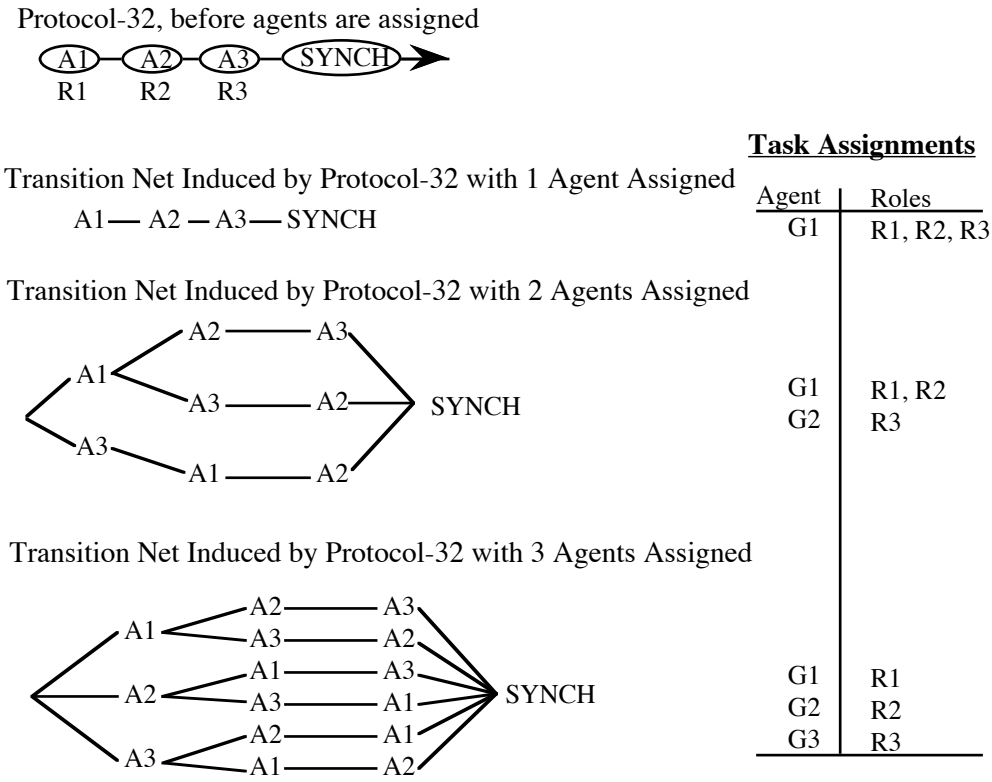


Figure 10. Transition Networks Induced by a Simple Protocol.

The shape of the transition net varies depending upon how many agents are available for parallel activity. Our system does not explicitly perform this transformation; the plan recognition process implicitly implements equivalent semantics. In Figure 7 the nodes are labeled with the action leading from the previous state into the node.

Several additional mechanisms were implemented which improved the expressive power of the formalism and made it easier to use. Conditional actions extended the concept of optional actions and were used to restrict protocol actions to certain simulation states. Many actions require this context sensitivity, such as:

- *Start-IV*, unless the IV is already in place;
- *Start-Compressions*, unless CPR is in progress;
- *Start-Ventilation*, unless ventilation is being done;
- *Charge-Defibrillator*, when defibrillator is not charged;
- *Stop-Compressions*, when CPR is in progress

<Chris --Where does this below belong?>>>

A subprotocol calling mechanism, with argument passing, was implemented to allow for modular construction of the protocols. Semantically, a subprotocol was interpreted as if it were inserted to replace the actions referring to the subprotocol.

The Bias Mechanism

The simulation provides different contexts which may be of differing relevance to teaching a particular topic. For a specific student, the highest priority is given to simulation contexts in which student model suggests will lead to the most important student learning. Heuristics for judging how much a student can be expected to learn in a given context are based upon the evaluation of the student's comprehension of the topics relevant to that context. If the student already understands the relevant topics then little additional learning is possible.

The bias mechanism was able to direct the simulation toward a context in which learning was expected to be successful. In this system, the simulation context was completely defined by the cardiac rhythm. Consequently it was necessary to map learning needs based upon topics into goals for the system to reach a cardiac rhythm. This was accomplished by transforming the priority of a topic into a priority for rhythms.

Each state transition in the simulation is associated with a different probability as shown in Figure 11. The nodes represent states of cardiac arrest or arrythmias and the arcs represent the probabilities of moving to a new physiological state following a specified treatment event or other significant occurrence during the simulation. The left side of Figure 11 represents the normal traversal of the system from one arrhythmia (VFIB) to alternative possible arrythmias (VTACH, ASYS and BRADY). The Tutor alters the probabilities of traversal to increase the probability that a specific learning opportunity will be available to the student. Biasing the simulation to reach states with good learning opportunities is a novel way to implement goal directed behavior.

Each state transition in the simulation is associated with a different probability as shown in Figure 11. The nodes represent states of cardiac arrest or arrythmias and the arcs represent the probabilities of moving to a new physiological state following a specified treatment event or other significant occurrence during the simulation. The left side of Figure 11 represents the normal traversal of the system from one arrhythmia (VFIB) to alternative possible arrythmias (VTACH, ASYS and BRADY). The Bias Mechanism alters the probabilities of traversal to increase the probability that a specific learning opportunity will be available to the student. Biasing the simulation to reach states with good learning opportunities is a novel way to implement goal directed behavior.
The system employs planning technology as a knowledge-based critic to determine goal states. The goals in this directed graph correspond with simulation states that have been identified as providing an opportunity for useful tutoring actions. However, the tutor must reason about interrelations among these goals: several goal states in the graph may satisfy a single tutoring goal; tutoring goals may fail even though the simulation reaches the chosen state; and all tutoring goals need to be satisfied eventually.

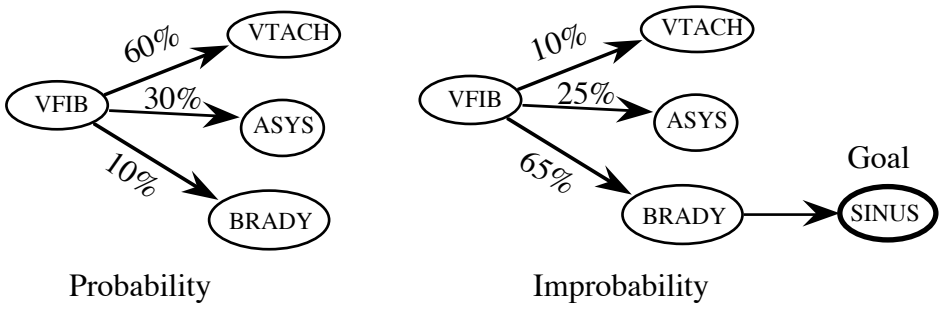


Figure 11. The Bias Mechanism.

The bias mechanism was implemented by searching forward from the current state whenever the simulation reaches a branch point. Normally, the simulation would choose a direction randomly based upon the domain specific probabilities of the outcome. Before making this random selection, however, the bias mechanism searches several states forward to determine if a high priority simulation state is easily accessible. When a goal, or high priority state, is accessible the bias mechanism considers altering the base probabilities of the choice point. The choice is then made randomly based upon the improbable resulting statistics.

Events that occur within the simulation (including user generated events) cause simulation state changes that depend partially upon the underlying probabilities of the simulation model, and partially upon an improbability factor obtained by predicting which high priority topics will be reinforced by each potential state change. Alteration of the maximum allowed improbability causes the simulation to vary from being model directed (driven by the domain statistics) to being goal directed (driven by student learning needs). Intermediate values of allowed improbability cause the simulation to move toward profitable learning states with reasonable speed without making the simulation unrealistically predictable.

The Planning Mechanism

The student model in the Cardiac Tutor is a central data base for all of the information the system has about an individual student. It is integrated with general knowledge about students and with domain knowledge to form conclusions about the best approach for teaching each particular student. Student modeling is a critical component of intelligent tutors, because it allows the system to focus on the specific needs of each individual.
Methods are needed to deal with a user's unknown skills and knowledge, and with change, contradictions, and inconsistencies about this knowledge.

Implementing a student model requires: 1) a structure in which to record inferences about the user; 2) a mechanism for building a student specific model within this theory; and 3) knowledge about how to apply this model to other aspects of the system's behavior. The student model requires both a complex representation and a sophisticated control structure in order for the tutor to be responsive and flexible with students.

The student model was built by comparing student actions with expert protocols (similar to plans or scripts) representing expert behavior, Figure 12. Medical interventions by the student were mediated by the computer and compared to protocols representing expert behavior. The computer offers automated tutorial help in addition to recording, restoring, critiquing and grading student performance. It customizes the simulation to suit different previous levels of achievement and assists the learning process dynamically. Good or improved performance is noted with positive feedback; incorrect behavior is categorized and commented upon.

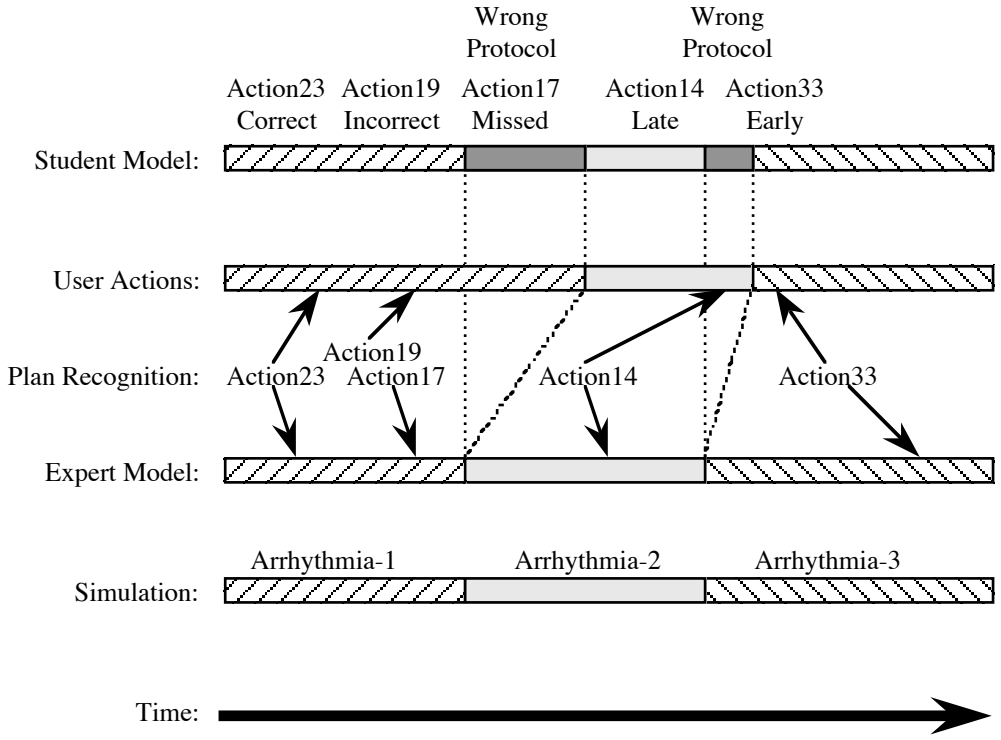


Figure 12. Functionality of the Simulation and Planning Mechanisms.

Figure 13 shows a time varying trace of the integrated simulation, planner, plan recognition system, and student-model reflecting the actions of the student, the system and their effects on each other. The bottom line, *Simulation*, represents clinic reality, in this case, the independent succession of rhythms of the heart during a cardiac arrest. Sometimes the heart spontaneously changes state and goes into one of several abnormal rhythms or arrhythmias. The student is expected to respond to the state changes correctly, *Expert Model*. However, the student's response, *User Actions*, is frequently inconsistent with the behavior of the expert model. During such inconsistencies, a *Plan Recognition* phase predicts what an expert would do and compares this with the student's actions.

The simulation proceeds in a randomized, but clinically plausible, sequence taking into account all interventions that have been applied thus far. All student actions, including incorrect actions, affect the simulation making it possible for the simulation to reach states that are not anticipated by any protocol. General planning

techniques are used so the system can make reasonable interpretations of the protocols in novel situations. Without this flexibility a student mistake might leave the system with inconsistent or impossible expectations leading to a cascade of spurious exceptions. Dynamic feedback is based on real-time comparisons between student and expert actions, in the context of the simulation. Because the system's recommendations reflect the current context created by the simulation model, the recommendations are robust.

Conclusions

This paper addressed issues in reasoning about multiple agents and using a real-time simulation for training. It also described the semantics of the protocol mechanism, the simulation and tutoring mechanisms.

Dynamic construction of the student model involves monitoring student actions during the simulations and evaluating these actions in comparison with an expert model encoded as a multi-agent plan. The plan recognition techniques are novel and allow the expert knowledge to be expressed in a form that is natural for domain experts.

Multiple agent and planning technology enabled the Cardiac Tutor, unlike typical teaching systems, to go beyond simply classifying student actions as correct or incorrect by specifying how an incorrect user action relates to the suggested action. Partially correct actions can be detected by matching a structured description of the user's actions with a structured expert model. Actions can be partially correct when performed too early, too late, out of order or using invalid parameters. (An example of an action with an invalid parameter is giving a correct drug in the wrong dose). Students can also correctly follow the wrong protocol. The system detects when a student correctly follows the wrong protocol, but this capability was not implemented. Partially correct student actions were given less weight than fully incorrect actions, although, the specific way the actions were incorrect was not stored in the student model because the system had no mechanism to use such information. The system responds to idiosyncratic student activity, e.g., by suggesting that the wrong protocol was being followed in Action 14, Figure 12 where the student's action was too late as judged by the expert model.

The student model is rich enough to reason effectively about tutoring goals and to identify goals toward which the simulation should be directed. The system selects problems for the purpose of achieving specific learning goals and reasons about and manipulates the environment to present explanations and content in a favorable context ensuring that problems continue to interest and challenge the student. It improves the tutoring result by biasing the environment as the system moves on toward new contexts creating opportunities for improved student learning.

The tutor implements mechanisms for goal selection, plan formation, and plan instantiation within situated contexts. It includes an accurate descriptive model of the emergency room environment and general patient status, combined with a causal model of cardiac function and related physiologic systems.

Embedding the simulation language within the Lisp implementation significantly reduced the effort required for implementation of the simulation mechanism, and was suitable for research, but resulted in some obscure interactions making it difficult for programmers to develop a clear sense of the scope of the simulation language. Much of the mechanism for a Lisp interpreter was duplicated in the simulation engine but macro definitions from the underlying Lisp implementation resulted in the simulation language inheriting some peculiarities.

Several contributions of this research have been articulated in this chapter, including: 1) integration of a simulation, planner, plan recognizer and student model; 2) development of a student model to direct the curriculum towards desirable learning states; and 3) minimizing negative consequences of inaccuracy in the student model.

References

American Heart Association, *Textbook of Advanced Cardiac Life Support* (second edition), Dallas, Texas (1987).

American Heart Association, R. O. Cummins, editor, *Textbook of Advanced Cardiac Life Support*, Dallas, Texas (1994).

American Medial Association, *Journal of the American Medical Association*, **268** (16) (1992).

Anderson, J. R., Cognitive Psychology and its Implications, W. H. Freeman and Company, New York (1990).

Broverman, C., *Constructive Interpretation of Human-Generated Exceptions During Plan Executions,* Ph.D. Thesis, Technical Report 91-9, Computer Science Dept., University of Massachusetts, Amherst (1991).

Eliot, C. R., and Woolf, B. P., Reasoning about the User within a Simulation-based Real-time Training System, Proceedings of the Fourth International Conference on User Modeling pp. 121-126 (1994).

Eliot, C. R., and Woolf, B. P., An Adaptive Student Centered Curriculum for an Intelligent Training System, *Journal of User Modeling and User Interaction*, pp. 67-86 (1995).

Eliot, C. R., and Woolf, B. P.. Iterative Development and Validation of a Simulation-based medical Tutor, Internatonal Conference on Intelligent Tutoring Systems, University of Montreal, Submitted.