Anchored Interactive Learning Environments

Thad Crews
Department of Computer Science
Western Kentucky University
Bowling Green, Kentucky 42101

Gautam Biswas, Susan Goldman, John Bransford
Dept. of Computer Science & Learning Technology Center
Vanderbilt University, Nashville, Tennessee 37235

Abstract

Advances in computer technology and multi-media systems have led to widespread interest in computer-based instruction and learning environments. The use of video, animation, graphics, and simulation allow the presentation of material in realistic contexts, thus addressing the problems of *inert knowledge* while promoting *constructive* and *generative* learning. But the true potential and benefits of these systems are yet to be realized. Cognitive studies on learning and transfer suggest that concepts acquired in a single context often remain coupled to that context, and are not readily transferred and accessed in other problem solving situations. These findings point to the limitations of traditional, single-context, computer-based instructional environments. This paper introduces an emerging architecture for instruction and learning, Anchored Interactive Learning Environments (AILE). AILE extend well-developed theories of educational practice into the domain of interactive computer assisted instruction. The resulting computer-based environments facilitate sustained learning by active learners. The design and implementation of AdventurePlayer, an Interactive Learning Environment (ILE) anchored in the *Rescue at Boone's Meadow* (RBM) episode from the Adventures of Jasper series is presented. An experimental study demonstrates the effectiveness of this system in the trip planning domain.

To appear: Intl. Journal of AI in Education, vol. 8, 1997.

Introduction

From very early in the computer age, researchers have hypothesized the computer's enormous potential for assisting in the education process. Early research efforts were hampered by a number of factors: lack of powerful hardware, inadequate authoring software for developing computer-based representations of domain and pedagogical knowledge, and the inability of researchers to develop a unified framework for computer-based instruction and learning. Recent technological advances in computer hardware, software and multi-media systems, by themselves have been insufficient to realize the dreams of a computer-driven revolution in educational practice. Significant benefits will start to accrue when computer-based technologies are systematically incorporated into a system of tools that support thinking and learning activities.

The paper develops a new approach for computer-assisted instruction designed to facilitate generative and sustained learning in an interactive multi-media environment. This approach, based on well-developed theories of educational practice, is applied to design and implement an *Interactive Learning Environment (ILE)*, AdventurePlayer, that is anchored to *Rescue at Boone's Meadow*, one of the twelve video-based episodes in the *Adventures of Jasper Woodbury* series developed to teach mathematics problem solving in realistic contexts (CTGV, 1997). The success of AdventurePlayer leads us to propose a new architecture for learning and problem solving, *Anchored Interactive Learning Environments (AILE)*, to facilitate sustained learning in multiple domains by active learners.

Anchored Instruction

The Cognition and Technology Group at Vanderbilt (CTGV) is an interdisciplinary team of researchers at the Learning Technology Center, Peabody College for Education at Vanderbilt University. Members of CTGV have developed and tested a variety of technology-based programs that are consistent with *constructivist theories* (e.g., von Glasersfeld, 1987; Bransford, Goldman, & Vye, 1991; Resnick & Klopfer, 1989; Scardamalia & Bereiter, 1991; Spiro et al., 1991) and *generative learning* in meaningful contexts (CTGV, 1990; 1991; 1992a; 1992b; 1992c). The essence of this approach is to *"anchor"* or situate instruction in the context of meaningful problem solving environments. These anchoring environments are referred to as *macrocontexts* as they involve complex situations that require students to formulate and solve a set of interconnected subproblems (Bransford, Sherwood, & Hasselbring, 1988). Each macrocontext facilitates sustained exploration as it may be revisited from many perspectives over periods of weeks and months. In addition, the anchors (and instructional activities that accompany them) offer additional cross-curricular extensions.

CTGV is experimenting with anchored instruction programs in a variety of areas including mathematics (e.g., CTGV, 1991b, 1992, 1993), science (e.g., CTGV, 1993; Goldman et al. 1996; CTGV, 1991), and literacy (e.g., Bransford et al, 1996; CTGV, 1991; Sharp et al., 1992). These anchoring environments are designed to invite the kinds of thinking and reasoning necessary for students to develop (i) the general skills and attitudes necessary for

effective problem solving, and (ii) the specific concepts and principles that allow them to think effectively about particular domains. The focus has been on promoting *thinking* and *reasoning* in problem solving situations because there is considerable evidence that today's students are not particularly strong in these areas (e.g., Bransford, Goldman, & Vye, 1991; Nickerson, 1988; Resnick, 1987). CTGV's anchored instruction approach shares a strong resemblance to traditional constructivist approaches, such as theme-based learning (Dewey, 1933) and case-based learning (Gragg, 1940).

CTGV has been developing a series of macrocontexts with corresponding classroom material for the purpose of supporting anchored instruction learning activities (CTGV, 1997). These macrocontexts incorporate seven important cognitive learning and instructional design principles that were developed for mathematics problem solving in the *Adventures of Jasper Woodbury* series. Modifications of these principles for science and literacy are discussed in CTGV (1992a; 1997).

Design Principles

The principles that govern the design of anchored learning are outlined below (CTGV, 1997).

- *Generative Learning Format.* The macrocontext storyline creates a meaningful context for problem solving. The end of the story, however, is generated by the student through the resolution of the challenge. Having students generate the story's ending provides motivation: students like to determine for themselves the story's outcome. An additional advantage of the generative format is that students must become active learners when generating and solving the subproblems required by the challenge. Research findings suggest that there are important benefits to having students generate information (e.g., Soraci, et al., 1994).

- *Video-Based Presentation Format.* The video medium allows students to comprehend complex and interconnected problems much better than if the information were presented in a text form. This is especially true for students who have difficulty with reading. The video format allows characters, actions, and settings to be depicted in a rich, vivid, and realistic manner that is hard to achieve in text-only presentations. A second advantage of the video-based format is that it provides the ability to weave in related background information which might motivate the study of other problems in mathematics and other domains.

- *Narrative Format.* The video narrative is designed to contain setting information, a slate of characters, an initiating event, and consequent events. The challenge at the end of the video follows naturally, creating for students the impression that they are solving a realistic problem rather than responding to a lecture on video. Furthermore, the more vivid and graphic depiction of events creates for students a more authentic use of mathematical concepts (e.g., Brown, Collins, & Duguid, 1989).

- *Problem Complexity.* The *challenge* presented to students is a complex problem with many interrelated steps. The complexity is intentional and based on a very simple

premise: *students cannot be expected to learn to deal with complexity in the real world unless they are trained to do so.* Unfortunately, traditional classroom activities do not routinely provide students with the opportunity to engage in the kind of sustained mathematical thinking necessary to solve complex problems. The video presentation of the challenge does not hide the complexity of the task, but at the same time makes is look interesting and solvable.

- *Embedded Data Design.* An important design feature of the mathematics macrocontexts is the *embedded data* format. All the data necessary to solve the challenge are seamlessly embedded in the video story along with a great deal of extraneous information. Unlike typical word problems, the video does not explicitly identify the mathematical problems that need to be solved to complete the challenge. The result is that students must first identify and understand the problem, determine what information is relevant, remember where this information was presented, and then extract that information from the story. In other content areas, the macrocontext introduces students to additional resources they can use to gather necessary data (Goldman, et al.; Sherwood, et al., 1995).

- *Opportunities for Transfer.* Cognitive science literature on learning and transfer suggests that concepts acquired in only one context tend to be welded to that context and hence are not likely to be spontaneously accessed and used in new settings (Bransford et al., 1989). The adventures in the Jasper series are designed so that there are at least three episodes for each problem type: trip planning, statistics and business planning, algebra and geometry. This provides students with the opportunity to use and reuse mathematical concepts in a variety of contexts, thus considerably increasing the likelihood of skill transfer to new situations and reducing the likelihood of *inert* learning. There are also a set of *analog* problems associated with each adventure, which help reinforce and extend mathematical concepts that students use in the original adventure.

- *Links Across the Curriculum.* Each video story contains all the data necessary to solve the challenge. In addition, the story also provides many opportunities to introduce topics from other subject matters. For example, in the trip planning episodes, maps are used to help figure out the solutions. These provide a natural link to geography, navigation, and famous events in which trip planning was an important component, such as Charles Lindbergh's solo flight across the Atlantic.

These seven design principles mutually influence one another and operate as a gestalt rather than as a set of independent features. For example, the use of video brings the world into the classroom in a manner that motivates students. It makes complex mathematical problem solving accessible to students who have difficulties comprehending complex situations from text material. The narrative format, the generative design of the stories, the complexity of the challenge, and the fact that the adventures include embedded data present learning opportunities on subgoal generation, finding relevant information, and engaging in logical decision making tasks while keeping the complexity of the task manageable. The narrative format also makes it easier to embed information and allude to related problems that provide opportunities for links across the curriculum.

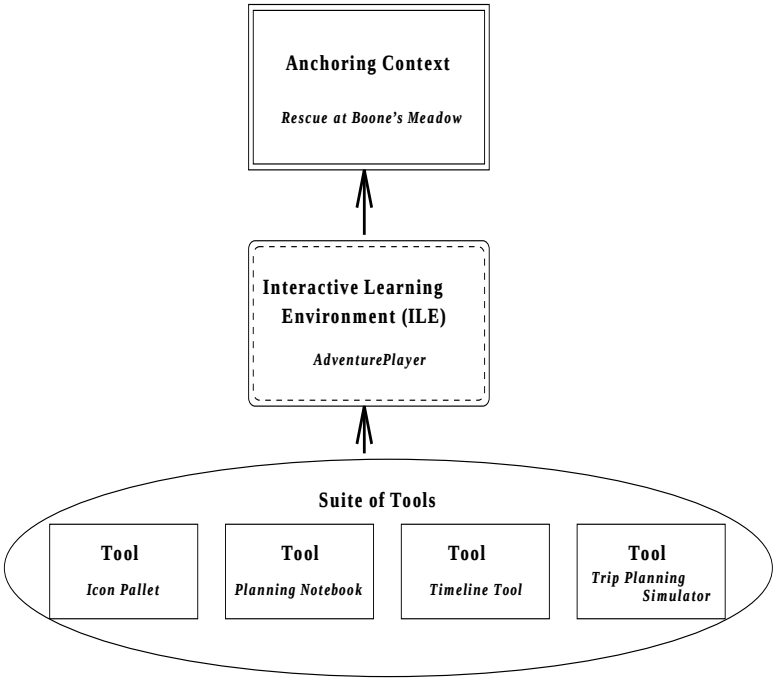

Figure 1: AILE for Rescue at Boone's Meadow

Our research integrates computer-based Interactive Learning Environments (ILE's) that facilitate guided discovery and generative learning with the video-based Jasper adventures (macrocontext). The overall architecture, Anchored Interactive Learning Environments (AILE), facilitates sustained learning by active learners while also providing the necessary guidance and assistance for less motivated learners. We instantiate this architecture in the context of a trip planning adventure, *Rescue at Boone's Meadow (RBM)*.

Anchored Interactive Learning Environments (AILE)

The AILE architecture for the Jasper adventure, Rescue at Boone's Meadow (RBM), is illustrated in the Figure 1. AdventurePlayer, the Interactive Learning Environment (ILE), is invoked by students in the context of the anchor or macrocontext, in this case the trip planning problem. The ILE provides a computer-based problem solving environment that facilitates constructivist and generative learning. This is achieved by providing students with a suite of *tools* that facilitate problem solving while making the important concepts associated with the problem explicit. Tools are designed with built-in *scaffolds* to assist students in bridging gaps in their knowledge of complex aspects of the problem solving process. These scaffolds are gradually removed as students progress in their problem solving tasks. Some tools are also designed to provide *coaching* feedback in the manner suggested by Burton and Brown (1979).

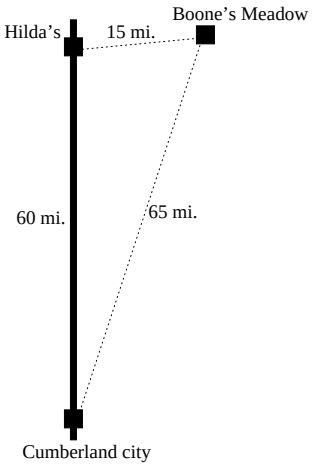

Figure 2: Rescue at Boone's Meadow Map

The Macrocontext: Rescue at Boone's Meadow

Rescue at Boone's Meadow is one of 12 video-based anchors in the Jasper series designed to teach mathematical problem finding and problem solving in traditional classroom environments for students in grades five and higher (CTGV, 1992a; CTGV, 1997). The 12 adventures are organized in triples and cover complex trip planning, statistics and business plans, geometry, and algebra. Each of these adventures provides a very rich environment that creates *"multiple opportunities for problem solving, reasoning, communication, and making connections to other areas such as science, social studies, literature, and history"* (CTGV, 1997). *Rescue at Boone's Meadow* is one of three adventures that deals with complex trip planning.

The adventure's storyline begins with one of the primary characters, Larry Peterson, flying his ultralight into Cumberland City. Larry is teaching his friend Emily Johnson how to fly. During Emily's lessons, she (and the viewers) learn much about the ultralight, including information on payload, fuel capacity, fuel consumption, speed, landing requirements, and how the shape of the wing produces lift. To celebrate Emily's first solo flight, Larry and Emily join Jasper Woodbury for dinner at a local restaurant. During dinner, Jasper reveals that he will soon be taking his annual fishing trip. He plans to drive to Hilda's gas station, park his car there, and then hike 15 miles to his favorite fishing spot in the woods, a remote location known as Boone's Meadow.

While camping in the Meadow, Jasper finds a bald eagle that has been wounded by a gunshot. He radios for help. Emily consults the local veterinarian, Doc Ramirez, who warns that time is the critical factor in saving the eagle. Emily consults a map on the wall that reveals there are no direct roads leading to Boone's Meadow from either Cumberland City or Hilda's. She also learns the distances from the City to Hilda's gas station and Boone's Meadow are 60 and 65 miles, respectively (see Figure 2). The video narration ends with the challenge: *"What is the fastest way to rescue the eagle, and how long will it take?"* At this point, students shift from passive viewing to active problem solving.

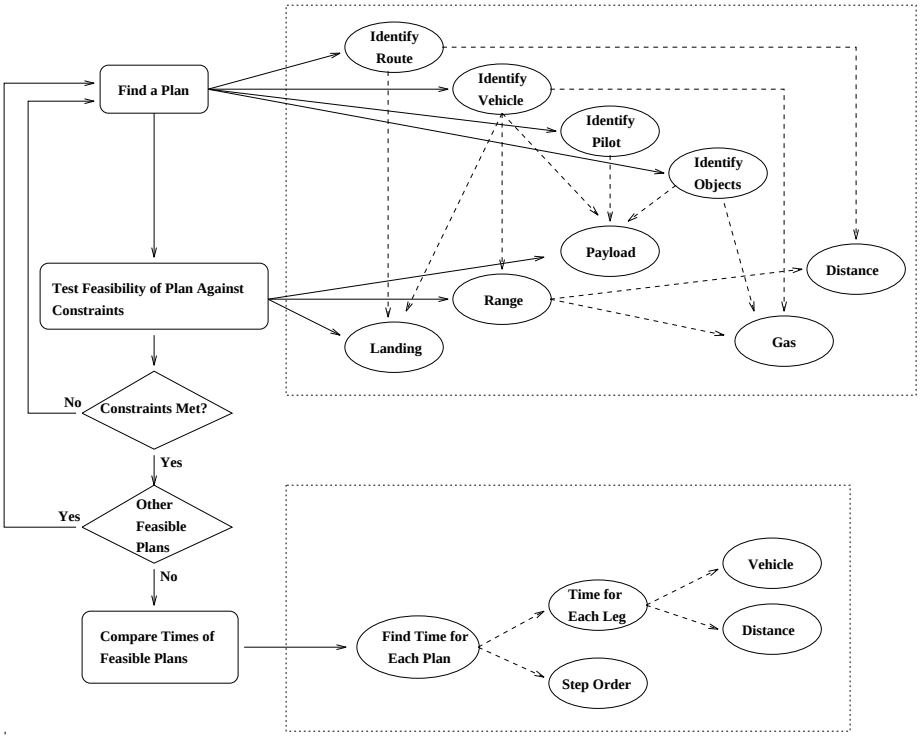

Figure 3: RBM Planning Net (adapted from Goldman et al., 1991)

The RBM challenge of rescuing the eagle appears simple, but in fact requires a variety of problem solving skills. To find the quickest way to rescue the eagle, one must generate multiple rescue plans, validate them against the constraints of the domain, and determine how long each plan would take to implement (see Figure 3). There are a large number of distinct rescue plans that the student may consider since there are three different people who can assist in rescuing the eagle (Emily, Larry, and Jasper), three methods of travel (Emily's truck, Larry's ultralight, and walking), and multiple possible route combinations.

Each plan proposed by the student must be evaluated with respect to the appropriate constraints. For example, if the ultralight is used for the rescue, several constraints, such as payload, landing requirements, and range of the ultralight, will determine the feasibility of each leg of a given route. An additional complication is that payload and range are interrelated constraints. The range of the ultralight will depend upon the amount of available gas, and the weight of the gas must be considered when calculating payload.

Once the student has assessed the plan as feasible, the total time required to complete the plan must be determined. The student will then select the plan that satisfies the challenge of rescuing the eagle in the shortest possible time. The optimal solution requires that

- both Emily and Larry be involved,
- both the truck and the ultralight be used, and
- the rescue route include a stop at Hilda's gas station.

Furthermore, to satisfy payload constraints, the optimal solution requires that Emily pilot the ultralight while Larry drives the truck. Computations also determine that additional gas must be taken on the ultralight to enable it to fly two consecutive legs from Cumberland City to Boone's Meadow, and then from Boone's Meadow to Hilda's.

It is clear from the above discussion that the RBM challenge is quite complex. In protocol studies conducted in the fifth and sixth grades in middle school (CTGV, 1997; Goldman, et al., 1991), students often came up with a solution in a relatively short amount of time. Unfortunately, most student solutions are non-optimal and incomplete. Such data indicate students may benefit from appropriate technology-based learning tools that promote learning and reflection by providing appropriate feedback during the problem solving process.

AdventurePlayer, the Interactive Learning Environment

In our design of AdventurePlayer as an ILE, we made a conscious effort to incorporate beneficial characteristics of both Cognitive Tools (Lajoie, 1993) and Intelligent Tutoring Systems (Wenger, 1987). The goal was to support discovery learning through reflective interaction and constrained curriculum-driven learning so that students developed both general and domain specific thinking and problem solving skills. To achieve this, we adopted four principles that governed the design and implementation of AdventurePlayer.

[Principle 1] *ILEs should implement the principles of generative and active learning.*

Regian and Shute (1992) point out that while there are many published accounts of instructional system design and development (e.g., Wenger, 1987; Lajoie, 1993), the range of domains for which they have been built is narrow, and their implementation and use is often guided by intuition rather than theory. As a result, the student's established learning routine is often altered to accommodate the educational software. The design and development of ILEs should be based on sound learning theories. ILEs should be designed to encourage students' active participation in all their learning activities, including planning, goal setting, knowledge organization, problem solving, and assessment. (c.f., Papert, 1980; Collins & Brown, 1988; Reusser, 1993).

[Principle 2] *ILEs should provide anchored learning opportunities.*

Anchoring problem solving activity in an interesting and realistic environment allows students to experience the kinds of complex, challenging problems that experts encounter. Therefore, anchored instruction invites the type of deep thinking and reasoning necessary for the development of effective problem solving skills. The problem complexity encourages sustained exploration and cross-curricular extensions. AdventurePlayer is designed to facilitate deep thinking and reflection while solving complex but realistic trip planning challenges.

[Principle 3] *ILEs should facilitate and allow guided-discovery learning.*

Guided discovery learning implies that the locus of control is more with the student. However, environments with total student autonomy can lead to poor results, especially in situations where student comprehension is low (Brown & Campione, 1996). In these

situations, it is important that the system help students set cognitive goals, facilitate problem comprehension, provide tools for developing self-monitoring and knowledge organization skills, and to learn how to learn in general (c.f., Scardamalia, 1989; Nathan & Resnick, 1993). It may become necessary, therefore, for the system to intervene during times of excessive student floundering. During these learning plateaus, the system may temporarily provide instruction in the form of coaching to make the student aware of the learning issues associated with the observed floundering.

The notion of coaching was first introduced by Burton & Brown (1979). The purpose of a coach is not to lecture, but to foster the learning inherent in the activity itself by pointing out existing learning opportunities and by transforming failures into learning experiences. WEST (Burton & Brown, 1979), designed as an Intelligent Tutoring system, coached based on an empirical student model. The ANIMATE system (Nathan & Resnick, 1993) provided state-based coaching feedback. Task-based feedback was provided in the TAPS (Derry & Hawkes, 1993) and HERON (Reusser, 1993) systems. Which approach is most effective for different learning contexts remains an open question. Each approach, however, offers significant instructional opportunities that support discovery learning.

[Principle 4] *Knowledge of the domain and problem solving task should be made explicit in multiple ways in the representations and interfaces used in the system.*

Representation of knowledge is an important part of any learning process (cf., Larkin & Simon, 1987; Roschelle, 1990). These representations may be textual, algebraic, and diagrammatic. Such representations of concepts and ideas should be supported whenever possible, even when this information is redundant to other representations. Simon (1995) points out that *"representations may be equivalent in the knowledge embedded in them without being equivalent in the power and speed of the inference processes they enable. They may be informally equivalent without being computationally equivalent." (p. xii.)*

Providing multiple representations for information increases a student's opportunity to understand and apply knowledge. Furthermore, whenever possible, students should be given opportunities to actively construct representations of their knowledge, thereby making explicit their mental models of the problem solving situation and solution.

System Design

The AdventurePlayer system facilitates construction evaluation, and reasoning about solutions to trip planning problems. Trip planning can be cast as a traditional planning problem (c.f., STRIPS, Fikes & Nilsson, 1971; NOAH, Sacerdoti, 1974), where the task is to identify a sequence of actions (e.g, fly the ultralight to the Meadow, pick up the eagle, fly back to the City) that results in the successful achievement of the desired goal (e.g., bring eagle to hospital in City). An analysis of the RBM adventure yields five basic actions (see Figure 4) that may be presented as STRIPS-like operators with preconditions and effects (Fikes & Nilsson, 1971) . Three of these operators are travel actions: *Drive, Fly*, and *Walk*. The other two operators, *Fuel* and *PutIn*, enable gasoline transfer (e.g., fill up the truck's gas tank) and object movement (e.g., put gas can in ultralight), respectively. Other trip planning adventures may necessitate the use of additional operators, such as *Swim, Sail, Run, Bike,*

```
PutIn(Object1,Object2)
    Preconditions
        Whereis(Object1) = Whereis(Object2)
        CanHold(Object2,Object1)
    Postcondition
        Holding(Object2,Object1)

Drive(Means,From,To)
    Essential Preconditions
        Isa(Means,Driveable)
        RoadExists(From,To)
        Whereis(Means) = From
        Holding(Means,driver)
    Computed Preconditions
        NotLessThan(Means::Fuel,FUELNEEDED)
    Postconditions
        Whereis(Means) = To
        Means::Fuel -= FUELNEEDED

Fuel(Source,Dest[,AMOUNT])
    Preconditions
        CanHold(Source,Gasoline)
        CanHold(Dest,Gasoline)
        GreaterThan(AMOUNT,0)
        AtLeast(Source::Fuel,AMOUNT)
        AtMost(Dest::Fuel + AMOUNT,Dest::MaxFuel)
    Postconditions
        Source::Fuel -= AMOUNT
        Dest::Fuel += AMOUNT

Fly(Means,From,To)
    Essential Preconditions
        Isa(Means,Flyable)
        Whereis(Means) = From
        Holding(Means,pilot)
    Computed Preconditions
        NotLessThan(Means::Fuel,FUELNEEDED)
        LessThan(Means::Payload,Means::MaxPayload)
    Postcondition
        Whereis(Means) = To
        Means::Fuel -= FUELNEEDED

Walk(Means,From,To)
    Essential Preconditions
        Isa(Means,Person)
        Whereis(Means) = From
    Postcondition
        Whereis(Means) = To
```

Figure 4: The Rescue at Boone's Meadow Operators

Pay, Inflate, and Burn depending on the storyline and available objects.

AdventurePlayer assists in plan generation and plan evaluation for solving complex trip planning problems. Plan generation is the task of selecting and sequencing actions to achieve a *desired goal*. Plan evaluation involves validating selected actions by checking if each action's preconditions are satisfied. Unsatisfied preconditions are established as *subgoals*, and this invokes further plan generation. In addition, the determination of the *optimality* of a solution in the plan evaluation phase requires computing plan costs in terms of chosen units (e.g., time, dollars, etc.) and comparing alternate plans in terms of these costs.

When the student has identified a planning action, AdventurePlayer responds to that action according to the following three principles:

The Valid Action Principle. Any action that is valid in the real world (regardless of its usefulness to solving the problem at hand) is allowed by the system.

The Invalid Action Principle. Any action that is not valid in the real world due to physical limitations imposed by the domain is not allowed by the system. In addition, the system provides a message explaining why the action is invalid.

The Incomplete Action Principle. Any action that is incomplete is allowed by the system unless there is direct evidence that the action is invalid in the current scenario.

An example of a valid action would be to walk from Cumberland City to Boone's Meadow. This particular action may not contribute to the final problem solution; nevertheless, it constitutes an activity that can be performed in the real world and, therefore, is allowed by the system. However, an attempt to drive from the City to the Meadow is invalid because the video states that there is no road access to the Meadow. The system would not

allow the requested action, informing the student with an explanation, such as *"There is no driveable road from Cumberland City to Boone's Meadow."*

An incomplete action is an action with one or more unsatisfied (but possibly satisfiable) preconditions. An incomplete action is accepted by the system in the same manner as valid action, but internally the action is labeled as incomplete. For example, flying the ultralight requires the specification of a pilot and the inclusion of enough gas in the ultralight's tank to complete the flight. A fly action without the specification of the pilot (or with insufficient fuel) is allowed by the system during initial planning. However, in order to generate a complete final solution, the student at some later point must address the missing information for this and any other incomplete actions.

There are two reasons for handling incomplete actions in this manner. First, it allows students to plan hierarchically by focusing initially on the general plan and filling in details later (c.f., Sacerdoti, 1974). The second reason for allowing incomplete (as well as valid but non-optimal) actions is related to the objective of providing a constructivist learning environment rather than a traditional interactive tutoring system with strong system control. It has been shown that negative learning can result from constraining student performance too tightly (Anderson, Boyle, Corbett, & Lewis, 1990).

AdventurePlayer incorporates a number of tools to facilitate the complex trip planning problem solving task: an icon pallet, a planning notebook, a timeline representation, a plan simulator, and a planning coach (see Figure 5). Each of these components are described briefly below.

AdventurePlayer Tools

Tools are invoked by students from within a particular ILE to help solve problems and achieve goals. For example, a *timeline tool* may be used for trip planning or scheduling events at a business meeting. Likewise, a general tool for creating and manipulating graphs may be useful in ILE where data collection and analysis are required for problem solving. Our tools are designed to provide scaffolding (Vygotsky, 1978) to help students initially cover gaps in their detailed knowledge while not loosing sight of the big picture or the overall problem they are attempting to solve. This is valuable as anchoring contexts are, by design, quite complex and may require problem solving skills that are initially beyond the student's current skill level. The set of tools designed in the AdventurePlayer context (see Figure 5) are described in detail below.

Icon Pallet

The icon pallet is always visible in the AdventurePlayer interface. All objects relevant for the adventure are represented in the pallet as icons, and students may double-click on any icon to receive relevant information about that object. For example, if one clicks on the *ultralight* icon, parameters, such as *maximum payload*, *size of its gas tank*, *rate of gas consumption*, and *flying speed* appear on an information window (Figure 5). Information retrieval is necessary

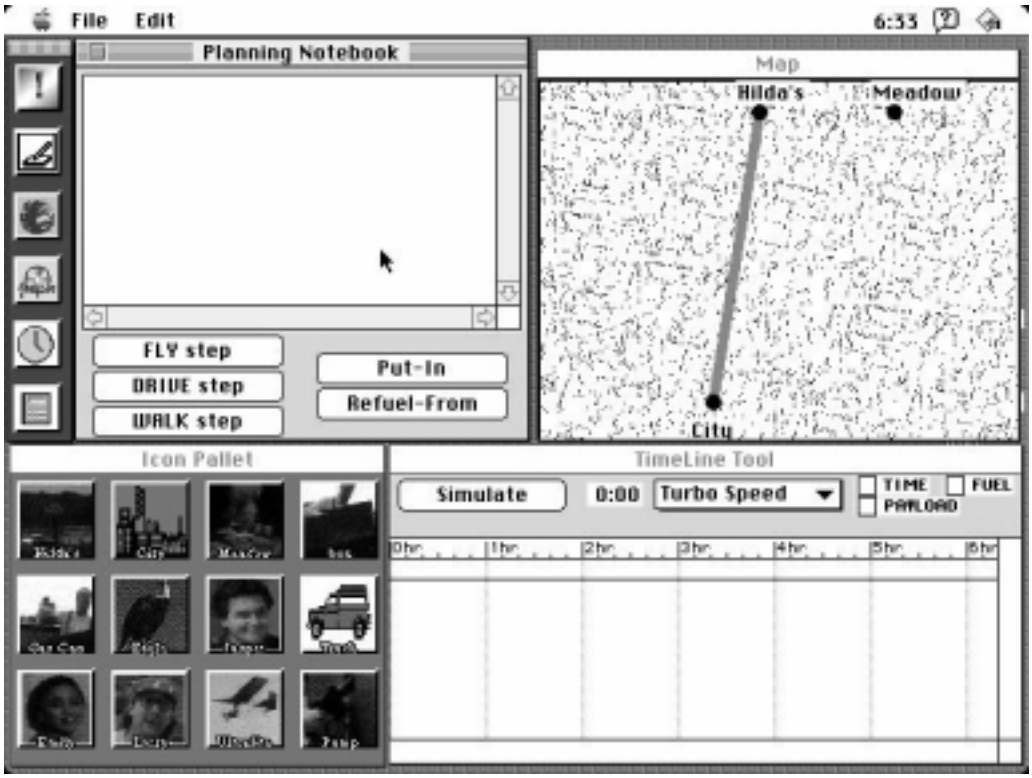

Figure 5: AdventurePlayer: An Interactive Learning Environment

during the planning process, since the anchoring macrocontext contains more factual data than a student can retain in short term memory.

Planning Notebook

The planning notebook is a template-based graphical tool for communicating and representing planning actions. The notebook allows students to generate and instantiate planning actions (including actions that are incomplete and/or non-optimal). At the bottom of the notebook is a set of buttons corresponding to all actions suggested by the anchoring macrocontext. "Travel" actions (e.g., Drive, Fly, Walk) involve movement of an object from one location to another. "Enabling" actions (e.g., Put-In, Refuel-From) involve the movement of objects and substances from one object to another at the same location. When a button is selected, a template for the selected action appears in the notebook with default slot values. Travel actions are displayed in bold while enabling actions appear in plain font and are indented. The slots are updated by dragging icons from the icon pallet and dropping them onto the target slot. Figure 6 shows a partial plan with an uninstantiated "Fly" action, and Figure 7 shows the fully instantiated optimal solution to the RBM adventure.

The template-based textual notebook for communicating and representing planning actions provides the following advantages:

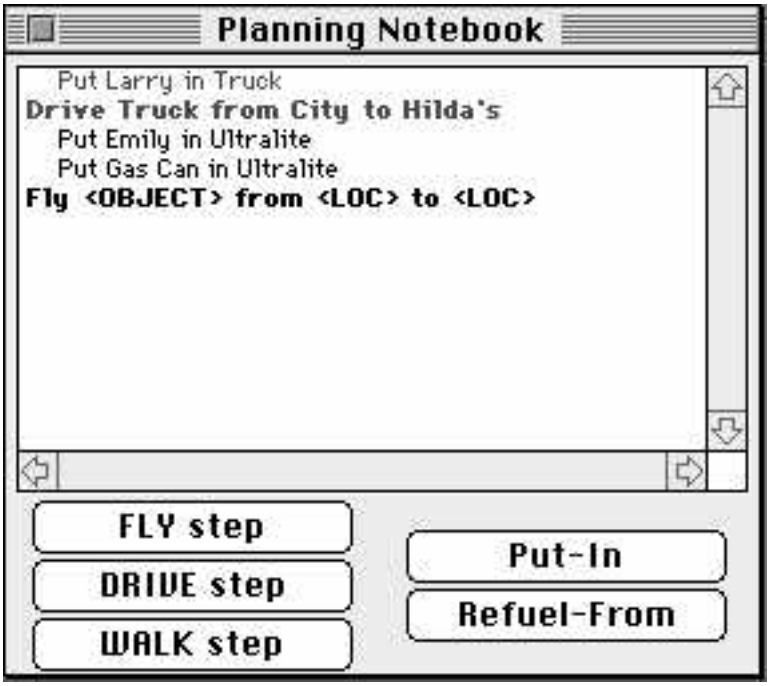

Figure 6: Planning Notebook with a partial plan

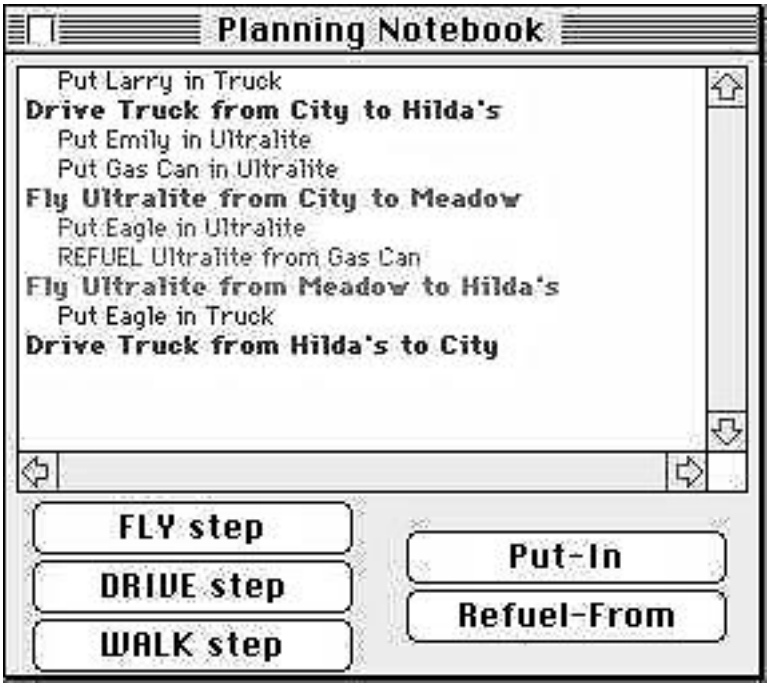

Figure 7: RBM Optimal Solution

- The planning notebook facilitates non-linear planning. For example, a student may chose to drive the eagle from Hilda's gas station to the City even though neither the vehicle nor the eagle is (yet) at the station.
- The planning notebook facilitates abstract planning. For example, a student may establish an abstract "Fly" action and incorporate it into the plan without immediately filling in the slots.
- The planning notebook facilitates intuitive plan modification. Changing the step "Fly ultralight from Meadow to City" to "Fly ultralight from Meadow to Hilda's" is as easy as dropping the Hilda's icon on the destination slot. Likewise, the action sequences may be reordered by traditional drag-and-drop actions.
- The planning notebook provides visual distinctions of precondition actions through font changes, indentation, and color highlighting. When a travel action becomes fully specified, the action becomes highlighted with a color and may then be dragged to the timeline (discussed below) where the student can specify temporal information regarding the step.

Timeline

The timeline tool allows students to construct a diagrammatic representation of their plan. Fully specified travel actions in the notebook may be dragged into the timeline where they are represented by a *timebar* which may be moved and resized to reflect (a) the start of the action, and (b) the duration of the action. This provides students with graphical and visual mechanisms (positioning and sizing) to make explicit the temporal issues associated with planning.

Early work on this project revealed that text-only plan representations created ambiguities with respect to step ordering. The timeline makes students specify explicitly when steps occur, thereby forcing them to think if steps should occur in parallel or in linear sequence. Such dependencies expose students to issues related to scheduling complex task operations.

The timeline's representation is both intuitive and expressive enough to specify sequential actions (i.e., action-2 cannot begin until action-1 is completed) and parallel actions (action-2 does not depend on action-1, therefore, it can commence before action-1 is complete). For example, Figure 8 illustrates the relative lengths of each travel action as well as the ability to execute the initial fly and drive actions in parallel.

Plan Simulator

Simulation is a powerful pedagogical tool that has been used successfully in various instructional systems such as STEAMER (Williams, 1981), ThinkerTools (White, 1993), ANIMATE

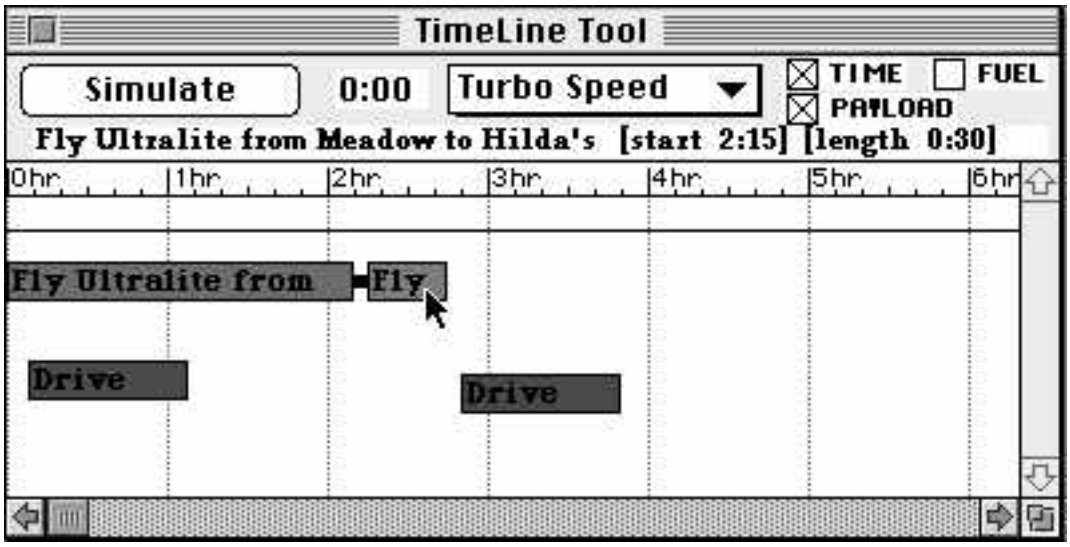

Figure 8: Timeline with four travel actions

(Nathan & Resnick, 1993), and the Envisioning Machine (Roschelle, 1990). AdventurePlayer's simulator is state-based, driven by the ordering and size of the timebars in the timeline. The simulation makes explicit parameters and concepts that are important for problem solving. For example, if a student chooses to fly the ultralight with insufficient fuel, the ultralight takes off, but crashes at some point before it reaches its destination. Rate-time-distance relations are also illustrated. The time allocated by the student to complete a travel action is made explicit by the length of the timebar. When simulating time, the simulation will end too quickly if the bar length is too short. Likewise, the simulation will go beyond the target if the timebar is too long. The simulator supports qualitative understanding of the temporal aspects of the plan by having objects move on the screen at speeds that are directly proportional to their actual speed and for a time that is directly proportional to the length of the appropriate timebar. When a student generates a complete but non-optimal plan, the simulation feedback provides pointers to the steps that require the most time. As a general heuristic, the student is requested to direct attention and focus replanning activities on those steps.

Another important feature of the simulation environment is the amount of control given to the student. The student not only determines when to perform the simulation, but also controls the degree of feedback generated by the simulation. In the AdventurePlayer context, the student may select to receive or ignore feedback on the domain constraints payload, fuel, and time. Providing the student with the ability to control feedback conforms to the third ILE principle discussed earlier. Furthermore, the principle of minimal help (Reusser, 1993) is taken a step further since the student–not the system–decides what feedback is relevant.

Trip Planning Coach

Because of the complexity of the RBM challenge and similar optimal trip planning problems, AdventurePlayer contains a component for coaching students on optimal planning strategies. Most traditional planning systems are not designed for generating optimal solutions. Those that do address optimal planning typically attempt to optimize the planning process rather than the plan itself. The planning space for an optimal solution is quite large; generating an optimal solution for even the simple "blocks world" domain has been shown to be NP-hard (Chenoweth, 1991; Gupta & Nau, 1991). For this reason, any system that plans optimal solutions could benefit from the use of heuristic search techniques to constrain the search process.

The AP* search algorithm (Crews & Biswas, 1993a) was developed in an effort to combine the advantages of hierarchical planning and best-first search to generate plans that are optimal for some prespecified cost criteria. Furthermore, the AP* algorithm was designed to be transparent in its problem solving performance, an important feature from a tutoring perspective, as previously demonstrated by SOPHIE (Brown, et al, 1982) and a number of other intelligent tutoring projects (Wenger, 1987).

The AP* algorithm is an extension of the A* algorithm (Hart et al., 1968) for application to hierarchical planning domains in which an optimal goal is desired. The h' function is defined to be the cost of executing the operator assuming that only the most abstract level of preconditions need to be satisfied. The addition of more detailed preconditions can only increase the cost of the operator, thus guaranteeing that h' will never overestimate h, and, therefore, that the AP* algorithm is admissible. Branching is constrained by means of domain heuristics. The two heuristics used for AdventurePlayer's trip planning coach are:

Heuristic 1: *Direct Route Heuristic* – Consider the direct route to the destination using the best available vehicle.

Heuristic 2: *Best Vehicle Heuristic* – Consider an indirect route to the destination if a better vehicle can be employed on the alternate route.

Algorithm: AP*

1. Start with the search tree containing only the initial node, called *ROOT*. For this node, $g = 0$, compute the h' value, and set $f' = h' + 0 = h'$. Set the *SOLVED* list to nil.
2. Until a goal state has been reached, or the f' of *ROOT* is $=$ *FAILURE*, repeat the following procedure:
   - Move down the search tree from *ROOT* to a leaf node by selecting the appropriate child for each non-leaf node as follows:
     - If the node is an OR node, select the lowest cost child node.
     - If the node is an AND node, select the first unsolved child node.

     Call this leaf node the *BEST_NODE*.

- If *BEST_NODE* is immediately executable, do the following. Execute *BEST_NODE* to generate a new state. Update the search space by placing *BEST_NODE* on the *SOLVED* list. If *BEST_NODE* is an OR child, or if *BEST_NODE* is the last AND child of a node, put *BEST_NODE*'s parent on the *SOLVED* list and repeat this propagation until either a parent node is not solved or *ROOT* is placed on the *SOLVED* list. Repeat step 2.
- Since *BEST_NODE* is not immediately executable, attempt to generate successor nodes for *BEST_NODE*.
  - If no successors are generated, assign *FAILURE* to the h' of *BEST_NODE*.
  - If successors are able to be generated, make *BEST_NODE* an OR node. For each *SUCCESSOR*, do the following:
    * Set *SUCCESSOR* to point back to *BEST_NODE*.
    * Compute $g(SUCCESSOR) = g(BEST\_NODE)+$ the cost of getting from *BEST_NODE* to *SUCCESSOR*.
    * If *SUCCESSOR* is an AND node, create *SUB_NODE* for each sub-node of *SUCCESSOR* and set $h'(SUCCESSOR) = \sum h'(SUB\_NODE)$.
    * Update the h' for *BEST_NODE* to be the minimum h' of all *SUCCESSOR*s.
  - Update all h' values between *BEST_NODE* and *ROOT* by the following process. If the parent of *BEST_NODE* is an OR node, set h' of the parent node to be the minimum of its children's h' values. If the parent of *BEST_NODE* is an AND node, set h' of the parent node to be the sum of its children's h' values. Repeat this propagation of information until *ROOT* has been updated. Repeat step 2.

The search space generated by the AP* algorithm is useful for coaching purposes. Consider a student whose plan is to fly to the Meadow and back. There are two problems with the student's plan. First, there is not enough fuel in the ultralight to fly to the Meadow and back. The second problem is that even if there were enough gas in the ultralight, this solution would not be the optimal one. The second point is more interesting from a coaching perspective (simulation alone provides all necessary feedback pertaining to the lack of fuel.)

AdventurePlayer's trip planning coach examines the AP* search space looking for the critical point where the student's solution deviates from the optimal solution (Figure 9). In this case, examination of the critical point reveals that the optimal solution followed a branch generated by the Best Vehicle heuristic. The coach may use this knowledge when interacting with the student.

In the process of making the student think of a better solution, a coach should avoid giving too specific a response (i.e., "It would be faster to fly the eagle to Hilda's, put the eagle in the truck, then drive the truck from Hilda's to the City."). While this feedback would likely result in the student developing the optimal solution, it is doubtful the interaction would help the student develop general trip planning skills. Instead, coaching should focus on the general skills behind the desired behavior. For this reason, AdventurePlayer's coaching may occur on a variety of levels designed to stimulate productive student reflection:

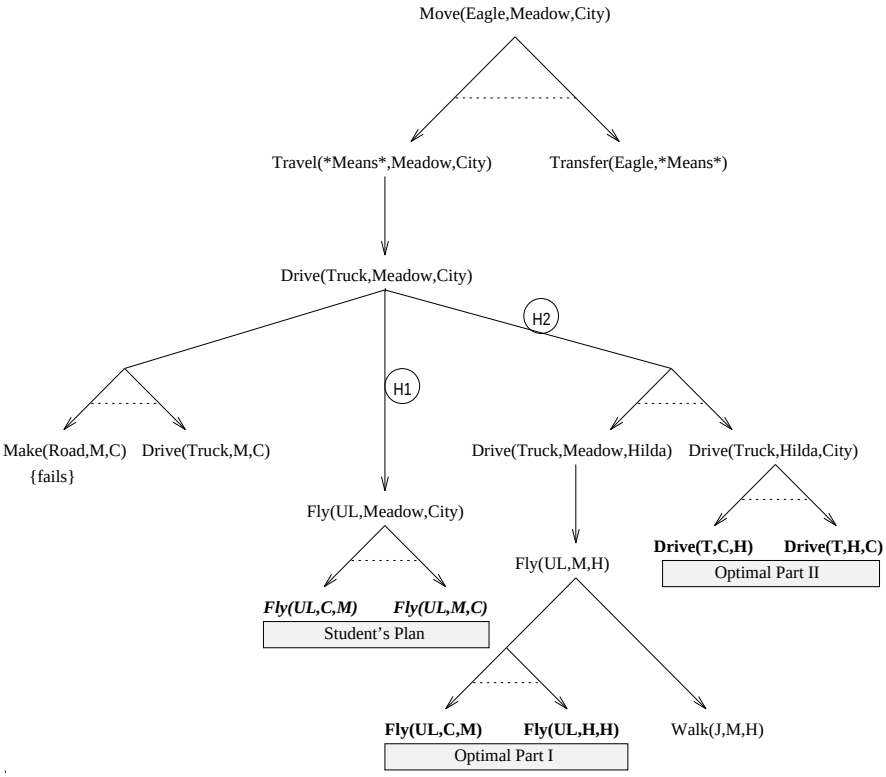

Figure 9: AP* search space with non-optimal student solution

Coaching Level 1: Consider - *"Your solution takes 5 hours. Is that reasonable?"*

Coaching Level 2: Question - *"Is your plan the fastest possible plan to rescue the eagle?"*

Coaching Level 3: Challenge - *"I have a solution that is 45 minutes faster."*

Coaching Level 4: Hint - *"Your plan uses the ultralight. Consider how a faster vehicle could be used in your plan."*

Coaching Level 5: Instruct - *"Try using the truck to get the eagle from Hilda's to the hospital. If you do not understand, ask your teacher."*

System Evaluation

AdventurePlayer was evaluated in terms of its utility for assisting students in solving the *Rescue at Boone's Meadow* Adventure. In particular, the study provided empirical data regarding the problem solving benefits resulting from the timeline, simulation, and coaching components. An earlier study (CTGV, 1997; Goldman et al., 1991) demonstrated that sixth grade students working as individuals without any computer assistance demonstrated weaknesses in plan generation, plan evaluation, and plan optimization skills. Our hypothesis

was that AdventurePlayer's timeline, simulation, and coaching components would assist students in these essential planning tasks.

The study involved two sixth grade classes in a Bowling Green, Kentucky public elementary school. For the study, the set of students were randomly distributed into two groups. Group 1 served as the control group and used a barebones version of the AdventurePlayer environment that included the planning notebook and icon pallet tools. The map depicting the three locations, the City, Boone's Meadow, and Hilda's gas station was also depicted on the AdventurePlayer interface. This system was labeled the core system. It facilitated abstract and hierarchical plan generation, but provided no mechanisms by which the students could verify their solutions. Group 2 used the complete ILE environment that included the core system plus the interactive timeline and simulation tools. Coaching was also available for Group 2 students. This feedback was provided only if the student generated a complete but non-optimal solution. Since the coach had no effect on the initial problem solving activities, the effects of the timeline and simulator components on student problem solving could be ascertained by comparing groups 1 and 2 up to the point where coaching might occur in Group 2.

The study was conducted in a two week period over a total of 7 class periods. A pretest (see Appendix 1) was administered at the beginning of the first class period after a brief introduction. The results of the pretest were used to establish balanced groups. Students worked individually on assigned computer systems. A training session, where students followed scripted directions for entering a worked example, helped students become familiar with the AdventurePlayer interface. After the computer intervention was completed for all students, the posttest was administered in the original classroom. The posttest was identical to the pretest, and again students were given as much time as they required to complete the test.

Results

We compared group performance in the following categories: (a) plan generation, (b) plan evaluation, (c) plan optimization, and (d) improvement in posttest scores. The results are summarized below.

Plan Generation

Plan generation requires the selection and sequencing of useful actions that contribute to achieving a desired goal. Often the successful execution of actions require other preconditions to be satisfied. These preconditions are established as subgoals and additional action selection is required so as to achieve the overall goal. Generating working plans in the RBM domain is a complex task for middle school students, and in many cases they have been observed to generate incomplete plans (Goldman et al., 1991). For the purpose of this study, a *complete plan* was defined as one that, if implemented, would successfully accomplish the goal of rescuing the eagle. An RBM plan may be incomplete for a variety of reasons:

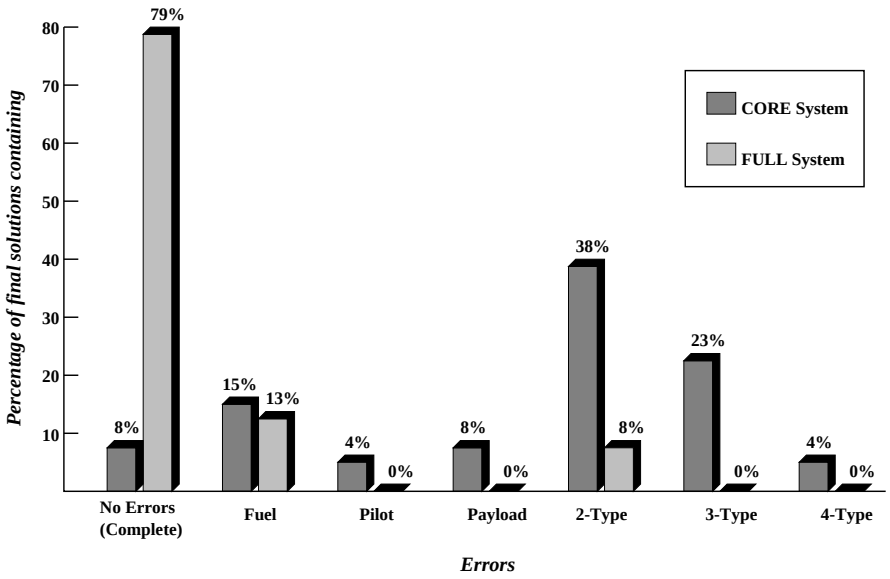

Figure 10: Classification of students' final solutions

- Domain constraints such as pilot, payload, and fuel may be unsatisfied.
- Physical conditions may be violated, such as driving where there is no road, or an interaction between two objects that are at two different locations (e.g., placing the eagle in the truck when the eagle is at the Meadow and the truck is at the City).
- Logical errors may exist whereby the plan could execute to completion but fail to transport the eagle to the desired location.

Analysis of students' final solutions revealed that 19 of 24 (79%) students using the full environment derived a complete solution, whereas only 2 of 26 (8%) students using the core system derived a complete solution. These data are for solutions *prior to* any coaching by the system. A χ^2 test indicated that the frequencies of the complete solution were significantly related to condition, i.e., $\chi^2(1, 50) = 26.17, p < .00001$ (first set of bars in Figure 10).

Students' final solutions were incomplete for a variety of reasons. In plans containing only a single error, insufficient fuel was the cause for planning errors for 4 students in the core group and 3 students using the full system, excessive payload was the cause for planning errors for 2 students in the core group, and 1 student in the core group failed to identify the ultralight's pilot in his plan. A majority of the plans created by students in the core group (65%) contained multiple errors related to fuel, pilot, payload, physical, and logical constraints. For example, 38% of the core group's plans and 8% of full group's plans had two errors. The remaining student plans created by the core group contained three or more errors whereas none of the students who used the full system had more than two errors. These data are not influenced by the coach. Coaching did not occur until after students had identified a complete plan. This analysis of students (pre-coaching) final solutions suggests that the state-based feedback provided by the simulation tool helped full system students in generating complete, valid plans.

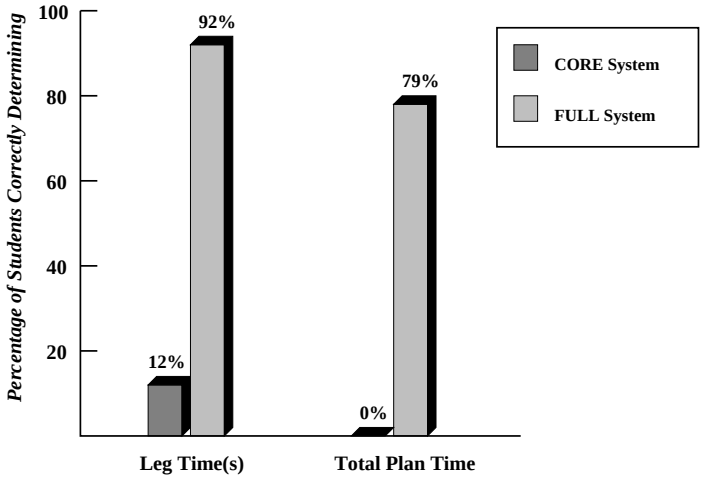

Figure 11: Percentage of students who evaluated plan times correctly

Plan Evaluation

The RBM challenge requires students to evaluate their plan in terms of the actual time required to complete the plan. Only 3 of the 26 (12%) core system students evaluated time correctly, even for a single leg of the plan, whereas 22 of the 24 (92%) full system students evaluated at least one leg time correctly. The first set of bars in Figure 11 illustrates that the timeline and simulation increased students' ability to determine individual leg times. A χ-square test established the significance of the result, i.e., $\chi^2(1, 50) = 32.05, p < .00001$.

Furthermore, not a single core system student evaluated total plan time correctly, whereas 79% of full system students were able to derive the total plan's time correctly. Note that these data are prior to coaching intervention for students using the complete environment. The result illustrated in the second set of bars in (Figure 11) was significant, $\chi^2(1, 50) = 39.22, p < .00001$.

It should be noted that many full system students appeared to have used trial and error when determining leg times rather than compute each time from the rate-time-distance (RTD) formula. They did this by resizing the bars in the time line tool arbitrarily. When the simulator provided time specific feedback (e.g., *time too long* or *time too short*) students picked up on this cue to resize the bar accordingly. This process was repeated a number of times until the students converged on the right bar size. The combination of the time line tool and simulator feedback worked well as a scaffold for students who were weak in RTD calculations. However, it was surprising to see that students who had done the RTD calculations in the pretest correctly also preferred to use the trial and error approach rather than use the formula to compute the time.

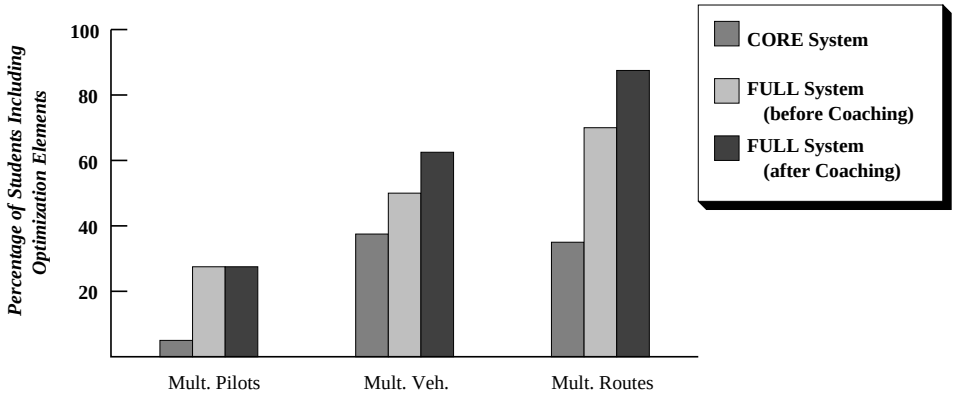

Figure 12: Optimization by considering multiple plans

Plan Optimization

The plan generation and plan evaluation predictions described above concern Adventure-Player's utility in assisting students to generate a complete plan and determine the time required to execute that plan. However, the challenge presented to students requires them to identify the *fastest* way to rescue the eagle. Plan optimization involves considering multiple plans in order to select the best plan. System trace analysis reveals that for the core system students, 4% considered multiple pilots, 38% considered multiple vehicles, and 34% considered multiple routes (Figure 12). For the full system students, 29% considered multiple pilots, 50% considered multiple vehicles, and 71% considered multiple routes without any coaching. After coaching, 62% of the full system students considered multiple vehicles and 88% considered multiple routes.

While the timeline and simulator may increase the number of plans considered by students, they do not guarantee that the student will derive the optimal solution. For this reason, AdventurePlayer contains a coaching component designed to assist students who generate non-optimal solutions. This trip planning coach provides general trip-planning feedback based on the AP* algorithm. The coach was available only to full system students, and was activated only when a student generated a complete but non-optimal plan. Of the full system students, 7 generated the optimal solution without any coaching and 5 did not generate a complete solution. The remaining 12 students received coaching, after which 75% (i.e., 9 out of 12) achieved the optimal solution (Figure 13).

When the student had a complete but non-optimal plan, the system posed the following question to the student: *Is your plan the fastest possible plan for rescuing the eagle?* If the student replied *NO*, the student was allowed to continue problem solving without any coaching. If the student replied *YES*, and that student's plan was non-optimal, the coach intervened with appropriate advice.

As an example of coaching interaction, consider the situation with student 23. His non-optimal plan involved flying the ultralight from the City to the Meadow, from the Meadow to Hilda's, and from Hilda's to the City. The optimal solution is similar except that the final leg involves driving the truck from Hilda's to the City rather than flying the

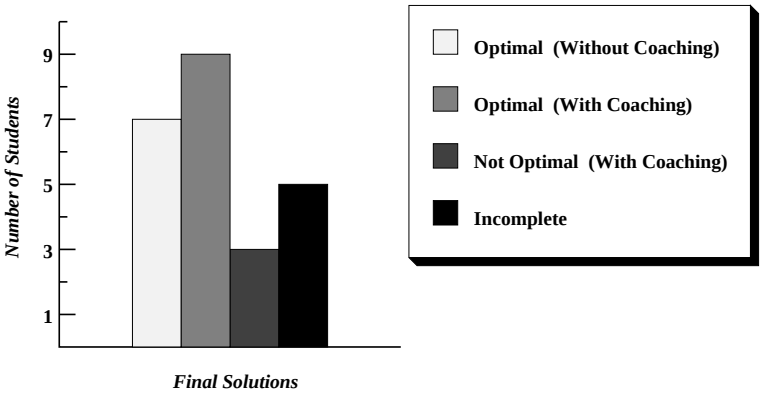

Figure 13: Final solutions for Group 2 students

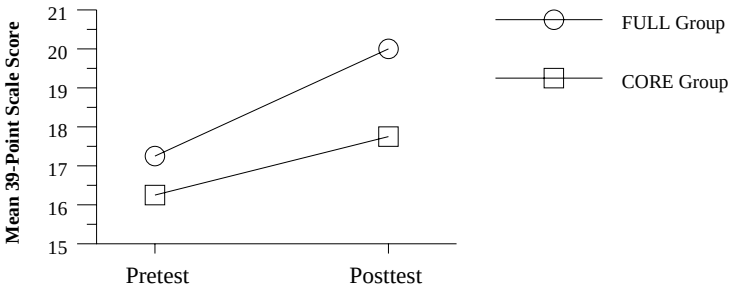

Figure 14: Pretest/Posttest overall scores

ultralight along that leg. The coach offered the following hint, making no mention of specific steps or locations:

> *Your plan uses the ultralight to rescue the eagle. Consider using a faster vehicle to reduce your plan's total time.*

After considering the coach's advice, student 23 was able to generate the optimal solution without further prompting.

Post-test Improvement

At the conclusion of this study the pretest was readministered as a posttest. The scores were then analyzed with a mixed model Analysis of Variance (ANOVA). The scores on the pre and post test were submitted to a preliminary mixed model Analysis of Variance (ANOVA) in which treatment group (core versus full) and gender were between-subject variables and time of test (pretest versus posttest) was the within subjects variable. There was no main or interaction effects of gender ($F(1,47) < 1$) so the data were reanalyzed using the gender variable.

Figure 14 shows that each group showed significant improvement from pre to post test, $F(1,47) = 13.75, p < 0.01$ for the main effect of time of test. There was no sig-

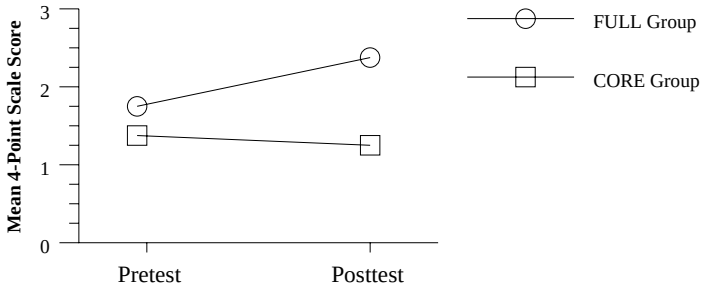

Figure 15: Pretest/Posttest rate-time-distance scores

nificant treatment by time of test interaction ($F(1,47) = 1.92, p = 0.172$) indicating the amount of improvement in each group was similar. Although there was significantly greater benefit to the full system compared to the core treatment on the overall score, examination of the means for the individual questions suggested that the full system students had outperformed the core system students on the range calculations and the identification of time-relevant variables questions (Questions 3 and 5 in Appendix A). The specific nature of the increased improvement is consistent with the differences between the complete and core systems. Students improved understanding of range may be attributed to the feedback received from the simulator, especially when the associated fuel feedback was turned on. Likewise, the simulation environment made visually explicit the role of relative speeds of objects and the distance between locations as the factors that determined the time required to complete a leg. Therefore, the simulation environment could also be expected to improve students' understanding of rate and distance as variables directly related to time computations. The cumulative scores for these two pre-post questions were submitted to ANOVA with treatment (core versus full) as the between-subjects factor and scores (pretest versus posttest) as the within-subjects factor. There was a significant interaction between treatment and pretest/posttest scores ($F(1,47) = 4.72, p < .05$). Figure 15 illustrates that the treatment groups performed similarly at pretest, but at posttest, full system students outperformed core system students on these two questions. These data support the claim that the timeline and simulation components help students understand some of the issues associated with range and time.

Discussion

This study provides evidence that the AdventurePlayer ILE assisted students in solving a complex trip planning problem. As predicted, the overall environment that included the timeline and simulation tools and the coaching components resulted in improved student performance in plan generation and evaluation. In the experimental study, 79% of the students using the full ILE generated complete plans compared to only 8% of the students who used the core system. Students using the timeline and simulator did better in computing individual leg times (92% vs. 12%) and the total plan time (79% vs. 0%). Furthermore, students using the full ILE did better at optimizing plans than core system students by considering multiple pilots, multiple vehicles, and multiple routes. Coaching of the full system

students who produced non optimal plans led to 75% producing an optimal plan. While overall posttest scores showed no significant effect, full system students did demonstrate significant improvements on posttest problems involving range calculations and the identification of time-relevant variables, the specific components addressed by the differences between the full and core systems.

These results are encouraging. The RBM challenge is quite complex, designed to be solved in groups over a period of days. Yet for students working individually on the AdventurePlayer ILE, 19 of 24 were able to derive a *complete* rescue plan, and 16 of 24 were able to derive the *optimal* rescue plan.

An interesting observation made in this study, was that students using the full AdventurePlayer environment made frequent use of the RTD scaffold provided by the timeline tool. This was an intended effect. One of the characteristics of an ILE is that it reduce the conceptual and computational burden of problem solving to give weaker students an opportunity to think about the problem and develop a solution without being overwhelmed by its complexity. As discussed, *scaffolds* can be gradually removed from the environment as students progress in their problem solving activities.

AILE and other Instructional Frameworks

AILE is an emergent design whereby student learning and problem solving occurs within an anchored instruction context, thereby affording guidance to students generative learning efforts and at the same time reducing the likelihood of students developing inert knowledge. The AILE design allows for ILE exploration which is guided by some aspect of an anchoring macrocontext. Likewise, students learn to use general tool components in specific problem solving situations. Thus both general and specific knowledge and skills are presented in a variety of contexts.

Two approaches that are similar to AILE are Increasingly Complex Microworlds (ICMs) and ThinkerTools. ICMs, initially proposed by Fischer, Brown, & Burton (1978), embody a basics first instructional approach whereby students begin by exploring relatively simple microworlds. Students are required to reach a specified level before they are allowed to explore more complex microworlds. As such, exploration and learning necessarily assumes a bottom up approach, where students successively learn and refine component problem solving tasks before they combine them to solve more complex tasks. The AILE approach follows a more flexible guided generation model of teaching (cf., CTGV 1991) that emphasizes both top down (complex problem solving in the anchoring macrocontext) and bottom up (working in a simplified domain) generative activities of students. As a result, AILE students are able to develop problem solving skills in a self-determined order, which increases student ownership of new knowledge. Because student discovery learning is anchored in a realistic situation, there is a greater likelihood the knowledge generated by the student will be active rather than inert (Bransford, Sherwood, & Hasselbring, 1988).

The ICM design has been expanded and deployed by White (1993) in her Thinker-

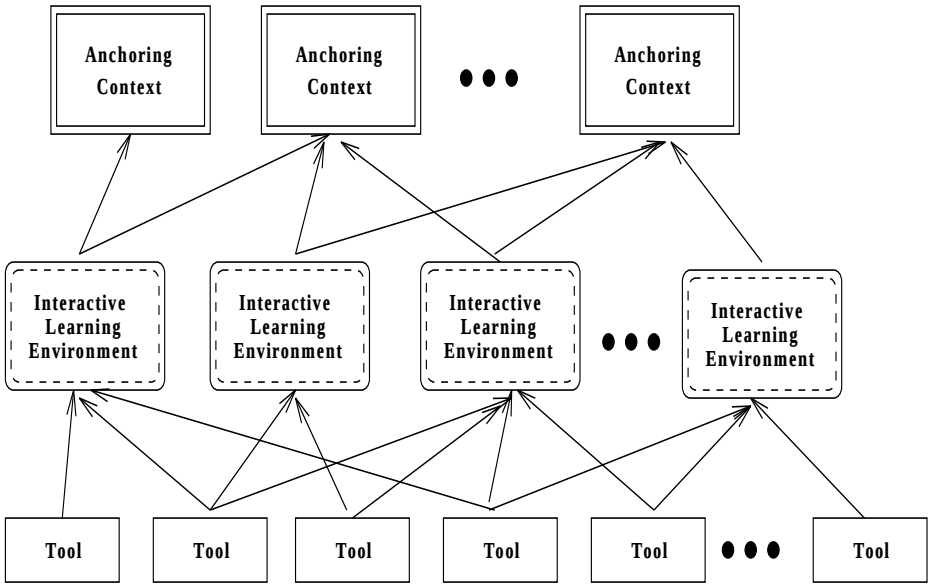

Figure 16: Generalized AILE architecture

Tools system. White's ThinkerTool design allows students to engage in discovery learning on a variety of topics within a simplified domain. As students' skills improve, more realistic constraints are added, and the complexity of the problem solving increases. AILE anchoring contexts on the other hand are sufficiently complex to facilitate prolonged exploration without any additional complexity. However, as with ThinkerTools, further complexity may be introduced regarding areas known to be problematic for learners. As an example, crosswinds may be introduced to the RBM adventure adding complexity to the flying calculations. There is already evidence that this added complexity will produce interesting learning opportunities (CTGV, 1993).

The Next Steps: Generalizing the ILE Concept

The success of the ILE concept in the context of the RBM adventure naturally leads to the question of how the AILE concept may be extended to an encompassing instructional architecture that integrates anchored instruction with constructivist educational theories. The goal is that the computer-based AILE architecture incorporate the design principles discussed earlier to facilitate generative sustained learning of active participants while also providing the necessary guidance and assistance for less motivated learners.

A generalization of the RBM AILE of Figure 1 is illustrated in Figure 16. There are several ways ILE's and tools can be arranged in the AILE architecture. One is to help students develop progressively more active knowledge by allowing them to encounter tools and concepts in a wide variety of anchoring contexts. For example, students can learn about the "rate of travel", i.e., speed of vehicles such as planes versus trucks in the RBM context. They can then transfer to another Jasper adventure, such as *The Big Splash* (CTGV, 1997;

Vye, et al., 1997) and learn about rate in the context of time required to fill a pool of water, and the price per unit volume. As the concept of *rate* is introduced in multiple contexts, students can see that similar tools (e.g., graphing tools and specialized calculators) can be used across these contexts. Examples of the selection and then invention of tools in non-ILE environments are discussed in Bransford, et al. (in press). This research in now being extended to designing and developing computer-based ILE environments that assist students in these tasks.

Another way to arrange ILE's and tools in the general AILE configuration is to approach each anchor from multiple perspectives. For example, in addition to mathematics, RBM can be explored from the perspective of science (e.g., endangered species, principles of flight) and social studies (the evolution of laws to protect endangered species). Students can then transfer to other Jasper adventures, such as *Bridging the Gap* (see CTGV, 1997), where similar perspectives can be brought to bear on the adventure. Over time, students learn to use, adapt, and invent tools that have multiple applications.

A key feature that characterizes all of the ILE's is that problem solving methods are instantiated by a set of tools. Tools are the most general components in the AILE architecture in the sense that they deal with basic problem solving mechanisms, such as calculators for computations, equation solvers in algebra, graphing tools for studying functional relations, and planning notebooks for describing sequences and sets of activities that collectively achieve a goal. They are invoked by students from within a particular ILE to help solve particular problems linked to the overall challenge. For example, the timeline tool used in the AdventurePlayer trip planning ILE would also be useful in problem solving situations that require scheduling of resources and events over time to complete a particular task. A general graphing tool can be used in many ILE environments for studying functional relations, such as the rate problems discussed above.

Overall, the Jasper project is tackling a number of innovative ideas in instruction and learning, curriculum design, thinking and reasoning in complex problem solving, and formative assessment. AdventurePlayer has been a first effort to develop a computer-based exploratory environment in accordance with the four design principles of ILEs. Future work will focus on the development of other ILEs and tools associated with the other anchors of the Jasper adventures. Experiments will involve more systematic studies of transfer between episodes and domains, and the development of more sophisticated formative assessment schemes to aid active learning processes.

Acknowledgments: This research is partially supported by grants from the National Science Foundation (MDR-9252990). We acknowledge the contributions of a number of other CTGV members, in particular Sashank Varma, Mitch Nathan, Susan Williams, Stephen Owens, and Nancy Vye.

References

[1] Anderson, J., Boyle, C., Corbett, A., & Lewis, M. (1990) Cognitive modeling and

intelligent tutoring, *Artificial Intelligence,*, 42, pp. 7-49.

[2] Bransford, J., Sherwood, R., and Hasselbring, T. (1988) The video revolution and its effects on development: Some initial thoughts. In G. Foreman and P. Pufall (eds.), *Constructivism in the computer age,* pp. 173-201, Lawrence Erlbaum Associates, Hillsdale, NJ.

[3] Bransford, J., Goldman, S., and Vye, N. (1991) Making a difference in people's abilities to think: Reflections on a decade of work and some hopes for the future. In L. Okagaki and R.J. Sternberg (eds.), *Directors for development: Influences on children,* pp. 147-180, Lawrence Erlbaum Associates, Hillsdale, NJ.

[4] Bransford, J., Zech, L., Schwartz, D., Barron, B., Vye, N., & the Cognition and Technology Group. (1996) Fostering mathematical thinking in middle school students: Lessons from research. in *The Nature of Mathematical Thinking,* R.J. Sternberg and T. Ben-Zeev (eds.), pp. 203-250, Lawrence Erlbaum Assoc., Mahwah, NJ.

[5] Bransford, J., Zech, L., Schwartz, D., Barron, B., Vye, N., & the Cognition and Technology Group. (in press) Designs for environments that foster mathematical thinking. in *Symbolizing, Communicating and Mathematizing: Perspectives on Discourse, Tools, and Instruction Design.* T. Cobb (ed.).

[6] Brown, A. & Palinscar, A. (1989) Guided, cooperative learning and individualized knowledge acquisition. In Resnick, L.(ed.), *Knowing, Learning, and Instruction,* Hillsdale, NJ: Lawrence Erlbaum Associates.

[7] Brown, J.S., Collins, A., & Duguid, P. (1989) Situated cognition and the culture of learning. *Educational Researcher*, 18, pp. 32-41.

[8] Burton, R., & Brown, J. (1982) An investigation of computer coaching for informal learning activities. In Sleeman, D. & Brown, J. (Eds.) *Intelligent Tutoring Systems,* Academic Press, London.

[9] Burton, R., Brown, J., & Fischer, G. (1984) Analysis of skiing as a success model of instruction: manipulating the learning environment to enhance skill acquisition. In Rogoff and Lave (Eds.) *Everyday cognition: its development in social context.* Harvard University Press, Cambridge, MA.

[10] Chenoweth, S.V. (1991) On the NP-Hardness of Blocks World, *Proc. AAAI-91,* AAAI Press, Menlo Park, CA, pp. 623-627.

[11] Cognition and Technology Group at Vanderbilt. (1990) Anchored instruction and its relationship to situated cognition. *Educational Researcher,* 19(6), pp. 2-10,

[12] Cognition and Technology Group at Vanderbilt. (1991) Technology and the design of generative learning environments, *Educational Technology*, 31, pp. 34-40.

[13] Cognition and Technology Group at Vanderbilt. (1992(a)) An anchored instruction approach to cognitive skills acquisition and intelligent tutoring. In J. W. Region & V. J. Shute (Eds.), *Cognitive approaches to automated instruction,* Hillsdale, NJ: Lawrence Erlbaum Associates, pp. 135-170.

[14] Cognition and Technology Group at Vanderbilt. (1992(b)) The Jasper experiment: An exploration of issues in learning and instructional design, *Educational Technology Research and Development*, 40, pp. 65-80.

[15] Cognition and Technology Group at Vanderbilt. (1992(c)) The Jasper series as an example of anchored instruction: Theory, program description, and assessment data. *Educational Psychologist,* 27, pp. 291-315.

[16] Cognition and Technology Group at Vanderbilt. (1993) The Jasper series: Theoretical foundations and data on problem solving and transfer. In L.A. Penner, G. M. Batsche, H. M. Knoff, & D. L. Nelson (Eds.), *The challenge in mathematics and science education: Psychology's response,* Washington, DC: American Psychological Association, pp. 113-152.

[17] Cognition and Technology Group at Vanderbilt. (1994) From visual word problems to learning communities: Changing conceptions of cognitive research. In K. McGilly (Ed.), *Classroom lessons: Integrating cognitive theory and classroom practice,* Cambridge, MA: MIT Press/Bradford Books, pp. 157-200.

[18] Cognition and Technology Group at Vanderbilt. (1997) The Jasper Project: Lessons in Curriculum, Instruction, Assessment, and Professional Development. Lawrence Erlbaum Associates, Inc., Mahwah, NJ.

[19] Crews, T. & Biswas, G. (1993(a)) A tutor for trip planning and mathematical problem solving, *Proc. World Congress on AI in Education,* Edinburgh, Scotland, pp. 346-353.

[20] Crews, T. & Biswas, G. (1993(b)) Towards an optimal planner for tutoring systems, *Proc. 5th Midwest AI and Cognitive Science Conference,* Chesterton, IN, pp. 46-52.

[21] Dewey, J. (1933) How we think, a restatement of the relation of reflective thinking to the educative process. Heath, Boston, MA.

[22] Fischer, G., Brown, J., & Burton, R. (1978) Aspects of a theory of simplification, debugging, and coaching, *Proc. Second Annual Conf. of the Canadian Society for Computational Studies of Intelligence,* Toronto, CA, pp. 139-145.

[23] Goldman, S. R., Vye, N. J., Williams, S. M., Rewey, K., Pellegrino, J. W., & the Cognition and Technology Group at Vanderbilt. (1991) Solution space analyses of the Jasper problems and students' attempts to solve them. Paper presented at the *American Educational Research Association,* Chicago, IL.

[24] Gupta, N. & Nau, D. (1991) Complexity results for blocks world planning, *Proc. Ninth National Conference on Artificial Intelligence (AAAI)*, Los Altos, CA:Morgan Kaufmann, pp. 629-633.

[25] Fikes, R. & Nilsson, N.J. (1971) STRIPS: A new approach to the application of theorem proving to problem solving, *Artificial Intelligence*, 2, pp. 189-208.

[26] Goldman, S.R., et al. (1996) Anchoring science instruction in multimedia learning environments. In S. Vosniadou, E. De Corte, R. Glaser, and H. Mandl (eds.), *International perspectives on the design of technology supported learning environments,* pp. 257-284, Lawrence Erlbaum Associates, Hillsdale, NJ.

[27] Hart, P., Nilsson, N., & Raphael, B. (1968) A formal basis of the heuristic determination of minimum cost paths. *IEEE Transactions on SSC* 4:100-107.

[28] Lajoie, S. (1993) Computer environments as cognitive tools for enhancing learning, In Lajoie, S., & Derry, S. (Eds.): *Computers as Cognitive Tools,* Hillsdale, NJ: Lawrence Erlbaum Assoc.

[29] Larkin, J., & Simon, H. (1987) Why a diagram is (sometimes) worth ten thousand words. *Cognitive science, 11,* pp. 65-99.

[30] Nathan, M. & Resnick, L. (1993) Less may be More: Unintelligent tutoring based on psychological theories and experimentation, Vosniadou, De Corte, Glaser, & Mandl (Eds.) *Psychological and Educational Foundations of Technology-Based Learning Environments.*

[31] Nickerson, R.S. (1988) On improving thinking through instruction. *Review of Research in Education,* 15, pp. 3-57, 1988.

[32] Papert, S. (1980) *Mindstorms: Children, computers and powerful ideas.* New York: Basic Books.

[33] Resnick, L.B. (1987) *Education and learning to think.* National Academy Press, Washington, D.C.

[34] Resnick, L.B. and Klopfer, L.E. (eds.). (1989) *Toward the thinking curriculum: Current cognitive research.* ASCD, Alexandria, VA.

[35] Reusser, K. (1993) Tutoring system and pedagogical theory. In Lajoie & Derry (Eds.) *Computers as Cognitive Tools.* Hillsdale, NJ: Lawrence Erlbaum Assoc.

[36] Roschelle, J. (1990) Designing for conversations. *AAAI Spring Symposium on Knowledge-Based Environments for Learning and Teaching,* Stanford, CA.

[37] Sacerdoti, E. (1974) Planning in a hierarchy of abstraction spaces, *Artificial Intelligence*, 5, pp. 115-135, Amsterdam, Netherlands: North Holland.

[38] Scardamalia, M., & Bereiter, C. (1991) Higher levels of agency for children in knowledge building: A challenge for the design of new knowledge media, *Journal of the Learning Sciences*, 1, pp. 37-68.

[39] Sherwood, R.D., Petrosino, A.J., Lin, X., Lamon, M., & the CTGV. (1995) Problem based macrocontexts in science instruction: Theoretical basis, design issues, and the development of applications. In *Towards a cognitive science perspective for scientific problem solving,* D. Lavoie (ed.), pp. 191-14, National Association for Research in Science Teaching, Manhattan, KS.

[40] Shute, V.J., Glaser, R., & Raghavan, K. (1989) Inference and Discovery in an exploratory laboratory. In P.L. Ackerman, R.J. Sternberg, & R. Glaser (eds.), *Learning and individual differences,*, New York: W.H. Freeman, pp. 279-326.

[41] Spiro, R., Feltovich, P., Jacobson, M., & Coulson, R. (1991) Cognitive flexibility, constructivism, and hypertext: random access instruction for advanced knowledge acquisition in ill-structured domains. *Educational Technology,*vol. 31 (5), pp. 24-33.

[42] Soraci, S.A., Jr., Franks, J.J., Bransford, J.D., Chechile, R.A., Belli, R.F., Carr, F., & Carlin, M.T. (1994) Incongruous item generation effects: A multiple-cue perspective. *Journal of Experimental Psychology: Learning, Memory, and Cognition,* vol. 20, pp. 1-12.

[43] von Glaserfeld, E. (1987) Learning as a constructive activity. In C. Janvier (ed.), *Problems of representation in the teaching and learning of mathematics,* pp. 3-17, Lawrence Erlbaum Associates, Hillsdale, NJ.

[44] Vye, N.J., Goldman, S.R., Voss, J.F., Hmelo, C., Williams, S., & CTGV. (1997) Complex mathematical problem solving by individuals and dyads. *Cognition and Instruction,* 15, pp. 435-484.

[45] Vygotsky, L. (1978) *Mind in Society: the development of higher psychological processes,*, Cambridge, MA: Harvard Univ. Press.

[46] Wenger, E. (1987) *Artificial Intelligence and Tutoring Systems.* Los Altos, CA: Morgan Kaufmann.

[47] White, B.Y. (1993) ThinkerTools; Causal models, conceptual change, and science education, *Cognition & Instruction*, 10, pp. 1-100.

[48] Williams, M., Hollan, J., & Stevens, A. (1981) An overview of STEAMER; an advanced computer-assisted instruction system for propulsion engineering, *Behavior Research Methods and Instrumentation,* 13(2), pp. 85-90.