International Journal of Artificial Intelligence in Education 15 (2005) xxx–xxx
IOS Press

The Andes Physics Tutoring System: Lessons Learned

Kurt VanLehn, **Collin Lynch**, *LRDC, University of Pittsburgh, Pittsburgh, PA, USA*
{VanLehn, collinl}@pitt.edu

Kay Schulze, *Computer Science Dept., US Naval Academy, Annapolis, MD, USA*
schulze@artic.nadn.navy.mil

Joel A. Shapiro, *Dept. of Physics and Astronomy, Rutgers University, Piscataway, NJ, USA*
Shapiro@physics.rutgers.edu

Robert Shelby, *Physics Department, US Naval Academy, Annapolis, MD, USA*

Linwood Taylor, *LRDC, University of Pittsburgh, Pittsburgh, PA, USA*
lht3@pitt.edu

Don Treacy, *Physics Department, US Naval Academy, Annapolis, MD, USA*
treacy@usna.edu

Anders Weinstein, *LRDC, University of Pittsburgh, Pittsburgh, PA, USA*
andersw@pitt.edu

Mary Wintersgill, *Physics Department, US Naval Academy, Annapolis, MD, USA*
mwinter@usna.edu

Abstract. The Andes system demonstrates that student learning can be significantly increased by upgrading only their homework problem-solving support. Although Andes is called an intelligent tutoring system, it actually replaces only the students' pencil and paper as they do problem-solving homework. Students do the same problems as before, study the same textbook, and attend the same lectures, labs and recitations. Five years of experimentation at the United States Naval Academy indicates that Andes significantly improves student learning. Andes' key feature appears to be the grain-size of interaction. Whereas most tutoring systems have students enter only the answer to a problem, Andes has students enter a whole derivation, which may consist of many steps, such as drawing vectors, drawing coordinate systems, defining variables and writing equations. Andes gives feedback after each step. When the student asks for help in the middle of problem-solving, Andes gives hints on what's wrong with an incorrect step or on what kind of step to do next. Thus, the grain size of Andes' interaction is a single *step* in solving the problem, whereas the grain size of a typical tutoring system's interaction is the *answer* to the problem. This report is a comprehensive description of Andes. It describes Andes' pedagogical principles and features, the system design and implementation, the evaluations of pedagogical effectiveness, and our plans for dissemination.

Keywords:

INTRODUCTION

For almost as long as there have been computers, there have been computer-based tutoring systems. One type of system, called an intelligent tutoring system, has produced impressive gains in laboratory studies (Shute & Psotka, 1996). Nonetheless, with only a few exceptions, intelligent tutoring systems are seldom used in ordinary courses. We believe the lack of acceptance is due in part to their tendency to require extensive modification of the course content. Building an intelligent tutoring system involves making detailed analyses of the desired thinking (called cognitive task analyses) so that the desired reasoning can be represented formally in the system and used to discriminate desired student thinking from undesirable student thinking. The cognitive task analysis often leads to insights into how to improve instruction, and the insights are incorporated in the tutoring system. Consequently, the tutoring system teaches somewhat different content than an ordinary version of the course. Instructors, alone or in committees, control the content of the course, and they may not wish to adopt these content changes.

One approach is to include the tutoring system as part of a broader reform of the instruction, and convince instructors that the whole package is worth adopting. This is the approach taken by Carnegie Learning (www.carnegielearning.com) with successful Cognitive Tutors. They sell a whole curriculum that is consistent with recommendations from national panels and incorporates instruction developed by award-winning teachers. The same approach has been used by successful industrial and military deployments of intelligent tutoring systems, such as the Radar System Controller Intelligent Training Aid (from Sonalysts; www.sonalysts.com), the Tactical Action Officer Intelligent Tutoring System (from Stottler-Henke; http://www.shai.com/solutions/training/taoits.htm), or the Aircraft Maintenance Team Training (from Galaxy Scientific; http://www.galaxyscientific.com/areas/training/its1.htm). The designers work with subject matter experts to devise both improved content and a tutoring system to go with it.

However, getting instructors and institutions to adopt curricular reforms is notoriously difficult, with or without an accompanying tutoring system. Scientific evidence of greater learning gains is only part of what it takes to convince stakeholders to change.

Moreover, the technology of intelligent tutoring systems does not in itself *require* content reform. It should be able to aid learning of almost any content.

The goal of the Andes project is to demonstrate that intelligent tutoring can be decoupled from content reform and yet still improve learning. This requires that Andes be *minimally invasive*. It should allow instructors to control the parts of the course that they want to control, and yet it should produce higher learning gains than ordinary courses.

One task that instructors seem happy to delegate is grading homework. In recent years, many web-based homework (WBH) grading services have become available. Ones that serve college physics courses (as does Andes) include WebAssign (www.webassign.com), Mastering Physics (www.masteringphysics.com), CAPA (homework.phys.utk.edu), Homework Service (hw.utexas.edu/overview.html), OWL (ccbit.cs.umass.edu/owl), WebCT (www.webct.com), Blackboard (www.blackboard.com) and WWWAssign (emc2.acu.edu/~schulzep/wwwassign). These WBH services have students solve problems assigned by instructors, so the instructors still have control over that important feature of their courses. Students enter their answers on-line, and the system provides immediate feedback on the answer. If the answer is incorrect, the student may receive a hint and may get another chance to derive the answer.

In principle, students could submit a whole derivation of their answer and get feedback on each step. For instance, students using the Language, Proof and Logic textbook (www-csli.stanford.edu/LPL/) may submit proofs in formal logic and receive feedback at the level of individual sentences in the proof. However, most WBH services have student submit only a simple answer, such as a number, a menu selection, or an algebraic formula.

These services have grown rapidly. Thousands of instructors have adopted them. Universities have saved hundreds of thousands of dollars annually by replacing human graders with these services (Dufresne, Mestre, Hart, & Rath, 2002). These trends suggest that unlike tutoring systems, homework helpers will soon be ubiquitous. They are minimally invasive. In particular, they can be used with traditional classes as well as ones where only a small portion of the homework problems are traditional, because instructors can author their own homework activities as well.

However, the impact of WBH on learning is not clear. On the one hand, there are two reasons why WBH might be better than paper-and-pencil homework (PPH). First, instructors often cannot afford to grade every PPH problem that they assign, so students may not do all their assigned homework. With WBH, every problem is graded, and the homework grades often count in the student's final grade, so students do more homework than they would with PPH. This suggests that WBH should produce more learning than PPH. Secondly, with WBH, students receive immediate feedback on their answers, which might motivate them to repair their derivations and hopefully the flaws in the knowledge that produced them. Of course, students could also look in the back of the book for the answers to some problems, but anyone who has graded homework knows that not all students do that, or if they do, they don't bother to correct their mistakes.

On the other hand, there is at least one reason why PPH might be better than WBH. With WBH, students only enter answers and not the derivation of those answers. When humans grade PPH, they often score the derivations and, at least in physics courses, producing a well-structured, principled derivation usually counts more than getting the right answer. This grading practice is intended to get students to understand the physics more deeply. Of course, PPH students do not always read the graders' comments on their derivations, but the fact that they know their derivations are being graded might motivate them to produce good ones anyway. Because the WBH systems cannot grade the student's derivations, students might try less hard to produce good derivations and thus learn more shallowly than they would with PPH.

Clearly, experiments are needed that compare WBH with PPH. Although there are many studies of WBH courses, they mostly report how the WBH service was used, the results of questionnaires, and correlations between WBH usage and learning gains. Only three studies of physics instruction have compared learning gains with WBH vs. PPH.

In the first study (Dufresne et al., 2002), the same professor taught a PPH class and two WBH classes in consecutive semesters. A subset of the exam problems were the same in all three classes, which allows them to be compared. One WBH class's mean exam score was 0.44 standard deviations higher than the PPH class, but the other WBH class's mean exam score was not significantly different from the PPH class. However, a simple explanation for these results may be that the PPH students didn't do much homework. For the PPH class, most assigned homework was not collected and graded, whereas for the WBH classes, all homework was scored by the WBH service and the scores counted in the student's final grade. Moreover, when students self-reported their homework times, the PPH student spent much less time than the

WBH students. Of the PPH students, 62% said they spent less than 2 hours per week and 4% said they spent more than 4 hours. Among students in the higher scoring WBH class, 11% reported spending less than 2 hours, and 46% said they spent more than 4 hours. Thus, it is likely that the gains in the WBH class were due at least in part to the requirement that students hand in homework, which resulted in them solving more problems, spending more time on task, and learning more.

In the second study (Pascarella, 2002) and the third study (Bonham, Deardorff, & Beichner, 2003), all the PPH problems were graded and the scores counted toward the student's course grade. In the Pascarella study, half the students in a large college physics class (500+) solved homework on a WBH while the other half solved problems on paper. After the midterm, the students switched. Those that used WBH now used PPH and vice versa. In the Bonham et al. study, two classes were used. Students in several sections of a calculus-based class received very similar homework assignments. Students in a PPH section and a WBH section of an algebra-based class received identical assignments. In both the Pascarella and Bonham et al. studies, dependent measures included multiple-choice and open-response exam questions, and the Force and Motion Concepts Inventory (Thornton & Sokoloff, 1998). When SAT and GPA (grade point average) were factored out, none of the measures showed a difference between PPH students and WBH students. In the Bonham et al. study, this included a rather detailed scoring of the open-response exam questions designed to detect benefits of using PPH, where students were required to show all their work and were graded more on the derivation than on the answer. A video-based longitudinal study of some of the Pascarella students (Pascarella, 2004) suggested that WBH encouraged shallower thinking than PPH while doing homework, but the sample size was too small to draw reliable conclusions.

This is moderately good news. The WBH answer-only format probably does not hurt students relative to the PPH format even though students only enter answers and not derivations when doing WBH. Perhaps WBH's immediate feedback compensates for its answer-only format. Most importantly, WBH allows all assigned problems to be graded. Thus, when WBH is compared to PPH courses that do not grade all assigned problems, WBH probably causes students to do more of their assigned problems and thus learn more.

The goal of the Andes project is to retain the minimal invasiveness of WBH, but increase the learning gains of students. The key idea is simply to have students enter their derivations just as they do with PPH, but Andes gives immediate feedback and hints as each step is entered. Evaluations indicate that Andes homework has elicited more learning from students than PPH. This document describes how Andes behaves, its design and implementation, and its evaluations.

A brief history

The Andes project originated with an Office of Naval Research management initiative to forge close relationships between ONR and the Navy's academic institutions. In particular, there was an interest in trying out artificially intelligent tutoring technology, a longstanding research area for ONR, at the Naval Academy. Dr. Susan Chipman of ONR supported a summer symposium series for interested Academy faculty in which many speakers from the intelligent tutoring research community spoke. Two projects resulted, an extensive implementation of Ken Forbus' existing CyclePad software in the thermodynamics curriculum at the Academy, and a more

ambitious project to build a new physics tutor on the foundations of the Cascade and Olae projects, while also conducting research on issues of instructional strategies in intelligent tutors.

Cascade was a rule-based cognitive model of physics problem solving and learning (VanLehn, 1999; VanLehn & Jones, 1993b, 1993c; VanLehn, Jones, & Chi, 1992). It provided one key ingredient of an intelligent tutoring system: a cognitive task analysis in the form of highly detailed rules that were capable of solving many physics problems in a variety of correct and incorrect ways.

Olae, which was built on top of Cascade's cognitive model, was an on-line assessment system (Martin & VanLehn, 1995a, 1995b; VanLehn & Martin, 1998; VanLehn & Niu, 2001). It provided two more key ingredients: a graphical user interface and a student modeling module.

In order to convert these ingredients into an intelligent tutoring system, we need to add two new capabilities: feedback on student work and hints. Beyond these technical additions, the project needed to include physics instructors who were dedicated to designing the tutoring system and evaluating it in their classes. A team was assembled including the four Naval Academy professors listed above and a collection of post-docs, programmers and graduate students at LRDC, some of whom are listed above.[1]

The main challenge was to create a minimally invasive tutoring system. Unlike the highly successful intelligent tutoring projects at CMU (Anderson, Corbett, Koedinger, & Pelletier, 1995; Koedinger, Anderson, Hadley, & Mark, 1997), the Andes project was not empowered or interested in changing the curriculum significantly. The Andes instructors only taught a portion of the USNA physics course. Their sections had to use the same textbooks, same final exams, similar labs and similar lectures as the non-Andes sections of the course. Thus, the challenge was to improve student learning while coaching only the student's homework. That is, the tutoring had to be minimally invasive. While challenging, we hoped that this would facilitate incorporating Andes into existing physics classes around the world.

The first version, Andes1, was based on Olae's student modeling technique. The technique used Bayesian networks to infer the probability of mastery of each rule. That is, a rule was assumed to be in one of two states: mastered or unmastered. For a given student, the probability of the rule being in the mastered state reflected all the evidence that had been gathered so far about this student's behavior. Initially, all Andes1 knew about a student is that the student is a member of some population (e.g., US Naval Academy freshmen), and this established the prior probability of each rule. As the student used Andes1, it noted which actions tended to be done by the student, and this increased the probabilities on the rules that derived those actions. For actions that could be done but were not, Andes1 gradually reduced the probability on the rules that derived those actions. The Bayesian networks handled these probabilistic calculations efficiently and correctly.

At the time, Andes1 was one of the first large scale applications of Bayesian networks, and arguably the first application of Bayesian networks to intelligent tutoring systems. The same technique was used in the Self-Explanation Coach (Conati & VanLehn, 2000; Conati & VanLehn, 2001) in order to assess rule mastery by observing which lines in a worked example were studied by students. As a early application of Bayesian networks to student modeling,

---

[1] Andes project "alumni" include Drs. Patricia Albacete, Cristina Conati, Abigail Gertner, Zhendong Niu, Charles Murray, Stephanie Siler, and Ms. Ellen Dugan.

many technical challenges were discovered and surmounted. These results are described in (Conati, Gertner, & VanLehn, 2002).

Andes1 was evaluated twice at the Naval Academy, with strongly encouraging results. On the second evaluation, the mean post-test exam score of the students who did their homework on Andes was approximately 1 standard deviation higher than the mean exam score of students who did the same homework with pencil and paper (Schulze et al., 2000). We could have stopped there, but we wanted to find out why Andes1 was so effective, and in particular, whether its novel student modeling technique was accurate and perhaps even partially responsible for its success.

We conducted several studies to analyze the effectiveness of Andes1 (VanLehn et al., 2002; VanLehn & Niu, 2001). The bottom line was that the Bayesian student modeling technique was *not* the source of Andes1's power. It was indeed a highly accurate assessment of student mastery (VanLehn & Niu, 2001), but the rest of the tutoring system didn't really have much use for such an assessment. Tutoring systems often use assessments to decide which problem a student should do next, or whether the student has done enough problems and can go on to the next chapter. However, Naval Academy students were assigned specific problems for homework, so Andes did not need to select homework problems, nor was it empowered to decide whether the student should go on to the next chapter. So Andes was creating an assessment, but not using it to make the usual sorts of decisions one would make with such an assessment.

The studies also reveal some significant pedagogical flaws in the hints given to students. In order to revise the hint system more easily, we eliminated the Bayesian networks while retaining the non-probabilistic aspects of the student modeling module. We also incorporated two new mathematical algorithms that vastly improved the combinatorics and simplicity of the system. (Shapiro, 2005) The new system, Andes2, also featured a new, more concise knowledge representation and many other improvements, just as one would expect when redesigning a prototype from scratch.

Andes2's initial evaluation was promising, so we invested several years into scaling it up. It now covers most of the Autumn semester physics course (mostly mechanics) and about half the Spring semester (mostly electricity and magnetism). At this writing, Andes has 356 problems which are solved by a knowledge base of 550 physics rules.

Within the last year, many "convenience" features were added, such as electronic submission of homework and automated problem scoring. These make Andes2 competitive with WBH grading services. Andes is now freely available, and may be downloaded or used as a web-based service (http://www.andes.pitt.edu).

A preview of the paper

It is important to realize that unlike many of the articles, this one is not reporting tests of a hypothesis or a new technology. Granted, we had to invent some new technology en route, and the evaluation of Andes is similar to the test of a hypothesis, but the ultimate purpose of the project was to see if a minimally invasive tutoring technology could increase learning in real-world classes. Thus, most of what we learned in this process is applied knowledge: what will students and instructors accept; what kinds of hints work; which algorithms scale and which do not; how to conduct a fair field evaluation; etc. This article attempts to summarize what has been learned from this effort. It has the following sections:

- Andes2 from the point of view of the student—what it looks like, what it does and what role it plays in the physics course.
- The knowledge and skills that Andes2 teaches.
- The pedagogical features of Andes2, including both mundane ones and ones requiring AI.
- The technology of Andes, including important knowledge structures and algorithms.
- The evaluations of Andes at the United States Naval Academy.
- The lessons learned.

THE FUNCTION AND BEHAVIOR OF ANDES

In order to make Andes minimally invasive, we tried to make its user interface as much like pencil and paper homework (PPH) as possible. A typical physics problem and its solution on the Andes screen are shown in Figure 1. Students read the problem (top of the upper left window), draw vectors and coordinate axes (bottom of the upper left window), define variables (upper right window) and enter equations (lower right window). These are actions that they do when solving physics problems with pencil and paper.

Unlike PPH, as soon as an action is done, Andes gives immediate feedback. Entries are colored green if they are correct and red if they are incorrect. This is called flag feedback (Anderson et al., 1995). In Figure 1, all the entries are green except for equation 3, which is red.

Also unlike PPH, variables are defined by filling out a dialogue box, such as one shown in Figure 2. Vectors and other graphical objects are first drawn by clicking on the tool bar on the left edge of Figure 1, then drawing the object using the mouse, then filling out a dialogue box like the one in Figure 2. Filling out these dialogue boxes forces students to precisely define the semantics of variables and vectors. PPH does not require this kind of precision, so students often just use variables in equations without defining them. If students include an undefined variable in an Andes equation, the equation turns red and a message box pops up indicating which variable(s) are undefined.

Andes includes a mathematics package. When students click on the button labeled "x=?" Andes asks them what variable they want to solve for, then it tries to solve the system of equations that the student has entered. If it succeeds, it enters an equation of the form <variable> = <value>. Although many students routinely use powerful hand calculators and computer-based mathematics packages, such usage requires copying the equations from Andes to their system and back. Andes eliminates this tedious and error-prone copying process. This is one reason that Andes is popular with students. Nonetheless, instructors can turn this feature off.

Andes provides three kinds of help:

- Andes pops up an error messages whenever the error is likely to be a slip. That is, the error is probably due to lack of attention rather than lack of knowledge (Norman, 1981). Typical slips are leaving a blank entry in a dialogue box, using an undefined variable in an equation (which is usually caused by a typographical error), or leaving off the units of a dimensional number. When an error is not recognized as a slip, Andes merely colors the entry red.

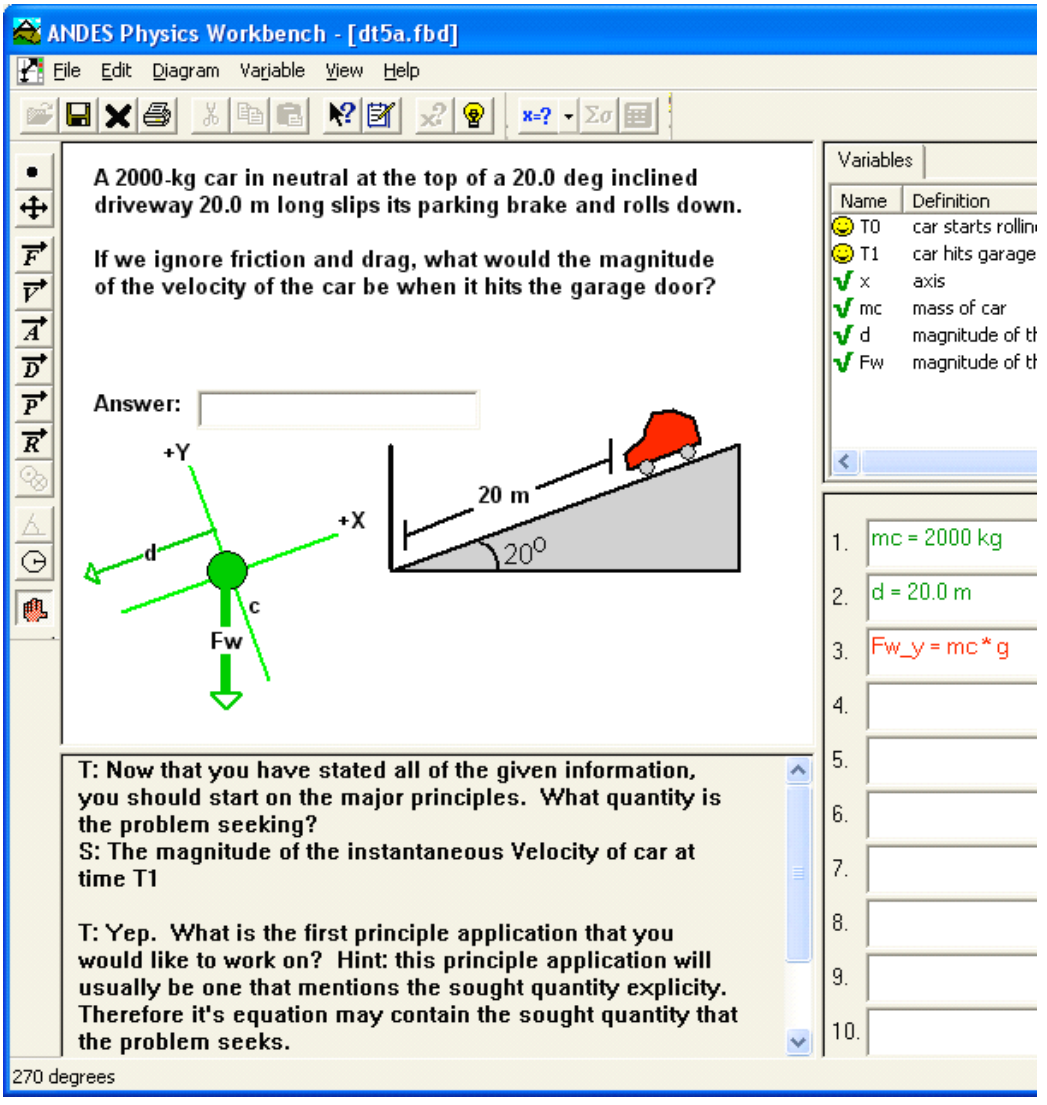

Fig. 1. The Andes screen (truncated on the right).

- Students can request help on a red entry by selecting it and clicking on a help button. Since the student is essentially asking, "what's wrong with that?" we call this *What's Wrong Help*.
- If students are not sure what to do next, they can click on a button that will give them a hint. This is called *Next Step Help*.

Thus, for errors that are likely to be careless mistakes, Andes gives unsolicited help, while for errors where some learning is possible, Andes gives help only when asked. This policy is intended to increase the chance that students will repair substantive errors without asking for

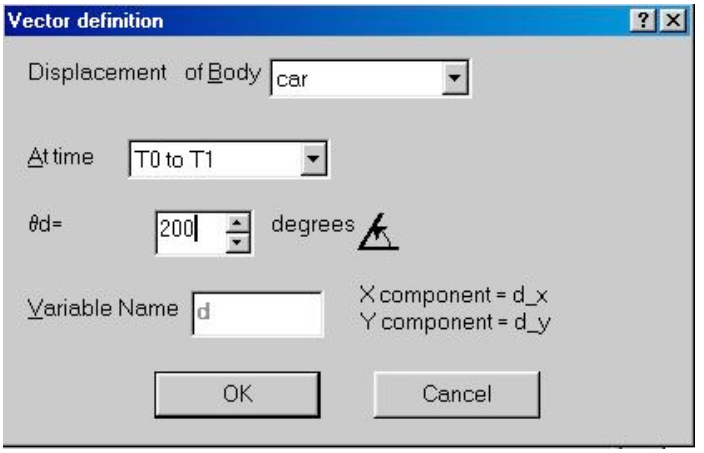

Fig. 2. A dialogue box for drawing a vector.

help. Self-repair may produce more robust learning, according to constructivist theories of learning (e.g., Merrill, Reiser, Ranney, & Trafton, 1992).

What's Wrong Help and Next Step Help usually generate a hint sequence. The hints are printed in the lower left window. In order to force students to attend to it, the other windows deactivate and turn gray. This avoids the problem found in eye-tracking studies of other tutoring systems where students simply would not look at a hint even though they knew it was there (Anderson & Gluck, 2001).

Most hint sequences have three hints. As an illustration, suppose a student who is solving Figure 1 has asked for What's Wrong Help on the incorrect equation *Fw_x = -Fs*cos(20 deg)*. These are the three hints that Andes gives:

- Check your trigonometry.
- If you are trying to calculate the component of a vector along an axis, here is a general formula that will always work: Let θV be the angle as you move counterclockwise from the horizontal to the vector. Let θx be the rotation of the x-axis from the horizontal. (θV and θx appear in the Variables window.) Then: V_x = V*cos(θV-θx) and V_y = V*sin(θV-θx).
- Replace cos(20 deg) with sin(20 deg).

After the first two hints, Andes displays two buttons labeled "Explain more" and "OK." If the student presses on "Explain more," they get the next hint in the sequence. If the "OK" button is pressed, the problem solving windows become active again, the lower left window becomes gray, and the student resumes work on the problem.

This three-hint sequence is typical of many hint sequences. It is composed of a pointing hint, a teaching hint and a bottom-out hint:

The pointing hint, "Check your trigonometry," directs the students' attention to the location of the error. If the student knows the appropriate knowledge and the mistake is due to carelessness, then the student should be able to pinpoint and correct the error given such a hint (Hume, Michael, Rovick, & Evens, 1996; Merrill et al., 1992).

The teaching hint, "If you are trying to calculate…," states the relevant piece of knowledge. We try to keep these hints as short as possible, because students tend not to read long hints

(Anderson et al., 1995; Nicaud, Bouhineau, Varlet, & Nguyen-Xuan, 1999). In other work, we have tried replacing the teaching hints with either multimedia (Albacete & VanLehn, 2000a, 2000b) or natural language dialogues (Rose, Roque, Bhembe, & VanLehn, 2002). These more elaborate teaching hints significantly increased learning in laboratory settings, but have not been tried in the field. Although a teaching hint allows "just in time learning," real-world students are sometimes more concerned about getting their homework done than with learning (Dweck, 1986).

The bottom-out hint, "Replace cos(20 deg) with sin(20 deg)," tells the student exactly what to do. Because Koedinger and Anderson (1993) found that their tutoring system's bottom-out hints often left students uncertain about what to enter, we have tried to make Andes' bottom-out hints as specific and clear as possible.

Andes sometimes cannot infer what the student is trying to do, so it must ask before it can give help. An example is shown in Figure 1. The student has just asked for Next Step Help and Andes has asked, "What quantity is the problem seeking?" Andes pops up a menu or a dialogue box for students to supply answers to such questions. The students' answer is echoed in the lower left window.

As the student solves a problem, Andes computes and displays a score. Most homework helpers make the score a function of the correctness of the student's answer and the number of hints received. Andes puts little weight on answers, because it provides such good help that students almost always get the answer right. Instead, it measures the proportion of entries that were made correctly (green). Counting hints tends to discourage them, so Andes only subtracts points when students ask for bottom-out hints. In addition to making the score a function of degree of correctness and number of hints, Andes tries to encourage good problem solving habits by awarding points for entering certain information explicitly. For instance, students get points for entering equations for fundamental principles that do not have given values or other values substituted into them. The overall score on a problem is continually displayed in the lower right corner. If students print their solution or use print preview, they see the subscores from which their score was computed.

Andes can be used both offline and online. When used offline, students print their homework and hand it in on paper. Instructors who grade such homework save time because they have Andes' subscores to start with and it is easier to read printed equations than handwritten ones. When Andes is used online, students submit their problem solutions via the Web. The Andes scores are sent to the instructor's grade book, which looks and acts like a spreadsheet. The cells contain the student's score on a problem as computed by Andes; clicking on the cell displays the student's solution. The grade book can be dumped to a tab-delimited file that can be read by commercial spreadsheets and databases.

WHAT STUDENTS SHOULD LEARN

Elementary physics is often considered to be a nomological (law-based) science in that many empirical patterns of nature can be explained with a few principles, such as Newton's laws, the law of Conservation of Energy, Maxwell's equations, etc. An explanation is just a deduction—an informal proof. Thus, it is unsurprising that all AI systems that solve physics problems, including Andes, consider a solution to be a proof (Bundy, Byrd, Luger, Mellish, &

Palmer, 1979; de Kleer, 1977; Elio & Scharf, 1990; Lamberts, 1990; Larkin, Reif, Carbonell, & Gugliotta, 1988; McDermott & Larkin, 1978; VanLehn et al., 2004; VanLehn & Jones, 1993b).

However, they have also found that the principles listed in textbooks are not the only ones needed for creating these proofs. Some of the essential inferences are justified by "principles" that never appear in textbooks. For instance, below are all the equations needed to solve the problem of Figure 1 and their justifications.

Fw_x = mc*a_x	Newton's second law along the x-axis
v2_x^2=v1_x^2+2*a_x*d_x	constant acceleration, so vf^2-vi^2=2ad
Fw = mc*g	Weight = mass * g, i.e., W=m*g
Fw_x = -Fw*sin(20 deg)	projection of Fw onto the x-axis
a_x = -a	projection of a onto the x-axis
v2_x = -v2	projection of v2 onto the x-axis
d_x = -d	projection of d onto the x-axis
g = 9.8 m/s^2	car is assumed to be near earth
v1_x = 0	For objects at rest, velocity components = 0
mc = 2000 kg	given
d = 200 m	given

The first equation is justified by Newton's second law, which all instructors would agree is a major physics principle. The second line is justified by an equation of translational kinematics that doesn't have a name but appears widely in textbook summaries of kinematics principles. The other equations are justified by "principles" that instructors considered less important. Some, such as "weight = mass * g," are special cases of more general laws. Others, such as the projection formulae for vectors, are considered parts of mathematics. Still others, such as the one that justifies v1_x=0, are considered common sense entailments of a proper understanding of physics concepts. Most instructors would object to our use of the term "minor principles" for these special case, mathematical or common sense justifiers. However, to an AI program, all these pieces of knowledge act just like the major principles—they justify inferences about physical situations.

If students are going to solve problems, then they must master all principles, both major and minor. This is the first instructional objective of Andes. Andes contains explicit representations of minor principles, and it will discuss them with students during hints. For instance, many textbooks don't mention that the magnitude of a tension force at one end of an ideal string is equal to the magnitude of the tension force at the other end of the string, so Andes explicitly teaches this minor principle if necessary.

All physics problem solving systems have knowledge bases that include principles, and in fact, they tend to agree on what those principles are, even the minor ones. However, if such a system knows only principles, then it may produce a huge list of equations, most of which are unnecessary, as it solves a problem. Experts and competent students clearly have some other knowledge than principles, and they use this knowledge to constrain their reasoning so that they produce just the principle applications required for solving the problem.

Unfortunately, there is as yet no consensus about the nature of this extra knowledge. Different AI models have used different kinds of knowledge to constrain the application of principles. Some use hand-authored search heuristics that could potentially be taught explicitly to students (Bundy et al., 1979; VanLehn et al., 2004). Some use search heuristics that are learned by machine learning algorithms and are completely inappropriate for teaching to students

(Lamberts, 1990; VanLehn & Jones, 1993c). Others use analogies to solve problems (VanLehn & Jones, 1993b) or to generalized problem solutions called cases (Elio & Scharf, 1990) or schemas (Larkin et al., 1988). All these systems are equally expert at solving problems. One might hope that psychological evidence, such as talk-aloud protocols, might help us determine which kind of knowledge is used by human experts. Thus, let us take a brief tour of the physics expert-novice literature.

The literature suggests that the main difference between expert and novice physics problems solvers is that experts can identify the major principles required to solve a problem before they solve it on paper. Here are representative findings:

When Chi, Feltovich and Glaser (1981) asked experts to group problems together that were "similar," experts would put Figure 1 in with other problems that used Newton's second law in their solution, whereas novices would group it with other problems that had inclined planes. That is, experts grouped problems according to the major principle applications in their solutions, whereas novices grouped problems according to surface features.

When Chi et al. (1981) asked subjects to state their basic approach to a problem like the one in Figure 1, the experts would say something like "I'd apply Newton's second law and the kinematics of constant acceleration," whereas novices would say something like, "I'd draw some vectors and write down the equations, being careful not to confuse sine and cosine." That is, experts mentioned the major principle applications and the novices did not.

Larkin (1983) and Priest (1992) studied the order in which equations were written down during paper and pencil problem solving. Experts tended to write equations in groups, where each group contained a single major principle application and several minor ones that were closely related to it. Novices tended to write equations in algebraic orders; equations were adjacent if they shared variables. This suggests that experts had planned a solution in terms of major principles, whereas novices were following some kind of search strategy that treated all equations equally, regardless of whether they came from applying a major principle or a minor one.

All these findings are consistent with the hypothesis that experts can study a problem for a while (Chi, Feltovich and Glaser's experts averaged 45 seconds per problem) and decide which major principles are required for its solution. Moreover, they can do so "in their heads" without writing anything down. Unfortunately, this result doesn't help us decide which of the cognitive models of expertise is best, since they can all be extended to identify major principles "in their heads."

Because there is no consensus on what experts know besides the principles, Andes does not explicitly teach any such knowledge. This is required by its goal of being minimally invasive. In particular, it does not teach search heuristics, cases or schemas. However, there is consensus that experts can identify the major principles required to solve a problem, even though it may take them almost a minute to do so. Thus, Andes tries to encourage students to think about problem solutions in terms of their major principles. This is its second instructional objective.

This instructional objective is well known in physics education research, and is often described as "achieving a *conceptual* understanding of physics problem solving." Students tend to get buried in the algebraic details of solutions. Physics educators would like students to be able to rise above the details, to see the major principles and concepts, and to understand how the physics constrains the behavior of the physical system.

Andes has many other instructional objectives, but they are less important than getting students to master the principles and to think about solutions in terms of major principles. The secondary instructional objectives will be introduced in the next section, when discussing the features that address them.

PEDAGOGICAL FEATURES

This section lists features of Andes that are intended to either facilitate student learning or to facilitate the instructor's use of the system. Although the evaluations suggest that Andes is effective, we do not know which of the many features listed below is responsible for its pedagogical power. That would be a fit topic for future research.

Features common to all homework helpers

This first section lists features that any homework helper should have, where a homework helper is a system that facilitates learning from homework without attempting to constrain or replace the rest of the course's learning activities. Examples of commercial homework helpers for physics include WebAssign (http://www.webassign.com) and Mastering Physics (http://www.masteringphysics.com). They include most of the features listed in this section.

Hundreds of problems

A homework helper should have a vast number of homework problems. This mundane feature is critically important. Students and instructors are unwilling to install software and learn how to use it if it will only help them for a few hours. Andes currently contains 356 problems that cover most of the primary topics of a full-year physics course. We are still adding problems to cover secondary topics, such as sound waves, that not all instructors teach.

Authoring tools

In order to facilitate defining problems, homework helpers often have authoring tools. A typical tool lets the author define a problem statement, a correct answer and perhaps a few wrong answers and hints.

In Andes, the author does not specify an answer, but instead defines the problem formally so that Andes can solve the problem itself. The author then checks that Andes recognizes all the correct derivations, and that it provides appropriate hints. If derivations are missing or hints are misleading, then the Andes' physics knowledge base must be revised.

When the Andes project began, we naively thought that once we had developed enough authoring tools, the instructors could augment the knowledge base on a part-time basis. This might work for small knowledge bases, but Andes currently has 550 rules. Maintaining such a large knowledge base is like maintaining any other large software system — it requires time, expertise and detailed familiarity with the software. Thus, our work process involves a full-time knowledge engineer (Anders Weinstein). The instructors choose problems, write the formal definitions, and take a first stab at getting Andes to solve them. If there are inadequacies, the

knowledge engineer takes over. When he finishes revising the knowledge base, the instructors again check the derivations and hints. If there are no inadequacies, the new problems are added to the deployed version of Andes.

Learning how to communicate (precisely) with Andes

Because Andes has a paper-like user interface, students can start solving problems with little training. Nonetheless, they need help mastering a few details. For instance, students often think that "sin(45)" refers to 45 degrees, but Andes will interpret it as 45 radians. Students must learn to use the degree symbol when they mean degrees.

Worse yet, instructors must also learn these details before they can begin to use Andes smoothly. Several instructors who were considering Andes for use in their courses were able to start solving problems without any user interface training at all. This is a tribute to its intuitive, paper-like interface. However, they soon ran into user interface details that stymied them, and rejected Andes as "too hard for students to use."

The underlying problem is that Andes requires mathematical and graphical notation to be used precisely. Communicating precisely is difficult for instructors as well as students, in part because they often understand each other even when the notation is imprecise (e.g., they both interpret "sin(45)" as sin(π/4)). Although it might be technically possible to make Andes as tolerant of imprecise notation as humans are, we believe that students should learn how to communicate precisely. This may prevent them from unintentionally doing vague or even sloppy thinking. This is one place where we deliberately violated our policy of being minimally invasive. Andes insists on more precision that is typically required of students.

User interface training is needed throughout the curriculum. As new physics topics are taught, new mathematical and graphical notations are introduced, and students must learn a few non-obvious details about them, such as how to draw vectors along the z-axis.

We tried many traditional user interface training methods: tutorials, manuals, a Microsoft-style help system, an animated talking agent and unsolicited pop-up hypertext. None worked particularly well.[2] In the last year, we have been using short videos. The student sees the Andes screen and hears an instructor explain how to use the system while watching the mouse move and the characters appear. These videos are the only user interface training method that has received positive reviews from the students.

In the current version of Andes, when students open a new chapter's problem set, it checks to see if they have viewed the training video for that chapter. If not, it strongly suggests that they do. The video demonstrates both how to use any new notations and how to solve a simple problem. Thus, user interface training is combined with an initial example of problem solving. Because studying examples is highly popular with students (LeFevre & Dixon, 1986), they eagerly view these videos and thus get user interface training just when they need it.

---

[2] Professors at the Naval Academy encourage students to visit their offices at any time. Students experiencing user-interface problems would often do so, or would talk to the instructors after class. Thus, we enjoyed a rather unique window into user interface training failures that occur in a field setting.

Macro-adaptation and student modeling

Many tutoring systems, including both intelligent and non-intelligent ones, take responsibility for selecting problems. When the student finishes one problem, the system decides which problem the student should do next or whether the student has done enough problems for this textbook chapter and can go on to the next. Letting the system choose problems to fit the student's needs is called macro-adaptation (Shute, 1993), and the special case of letting the system move the student on to the next chapter is called mastery learning (Bloom, 1984). Macro-adaptation requires that the system maintain an assessment of the student that includes at least an estimate of the student's competence on the current chapter's instructional objectives. The assessment can include other information as well, such as the student's propensity to use hints. This evolving assessment of the student is often called the student model (VanLehn, 1988).

Mastery learning is typically used only in self-paced courses, whereas many physics courses are class-paced. In a class-paced course, all students hand in their homework at the same time, move on to the next chapter at the same time, take exams at the same time, etc. In a self-paced course, students work for as long as necessary in order to master a chapter. Thus, they hand in homework, take exams and finish the course at different times. A self-paced course is the perfect home for mastery learning and other forms of macro-adaptation. On the other hand, when mastery learning is used in a class-paced course, one student might have to work many more hours per week than another student in order to stay up with the class. Such difficulties have thwarted widespread adoption of mastery learning in class-paced courses (Kulik, Kulik, & Bangert-Drowns, 1990).

Andes has not yet been used in a self-paced course, so it currently does not support macro-adaptation and hence does not need to maintain a student model. In particular, the Bayesian student model that was so prominent in Andes1 is no longer useful, so it has fallen into disrepair.

Scoring

Mastery learning involves a kind of *medium stakes* assessment—the system's assessment of the student controls whether the student "passes" a chapter and moves on to the next. Physics courses also require *high stakes* assessments, which determine the students' course grades and whether they pass or fail the course.

Student modeling is typically not used for high stakes assessments. Because it is usually based on homework performance, and homework is typically unsupervised, instructors do not know how much help students are getting while doing their homework. For instance, students may do their homework in small groups, or routinely copy from their friend's homework. If the tutoring system were used in a proctored setting (e.g., during recitations), then the student models developed during that time could be used for high stakes decision making. [3]

Nonetheless, Naval Academy students and instructors have practically demanded that Andes score the students' homework. This is a zero stakes assessment, as nothing official is decided on

[3] Indeed, a course that replaced recitations sessions that merely "went over the homework" with proctored use of Andes (or another ITS) could increase the attendance at recitations, increase the number of problems solved per week, eliminate in-class exams, provide assessments on a weekly basis, and increase learning.

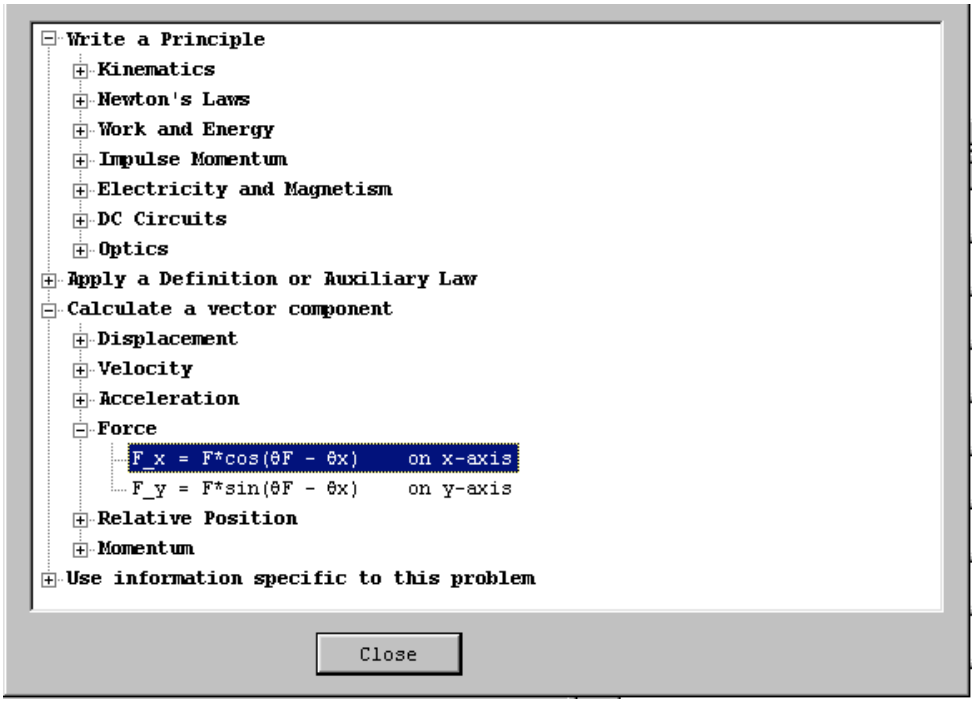

Fig. 3. The Cheat Sheet is an index to unintelligent help.

the basis of these grades. Apparently, students just want to know "how they are doing," and so do the instructors (cf. Schofield, 1995).

Andes computes several measures of performance, such as the proportion of entries that were correct (green) and the number of requests for bottom-out hints. Andes also computes an overall score that is a weighted sum of the individual measures. These measures depend only on the students' behavior on the current problem. That is, they are per-problem rather than cumulative.

Some students seem to be highly motivated by their Andes score, even though they understand that nothing depends on it. We have recently begun to use the score as a "soft" motivation for changing behavior, as described later.

Unintelligent help

Like most conventional tutoring systems, word processors and other end-user software, Andes includes help that is not sensitive to the current problem solving state. Such "unintelligent" help consists of text and other passive material that the student searches for useful information. An example is the Andes Cheat Sheet, which is shown in Figure 3. It is a hierarchical menu of all the equation-generating concepts known to Andes. Selecting an equation and clicking on the "More help on selection" button brings up a short hypertext, such as one shown in Figure 4. Although students usually have similar cheat sheets available in paper form, we wanted to monitor students usage of them so we provided one on-line.

Required reasoning

The pedagogical features discussed in the preceding section could appear in many kinds of homework helpers, even ones that just ask students for the bare answers to problems. Starting

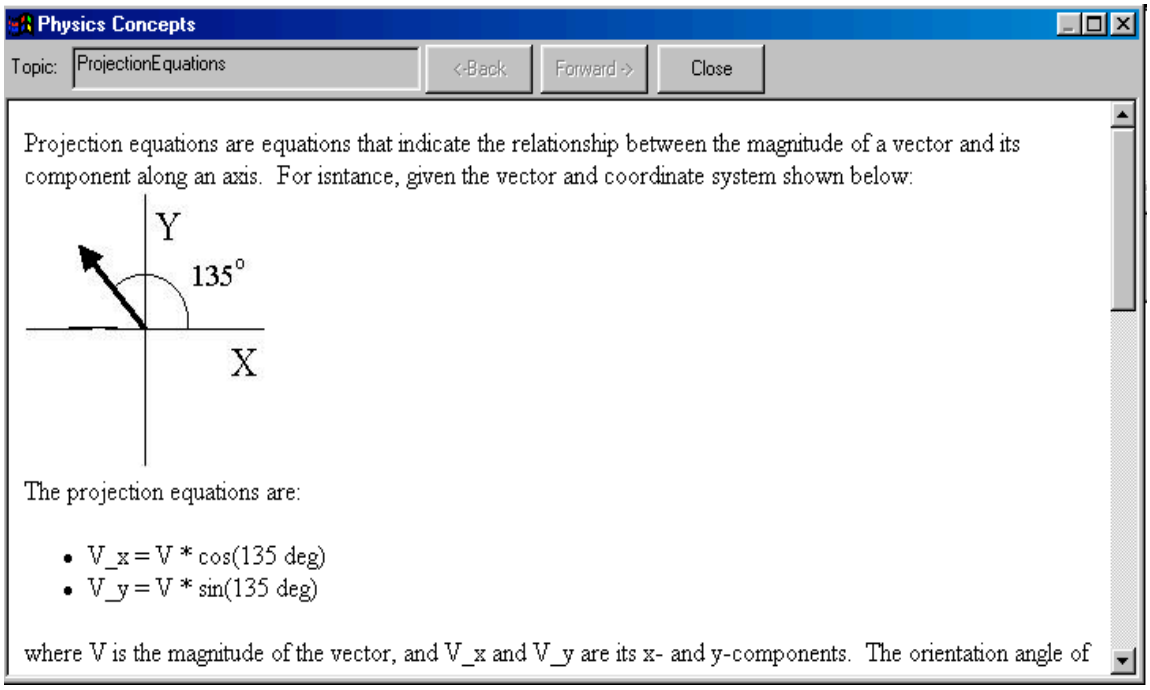

Fig. 4. Passive instruction (unintelligent help) popped up by the Cheat Sheet.

with this section, we discuss features that are applicable only when the student solves a problem in multiple steps that the tutor can observe.

When an instructional system can observe multiple steps, then it has the opportunity to

A. 3x+7 = 25

3x = _____

x = _____

B. 3x+7 = 25

_____ = _____

_____ = _____

_____ = _____

x = _____

C. 3x+7 = 25

_____ = _____ Justification: _____

x = _____ Justification: _____

Fig. 5. Three user interfaces for an equation solving tutoring system.

constrain them. That is, it can attempt to guide the students' reasoning while solving a problem by restricting the intermediate steps that the student can enter. Typically, this is done by providing type-in boxes or menu selections for intermediate steps. As an illustration, Figure 5 shows 3 user interfaces for entering the intermediate steps of solving an equation. User interface A encourages one particular strategy, namely subtracting the 7 then dividing by 3. However, it also blocks another strategy, namely dividing by 3 then subtracting 7/3. User interface B allows students to put any expressions they want in the blanks, as long as the resulting equations are mathematically equivalent to the first equation, so this user interface allows students to use many kinds of problem solving strategies, including inefficient ones. User interface C requires the student to enter "3x" in the first blank, "25-7" or "18" in the second blank, and "subtract 7 from both sides" in the third blank. On the next row, the student must enter "18/3" or "6" in the first blank and "divide by 3 on both sides" in the second blank. Like the user interface A, this one constrains the student to follow a particular strategy. However, it also requires the student to identify the mathematical justification for each step. Figure 5 illustrates just 3 of many possibilities that multi-step tutoring systems have for both constraining students' reasoning and for making parts of it explicit that are normally not written down.

Many tutoring systems have user interfaces that constrain student reasoning, but their impacts on learning are mixed. For instance, Singley (1990) found that having calculus students pursue a goal-directed problem solving strategy and explicitly entering their intermediate goals seems to have accelerated their learning. However, a similar manipulation seems to have failed with a geometry tutoring system (Koedinger & Anderson, 1993) and yielded only a mild positive effect with a propositional logic tutoring system (Scheines & Sieg, 1994). Constraining student reasoning is probably just as complex a design issue as feedback or hints. We do not yet have an empirically grounded theory of which constraints are beneficial to whom at what times in their learning.

Andes puts little constraint on students' reasoning. Its user interface is like interface B of Figure 5—students can fill in the blanks any way they want as long as their equations and other entries are true statements about the physical quantities in the problem.

We left the user interface unconstrained so that it would be similar to pencil and paper. In general, one gets higher transfer from training to testing when the user interfaces are similar (Singley & Anderson, 1989). Although the unconstrained Andes user interface might make learning slower compared to a constrained user interface, the transfer literature suggests that mastery of physics on the Andes user interface should almost guarantee mastery on pencil and paper. Moreover, keeping the user interface unconstrained makes Andes less invasive.

Although the general policy was to leave the Andes user interface unconstrained in order to increase transfer and reduce invasiveness, several deliberate exceptions to the policy have been implemented. This section discusses those exceptions and their motivations.

Variables must be defined

When using pencil and paper, students can define variables by writing, for example, "Let v1 be the instantaneous velocity of the cart at time 1." However, students rarely write such definitions. They prefer to use the variables in equations without defining them. Andes does not permit this. Variables must be defined by filling out a dialogue box. If an undefined variable is used in an equation, an error message pops up identifying the undefined variables in the equation.

Andes requires that a variable has a unique definition. For instance, suppose an object is moving with constant velocity from time 1 to time 2. Students and instructors often want to use the same variable to denote 3 distinct quantities that happen to be equal: the average velocity over time 1 to 2, the instantaneous velocity at time 1 and the instantaneous velocity at time 2. Andes does not permit such variable overloading (Liew, Shapiro, & Smith, 2004). One must pick one of the definitions for the variable.[4]

There are technological reasons for requiring students to define variables. If a variable is not defined explicitly, it is difficult to identify its definition until it has appeared in several equations. Even human graders find this difficult. Work has begun on automating this kind of variable identification (Liew et al., 2004), but it remains a difficult technological problem[5]. To avoid having to solve it, Andes simply requires that variables be defined.

However, the main reason for requiring variables to be defined is that this precision may help novices learn the distinctions between related concepts. For instance, the dialogue box for velocity requires that students always choose between "instantaneous" and "average." This reminds students of the distinction, which can often become vague and muddy in their minds, especially when reading or listening to problem solutions that use variable overloading. As another example, the dialogue box for defining an individual force requires the student to specify both the object that the force acts on and the object that the force is due to. This reinforces the concept that an individual force is always an interaction between two objects (Reif, 1987). On the other hand, the dialogue box for defining a net force only mentions one object. The two dialogue boxes look different in order to emphasize the distinction between net forces and individual forces. In short, although requiring students to define variables is a restriction on their reasoning style, we believe it accelerates their learning of certain conceptual distinctions.

Students often complain about the requirement that they define variables. They believe that they have already mastered the appropriate concepts and do not need the practice. However, students are not always accurate at self-monitoring, and poorer students often overestimate their competence.

An appropriate extension to Andes would be to resurrect its student modeling capability so that it could track mastery of definitional concepts. This would allow it to fade its definitional scaffolding by providing pre-defined variables to students who have mastered the relevant concepts.

In the meantime, Andes provides predefined variables in pedagogically appropriate situations. On complex problems, it provides predefined variables for quantities that are likely to be well understood due to practice on simpler problems. For instance, on electrical circuits with multiple resistors, Andes predefines a variable for each resistor's resistance (e.g., R1, R2…)

---

[4] When several variable definitions are equal, in that they have the same value, then one only needs to define a variable for one of them. As described later, Andes checks the correctness of equations by replacing each variable in an equation with its numerical value. In this example, since all three definitions—v12, v1 and v2—have the same value, one only needs to define a variable for one of them.

[5] The problem is clearly NP-hard, but amenable to constraint-satisfaction techniques. The problem is: Given a set of V variables, D definitions and E equations, assign definitions to variables in such a way that the maximal number of equations are true. There are approximately D^V possible assignments.

because such problems are not usually assigned until students have solved several problems with a single resistor or a pair of resistors.

Vectors

Some quantities, such as mass, are scalars whereas other quantities, such as velocity, are vectors. Although equations can be written with vectors, introductory physics students are not taught methods for solving systems of vector equations. Instead, students are taught to draw a coordinate system, define the components of each relevant vector along the relevant axes, and then write scalar equations in terms of the vector components. Scalar equations are just ordinary, high-school algebra equations, which students should know how to solve.

In Andes, students may not enter vector equations. They do not need to, because their solution methods involve only scalar equations. Moreover, during the Cascade project (VanLehn & Jones, 1993b, 1993c; VanLehn et al., 1992), we observed students misinterpreting vector equations as scalar equations and becoming quite confused. Thus, in Andes, all equations are scalar.

Whenever students want to define a scalar quantity that is related to a vector, such as its magnitude or its component along the x-axis, they must draw the vector itself. For instance, in Figure 1, in order to write the given information that "d = 20.0 m", the student had to draw the displacement vector d.[6] This is a constraint on students' reasoning because paper and pencil does not require students to draw vectors.

In the first version of Andes, students could define vector-related scalars via a menu or by drawing the vector. The 1999 evaluation indicated that few students used the vector drawing option. They would draw nothing, and this seemed to cause many errors, such as missing negative signs in projection equations, sine vs. cosine errors, or angle arguments that were off by multiples of 90 degrees. During the 2000 evaluation, instructors strongly recommended that students draw vectors instead of using the menu, and their paper-and-pencil exams gave points for drawing vectors. Nonetheless, students still preferred to use the menus rather than drawing the vectors, and the errors persisted. So we removed the menu option, endured the student's complaints, and noticed a decrease in errors.

While reducing errors is perhaps reason enough for requiring vector drawing, there is another motivation that is subtle but more important. It will be covered in the next section.

Explicating principle applications

As discussed earlier, one of Andes' major instructional objectives is to encourage a conceptual understanding of physics problem solutions. There is no consensus yet in cognitive science about what this means exactly, but it certainly involves being able to identify the major principle

---

[6] A common convention is to use the same letter in boldface to represent a vector and in plain font to represent the magnitude of that vector. Andes does not permit vector variables in equations, so the vector sense of the symbol appears only in diagrams and not in equations. Although this overloading may cause students confusion, we chose to conform to the convention simply because it is so universally followed.

applications required for solving a problem. Thus, students' solutions should somehow highlight the major principles and their role in solving the problem.

Several approaches have been tested for emphasizing major principle applications. Dufresne et al. (1992) had students explicitly identify the major principles required for solving a problem by working through a decision tree with questions like "Does the moving object have any non-conservative forces acting on it?" After answering enough questions, the system would state the major principle and its equation. Leonard et al. (1996) had students solve a problem and then write a short essay describing their basic approach to solving it. Katz et al. (2003) had students solve a problem, then answer short essay questions about the major principle applications. Aleven et al. (2002) had students identify the principle that justified each line of a derivation, using a user interface similar to C in Figure 5.

All these approaches increased students' conceptual understanding of problems, but we decided against including them in Andes for two reasons. (1) Some of the techniques require the tutoring system to have natural language understanding capabilities that are currently beyond the state of the art (but see Aleven, Popescu, & Koedinger, 2003; Graesser, VanLehn, Rose, Jordan, & Harter, 2001). (2) Students do not normally do these activities as part of problem solving, so they would only be willing to do them with Andes if their instructors explicitly endorsed them and assessed them. For instance, the students of Leonard et al. (1996) knew that essay writing would be on their exams. This might require instructors to reform their instruction to a greater degree than they would be willing or able to do. For instance, in large physics classes, it is difficult to grade exams that include even a few essays per student.

Andes' approach is based on the observation that when a principle is applied, the resulting equation can be written all by itself or by combining it with other equations. For instance, when applying Newton's second law to the problem of Figure 1, the student can write the primitive form *Fw_x = mc*a_x* or a composite form, such as *–Fw*sin(20 deg)=-mc*a*, which combines the primitive form with *Fw_x=-Fw*sin(20 deg)* and *a_x=-a*.

In order to encourage thinking of solutions in terms of major principles, Andes encourages students to write the first application of a major principle in its primitive form. If they write it in composite form, a warning message pops up and they lose points in the scoring rubric. However, the composite equation is still colored green, because it is still a true equation for the problem. The intent is that students learn which principles are major, and that they get into the habit of including a clear statement of them in their problem solutions. Granted, this is a constraint on their problem solving, but a fairly lightweight one given the importance of the potential learning outcomes.

For instance, if a student enters *a*mc = Fw*sin(20 deg)* for the problem in Figure 1, then Andes employs the following hint sequence:

- Although equation 3 is correct, you have not yet entered a fundamental vector principle being used in component form on a line by itself.
- It is good practice to identify the fundamental vector principles you are using by writing them purely symbolically in component form before combining them with other equations or given values. Select "Review Physics Equations" on the Help menu to view a list of principles and their standard forms.
- A good solution would show the following in component form: Newton's second law.

What Andes would prefer the student to have entered is either *a_x *mc = Fw_x,* or *Fw_x = a_x*mc,* or *a_x=Fw_x / mc*, etc.

To summarize, the Andes user interface was designed to be as unconstraining as paper and pencil in order to increase transfer to pencil and paper. However, there are three exceptions to this policy, all of which are intended to increase student learning. (1) All variables must be defined precisely before being used. (2) All vectors must be drawn in order to define the associated variables. (3) Students are warned if they apply a major principle in composite form, that is, as an algebraic combination of the major principle's equation and other equations.

Feedback and hints

This section discusses feedback, which is a comment by the tutoring system on the correctness of the student's work, and hints, which are comments by the tutoring system on how to produce better work.

Flag feedback

Whenever the subject enters a step in the solution, Andes colors it red if it is incorrect and green if it is correct. This is called flag feedback (Anderson et al., 1995), because it provides a merely binary "flag" (correct vs. incorrect) rather than a hint, a description of the error, or any further information. Studies show that flag feedback can dramatically improve the efficiency of learning (Anderson et al., 1995).

However, "correct" is a subtle term with two common meanings:

1. Correct means that the entry is valid. A valid definition is one that defines a quantity that actually exists. A valid statement is one that follows logically from the givens of the problem and the principles of the task domain.
2. Correct means that the entry is valid and it appears in a solution to the problem.

The advantage of the second definition of correctness is that it prevents students from wandering down paths that do not lead to a solution. Thus, it can make their problem solving more efficient. However, this also implies that they will not have experienced search, that is, with going down a false path, detecting that they are lost, and recovering. If all their homework is done with immediate feedback that keeps them on solution paths, they might first have to cope with search on a test problem. Thus, the first definition of correctness might give students most experience with search, and thus raise their scores on test problems that invite traveling down false paths.

Unfortunately, there appears to be no empirical work comparing the two types of feedback. That would be a good topic for further research.

Andes takes a compromise position, and uses both meanings for correctness. A definition is considered correct if the defined quantity appears in the solution (meaning 2). A statement is considered correct if it is valid (meaning 1). For instance, in an electrical circuit problem, it is "incorrect" to define a variable for the mass of the battery. Certainly the battery has a mass, but its mass is irrelevant to calculating voltages, currents and other electrical quantities. On the other hand, if a Newton's law problem can be solved by resolving forces along the x-axis, then an equation that balances forces along the y-axis is valid so it turns green even though it does not

appear in the solution of the problem. By using both meanings for "correct," Andes allows the student some experience with search without too much loss of efficiency.

What's Wrong Help

When an entry is red, students can select it and click on the "what's wrong with that" button. As mentioned earlier, this is called What's Wrong Help.

What's Wrong Help is always solicited help; never unsolicited. This makes it different from some other tutoring systems. Some tutoring systems give What's Wrong Help on every error without being asked (e.g. Nicaud et al., 1999). Other tutoring systems count the number of incorrect attempts and give What's Wrong Help after the third failed attempt (Anderson et al., 1995). Although we think unsolicited What's Wrong Help might be a good feature, it is technically difficult for Andes. In the Cognitive Tutors (Anderson et al., 1995), entry boxes almost always have a goal associated with them in that only specific text, formulae or other values are correct for that box. Although this is true of the dialogue boxes used in Andes to define vectors and variables, the equation boxes in Andes can be filled by any equation. Whenever a student enters an equation, Andes can make no assumptions about what equation they are attempting to enter. Thus, it makes no sense to count failed attempts at filling an equation box and offer unsolicited help after the third one. The students might be trying three different goals, instead of making three attempts at achieving the same goal.

Because Andes does not know what equation a student is trying to enter when they ask for What's Wrong Help on a red equation, it is difficult to decide what kind of hint to give. Andes1 tried to guess which correct equation the student intended by using a syntactic distance technique (Gertner, 1998). Despite considerable tuning, this technique never worked well. Moreover, even when Andes1 identified a correct equation that the student intended to enter, its hints could only point out the difference between the two. For instance, if the equation the student intended to enter was $Fw = mc*g$ (see Figure 1) and the student actually entered $Fw = -mc*g$, then Andes1 could only say "Sign error: You need to change –mc*g to +mc*g." When receiving such a "hint," students rewrote their equation as directed, watched it turn green, and were totally mystified about what the new equation meant and how Andes1 derived it. That is, because Andes1 used a syntactic, knowledge-free technique for analyzing errors, it gave syntactic, knowledge-free hints on how to fix them. Such hints probably did more harm than good.

Andes2 gives hints on errors only when the hint is likely to have pedagogical value. It has a set of error handlers, and each recognizes a specific kind of error and gives a hint sequence that helps the student learn how to correct it. For instance, in the case above, Andes would give the following hint sequence:

- You probably recall the vector equation 'W_vec = m*g_vec' where both W_vec and g_vec are vectors, namely the weight and the gravitational acceleration. This vector equation implies that the vectors have the same direction (both are downward). It also implies that their magnitudes are proportional, that is, that W=m*g where W and g stand for the MAGNITUDES of the weight and gravitational acceleration. Your equation has an unnecessary minus sign in it. Just remember that everything is positive: mass is a positive number, and because W and g stand for magnitudes, they are positive numbers, too.
- Change your equation to Fw = mc*g

Incidentally, this hint sequence has only a teaching hint and a bottom-out hint. It omits the pointing hint because a hint like "check your signs" would allow students to guess what to do to fix their equation and thus not get the teaching hint even if they needed it. At any rate, the main point is that whenever an error handler recognizes an error, the hint sequences are likely to be highly appropriate.

If no error handler matches an equation error, then Andes asks the student what equation the student was trying to enter and displays the Andes Cheat Sheet, which was shown earlier in Figure 3. Students can search through its hierarchical menu of equation-generating principles. When the student has found the principle that she wants to apply and clicked on the OK button, Andes tells the student whether it plays a role in this problem's solution. If it does, Andes helps the student apply it. If it doesn't, Andes suggests asking for Next Step Help. Thus, when What's Wrong Help does not know which equation the student intends to enter, because it cannot recognize the error in the student's equation, then it asks the student what principle the student is trying to apply.

Hint sequences

Whenever What's Wrong Help recognizes an error, Andes has an opportunity to help students learn. Thus, it gives a sequence of hints intended to accelerate learning. As mentioned earlier, the hint sequence usually has a pointing hint, a teaching hint and a bottom-out hint. Some pointing hints direct the student's attention to the location of the error in case the student already knows the relevant principle but has not noticed that it applies. A teaching hint is a short statement of the relevant principle or concept. The bottom-out hint indicates exactly what the student should enter. Some hint sequences also contain reminder hints, which provide a piece of a relevant principle in the hope that the student will be able to recall and apply the rest of the principle. Designing hint sequences is currently more of an art than a science, so hint sequences often contain material that is clearly helpful but difficult to classify in general terms.

There are several reasons why such hint sequence may increase learning. One is that expert human tutors use hint sequences containing similar hints (Hume et al., 1996). The psychology of human memory also suggests that such hints will facilitate learning:

- The pointing and reminding hints mention cues that should jog student memory retrieval. If the hints succeed, the student's generation of the item strengthens the association between the cues and the principles more than having the item presented. This is known as the generation effect (Slamecka & Graf, 1978).
- The statement of the principle is made in the context of these cues, which should facilitate subsequent retrieval. This is known as encoding specificity (Tulving & Thomson, 1973).
- If the student doesn't understand the principle well enough to apply it, the bottom-out hint serves as a just-in-time example. It too is done in the context of the cues, thus facilitating subsequent retrievals.

Bottom-out hints can be abused (Aleven & Koedinger, 2000). Students sometimes click through the other hints quickly in order to reach the bottom-out hint. Although sometimes this is warranted, it is generally a sign that the student has a performance orientation instead of a learning orientation (Dweck, 1986). In the belief that students with a performance orientation often pay close attention to their scores, Andes takes points off the student's score for every

bottom-out hint. We are considering adding a 20 second delay before giving a bottom-out hint, because this significantly reduced help abuse in a calculus tutor being developed in our lab (Murray & VanLehn, 2005). How to discourage help abuse without also discouraging appropriate help seeking is complex and requires further research (Aleven, Stahl, Schworm, Fischer, & Wallace, 2003).

Next Step Help

When students click on the Next Step Help button, which looks like a light bulb (Figure 1), Andes selects a step and gives the student hints about it. As with What's Wrong Help, if students click on "Explain further" enough times, they will receive a bottom-out hint that tells them exactly what to do.

Next Step Help is essential. Students sometimes enter a sequence of incorrect entries, and sometimes none of their attempts matches an error handler so they can get no What's Wrong Help. In some problems, students enter the given values from the problem statement, and then have no idea what to enter next. In this situation and many others, Next Step Help is essential.

Two versions of Next Step Help have been developed and evaluated. The first version attempted to recognize the student's plan for solving the problem and hint the next step in that plan (Gertner, Conati, & VanLehn, 1998; Gertner & VanLehn, 2000). The Andes1 Bayesian net was used for both plan recognition and next step selection. This approach was used in the 1999 and 2000 experiments. After the 2000 experiment, we evaluated it by randomly selecting episodes where students asked for Next Step Help. For each episode, the student's screen was printed just prior to Andes' first hint, the hard copies were given to 3 physics instructors, and the instructor wrote the sequence of hints they would give at this point. Although the instructors sometimes disagreed on what step to hint next, they did agree in 21 of the 40 episodes. Unfortunately, Andes tended to disagree with the humans. Its hint sequence agreed with the consensual one in only 3 of 21 cases. Thus, it appears that Andes1's Next Step Help is not consistent with expert human help.

The problem, in the opinion of instructors, was that students often had no coherent plan for solving the problem. They had generated a few correct entries, often with significant help from Andes, but there appeared to be no pattern in their selection of steps. For these students, the instructors wrote hint sequences that identified the major principle and the first step toward applying that principle. Instead of trying to recognize the student's plan, which is what Andes1 did, they hinted the first step in their own plan for solving the problem.

The next version of Next Step Help replicated the instructors' hint sequences. It was based on the assumption that if students are lost and asking for Next Step Help, then they probably have no coherent plan, so they should get a hint on the first relevant step in the instructors' plan for solving the problem. In mechanics, for instance, the instructors' plan usually starts with drawing a body, drawing coordinate axes and entering the given information. If the student has not completed all these "initialization" steps, Andes hints at the first missing one. If all the initialization steps have been completed, then Andes usually conducts the following dialogue:

- First, it asks the student to select the quantity the problem is seeking (see Figure 1, lower left window).
- The student selects a quantity from a hierarchical menu similar to the one used to define quantities. If an inappropriate quantity is selected, Andes keeps asking until

the student successfully selects a sought quantity or they have failed three times, in which case Andes identifies a sought quantity.

- Andes says, "What is the first principle application that you would like to work on? Hint: this principle application will usually be one that mentions the sought quantity explicitly. Therefore, its equation may contain the sought quantity that the problem seeks."
- The student selects a principle from the Andes Cheat Sheet, shown earlier in Figure 3. For Figure 1, the student should select "v_x^2 = v0_x^2 + 2*a_x*d_x [a_x is constant]"
- Andes determines the first step required for applying that principle that the student has not yet done and starts the hint sequence associated with that step. For Figure 1, it would hint drawing the car's acceleration.

Depending on the state of the student's problem, this dialogue is modified. Here are the major special cases:

- If the student has already gone through the dialogue and started applying a principle, but not all of the steps have been completed, then Andes starts by suggesting that the student finish applying the principle (e.g., "Why don't you continue with the solution by working on writing the constant acceleration equation v_x^2 = v0_x^2 + 2*a_x*d_x for the car from T0 to T1."). If the student clicks on Explain Further, Andes starts the hint sequence for one of the uncompleted steps.
- If the student has completed the application of a principle that mentions the sought quantity, but the set of equations entered by the student is not yet solvable, then Andes selects a principle application and suggests it (e.g., "Why don't you try applying Newton's second law to the car from T0 to T1."). If the student clicks on Explain Further, it hints the first step in the principle application.
- If the set of equations entered by the student is solvable but not yet solved, then Andes suggests using the solve-equations tool.
- If the set of equations has been solved but the student has not yet entered the appropriate values into the answer boxes, then Andes suggests doing so.

In summary, the new version of Next Step Help does not try to recognize the student's plan, but instead it engages the student in a brief discussion that insures the student is aware of the principle that is being used to solve the problem, then selects a step the student has not yet done, and hints that step.

This design is a compromise between brevity and completeness. The discussion of the solution plan could be much more complete and thorough. For instance, in Figure 1, the sought quantity is a magnitude and thus does not actually appear in the equation that results from applying the constant acceleration principle. That equation is *v2_x^2 – v1_x^2 = 2 * a_x * d_x*. It contains *v2_x*, not *v2*. Andes glosses over these details in order to focus the student's attention on the major principle and not get bogged down in minor principle applications.

Secondly, when a problem requires multiple major principle applications, and the student has already applied the major principle that contains the sought quantity, then Andes will suggest applying one of the remaining principles but it will skip the first 4 steps of the dialogue above. For instance, Figure 1's solution requires a second major principle application, namely Newton's second law, but if the student needs help on applying it, Andes just asserts that it needs to be applied. The first version of Andes2 had a longer dialogue that motivated this selection. In the

case of Figure 1, it would elicit the fact that the equations written so far left the acceleration *a* unknown, then it would ask the student to select a major principle that contains *a*. The instructors felt that by the time students were given problems with multiple applications of principles, they may not need such detailed problem solving instruction. The instructors preferred a shorter dialogue that drew the students' attention immediately to the major principle that needed to be applied.

Essentially, given that Andes does Next Step Help by suggesting a major principle application, we had to choose whether to motivate each principle application via identifying sought quantities, or simply assert that the principle needed to be applied. We choose to motivate the initial principle application, but just assert the others.

To summarize this whole section on feedback and hints, Andes offers three kinds:

- Flag Feedback: Andes turns entities red or green. Errors that are probably slips (i.e., due to carelessness rather than lack of knowledge) also generate an unsolicited pop-up error message.
- What's Wrong Help: If the student asks what's wrong with an entry and Andes can recognize the error, it gives hints intended to help the student learn to recognize the error and avoid it in the future.
- Next Step Help: If the student asks what to do next, Andes insures that they are aware of a major principle required to solve the problem, then hints one of its steps the student has not yet done.

Algebraic manipulation tools

Our hypothesis, shared with many physics instructors but certainly not all, is that students should be relieved of algebraic manipulation tasks so that they can concentrate on learning physics. Partly, this is just a matter of time on task. It takes less time to do a physics problem when the tutoring system does the algebra, so an instructor can assign more physics problems per week, and this should increase the student's learning of physics. Perhaps more subtly, when students are doing their own algebraic manipulation, they often blame it for errors that are actually due to flaws in their physics knowledge. For instance, if students who are doing their own algebraic manipulation manage to fix an equation by deleting or adding a negative sign, then they probably think of this as having caught a careless mistake in their algebraic manipulations; they don't even consider whether they might have misapplied a physics or mathematical principle such as projection. When the tutoring system does the algebraic manipulation, students know that every error is due to their own misunderstandings of physics, and this should accelerate their learning.

Andes has an algebraic manipulation tool. If the student clicks on a button labeled "x=?" Andes says "Choose the variable you would like to solve for." and offers a menu of variables. When the student has selected one, Andes attempts to solve the system of equations that the student has entered. If it succeeds, it writes a new equation of the form <variable> = <expression> where <variable> is the one that the student chose, and <expression> is a simplified algebraic expression that does not contain <variable>. Usually, <expression> is just a dimensioned number. If Andes cannot find a unique solution, it reports "Unable to solve for <variable> or multiple solutions were found."

Students are not required to use the algebraic manipulation tool, but most do. In part this is caused by the way Andes does flag feedback on equations. Students like to solve equations by

plugging in numbers whenever possible. Instructors often discourage this practice, because it obscures the logic of the derivations. For instance, instead of writing *mc=2000 kg*, *Fw=mc*g* and *Fw_x=Fw*cos(250 deg)*, a student might write on paper:

- Fw = 2000 kg * 9.8 m/s^2 = 19600 N
- Fw_x = 19600 N * cos(250 deg) = −6704 N

This kind of derivation makes it difficult to see what principles are being applied. Nonetheless, it is popular with students. However, when students try to follow this practice in Andes, they discover that they have to enter

- Fw_x = −6703.5948 N

or else the equation turns red. Andes insists on approximately 9 digits of precision when numbers are entered in the Equation window. If students round numbers, the equations turn red. (On the other hand, rounding to three significant digits is expected when numbers are entered in the Answer slots.) This high degree of precision rapidly discourages students from using the number-passing style of problem solving that obscures the physics reasoning. Instead, students tend to enter all their equations with variables instead of numbers, then use the equation solving tool. It plugs in the numbers, does the algebra, and reports answers as high precision numbers. The students copy the numbers to the Answer slots, rounding them to 3 significant digits. Although this is a constraint on the student's problem solving, and thus constitutes invasiveness, we believe the pedagogical benefits warrant it.

DESIGN AND IMPLEMENTATION

This section describes how Andes is implemented. Subsequent sections do not depend on it, so it may be skipped if the reader wishes.

Before the Andes system is distributed, it computes and stores detailed solutions to all the problems. This facilitates the computations that must be performed while the student is being tutored. The first phase is called "solution graph generation" because it produces one file per problem, and the main data structure in the file is called a *solution graph*.

Problems often have multiple solutions. For instance, the problem of Figure 1 has two solutions: one using Newton's second law and kinematics, and another using Conservation of Energy. When Andes generates the solution graph of a problem, it computes all possible solutions permitted by its knowledge base. This completeness is essential. If Andes failed to generate a valid solution to a problem and a student defined a quantity that appeared only in that solution, then the student's entry would be colored red, with unfortunate pedagogical results.

When the Andes project began, solution graphs were precomputed in order to reduce the computation required during tutoring. With today's computers, this is no longer necessary. In fact, all computation is done at runtime by Pyrenees, another physics tutoring system that uses the same knowledge base (VanLehn et al., 2004), and the response time is quite similar to Andes' response time. However, we have retained the two-phase architecture because it makes it much easier to maintain and extend the physics knowledge base. When the knowledge engineer makes a change to the knowledge base, solution graph generation produces a new set of solution graphs, one for each problem. The new solution graphs are compared to the old ones, which allow detection of unintended consequences, such as adding a new but invalid solution to a problem.

Given that there are currently 356 problems in Andes, this kind of semi-automated regression testing is essential.

This section describes how Andes implements each of its main features. For each feature, we describe the information stored in the solution graph file, how that information is computed and how it is used. As will be seen, different features require vastly different amounts of physics and algebra knowledge.

Implementing immediate feedback

This section describes how Andes implements immediate feedback. When the student makes an entry, Andes' immediate feedback consists of either coloring it green, coloring it red or popping up an error message such as "Undefined variable."

From an implementation perspective, there are two types of entries that students make when solving a problem: equations and non-equations, where non-equation entries include drawing vectors, drawing coordinate axes and defining scalar variables. The implementation of immediate feedback on non-equation entries will be discussed first.

In order to check non-equation entries, Andes first does a syntactic check. This catches many unintentional errors, such as leaving blanks in a dialogue box. These checks generate pop-up error messages such as "Please supply a time."

If the non-equation entry is syntactically correct, it is compared to the non-equation entries in the solution graph file. If an exact match is found, the student's entry is colored green; otherwise it is colored red. Thus, a non-equation entry is considered correct if and only if it is a component of at least one solution to the problem.

When a student enters an equation, Andes first checks its syntax. It parses it using a context-free parser. Because students can include units on numbers, the grammar is not the standard operator precedence one. We had to work with a corpus of several thousand equations in order to tune the grammar to accept all and only the syntactically legal equations, and to give reasonable error messages when it could not parse an equation. In the process of parsing the equation, Andes also detects undefined variables.

If the student's equation is syntactically well-formed, then Andes checks its dimensions. This often catches cases where students have forgotten to include the units of a dimensional number.

If the student's equation is dimensionally consistent, Andes completes its checking using a technique called color by numbers (Shapiro, 2005). The solution graph file for the problem contains the solution point, which is a list of every variable in the problem paired with its value when the problem has been solved. These values are substituted into the student's equation. This substitution produces an arithmetic equation. That is, the equation has all numbers and no variables. Color-by-numbers checks whether the equation balances, with due regard for numerical accuracy. If it does, the student's entry is colored green; and red otherwise.

For most problems, all the given values dimensioned numbers. However, some problems use a parameter instead of a number, e.g., "A truck of mass *m* collides…" For such problems, Andes pretends the problem statement used an "ugly" numerical value instead of a parameter. If the equation balances with this ugly value, it is highly likely that it would also balance if Andes had retained the symbolic parameter value instead. This same technique is used when quantities involved in the solution cancel out, and thus are not given values by the problem statement. For

instance, masses often cancel out in conservation of momentum problems, but they must nonetheless be mentioned in the equations required for solving the problem. Andes generates ugly values for such masses so that it can check student equations that contain mass variables.

Students are told that equations are colored green if and only if they can be derived algebraically from applications of the fundamental principles and the values given in the problem statement. Shapiro (2005) proved that color-by-numbers is equivalent to this definition; a sketch of the proof is presented here. Clearly if the equation fails to balance when the solution point is substituted in, then there cannot be a correct derivation of it. To prove the converse, we need to show that if an equation balances when the solution point is substituted in, then there exists an algebraic derivation of it. From the givens and fundamental principle applications, we can derive the solution point, which can be expressed as a set of equations of the form <variable> = <number>. Another branch of the derivation starts with the student's equation with numbers substituted into it. Because we assumed this equation balances, it is simply an arithmetically true statement that happens to have a lot of numbers in it, and for each number, there is an equation in the solution point of the form <number> = <variable>. We can substitute the solution point equations into the arithmetic equation, and produce a new equation that has variables and is exactly the student's equation. Thus, from a single arithmetically true equation, from the given values and from the applications of fundamental physics principles, we have derived the student's equation. This completes the proof that color-by-numbers is equivalent to algebraic derivability.

Color-by-numbers replaces search-based techniques used by other systems (e.g., Brna & Caiger, 1992; Yibin & Jinxiang, 1992). It replaces the technique used in Andes1, which was to pre-compute all algebraic combinations of the principle applications.

Color-by-numbers is similar to a technique used by CAI systems ever since Plato (Kane & Sherwood, 1980). However, the CAI technique applies to algebraic *formulas* whereas Andes' technique applies to algebraic *equations*. For instance, suppose a CAI system asks a fill-in-the-blank question such as "*v1_x* = _____" or "The x-component of the initial velocity is _____" and expects an algebraic expression as the answer. It can check the correctness of the student's entry by parsing it, checking its dimensions, substituting numbers for the variables and simplifying the resulting arithmetic formula. If simplification yields the expected number, then the student's answer is correct. Color-by-numbers essentially does this to both sides of the equation. Although it was always clear that the solution point could be substituted into an equation and the resulting arithmetic equation could be checked for balance, it was not clear what such a check really meant. Joel Shapiro proved that this technique was equivalent to finding a derivation for the equation. Andes may be the first system to apply the technique to equations instead of formulas.

In order to give immediate feedback, the solution graph file needs to contain the set of non-equation entries relevant to solving the problem and the solution point for the problem. In principle, this information could be provided by a human author instead of being generated by Andes. However the human author would need to supply high precision values for a large number of variables. As Shapiro (2005) discusses, numerical accuracy is important to the operation of color-by-numbers. If one were interested in a CBT-like approach where human authors provide solutions to every problem, it would be handy to include an equation solver like Andes. Then the author could enter a set of equations, which is much easier than entering a hundred or so 9-digit numbers.

Although color-by-numbers is simple and efficient, it is not perfect. A known problem is that incorrect subexpressions in the student's equation can be ignored if they are multiplied by an expression that evaluates to zero. For instance, suppose *v2^2 –v1^2= 2*a*s* is a correct equation for this problem, and that *v1=0* at the solution point. If the student writes *v2^2 +v1^2= 2*a*s* or even *v2^2 + v1*(v2-a*s) = 2*a*s*, then Andes will color the student's equation green, whereas a human grader would mark it wrong even though the equations can be derived algebraically from correct equations. Fortunately, this flaw occurs infrequently in practice. Shapiro (2005) reports that in the Autumn 2001 evaluation, in which 5766 problem solutions were entered, he found only one case where Andes marked green an equation that a grader would have marked wrong.

Implementing What's Wrong Help

When a red entry is selected and the student clicks on the What's Wrong Help button, Andes applies a large set of error handlers to the entry. Each error handler takes the student's entry as input. If it recognizes the error, it returns a hint sequence and a priority. If the handler doesn't recognize the error, it returns NIL. If several error handlers recognize errors in an incorrect entry, Andes chooses the hint sequence with the highest priority.

Error handlers are similar to West's issue recognizers (Burton & Brown, 1982), to Proust's bug recognizers (Johnson, 1990) and to the constraints of constraint-based tutoring systems (e.g., Mitrovic, 2003). All of these recognize errors and generate hint sequences.

Andes' error handlers work by making edits to the student's entry. If the edits generate a correct entry, then the error handler constructs an appropriate hint sequence that refers to the student's entry. For instance, suppose a student who is solving the problem in Figure 1 defines a zero-length velocity vector and asserts that it is the instantaneous velocity of the car at T1. The velocity is actually non-zero, so this is an error and it turns red. One edit is to change the time specification, so that the vector stands for the instantaneous velocity of the car at T0. Another edit is to change the qualitative value, so that the vector is non-zero. The second edit gets a higher priority, so Andes selects this hint sequence:

- Is the car at rest at T1?
- Since the car is not at rest at T1, the velocity vector needs to be non-zero.

Error handlers for equations simply edit the equations then call color-by-numbers on the edited equation. For instance, suppose a student solving the problem of Figure 1 enters the incorrect equation *Fw_x =-Fw*cos(20 deg)*. An error handler notes that there is a cosine in the equation, changes the equation to *Fw_x = -Fw*sin(20 deg)* and submits it to color-by-numbers, which indicates that the edited equation is correct. The error handler returns a high priority hint sequence comprised of the usual pointing hint, teaching hint and bottom-out hint:

- Check your trigonometry.
- If you are trying to calculate the component of a vector along an axis, here is a general formula that will always work: Let θV be the angle as you move counterclockwise from the horizontal to the vector. Let θx be the rotation of the x-axis from the horizontal. (θV and θx appear in the Variables window.) Then: V_x = V*cos(θV-θx) and V_y = V*sin(θV-θx).
- Replace cos(20 deg) with sin(20 deg).

Often, the same edit is done by different error handlers, but the error handlers have extra conditions that test features of the problem. For instance, sign errors are common in equations,

so there is a general sign-error handler whose hint sequence starts out with "Check your signs." This error handler has a low priority. There is a higher priority error handler that checks for a special kind of sign error. If the term with the sign error is the magnitude of a vector pointing opposite one of the coordinate axes, the hint sequence is (problem-specific text is substituted for the bracketed text):

- Think about the direction of the <vector>.
- Perhaps you are confusing the MAGNITUDE of the <vector> with its COMPONENT along the <x or y> axis. Because the vector is parallel to the <x or y> axis but in the negative direction, the projection equation is <equation>.
- Because the vector is parallel to the <x or y> axis and in the negative direction, replace <magnitude variable> with either -<magnitude variable> or <component variable>.

This bottom-out hint tries to do a little instruction. Because students often read only the bottom-out hint (Aleven & Koedinger, 2000), such instruction is included whenever it doesn't make the hint too long and difficult.

Andes does not try to find multiple errors in the same entry. For instance, Andes is not able to find the sign error and the trigonometric error in *Fw_x = Fw*cos(20 deg)* even though it could recognize each error individually. We could easily modify Andes to recognize multiple errors, but it would probably require advanced natural language dialogue planning to compose an understandable, pedagogically effect hint sequence. That would be an interesting topic for future work.

Any offer of What's Wrong help is based on guessing what the student was trying to do and what caused the error. Even error recognizers that give appropriate advice most of the time can sometimes give horribly misleading advice. As mentioned earlier, Andes1 used a much more liberal method for giving What's Wrong Help (Gertner, 1998), and it often provided misleading and even bizarre help. Informal evaluation of the log files suggests that if students get even a few cases of bad advice, they will stop asking for advice entirely, preferring to guess repeatedly instead. Thus, Andes2 includes an error recognizer only when we are confident that almost every application of it is warranted.

If an error is not recognized, Andes2 asks the student what the student was trying to enter. Thus, Andes2 pursues the policy that when there is any doubt about how to analyze an error, it is better to ask than to guess.

In order to implement What's Wrong Help, Andes needs only three sources of knowledge: the knowledge base of error handlers, one solution point per problem, and one set of defined quantities per problem. As mentioned earlier, one could imagine a human author providing this information on a per-problem basis, thus avoiding the knowledge engineering task of implementing a physics problem solver. Indeed, most constraint-based tutoring systems pursue exactly this approach (Mitrovic, Koedinger, & Martin, 2003). They have human authors provide solutions to each problem, and they provide a problem-independent knowledge base of error recognizers.

Next Step Help

The solution graph file for a problem contains a hierarchical plan for solving it. Actually, some problems have multiple solutions—Figure 1 can be solved either with conservation of energy or

with Newton's second law. These solutions can share parts, such as defining the given quantities. Thus, a solution graph file contains a data structure that represents plans for every solution and how they share their parts. This data structure is a directed graph, so it is called the *solution graph* for the problem. For example, Figure 6 is part of the information in the solution graph for the problem of Figure 1.

Although not shown above, there is a branch from step 3 to both step 6 and step 4. This represents that there are alternative solutions. These branches converge later; steps 5 and 6 both point to step 7.

The basic job of Next Step Help is to select one of the steps in the solution graph and give the student the hint sequence associated with that kind of step. Step selection is based on two design principles:

- On a depth-first, top-to-bottom walk of the solution graph, select the first step that the student has not yet done.
- At a branch in the solution graph, select the branch that has the largest proportion of steps done already by the student.

For instance, suppose the student has done all of steps 1, 2 and 3, and has also done step 6.4. Then Andes will walk all the way through steps 1, 2 and 3 without finding a step to hint. At the branch, it will count the number of steps done in 6 vs. the number of steps done in 4 and 5. It will see that 6 has one of its steps done whereas the other branch has none of its steps done. Thus, it takes the step 6 branch. It sees that step 6.1 has not been done, so it selects step 6.1 for hinting.

Once Andes has selected a step to hint, it generates a hint sequence. The hint sequence is comprised of two subsequences. The second subsequence is a hint sequence that is specific to the step itself. It typically consists of a reminder hint, a teaching hint and a bottom-out hint. The first subsequence is generated only when the selected step is part of a main step. For instance, step 1 is not part of a main step but step 6.1 is part of main step 6. The first subsequence makes sure that the student is aware of the main step. As discussed earlier, this might involve either eliciting the main step from the student ("What quantity is the problem seeking" then "What major principle should you apply to find it?"), or telling the main step to the student ("Let's apply Newton's second law to the car along the x-axis."), or referring backwards in the dialogue ("Why don't you continue applying Conservation of Energy to the car.").

In order to make this policy work, Andes must map the student's entries onto steps in the solution graph so that it can mark the steps "done." This is easily done for non-equation entries, but it is extremely difficult for equations. For instance, if the student enters *Fw_x = -m*g*sin(20 deg)*, then Andes must somehow figure out that this maps to steps 5.3.2 (writing *Fw_x = Fw*cos(250 deg)*) and step 5.4 (writing *Fw = m*g*). Andes1 solved this problem by precomputing all possible algebraic combinations of the steps, but this became infeasible as problems became more complex and numerous. For many months, this appeared an insurmountable problem.

1. Draw the body, which defines the mass variable, mc.
2. Define coordinate axes, rotated 20 degrees
3. Define given quantities
   - 3.1. Draw the displacement, d
   - 3.2. Enter the given value of displacement, d = 20 m
   - 3.3. Enter the given value of mass, mc = 2000 kg
4. Apply translational kinematics and get v1^2 = 2*a*d
   - 4.1. Draw the vector diagram
     - 4.1.1. Draw the initial velocity, vi
     - 4.1.2. Draw the final velocity, vf
     - 4.1.3. Draw the acceleration, a
     - 4.1.4. Draw the displacement, d (shared with step 3.1)
   - 4.2. Apply the fundamental principle: vf_x^2=vi_x^2+2*a_x*d_x
   - 4.3. Project the vectors onto the x-axis
     - 4.3.1. Apply projection: vi_x =0
     - 4.3.2. Apply projection: vf_x= -v2
     - 4.3.3. Apply projection: a_x= -a
     - 4.3.4. Apply projection: d_x= -d
5. Apply Newton's second law and get a= -g*cos(200 deg)
   - 5.1. Draw a free-body diagram
     - 5.1.1. Draw the force of gravity, Fw
     - 5.1.2. Draw the normal force, Fn
     - 5.1.3. Draw the acceleration, a (shared with step 4.1.3)
   - 5.2. Apply the fundamental principle: mc*a_x = Fw_x + Fn_x
   - 5.3. Project the vectors onto the x-axis
     - 5.3.1. Apply projection: Fn_x = 0
     - 5.3.2. Apply projection: Fw_x = Fw*cos(250 deg)
     - 5.3.3. Apply projection: a_x = -a
   - 5.4. Apply the weight law, Fw = mc*g
6. Apply conservation of energy and get vf^2 = 2*g*d*sin(20 deg)
   - 6.1. Define the kinetic energy at time T1: K1
   - 6.2. Define the potential energy due to gravity at T0: P0
   - 6.3. Apply the fundamental principle, P0 = K1
   - 6.4. Define height, h
   - 6.5. Apply the definition of gravitational potential energy: P0 = mc*g*h
   - 6.6. Apply trigonometry: h = d*sin(20 deg)
   - 6.7. Draw the final velocity, vf (shared with 4.1.2)
   - 6.8. Apply the definition of kinetic energy, K1 = _ * mc*vf^2
7. Solve the system of equations for vf and get vf = 11.59 m/s
8. Enter the value of vf into the answer box

Fig. 6. The solution graph for the problem of Figure 1.

Fortunately, a solution was found (Shapiro, 2005). The algorithm is called indy check because its main step is to check the independence of a set of multidimensional vectors. The vectors are the gradients of the solution graph equations and the student's equation. The algorithm computes the gradient of an equation by taking the partial derivative of each variable in

Table 1
Applying the indy check algorithm to a simple case

	Function f	∂f/∂m	∂f/∂g	∂f/∂Fw	∂f/∂Fw_x	gradient
A	Fw_x –Fw*cos(250º)	0	0	–cos250º	1	(0, 0, 0.342, 1)
B	Fw – mc*g	–g	–mc	1	0	(–9.8 ,–2000, 1,0)
C	mc – 2000	1	0	0	0	(1, 0, 0, 0)
D	g – 9.8	0	1	0	0	(0, 1, 0, 0)
S	Fw_x+mc*g*sin(20º)	g*sin20º	mc*sin20º	0	1	(3.352, 684, 0, 1)

the equation, evaluating the resulting expressions at the solution point, and using the resulting numbers as components of the gradient. In order to find a set of solution graph equations that can be combined to form the student's equation, the algorithm finds a set of gradients that can be combined via a weighted sum to form the gradient of the student's equation. The algorithm outputs all such sets.

As an example, suppose that the student's equation is *Fw_x = -mc*g*sin(20 deg)* and that the solution graph only has the equations *Fw_x = Fw*cos(250 deg), Fw = mc*g, mc = 2000* kg and *g = 9.8* m/s^2. Table 1 shows the gradients of each of the solution graph equations, A through D, and the student's equation, S. The equations are converted to functions of the form left-side – right-side. We take partial derivatives along each of the 4 dimensions, then plug in the solution point, which is *m = 2000*, *g = 9.8*, *Fw = 19,600* and *Fw_x = −6703.5948 N*. Now the algorithm searches for a subset of vectors {A, B, C, D} such that they combine linearly to form S. It finds that S = A−0.342*B (this is a vector equation), so it reports that the student's equation is an algebraic combination of the equations corresponding to A and B. Andes marks *Fw_x = Fw*cos(250 deg)* and *Fw = m*g* as "done" in the solution graph, thus preventing Next Step Help from selecting them when it is looking for a step to hint.

Shapiro (2005) showed that this algorithm is correct except for a special case. Although arguably rare in general, the special case shows up rather frequently in the particular problems that Andes uses. Heuristics are used to detect the special case, and it has caused no visible performance degradations.[7]

Any method of selecting a step to hint should probably consider two criteria: avoiding hinting steps that the student has already done, and trying to hint a step that is relevant to the student's current goals. Let us compare Andes to other systems along these two dimensions.

Andes has more trouble than most systems at identifying which steps have been done, because the solution graph contains primitive steps that the student can enter as a combination. This was a major impediment in the development of Andes' Next Step Help which indy check

[7] The most common version of the special case occurs when the solution graph has equations with sin(_) or cos(_), and it contains _=n*90° where n is an integer. When this occurs, indy check sometimes fails to include _=n*90° as one of the solution graph equations underlying the student's equation. This would fool Andes into thinking that the student had not yet entered _=n*90°. However, students don't actually write such equations. The equations are entered as a side-effect of drawing vectors. Since Andes knows exactly what vectors have been drawn, it knows not to give Next Step Help on a vector that has already been drawn no matter what indy check says. Other versions of the special case could occur in principle, but we have yet to see one that has not been caught by the heuristics and has caused a failure of Next Step Help.

resolved. Any task domain and user interface that allow students to enter combinations of primitive steps will probably face similar challenges.

Compared to other systems, Andes expends much less effort on trying to select a step that is appropriate to the student's current goals. The general problem is called plan recognition: given a sequence of observable actions, find a selection of hierarchical plans from a library that best fits the observed sequence, assuming that some plans are still in progress and hence that the student still intends to achieve those goals. Plan recognition has mostly been studied for intelligent user interfaces and natural language assistants, where it is known to be a difficult problem to solve even heuristically (e.g., Carberry, 1990). The problem is somewhat simpler in tutoring systems, because the system can assume that the student's top level goal is to solve the problem and that their plans are mostly the appropriate ones for the task domain. Nonetheless, plan recognition has still proved challenging for intelligent tutoring systems (Woodroffe, 1988). Some authors consider the challenge insurmountable (Self, 1990).

Unlike many task domains, the plans in Andes are weakly ordered. In particular, the order in which steps are done in Andes doesn't matter, except that variables must be defined before they are used in equations, and the set of equations must be complete before it is solved and the answer is entered. Thus, both Andes and the student often have considerable freedom in selecting a next step. As a continuation of the example from the beginning of this section, suppose that the student just completed steps 6.2 and 6.4, and had previously completed all of steps 1, 2 and 3. Thus, steps 6.1, 6.5, 6.6 and 6.7 are all legal. Intuitively, if the student has just defined the variables needed in an equation and has asked for Next Step Help, then the student has probably forgotten the equation and wants a hint on it, so 6.5 would be a good choice of a step to hint. A tutoring system that has great plan recognition capabilities would make this inference, and thus select step 6.5 for Next Step Help. This would satisfy the requirement that hints address the student's current goals.

Andes1 used probabilistic reasoning for plan recognition, and in a situation like this, it would indeed select step 6.5. However, the pattern of steps completed by the student was usually more random than in this example. In such cases, Andes1 tended to hint a step that was too far along in the solution. This made its hints seem confusing or even mystical. "How am I supposed to have known that?" was a common complaint. Andes2 took the more conservative strategy of hinting the first incomplete step in its own ordering of steps. Sometimes it hinted steps, such as 6.1, that were earlier than one that the student was working on. This made it seem pedantic. However, if students keep asking for Next Step Help, it eventually did hint the step they are seeking help on. Asking for Next Step Help repeatedly may have slowed students down, but it also exposed them to a rational sequence of steps. In the case of Andes1, repeatedly asking for Next Step Help yielded step selections that just got more and more confusing. We prefer step selections that are occasionally pedantic to ones that are occasionally confusing.

There are two situations where Andes2 does a limited form of plan recognition. Both occur when a problem has mutually exclusive solutions. (1) If the student has entered steps from one or more of the solutions, it hints steps from the solution that the student has made the most progress on, breaking ties in favor of the shortest solution. (2) If the student has not yet started any of the mutually exclusive solutions, Andes asks the student what plan the student would like to follow. In particular, it elicits a choice from the student during the first part of the hint

sequence by asking "What quantity is the problem seeking?" and "what principle should be applied to find it?"[8]

The lesson we learned is that when the task domain is weakly ordered, as is physics problem solving (VanLehn et al., 2004), most students who ask for Next Step Help have been entering steps in irrational orders. A plan recognition system that expects students to follow plans in the canonical depth-first, left-to-right order will be sorely misled by their performance. When the student's steps are randomly ordered, it is better to just ignore them and hint steps from the tutor's favorite ordering. Andes2 takes this a bit further by applying this preference even in the infrequent cases where students' steps are in a rational ordering that is different from the tutor's ordering. In such cases, Andes' impoverished plan recognition is an irritation but probably not a serious impediment to learning.

Solution graph generation

For each problem, Andes precomputes the solution graph using a knowledge-rich problem solver. The literature on expert-novice differences in physics problem solving, which was briefly reviewed earlier, suggested that experts organize their solutions around major principles, such as Newton's second law, Conservation of Energy or rotational kinematics. For each such principle, the Andes KB has a problem solving method. This section first describes problem solving methods (PSM), then how they are used to solve problems.

Every PSM incorporates a multi-step, hierarchical plan for generating an application of a major principle combined with associated minor principles. For instance, Figure 7 sketches the PSM for applying Newton's second law. When this PSM is applied to the problem of Figure 1, it produces main step 5 in the solution graph sketched earlier. The PSM also indicates what kinds of quantities the set of equations it generates will contain. The PSMs are similar to the schemas used in the Mecho physics problem solver (Bundy et al., 1979), the Fermi problem solver (Larkin et al., 1988) and human problem solving (Larkin, 1983).

In addition to the PSM, the knowledge base contains many smaller pieces of knowledge, represented as Horn clauses. These take care of inferences such as "If <object1> pulls on <object2> then there is a tension force on <object2> due to <object1>."

A problem is solved via a form of state-space search. Each state contains (1) a set of equations that have been generated already, (2) a set of quantities that are "known" and (3) a set of quantities that are "sought." In the initial state, the values of these slots are: (1) there is an equation for each of the given quantities, (2) the given quantities comprise the set of known

To apply Newton's second law to <body> at <time> along <axis>:
1. Draw a free-body diagram for <body> at <time> including <axis>
   1.1. For each force on <body> at <time>, draw it.
   1.2. Draw the acceleration of <body> at <time>
2. Write the fundamental equation in terms of components along <axis>
3. For each vector (i.e., acceleration and each force), write a projection equation for the vector along <axis>
4. For each minor principle of the form <force magnitude> = <expression> where the force is one of the ones on <body> at <time>, write the minor principle's equation.

Fig. 7. The PSM for Newton's second law.

quantities, and (3) the quantities sought by the problem statement comprise the set of sought quantities.

The search proceeds by iteratively choosing a PSM that contains one of the sought quantities and applying the PSM. When a PSM has been applied and a set of equations has been generated, the indy check algorithm is called in order to insure that none of the new equations can be generated from the equations already in the state. This insures that the equations in the state are always independent—none can be generated by algebraic combinations of the others. If the indy check succeeds, then a new state is produced by adding the new equations to it and by updating the sets of known and sought quantities. The search continues until the set of sought quantities becomes empty. The final state of each solution should contain N known quantities and N independent equations, so the equation solver is called to solve them and produce the solution point.

The search is carried through to exhaustion, thus generating all possible solutions to the problem. In the case for Figure 1, this would result in two final states. These are merged to produce the solution graph data structure which was shown earlier.

EVALUATIONS

Andes was evaluated in the U.S. Naval Academy's introductory physics class every fall semester from 1999 to 2003. This section describes how the 5 evaluations were conducted and their results.

The evaluation method

Andes was used as part of the normal Academy physics course. The course has multiple sections, each taught by a different instructor. Students in all sections take the same final exam and use the same textbook but different instructors assign different homework problems and give different hour exams, where hour exams are in-class, hour-long exams are given approximately monthly. (Hour exams are often called "quizzes" or "midterms" at other schools.) In sections taught by the authors (Shelby, Treacy and Wintersgill), students were encouraged to do their homework on Andes, print out their solutions and hand them in. All students had Academy-issued computers in their dormitory rooms, so all students had easy access to Andes on sufficiently powerful computers. Each year, the Andes instructors recruited some of their colleagues' sections as Controls. Students in the Control sections did the same hour exams as students in the Andes section.

Control sections did homework problems that were similar but not identical to the ones solved by Andes students. The instructors reported that they required students to hand in their homework, and credit was given based on effort displayed. Early in the semester, instructors marked the homework carefully in order to stress that the students should write proper derivations, including drawing coordinate systems, vectors, etc. Later in the semester, homework was graded lightly. Occasional short comments were given such as "Draw FBD", "Axes" or "Fill out F=MA by components" in order to continue the emphasis on proper derivations. In some classes, instructors gave a weekly quiz consisting of one of the problems from the

preceding homework assignment. This encouraged students to both do the assignments carefully and to study the solutions that the instructor handed out.

The same final exams were given to all students in all sections. The final exams comprised approximately 50 multiple choice problems to be solved in 3 hours. The hour exams had approximately 4 problems to be solved in 1 hour. Thus, the final exam questions tended to be less complex (3 or 4 minutes each) than the hour exam questions (15 minutes each). On the final exam, students just entered the answer, while on the hour exams, students showed all their work to derive an answer. On the hour exams, students were scored on (a) their vector drawing, (b) whether they defined variables and/or used standard, legible variable names and used them in preference to numbers in equations, (c) correctness of the equations written, (d) the correctness of the answer. Students were informed of the grading rubric well in advance.

The instructors believe that the open-response, complex problems of the hour exams were a more valid assessment of students' competence, but the multiple-choice final exam was necessary for practical reasons. The hour exam results will be reported first.

Hour exams results

It is important to check that the prior knowledge and skill of the Andes students was approximately the same as the prior knowledge and skill of the Control subjects. Students were not assigned randomly to condition, although it could be argued that they were assigned randomly to sections. Students were not given an appropriate pre-test. Thus, the standard methods for insuring equal prior competence were not used. However, over many years of experience, instructors have found that the students' physics grades are strongly correlated with their major and their overall college grade point average (GPA). The engineering majors are required to take a Statics course concurrently with their physics course, and the two courses share content, so the engineers tend to get high grades in their physics course. Also, the engineering and science majors tend to have better mathematical preparation than the other majors. Thus, one method of checking the equality of prior competence is to check that the two conditions have equal distributions of majors and equal mean GPAs.

In order to check for equivalent distribution of majors, the majors were classified as either Engineering, Science or Other. For instance, Table 2 shows the distribution of majors in the 2003 evaluation. A 3x2 Chi-squared test confirmed that the two conditions had the same distribution of majors ($p = 0.52$). The GPAs were also the same, according to a t-test on the means ($p=0.70$). Thus, these measures suggest that students in the Andes and Control conditions had equivalent prior knowledge and skill. Similar checks were made for every year, and equivalence was found in every case.

Table 2
Distribution of majors in the 2003 hour exam evaluation

	Engineering	Science	Other	Total
Andes	56	12	25	93
Control	24	9	11	44

Table 3
Results from hour exams evaluations

Year	**1999**	**2000**	**2001**	**2002**	**2003**
Number of Andes students	173	140	129	93	93
Number of control students	162	135	44	53	44
Andes mean (stan. dev)	73.7 (13.0)	70.0 (13.6)	71.8 (14.3)	68.2 (13.4)	71.5 (14.2)
Control mean (stan. dev)	70.4 (15.6)	57.1 (19.0)	64.4 (13.1)	62.1 (13.7)	61.7 (16.3)
P(Andes = Control)	0.036	< .0001	.003	0.005	0.0005
Effect size	0.21	0.92	0.52	0.44	0.60

Table 3 shows the hour exam results for all 5 years. It presents the mean score (out of 100) over all problems on one or more exams per year. In all years, the Andes students scored reliably higher than the Control students with moderately high effect sizes.[9] The 1999 evaluation had an effect size that was somewhat lower, probably because Andes had few physics problems and some bugs, thus discouraging students from using it. It should probably not be considered representative of Andes' effects, and will be excluded from other analyses in this section.

In order to calculate an overall effect size, it was necessary to normalize the scores across years so that they could be combined. The raw exam scores for each exam were converted to z-scores, and then the data points from years 2000 through 2003 were aggregated. The mean Andes scores was 0.22 (standard deviation: 0.95) and the mean score of Control students was -0.37 (0.96). The difference was reliable ($p < .0001$) and the effect size was 0.61.

It is often the case that educational software helps some types of students more than other types. For instance, it is often found that students with high prior knowledge learn equally well from the experimental and control instruction, but students with low prior knowledge learn more from the experimental instruction than from the control instruction. In order to check for this so-called aptitude-treatment interaction, we compared linear regression models of GPA vs. hour exam scores. We used the aggregated z-scores for the exams over years 2000 through 2002, because we lacked GPAs of student in the 2003 Control condition. Figure 8 shows the results. The regression lines of the two conditions are nearly parallel, indicating that there was little aptitude-treatment interaction.

In order to check for a different kind of aptitude-treatment interaction, we divided the students into engineering majors, science majors and other majors. For each group, we compared the mean exam scores of Andes students to Control students. However, the GPAs of the groups were statistically different. In order to factor out the differences in exam performance due to GPA differences, three regression equations were computed, one for each group of majors, that predicted the exam z-score of the students based on their GPA. We then calculated a residual score for each student by subtracting the predicted z-score from the raw z-score. Table 4 shows the means of the raw z-scores and the residual z-scores, with standard deviations in parentheses, and the effect size for the Andes vs. Control comparison based on the residual z-scores. The differences in benefits across majors were small.

[9] Effect size was defined as (AndesMean – ControlMean) / ControlStandardDeviation.

Table 4
Which groups of majors benefited the most from Andes?

	Engineering	**Science**	**Other**
Number of students	327	186	315
Andes: Raw score means (stand. dev.)	0.52 (0.86)	0.22 (0.94)	-0.12 (0.95)
Control: Raw score means (stand. dev.)	-0.14 (0.96)	-0.24 (1.00)	-0.59 (0.88)
Andes: Residual score means (s.d.)	0.15 (0.73)	0.16 (0.77)	0.18 (0.84)
Control: Residual score means (s.d.)	-0.32 (0.89)	-0.33 (0.86)	-0.30 (0.77)
Effect size for Residual scores	0.53	0.57	0.62
P-value for residual score comparison	<0.0001	<0.0001	<0.0001

From these comparisons, it is clear that the aptitude-treatment interactions are small. For both high and low GPA students, and for students of all majors, Andes is more effective than paper and pencil homework with an effect size of about 0.6.

The physics instructors recognize that the point of solving physics problems is not to get the right answers but to understand the reasoning involved, so they used a grading rubric for the hour exams that scored the students' work in addition to their answers. In particular, 4 subscores were defined (weights in the total score are shown in parentheses):

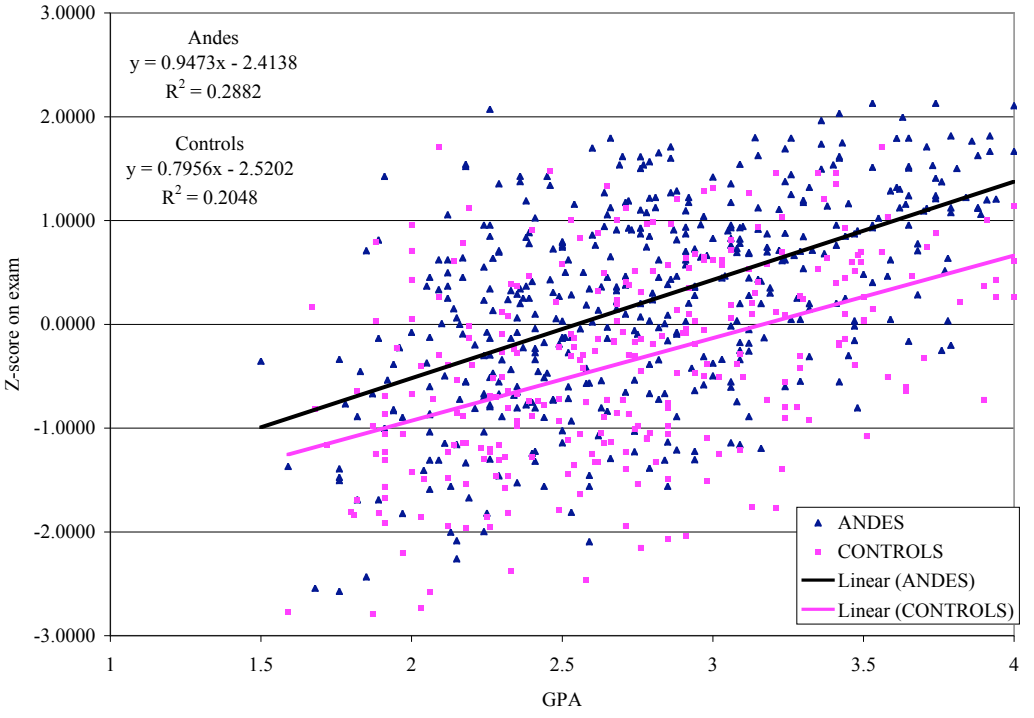

Fig. 8. Aptitude-treatment interaction.

- *Drawings:* Did the student draw the appropriate vectors, coordinate systems and bodies? (30%)
- *Variable definitions:* Did the student use standard variable names or provide definitions for non-standard names? (20%)
- *Equations:* Did the student display major principle applications by writing their equations without algebraic substitutions and otherwise using symbolic equations correctly? (40%)
- *Answers:* Did the student calculate the correct numerical answer with proper units? (10%)

Andes was designed to increase student conceptual understanding, so we would expect it to have more impact on the more conceptual subscores, namely the first 3. Table 5 shows the effect sizes, with p-values from two-tailed t-tests shown in parentheses. Results are not available for 2001. Two hour exams are available for 2002, so their results are shown separately.

There is a clear pattern: The skills that Andes addressed most directly were the ones on which the Andes students scored higher than the Control students. For two subscores, Drawing and Variable definitions, the Andes students scored significantly higher then the Control students in every year. These are the problem solving practices that Andes requires students to follow.

The third subscore, Equations, can also be considered a measure of conceptual understanding. However, prior to 2003, Andes was incapable of discriminating between good and poor usage of equations, so it is not surprising that the Andes and Control students tied on the Equations subscore in years 2000 and 2002. In 2003, Andes gave students warnings and points off on their problem scores if their first use of a major principle was combined algebraically with other equations. Although Andes could have required students to obey this problem solving practice, it only suggested it. This may explain why the Andes students still did no better than the Control students on the Equations subscore in 2003.

The Answers subscore was the same for both groups of students for all years even though the Andes students produced better drawings and variable definitions on those tests. This suggests that the probability of getting a correct answer depends strongly on other skills, such as algebraic manipulation, that are not measured by the more conceptual subscores and not emphasized by Andes. The tied Answer subscores suggest that the Andes students' use of the equation solving tool did not seem to hurt their algebraic manipulation on the hour exams.

In summary of the hour exam results, Andes students tended to learn more than Control students, with an overall effect size of 0.61. Andes was nearly equally beneficial for engineering majors, science majors and other majors, and for both high-GPA and low-GPA students. Breaking down the hour exam scores into subscores showed that Andes students scored higher than Control students on drawing and variable usage, which are two fundamental conceptual skills emphasized by Andes. Andes does not teach algebraic manipulation skills and has only

Table 5
Hour exam subscore effect sizes (and p-values for t-tests)

	2000	**2002a**	**2002b**	**2003**	**Average**
Drawings	1.82 (<.001)	0.49 (.003)	0.83 (<.001)	1.72 (<.001)	1.21
Variable definitions	0.88 (<.001)	0.42 (.009)	0.36 (.026)	1.11 (<.001)	0.69
Equations	0.20 (.136)	0.12 (.475)	0.30 (.073)	-0.17 (.350)	0.11
Answers	-0.10 (.461)	-0.09 (.585)	0.06 (.727)	-0.20 (.154)	-0.08

recently started suggesting how to use symbolic equations properly, so it had little apparent advantage over paper for teaching these skills.

Final Exam scores

Although the hour exams used complex, open-response problems instead of multiple choice problems, and this allowed instructors to use a grading rubric that valued conceptually clear derivations, it could be that some Control students did not believe that the grading rubric would be applied to the exams, and thus did not "show their work" on the exams. It could also be that the Control instructors put less stress than the Andes instructors on conceptually clear derivations. Thus, it is important to analyze the final exam results, because the finals required only answers and not derivations.

A final exam covers the whole course, but Andes does not. However, its coverage has steadily increased over the years. In 2003, Andes covered 70% of the homework problems in the course. This section reports an analysis of the 2003 final exam data.

First, we need to check that the Andes sections have a representative sample of the whole course population. As mentioned earlier, GPA and major seem to be the best measures of prior knowledge and skill. The Andes students' mean GPA was 2.92 (SD = 0.58), which was marginally significantly higher ($p = 0.0662$) than the non-Andes students' mean GPA of 2.80 (SD = 0.55). Moreover, the distribution of majors among the Andes students was statistically different from the distribution of majors among the non-Andes students ($p < .0001$, 3x2 Chi-squared test). In particular, there were relatively more engineers than in the Andes sections than in the non-Andes sections. Thus, it appears that the Andes students were *not* representative of the whole population. Thus, we had to use statistical techniques to factor out effects of the higher prior knowledge of the Andes students.

For each group of majors, we regressed the final exam scores of all students in the course against the students' GPAs. Of the 931 students, we discarded scores from 19 students with unclassifiable majors or extremely low scores. This yielded three statistically reliable 2-parameter linear models, one for each type of major. Each model expresses the relationship between general competence and the final exam score. For each student, we subtracted the exam score predicted by the linear model from the student's actual score. This residual score represents how much better or worse this student scored compared to the score predicted solely on the basis of their GPA and their major. That is, the residual score factors out the students' general competence. The logic is the same as that used with an ANCOVA, with GPA and major serving as covariates instead of pre-test scores.

Using these residual scores, we evaluated Andes' impact on students in each of the 3 groups of majors. As Table 6 indicates, the residual scores of the engineering and science majors were not statistically different with Andes than with paper homework. However, the other majors did learn more with Andes than with paper homework ($p<.016$; effect size = 0.5). Over all students, the Andes students mean residual score was higher than the mean residual score of the non-Andes students (effect size = 0.25; $p=0.028$).

Thus, Andes students overall learned significantly more than non-Andes students. The overall effect size was somewhat smaller for the final exam (0.25) than the hour exams (0.61). This may be partially due to the fact that roughly 30% of the final exam addressed material not covered by Andes. It may also be partially due to the format of the final exam. The final exam

Table 6
Residual scores on the 2003 final exam

	Engineers	**Scientists**	**Others**	**All**
Number of Andes students	55	9	25	89
Number of non-Andes students	278	142	403	823
Andes students mean (stand. dev.)	0.74 (5.51)	1.03 (3.12)	2.91 (6.41)	1.38 (5.65)
Non-Andes students mean (s.d.)	0.00 (5.39)	0.00 (5.79)	0.00 (5.64)	0.00 (5.58)
p(Andes=non-Andes)	0.357	0.621	0.013	0.028
Effect size	0.223	0.177	0.520	0.25

had students enter only their answers, whereas the hour exams had students show their work, which allowed graders to assess their conceptual understanding more directly.

Though we were gratified to see that Andes students learned more than non-Andes students, we were not surprised that Andes had little effect on the learning of the engineering and science majors, for two reasons. (1) In many studies, instructional manipulations tend to affect only the less competent students' learning, because highly competent students can usually learn equally well from the experimental and the control instruction (Cronback & Snow, 1977). (2) The engineering majors were concurrently taking a course on Statics, which has very similar content to the physics courses. This dilutes the effect of Andes, since it affected only their physics homework and not their Statics homework.

Questionnaire results

At the end of every evaluation, the Andes students filled out a questionnaire. The questions asked students about their overall acceptance of Andes and their ratings of individual features. This section discusses both kinds of results, starting with overall acceptance.

Table 7
Questionnaire results. Top number=favorable %. Bottom number=unfavorable %.

	2001	**2002**	**2003**
At 2 weeks, if you had not been required to use Andes, would you have used it?	40.6 59.4	56.0 44.0	34.7 65.3
At 12 weeks, if you had not been required to use Andes, would you have used it?	32.7 67.3	46.0 54.0	37.8 62.2
Was Andes more effective for doing homework vs. doing homework in a more traditional fashion and turning it in?	48.0 35.0	52.7 37.4	51.3 35.5
Do you think you learned more or less using Andes than if you had done the same exercises with pencil and paper?	35.0 39.8	46.2 40.7	45.3 38.7
The techniques learned with Andes helped me in solving test problems.	48.0 29.4	58.9 26.7	46.7 29.9
All SP211 students should be required to use Andes.	43.1 37.3	48.3 34.8	35.5 44.7
Average	39.9 46.7	51.4 39.6	41.9 56.0

Table 8
Responses (in percentages) to Autumn 2003 questions about features

	Agree	**Neutral**	**Disagree**
Do you feel that the immediate feedback feature of Andes helped you to complete correct solutions	71.4	14.3	14.4
Andes would be a more effective learning tool without the solve function.	14.0	18.6	67.4
Andes response to "What's wrong with that?" queries was helpful.	55.3	19.7	22.4
Andes response to "light bulb" help [Next Step Help] was helpful.	51.3	25.0	21.1
Andes response to "explain further" was helpful.	55.3	21.1	22.4
The "help topics" in the help menu was helpful	21.1	21.1	42.1

Table 7 shows how student acceptance changed over the years. Most questions had student select one of three answers, such as Agree-Neutral-Disagree or More-Equal-Less. Thus, Table 7 presents two numbers per cell. The top number is the percentage of students giving the response most favorable to Andes. The bottom number is the percentage of students giving the response least favorable to Andes. The difference between 100% and the sum of these two numbers is the percentage of students selecting the middle response (e.g., Neutral, Equal). In general, about 40% to 50% of the students seemed to like Andes and about 40% to 55% seemed to dislike it.

Students were also asked, "Was Andes more effective or less effective than other learning tools used at the USNA like WebAssign, Blackboard or the on-line Chemistry assignments. Circle the tool used for comparison." Across all three years, 69.2% of the students said Andes was more effective, 14% said Andes was less effective and the rest felt Andes was equally effective. This question's response suggests interpreting the low acceptance rates of Table 7 with caution. It may be that some students are not sold on the idea of using electronic homework helpers and would prefer to use paper and pencil. However, if required to use a homework helper, they prefer Andes over the more conventional ones.

Andes was not uniformly popular. When questionnaire results were broken down by majors, the engineers liked Andes the least. For instance, on the 2003 question "At 12 weeks, would you have used Andes if it were not required," 30% of the engineering majors said yes, versus 57% of the science majors and 53% of the other majors.

The questionnaire also asked students to evaluate individual Andes features. Table 8 shows responses from Autumn 2003 to the relevant questions. Most students liked flag feedback and the equation solving tool, about half the students found What's Wrong Help and Next Step Help useful, and a few students found the help menu useful that contained the equation cheat sheet and other unintelligent help.

Discussion of the evaluations

This section summarizes the evaluation results and compares them to those from Koedinger et al. (1997). That study tested a combination of an intelligent tutoring system (PAT) and a novel curriculum (PUMP). A revised version of this combination is now distributed by Carnegie Learning as the Algebra I Cognitive Tutor (www.carnegielearning.com). There are only a few in-school, semester-long, controlled studies of intelligent tutoring systems in the open literature,

and the widely-cited Koedinger et al. (1997) study is arguably the benchmark study against which others should be compared.

Using experimenter designed tests of conceptual understanding, Koedinger et al. (1997) found effect sizes of 1.2 and 0.7. The Andes hour exams were intended to measure both conceptual understanding and algebraic manipulation skills. When the hour exam scores were broken down into conceptual and algebraic components, Andes students scored significantly higher on the conceptual components that Andes addressed (Diagrams: effect size 1.21; Variables: effect size 0.69). In this respect, the results from the two studies are remarkably similar.

Standard tests may be less sensitive to conceptual understanding due to their multiple-choice format and different coverage. If so, one would expect them to be less sensitive to the benefits of tutoring. Indeed, the Koedinger et al., (1997) evaluation found smaller effects when using multiple-choice standardized tests: 0.3 for each of two tests. These results are comparable to our results for the multiple-choice final exam, where Andes students scored higher than non-Andes students with an effect size of 0.25. The smaller effect size was probably partly due to the facts that 30% of the homework problems are not yet covered by Andes.

Thus, the Andes evaluations and the Koedinger et al. (1997) evaluations have remarkably similar effect sizes: 1.2 and 0.7 on experimenter-designed tests and 0.3 on standard tests.

The Andes evaluations differed from the Koedinger et al. (1997) evaluation in a crucial way. The Andes evaluations manipulated only the way that students did their homework—on Andes vs. on paper. The evaluation of the Pittsburgh Algebra Tutor (PAT) was also an evaluation of a new curriculum, developed by the Pittsburgh Urban Mathematics Project (PUMP), which focused on analysis of real world situations and the use of computational tools such as spreadsheets and graphers. It is not clear how much gain was due to the tutoring system and how much was due to the reform of the curriculum. In our evaluation, the curriculum was not reformed. Indeed, the Andes students and the Control students were on the same course and used the same textbook. The gains in our evaluation are a better measure of the power of intelligent tutoring systems per se.

Moreover, the Andes students' self-reported times for doing their homework (approximately 3 hours per week) seem no higher than would be expected if they were doing their homework on paper. Thus, it seems that Andes can raise students' grades and their understanding of physics without requiring any more studying time from the students and without requiring instructors to reform their curriculum. This is extremely good news.

LESSONS LEARNED

This section describes what has been learned about the ITS development process and about human learning.

Lessons learned about systems development

Andes teaches introductory physics, but there are many similar topics that have similar cognitive task analyses. Indeed, any topic where problems are solved by producing a system of equations

and solving them is amenable to an Andes style treatment. This includes many topics in engineering and science.

In order to build such a tutoring system for use in the real world, we would do many things differently. This section presents the development plan that seems best given our experiences so far. It would have 3 phases:

1. Develop a tutor that has flag feedback, an integrated mathematical package, and many problems and videos, but offers no other help.
2. Add What's Wrong Help.
3. Add Next Step Help.

The key idea is to use the phase 1 tutor in a course, and then add features based on data collected from the log files. In order to get the phase 1 tutor used in a course, it should have enough problems to support several months' of homework. Students and faculty are rarely interested in learning new software if they can only use it for a week or two. Students like flag feedback and the built-in mathematical tools, so including them in the phase 1 version should facilitate getting the phase 1 tutor accepted in a course. Moreover, both features probably save students time without reducing their learning (Anderson et al., 1995).

For development of the phase 2 tutor, error handlers should be developed from the phase 1 tutor's data. Phase 1 log files should be mechanically sorted in order to identify high-frequency incorrect equations. Human graders should mark each equation with its error(s) and suggested hint sequence. From these, programmers should write error handlers. The error handlers should be tested on the log file equations to make sure that they do not misclassify errors.

In order to collect data for the phase 3 tutor, it would be helpful to determine when students typically ask for Next Step Help and what can be said to them then. This can be done in many ways, but perhaps the best would be to install a Help button on the phase 2 tutor that connects the student to a human tutor. To handle the load just before assignments are due, multiple human tutors would be needed. They would uses software similar to that used by call centers for technical support. A tutor pulls a help request from the queue, views a copy of the student's screen, and uses text messaging to help the student past the impasse. (A similar system was used during Andes development, albeit with only one tutee at a time.) The log files from these human tutoring episodes would be used to develop the phase 3 tutor's Next Step Help module. The human tutoring would also provide a valuable service to students.

There are several major design decisions that must be made when building an Andes-style tutor. The most important one is whether to generate the solution graph files by hand or to generate them via a rule-based problem solver, as Andes does. In order to generate the solution graph files by hand, authors should use the phase 1 tutor's user interface to solve every problem in every possible way. The authors must be careful to enter only equations that are applications of principles, and not algebraic combinations of applications of principles. The authors should mark each equation with a description of the principle application, as these are needed by Next Step Help to give hints on that equation. Authors may make mistakes during this complex, lengthy task. The mathematical package solves the authors' equations, so it may detect some errors.

On the other hand, one can implement a problem solver to solve all the problems. Errors made by the knowledge engineer while implementing the problem solver show up as missing or incorrect solutions, so they are more easily detected by domain experts checking the solution

graph files. In order to facilitate such checking, it is helpful if the phase 1 tutor can display each solution from the solution graph problem, one by one.

Frankly, we would probably always develop a problem solver. It is not as difficult as it once was, assuming that one uses Prolog or another programming languages developed for the purpose. Moreover, the development uncovers tacit domain knowledge (Shute & Psotka, 1996). The same tacit knowledge must be discovered and explicitly represented when hand-generating solution graph files, but we suspect it would not be represented with the same generality and elegance as it would be when incorporating it into a problem solver's knowledge base. The problem solver can also be used to look for strategies that could be explicitly taught to students by Next Step Help (VanLehn et al., 2004).

The second major design decision is how to provide the required mathematical expertise. We chose to write our own mathematical package because an appropriate package was not available when we started the project. (Shapiro, 2005) In a new development project, one would have to carefully assess both the technical and licensing issues involved in using an externally supported mathematical package, such as Mathematica or TK solver.

The third major design decision is what kind of problem scoring to provide, if any. Automated scoring of student's homework is popular with both students and faculty. However, given that feedback and hints virtually guarantee that students will produce a complete and correct derivation, it is not clear what the automated score should be based on. Andes provides grading rubrics designed by the instructors, but even they are not always happy with the rubrics' assessments. Discussions with other instructors suggest that these rubrics do not meet their instructional priorities. It would be wise to design flexible grading rubrics into the phase 1 tutor, and to refine them on the basis of the log files.

The most important system development lesson has been saved for last. It is absolutely critical that the instructors be active participants in the design process at every step of the way. The USNA instructors were essential to the success of Andes.

Lessons learned about human learning

The Andes evaluations were not intended to test hypotheses about human learning. Nonetheless, there are several hypotheses that are consistent with the evaluation results. It is worth articulating them so that they can be tested with carefully controlled laboratory studies. The three most plausible explanations for the benefits of Andes relative to the Control condition are listed below. The rest of this section considers each one in detail.

- Despite its goal of being minimally invasive, Andes did require a few problem solving practices (e.g., always drawing vectors) that were probably not required of the Control students, and these practices probably improved Andes' students' scores on the exams.
- Andes students repaired almost all the errors and impasses that they encountered while doing homework, whereas the Control students probably did not. This may have increased Andes student's learning in two ways. First, knowledge flaws were probably more frequently detected and remedied by Andes students than by Control students. Second, the number of successful applications of knowledge was probably higher for Andes students than for Control students, and these extra repetitions probably increased the accuracy and speed of knowledge application on exams.

- When the students repaired errors and impasses while doing their homework, Andes students received hints that emphasized principles whereas the Control students had to rely on textbook examples and other resources, which probably did not emphasize principles as much. This probably increased the depth and generality of the Andes students' knowledge.

Better problem-solving practices

Andes emphasized several problem-solving practices that were intended to both expedite learning and to reduce errors. They were:

1. Variables should be defined before being used in equations.
2. Numbers should include units, unless they are dimensionless.
3. Coordinate systems and bodies should be drawn.
4. In order to define a vector magnitude or component, one should draw the vector.
5. Major principle applications should be written first without algebraic substitutions.
6. When no major principles have been applied yet, consider one that contains a quantity sought by the problem.
7. Avoid the number-propagation style of problem solving.

The first 4 practices are required by Andes. Practice 5 is encouraged by warning messages and the problem scoring rubrics. Practice 6 is suggested by Next Step Help. Practice 7 was encouraged by requiring high precision for numbers used in the Equation Window.

Although the Control students were asked to apply these practices during the hour exams, the Andes students did so more frequently. This was clear from the exam subscores, where the difference between Andes students and Control students was larger for the rubrics that were most directly affected by these practices.

The traditional measure of success is whether an answer is correct or incorrect. This is affected by all kinds of errors, including unintentional mathematical mistakes. Competent solvers have learned to follow practices similar to these in order to avoid unintentional errors; for instance, when solving systems of equations, professional mathematicians' steps were actually somewhat smaller than those taken by students (Lewis, 1981). It is plausible that when Andes students used these practices on their exams, the practices decreased the error rate. This would explain why Andes students outscored non-Andes students on the final exams, where only the problem answers counted in the scores.

Repair frequency

Although the Andes and Control students were assigned similar homework problems, and homework was collected and scored in both conditions, it is likely that Andes students solved more homework problems *correctly* than the Control students. Unfortunately, we did not collect copies of the Control students' homework, so we do not know what proportion of their problems were solved correctly. Nonetheless, there are two pieces of indirect evidence.

First, the questionnaire asked the Andes students, "Would you have done the same number of exercises correctly with pencil and paper?" Averaged over the 2001, 2002 and 2003 evaluations, 58% of the students answered "Less," 18% answered "Same," and 24% answered

"More." If the students' opinions are accurate (a big "if"), then Andes students solved more problems correctly than the Control students.

Second, Andes students almost always submitted correct solutions to their homework. Unlike some tutoring systems, Andes do not force students to solve a problem correctly. A student could hand in a solution with the answer left blank or with red entries showing. However, if an Andes student handed in such a solution during the USNA evaluations, the Andes instructors would hand it right back. Andes students soon learned to complete all their problems correctly before handing them in. It is unlikely that the Control students did this, even when they could check their answers against ones from the back of their textbook.

There are two reasons why completing homework problems correctly should increase learning. First, completing a problem correctly means that students are probably fixing flaws in their knowledge that would otherwise prevent solution of the problem. This follows from earlier work which shows that when students reach an impasse (i.e., they don't know what to do next) or make an error (an incorrect entry), they can repair them by themselves, or with the aid of a textbook, a peer, a tutoring system or a human tutor. Sometimes the repair episode includes fixing a flaw in the students' knowledge or at least changing it (Brown & VanLehn, 1980). When students talk aloud during these repairs, they often realize that they have learned something (VanLehn et al., 1992). This is roughly the same kind of learning that occurs when reading a textbook—it changes the students' consciously accessible beliefs. In the parlance of ACTR (Anderson et al., 2004), this is declarative learning. So, the more problems that the students complete successfully, the more impasses and errors they repair successfully, so the more declarative learning they accomplish.

Second, unprepared impasses and errors often prevent a student from traveling down the rest of the path to a correct solution. In the extreme case, a student quits early. As a less extreme example, an uncorrected mistake early in the solution of Figure 1 may prevent the student from even getting to the point where Newton's second law should be applied, so the student misses the opportunity to practice the law's application. Certain kinds of learning are believed to be a by-product of operator application (e.g., Newell & Rosenbloom, 1981). In ACTR, this is called procedural learning (Anderson et al., 2004). The more times an operator is applied successfully, the more likely the student is to apply it successfully again in the future and the less time future applications will take. Therefore, if students repair their errors and impasses, then they should apply more of the intended operators successfully because they cover more of the correct solution paths. This additional practice should produce procedural learning that increases in speed and accuracy.

In short, the more frequently errors and impasses are repaired, the more frequently declarative and procedural learning occurs. Andes helps students detect errors by giving immediate feedback. It helps them repair errors with What's Wrong Help, and it helps them repair impasses with Next Step Help. These facilities allow instructors to require that homework problems be solved correctly. The Control students lacked these facilities, so it would have been unwise to require them to solve the homework problems correctly. Consequently, the Control students probably made fewer repairs than the Andes students, and thus engaged in less declarative and procedural learning.

Principle-based declarative learning at repairs

During studies of students learning physics from texts, it was noticed that some students repair impasses and errors in a rather shallow manner, often while referring to examples (VanLehn & Jones, 1993a; VanLehn et al., 1992). For instance, one student repaired his normal force error by noticing that the normal force in an example was drawn at an angle—not straight up. He noticed that the example's force was perpendicular to a surface, so he redrew his normal force to make it perpendicular to the surface in the problem he was solving. He clearly did not understand why this was the right thing to do. He even commented, "I wonder why they call this a normal force." This is an example of a *shallow* repair to an error.

A repair is shallow when students merely associate an action with some superficial conditions and do not try to use principles to justify the connection. Often such conditions are not general enough. In the illustration above, for instance, the surface was *below* the body in both the problem and the example. The student may have incorporated that feature into their beliefs about how to draw the normal force's direction. Thus, the student may be unsure how to draw the normal force when the surface is *above* the object. Worse still, students sometimes associate entirely inappropriate surface features with the action. Such shallow repairs can cause persistent, robust misconceptions (Brown & VanLehn, 1980; VanLehn, 1990). In short, it is important to reduce the frequency of shallow repairs.

Andes tries to do this by providing principle-based hints at impasses and errors. For instance, its hint sequence for the normal force error in Figure 1 is:

- Are you sure the normal force is straight up? (This is a pointing hint.)
- The normal force is perpendicular (normal) to the surface that causes it. Of course, the normal force is a 'pushing' force rather than a 'pulling' force, so it points away from the surface, not into it. (This is an instructional hint.)
- Because the normal force on the car is perpendicular to the driveway, make its direction be 110 degrees. (This is the bottom-out hint.)

The hints sequences, and especially the instructional hints, are intended to encourage deep, principle-based repairs. Although they are probably better than no hints at all, they are not perfect. The instructional hints should be short enough that students are willing to read them.[10] Nonetheless, some students still skip them, and it is likely that their repairs are shallow. In laboratory studies, replacing the instructional hints with more elaborate instruction elicited more learning (Albacete & VanLehn, 2000a, 2000b; Rose et al., 2001), which confirms the impression instructional hints sometimes fail to produce deep, principle-based repairs. Nonetheless, it still seems likely that Andes hints are better than no hints at encouraging deep learning. So this remains one potential source of its power advantage over the non-Andes conditions.

---

[10] This instructional hint would have to be much longer in order to thoroughly explain the normal force's direction. Whenever two objects are in contact, there is a contact force between them. However, when one object slides along another, the contact force is decomposed into two components: the component parallel to the surface is called the friction force and the component perpendicular (i.e., normal) to the surface is called the normal force. This decomposition is useful because the magnitudes of the two "forces" are related by F=_*N. There is no similar relationship for other contact forces, so they are not decomposed. This explanation is rarely found in textbooks, so Andes can perhaps be forgiven for excluding it from the instructional hint.

Summary

Andes has many features, both large and small, that are intended to accelerate students' learning. Thousands of design decisions were made, ranging from the wording of individual hints on up to how to provide algebraic manipulation tools. We cannot assign credit for its success to any one feature without much more experimentation.

Nonetheless, this section described three features that seem to us to be most plausible as the sources of its efficacy. They are:

- Constraints on students' problem solving practices, such as requiring the variables be defined and that vectors be drawn.
- Requiring that students hand in error-free solutions to all homework problems, and helping them produce such solutions by giving them immediate feedback and hints.
- Giving hints that encourage deep, principle-based repairs of errors and impasses.

Clearly, each of these represents a testable hypothesis. For instance, if we turned off immediate feedback and help but still required students to define variables, draw vectors, etc, how much learning would occur? This would isolate the amount of learning caused by the first feature listed above. Such findings would have implications for both physics education in general and the psychology of learning in general.

CONCLUSIONS AND FUTURE WORK

Research is often search—one repeatedly goes down a path, fails, backs up, and tries a different path; but with every failure, one learns. This paper has tried to document what has been learned from the Andes re-search.

Moreover, Andes is much better for having undergone this evolution. Multiple evaluations show that Andes is significantly more effective than doing pencil and paper homework. Moreover the costs are low: students seem to spend no extra time doing homework, and instructors do not have to totally revise their class in order to obtain these benefits for their students. Moreover, these results were not obtained through one or two laboratory studies, but through experiments in real USNA classrooms over five years.

It appears that we have succeeded in find a minimally invasive way to use intelligent tutoring systems to help students learn, namely, to deploy the technology as a homework helper. Moreover, Andes is probably more effective than existing homework helpers, such as WebAssign, Mastering Physics, and other web-based homework (WBH) services listed in the introduction. The existing evaluations, which were reviewed in the introduction, suggest that WBH is no more effective than paper-and-pencil homework, whereas Andes is significantly more effective than paper-and-pencil homework. To be certain that Andes is more effective than WBH, however, one should compare it directly to one of these systems.

For the immediate future, we have three goals. The first is to help people all over the world use Andes as the USNA has done, as a homework helper for their courses. Please see www.andes.pitt.edu if you are interested.[11]

The second goal is to develop a self-paced, open physics course based on Andes. It differs from plain Andes in that the course will include text and other passive material, and that it will require mastery learning. That is, a student must work on a chapter until mastery has been demonstrated. Mastery learning is widely believed to provide large, reliable benefits (Corbett, 2001), so combining it with Andes should create a remarkably powerful learning environment. However, the use of mastery learning means that the course must be a self-paced course, because different students will take different amounts of time to finish it. The course will also be open, in that instructors can modify the text and other passive material that their students see, and they can modify some of the pedagogical policies of Andes. We are currently looking for instructors who are interested in developing such a self-paced physics course with us.

Lastly, the Pittsburgh Science of Learning Center (www.learnlab.org) will use Andes to help create a physics LearnLab course. A LearnLab course is a regular course that has been heavily instrumented so that investigators can monitor the progress of individual students. Most LearnLab courses developed by the center will incorporate tutoring systems, in part because the systems to be used are known to enhance learning. However, the main reason for using tutoring systems in LearnLab courses is to allow researchers to test hypotheses by substituting a novel instructional module for one of the tutoring system's normal modules. This allows researchers to test hypotheses with the same rigor as they would obtain in the laboratory, but with the added ecological validity of a field setting. When the physics LearnLab is developed, it will be much easier to test some of the hypotheses about learning mentioned in the preceding section.

ACKNOWLEDGEMENTS

This research was supported by the Cognitive Sciences Program of the Office of Naval Research under grants N00019-03-1-0017 and ONR N00014-96-1-0260. We gratefully acknowledge the Andes Alumni: Drs. Patricia Albacete, Cristina Conati, Abigail Gertner, Zhendong Niu, Charles Murray, Stephanie Siler, and Ms. Ellen Dugan

REFERENCES

Albacete, P. L., & VanLehn, K. (2000a). The Conceptual Helper: An intelligent tutoring system for teaching fundamental physics concepts. In G. Gauthier, C. Frasson & K. VanLehn (Eds.) *Intelligent Tutoring Systems: 5th International Conference, ITS 2000* (pp. 564-573). Berlin: Springer.

Albacete, P. L., & VanLehn, K. (2000b). Evaluation the effectiveness of a cognitive tutor for fundamental physics concepts. In L. R. Gleitman & A. K. Joshi (Eds.) *Proceedings of the Twenty-Second Annual Conference of the Cognitive Science Society* (pp. 25-30). Mahwah, NJ: Erlbaum.

---

[11] Please plan to spend at least an hour evaluating Andes. It is critical that you view some training videos before trying to solve a problem.

Aleven, V., & Koedinger, K. R. (2000). Limitations of student control: Do students know when they need help? In G. Gauthier, C. Frasson & K. VanLehn (Eds.) *Intelligent Tutoring Systems: 5th International Conference, ITS 2000* (pp. 292-303). Berlin: Springer.

Aleven, V., & Koedinger, K. R. (2002). An effective metacognitive strategy: Learning by doing and explaining with a computer-based cognitive tutor. *Cognitive Science*, 26(2), 147-179.

Aleven, V., Popescu, O., & Koedinger, K. R. (2003). A tutorial dialog system to support self-explanation: Evaluation and open questions. In U. Hoppe, F. Verdejo & J. Kay (Eds.) *Proceedings of the 11th International Conference on Artificial Intelligence in Education, AI-ED 2003* (pp. 39-46). Amsterdam: IOS Press.

Aleven, V., Stahl, E., Schworm, S., Fischer, F., & Wallace, R. M. (2003). Help seeking and help design in interactive learning environments. *Review of Educational Research,* 73(2), 277-320.

Anderson, J. R., Bothell, D., Byrne, M. D., Douglass, S., Lebiere, C., & Qin, Y. (2004). An integrated theory of the mind. *Psychological Review,* 111(4), 1036-1060.

Anderson, J. R., Corbett, A. T., Koedinger, K. R., & Pelletier, R. (1995). Cognitive Tutors: Lessons Learned. *The Journal of the Learning Sciences*, 4(2), 167-207.

Anderson, J. R., & Gluck, K. (2001). What role do cognitive architectures play in intelligent tutoring systems. In D. Klahr & S. M. Carver (Eds.) *Cognition and Instruction: Twenty-five Years of Progress* (pp. 227-262). Mahwah,NJ: Erlbaum.

Bloom, B. S. (1984). The 2 sigma problem: The search for methods of group instruction as effective as one-to-one tutoring. *Educational Researcher*, 13, 4-16.

Bonham, S. W., Deardorff, D. L., & Beichner, R. J. (2003). Comparison of student performance using web and paper-based homework in college-level physics. *Journal of Research in Science Teaching,* 40(10), 1050-1071.

Brna, P., & Caiger, A. (1992). The application of cognitive diagnosis to the quantitative analysis of simple electrical circuits. In C. Frasson, G. Gauthier & G. I. McCalla (Eds.) *Intelligent Tutoring Systems, Second International Conference* (pp. 405-412). Berlin: Springer.

Brown, J. S., & VanLehn, K. (1980). Repair theory: A generative theory of bugs in procedural skills. *Cognitive Science,* 4, 379-426.

Bundy, A., Byrd, L., Luger, G., Mellish, C., & Palmer, M. (1979). Solving mechanics problems using meta-level inference, *Proceedings of the Sixth International Joint Conference on AI* (pp. 1017-1027). San Mateo, CA: Morgan Kaufmann.

Burton, R. R., & Brown, J. S. (1982). An investigation of computer coaching for informal learning activities. In D. Sleeman & J. S. Brown (Eds.) *Intelligent Tutoring Systems*. New York: Academic Press.

Carberry, S. (1990). *Plan Recognition in Natural Language Dialogue*. Cambridge, MA: MIT Press.

Chi, M. T. H., Feltovich, P., & Glaser, R. (1981). Categorization and representation of physics problems by experts and novices. *Cognitive Science,* 5(2), 121-152.

Conati, C., Gertner, A., & Vanlehn, K. (2002). Using Bayesian networks to manage uncertainty in student modeling. *User Modeling and User-Adapted Interactions,* 12(4), 371-417.

Conati, C., & VanLehn, K. (2000). Toward computer-based support of meta-cognitive skills: A computational framework to coach self-explanation. *International Journal of Artificial Intelligence in Education,* 11, 398-415.

Conati, C., & VanLehn, K. (2001). Providing adaptive support to the understanding of instructional material, *Proceedings of IUI 2001, International Conference on Intelligent User Interfaces.*

Cronback, L. J., & Snow, R. E. (1977). *Aptitudes and instructional methods: A handbook for research on interactions.* New York: Irvington.

de Kleer, J. (1977). Multiple representations of knowledge in a mechanics problem-solver, *International Joint Conference on Artificial Intelligence* (pp. 299-304). Cambridge, MA: MIT Press.

Dufresne, R. J., Gerace, W. J., Hardiman, P. T., & Mestre, J. P. (1992). Constraining novices to perform expert-like problem analyses: Effects on schema acquisition. *The Journal of the Learning Sciences*, 2(3), 307-331.

Dufresne, R. J., Mestre, J. P., Hart, D. M., & Rath, K. A. (2002). The effect of web-based homework on test performance in large enrollment introductory physics courses. *Journal of Computers in Mathematics and Science Teaching*, 21(3), 229-251.

Dweck, C. S. (1986). Motivational processes affecting learning. *American Psychologist, 41*, 1040-1048.

Elio, R., & Scharf, P. B. (1990). Modeling novice-to-expert shifts in problem-solving strategy and knowledge organization. *Cognitive Science*, 14, 579-639.

Gertner, A., Conati, C., & VanLehn, K. (1998). Procedural help in Andes: Generating hints using a Bayesian network student model. *Proceedings of the 15th National Conference on Artificial Intelligence*.

Gertner, A. S. (1998). Providing feedback to equation entries in an intelligent tutoring system for Physics. In B. P. Goettl, H. M. Halff, C. L. Redfield & V. J. Shute (Eds.) *Intelligent Tutoring Systems: 4th International Conference* (pp. 254-263). New York: Springer.

Gertner, A. S., & VanLehn, K. (2000). Andes: A coached problem solving environment for physics. In G. Gauthier, C. Frasson & K. VanLehn (Eds.) *Intelligent Tutoring Systems: 5th International Conference, ITS 2000* (pp. 133-142). New York: Springer.

Graesser, A. C., VanLehn, K., Rose, C. P., Jordan, P. W., & Harter, D. (2001). Intelligent tutoring systems with conversational dialogue. *AI Magazine,* 22(4), 39-51.

Hume, G., Michael, J., Rovick, A., & Evens, M. (1996). Hinting as a tactic in one-on-one tutoring. *Journal of the Learning Sciences,* 5(1), 23-49.

Johnson, W. L. (1990). Understanding and debugging novice programs. *Artificial Intelligence*, 42, 51-97.

Kane, D., & Sherwood, B. (1980). A computer-based course in classical mechanics. *Computers and Education*, 4, 15-36.

Katz, S., Connelly, J., & Allbritton, D. (2003). Going beyond the problem given: How human tutors use post-solution discussions to support transfer. *International Journal of Artificial Intelligence in Education*, 13, 79-116.

Koedinger, K., & Anderson, J. R. (1993). Reifying implicit planning in geometry: Guidelines for model-based intelligent tutoring system design. In S. P. Lajoie & S. J. Derry (Eds.) *Computers as cognitive tools*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Koedinger, K. R., Anderson, J. R., Hadley, W. H., & Mark, M. A. (1997). Intelligent tutoring goes to school in the big city. *International Journal of Artificial Intelligence in Education,* 8(1), 30-43.

Kulik, C., Kulik, J., & Bangert-Drowns, R. (1990). Effectiveness of mastery learning programs: A meta-analysis. *Review of Educational Research,* 60(2), 265-306.

Lamberts, K. (1990). A hybrid model of learning to solve physics problems. *European Journal of Cognitive Psychology,* 3(2), 151-170.

Larkin, J. (1983). The role of problem representation in physics. In D. Gentner & A. Stevens (Eds.) *Mental Models*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Larkin, J. H., Reif, F., Carbonell, J., & Gugliotta, A. (1988). Fermi: A flexible expert reasoner with multi-domain inferencing. *Cognitive Science*, 12(1), 101-138.

LeFevre, J., & Dixon, P. (1986). Do written instructions need examples? *Cognition and Instruction,* 3, 1-30.

Leonard, W. J., Dufresne, R. J., & Mestre, J. P. (1996). Using qualitative problem-solving strategies to highlight the role of conceptual knowledge in solving problems. *American Journal of Physics,* 64(12), 1495-1503.

Lewis, C. (1981). Skill in algebra. In J. R. Anderson (Ed.) *Cognitive skills and their acquisition* (pp. 85-110). Hillsdale, NJ: Lawrence Erlbaum Associates.

Liew, C. W., Shapiro, J. A., & Smith, D. E. (2004). Inferring the context for evaluating physics algebraic equations when scaffolding is removed, *Proceedings of FLAIRS2004*.

Martin, J., & VanLehn, K. (1995a). A Bayesian approach to cognitive assessment. In P. Nichols, S. Chipman & S. Brennan (Eds.) *Cognitively diagnostic assessment*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Martin, J., & VanLehn, K. (1995b). Student assessment using Bayesian nets. *International Journal of Human-Computer Studies,* 42, 575-591.

McDermott, J., & Larkin, J. H. (1978). Re-representing textbook physics problems, *Proceedings of the Second National Conference of the Canadian Society for Computational Studies of Intelligence*. Toronto, Ontario.

Merrill, D. C., Reiser, B. J., Ranney, M., & Trafton, J. G. (1992). Effective tutoring techniques: A comparison of human tutors and intelligent tutoring systems. *The Journal of the Learning Sciences,* 2(3), 277-306.

Mitrovic, A. (2003). An intelligent SQL tutor on the web. *International Journal of Artificial Intelligence in Education*, 13(2-4), 243-197.

Mitrovic, A., Koedinger, K. R., & Martin, B. (2003). A comparative analysis of cognitive tutoring and constraint-based modelling. In P. Brusilovsky, A. Corbett & F. d. Rosis (Eds.) *Proceedings of the Ninth International Conference on User Modeling, UM 2003* (Vol. LNAI 2702, pp. 313-322). Berlin: Springer-Verlag.

Murray, R. C., & VanLehn, K. (2005). Effects of dissuading unnecessary help requests while providing proactive help. In G. I. Mcalla & C.-K. Looi (Eds.) *Proceedings of Artificial Intelligence in Education*. Berlin: Springer.

Newell, A., & Rosenbloom, P. S. (1981). Mechanisms of skill acquisition and the law of practice. In J. R. Anderson (Ed.) *Cognitive Skills and Their Acquisition* (pp. 1-56). Hillsdale, NJ: Lawrence Erlbaum Associates.

Nicaud, J.-F., Bouhineau, D., Varlet, C., & Nguyen-Xuan, A. (1999). Towards a product for teaching formal algebra. In S. P. Lajoie & M. Vivet (Eds.) *Artificial Intelligence in Education* (pp. 207-214). Amsterdam: IOS Press.

Norman, D. A. (1981). Categorization of action slips. *Psychological Review*, 88(1), 1-15.

Pascarella, A. M. (2002). *CAPA (Computer-Assisted Personalized Assignments) in a Large University Setting.* Unpublished Doctoral Dissertation, University of Colorado, Boulder, CO.

Pascarella, A. M. (2004). *The influence of web-based homework on quantitiatve problem-solving in university physics classes.* Paper presented at the National Association for Research in Science Teaching (NARST), Vancouver, BC, Canada.

Priest, A. G., & Lindsay, R. O. (1992). New light on novice-expert differences in physics problem solving. *British Journal of Psychology,* 83, 389-405.

Reif, F. (1987). Interpretation of scientific or mathematical concepts: Cognitive issues and instructional implications. *Cognitive Science*, 11(4), 395-416.

Rose, C. P., Jordan, P. W., Ringenberg, M., Siler, S., Vanlehn, K., & Weinstein, A. (2001). Interactive conceptual tutoring in Atlas-Andes. In J. D. Moore, C. Redfield & W. L. Johnson (Eds.) *Artificial Intelligence in Education: AI-Ed in the Wired and Wireless future* (pp. 256-266). Washington, DC: IOS.

Rose, C. P., Roque, A., Bhembe, D., & VanLehn, K. (2002). A hybrid language understanding approach for robust selection of tutoring goals. In S. A. Cerri, G. Gouarderes & F. Paraguacu (Eds.) *Intelligent Tutoring Systems, 2002: 6th International Conference* (pp. 552-561). Berlin: Springer.

Scheines, R., & Sieg, W. (1994). Computer environments for proof construction. *Interactive Learning Environments,* 4(2), 159-169.

Schofield, J. W. (1995). *Computers, classroom culture and change*. Cambridge, UK: Cambridge University Press.

Schulze, K. G., Shelby, R. N., Treacy, D. J., Wintersgill, M. C., Vanlehn, K., & Gertner, A. (2000). Andes: An intelligent tutor for classical physics. *The Journal of Electronic Publishing*, 6(1).

Self, J. A. (1990). Bypassing the intractable problem of student modelling. In C. Frasson & G. Gauthier (Eds.) *Intelligent Tutoring Systems: At the crossroads of AI and Education* (pp. 107-123). Norwood, NJ: Ablex.

Shapiro, J. A. (2005). Algebra subsystem for an intelligent tutoring system. *International Journal of Artificial Intelligence in Education*.

Shute, V. J. (1993). A macroadaptive approach to tutoring. *Journal of Artificial Intelligence in Education,* 4(1), 61-93.

Shute, V. J., & Psotka, J. (1996). Intelligent tutoring systems: Past, Present and Future. In D. Jonassen (Ed.) *Handbook of Research on Educational Communications and Technology*: Scholastic Publications.

Singley, M. K. (1990). The reification of goal structures in a calculus tutor: Effects on problem solving performance. *Interactive Learning Environments*, 1, 102-123.

Singley, M. K., & Anderson, J. R. (1989). *The transfer of cognitive skill*. Cambridge, MA: Harvard University Press.

Slamecka, N. J., & Graf, P. (1978). The generation effect: Delineation of a phenomenon. *Journal of Experimental Psychology: Human Learning and Memory,* 4(6), 592-604.

Thornton, R. K., & Sokoloff, D. R. (1998). Assessing student learning of Newton's laws: The Force and Motion Conceptual Evaluation and the evaluation of Active Learning Laboratory and Lecture curricula. *American Journal of Physics,* 66(4), 338-351.

Tulving, E., & Thomson, D. M. (1973). Encoding specificity and retrieval processes in episodic memory. *Psychological Review*, 80, 352-373.

VanLehn, K. (1988). Student modeling. In M. Polson & J. Richardson (Eds.) *Foundations of Intelligent Tutoring Systems* (pp. 55-78). Hillsdale, NJ: Lawrence Erlbaum Associates.

VanLehn, K. (1990). *Mind Bugs: The Origins of Procedural Misconceptions*. Cambridge, MA: MIT Press.

VanLehn, K. (1999). Rule learning events in the acquisition of a complex skill: An evaluation of Cascade. *Journal of the Learning Sciences,* 8(2), 179-221.

VanLehn, K., Bhembe, D., Chi, M., Lynch, C., Schulze, K., Shelby, R., Taylor, L., Treacy, D., Weinstein, A., & Wintersgill, M. (2004). Implicit vs. explicit learning of strategies in a non-procedural skill, *ITS 2004*. Berlin: Springer.

VanLehn, K., & Jones, R. M. (1993a). Better learners use analogical problem solving sparingly. In P. E. Utgoff (Ed.) *Machine Learning: Proceedings of the Tenth Annual Conference* (pp. 338-345). San Mateo, CA: Morgan Kaufmann.

VanLehn, K., & Jones, R. M. (1993b). Integration of analogical search control and explanation-based learning of correctness. In S. Minton (Ed.) *Machine Learning Methods for Planning* (pp. 273-315). Los Altos, CA: Morgan Kaufmann.

VanLehn, K., & Jones, R. M. (1993c). Learning by explaining examples to oneself: A computational model. In S. Chipman & A. Meyrowitz (Eds.) *Cognitive Models of Complex Learning* (pp. 25-82). Boston, MA: Kluwer Academic Publishers.

VanLehn, K., Jones, R. M., & Chi, M. T. H. (1992). A model of the self-explanation effect. *The Journal of the Learning Sciences,* 2(1), 1-59.

VanLehn, K., Lynch, C., Taylor, L., Weinstein, A., Shelby, R. H., Schulze, K. G., Treacy, D. J., & Wintersgill, M. C. (2002). Minimally invasive tutoring of complex physics problem solving. In S. A. Cerri, G. Gouarderes & F. Paraguacu (Eds.) *Intelligent Tutoring Systems, 2002, 6th International Conference* (pp. 367-376). Berlin: Springer.

VanLehn, K., & Martin, J. (1998). Evaluation of an assessment system based on Bayesian student modeling. *International Journal of Artificial Intelligence in Education,* 8(2), 179-221.

VanLehn, K., & Niu, Z. (2001). Bayesian student modeling, user interfaces and feedback: A sensitivity analysis. *International Journal of Artificial Intelligence in Education,* 12(2), 154-184.

Woodroffe, M. R. (1988). Plan recognition and intelligent tutoring systems. In J. Self (Ed.) *Artificial Intelligence and Human Learning* (pp. 212-225). New York: Chapman and Hall.

Yibin, M., & Jinxiang, L. (1992). Intelligent tutoring system for symbolic calculation. In C. Frasson, G. Gauthier & G. I. McCalla (Eds.) *Intelligent Tutoring Systems, Second International Conference* (pp. 132-147). Berlin: Springer.