

**Technical Report**  
**Development of a cognitive tutor for learning truss analysis**  
**Paul S. Steif, Luoting Fu, Levent Burak Kara**

**1. Introduction**

Statics is an important foundational subject in many engineering majors. It is often a pivotal first course, in which many students weigh whether to proceed with an engineering major. In addition, learning of statics is also a concern to instructors in advanced level courses, such as capstone design, who would like students to be able to utilize what they learned in statics<sup>1</sup>. Conceptually, statics breaks important ground for engineering students. Drawing free body diagrams (FBDs) and imposing equilibrium in statics is a first foray into a style of analysis that is a cornerstone of engineering problem solving: isolation of a sub-system upon which balance laws are applied.

Students who complete statics should be able to effectively apply its methods to analyze realistic engineering systems. Some exposure to realistic systems comes from tackling problems often identified as “structures”, that is, trusses, frames and machines (mechanisms), which are addressed to some extent in most statics courses. Problems based on structures entail consideration of multiple interacting bodies simultaneously and are challenging because: (i) they draw upon much of the core conceptual knowledge in statics, which is always more difficult to draw up while solving problems, and (ii) they involve juggling several sub-analyses, and hence demand organization, strategy, and decision-making.

In typical classroom instruction, the basic concepts and approaches underlying the analysis of problems involving structures are presented, along with the solution of example problems. However, students then need to practice solving such problem themselves. It is widely recognized across a wide range of domains, that timely feedback during the practice of problem solving promotes learning<sup>2-7</sup>. While a human tutor can offer effective instruction during problem solving, offering such close monitoring via computer would benefit more students.

Many instructional interventions have been designed for statics, although computer-based ones interventions typically offer the most effective feedback when students exercise individual concepts or analysis skills. Recently, there have appeared computer systems that allow students to work on some simple statics problem more or less from start to finish, and provide feedback on individual steps<sup>8,9</sup>. There is also recent work to develop alternative user interfaces for computational tools for engineering mechanics education. A Truss Recognition tutoring system is being developed<sup>10</sup> in which students draw trusses with supports and loads using a mouse or pen. The strokes are recognized and interpreted by the system, which then analyzes the trusses to determine the loads in the bars. One goal is for students to be involved in more open-ended problem solving and more quickly get a sense of the consequences of a particular design. Sketch recognition is employed in another project<sup>11</sup> to develop a tutoring system for statics that allows students to solve problems by pen; the focus of that project is to develop alternative user interfaces for tutoring systems.

To follow, interpret, and give feedback to a user who is given wide latitude in problem solving, a tutoring system must have a model of the problem solver's thought process. Accordingly, researchers have developed Intelligent Tutoring Systems<sup>12,13</sup>, including even some relevant to mechanics of structures<sup>14-16</sup>. Intelligent computer-based instruction can be effective in increasing student learning beyond that achieved in normal classroom instruction<sup>17</sup>. Cognitive tutors<sup>18</sup> merged the ideas of intelligent tutoring systems with computational models of cognitive theories of human learning, memory, and problem solving<sup>19</sup>. Cognitive tutors are based on a cognitive model of a student interacting with problems in a domain.

Multi-body statics problems, in particular structures problems, offer challenges to tutoring systems, first because the solution space is highly bifurcated. Often, a number of sub-analyses must be conducted: in each sub-analysis the student chooses a sub-system, draws the FBD, and writes and solves equilibrium equations. Multiple combinations of sub-analyses may lead to the solution, sub-analyses can be conducted in various orders, a new sub-analysis might be started before the previous one is completed, and within each sub-analysis, the equations can be written down in different orders. The net effect is that one cannot predict the next solution step or steps, given what the solver has done so far. Structures problems present a second significant challenge to tutoring systems. In such problems, the solver needs to clearly delineate a subset of the original problem that is to be analyzed (the "subsystem"). Moreover, the solver must indicate the presence of forces, described through vectors and labels, at distinct points. The solver must be able to carry out these steps fluidly, and the tutoring system must properly recognize the user input.

So, to be effective, a tutoring system needs to be suited to the problem space of interest: here, multi-body statics. The system must grant the user graphical and textual tools to carry out solutions, including the designation of spatial regions and of symbols (vectors and labels), positioned relative to those regions, and textual (equation) input. In addition, the system needs to react to the user input in ways that are instructionally helpful, while granting the solver leeway comparable to that prevailing when solving on paper.

Among possible structures problems in statics, truss problems involve a reasonably complex, but still manageable, language of graphics and text. Furthermore, the pathways to solution can vary widely and still be correct. The underlying framework for solving statics problems, including the common errant pathways of students, has been studied<sup>20-23</sup>. This research enables us to formulate a simple cognitive model for solving truss problems and thereby interpret student work and offer instructional support in response to their graphical and textual input. But, there is an inherent conflict in computer instruction in such a complex domain, between the freedom students are given to devise solutions to problems and the ability of the computer to ascertain, judge, and give feedback on what students have done. On the one hand, one could give students a blank piece of paper (or a computer tablet) and ask them to conduct their analysis; but it would be quite challenging for a computer to interpret what students are drawing and respond to it. On the other hand, we could ask students a series of multiple-choice questions, with carefully chosen answers; while the responses are interpretable, they give no indication if students could independently solve a full problem on their own. This may be termed the *solution latitude-interpretation* trade-off.

In this technical paper, we describe the development of and form of a cognitive-style truss tutor, and we identify the trade-offs between fully free paper-and-pencil solving and the modestly constrained solving that the tutor permits. The constraints imposed still allow users to make the errors commonly committed by students in paper-and-pencil solving. The tutor is deemed a cognitive tutor in the sense that there is an underlying cognitive model that enables the tutor, wherever students are in their solution pathway, to offer feedback on errors they make and to make suggestions when they are stuck. The tutor also records essentially the entire student solution process; this enables us to track separately individual skills and determine if students gradually make fewer errors with practice on each skill. To carry out this tracking and analysis, we take advantage of the tools of the DataShop repository at the Pittsburgh Science of Learning Center (<http://learnlab.org/datashop>) provides a resource for educators and researchers to create, modify, and evaluate student models<sup>24</sup>. We use those tools to fit a statistical model from which one can compare how well different proposed collections of skills capture student learning, that is, explain variation in observed error rates of students using the tutor. Some results from an initial data set are presented.

## **2. Designing Tutor Based On Tasks Required and on Observations of Student Work**

In devising a tutor for statics, one should be aware of the general set of tasks expected of students. If one applies statics to a real physical system, then one needs to<sup>25</sup>:

1. Survey the physical system to recognize the various parts, how they are connected or supported
2. Select a subsystem, that is, a some portion of the system, for analysis
3. Draw a free body diagram of the subsystem
4. Write down equations of equilibrium for the subsystem
5. Solve equations for unknowns and potentially use those results in the analysis of subsequent subsystems

Different types of statics problems exercise these facets of the problem solving process to varying degrees. Certainly, the initial stage of surveying a physical system, the true modeling stage, can be the most difficult; it is also the most challenging for instructors to assign so students' efforts can be observed. In confining ourselves in this tutor to truss problems as they appear in textbooks, we accept that the problems are pre-modeled and focus on the remaining steps in statics.

To design a tutor that does not allow free form sketching, but partially constrains the solver, we should consider what errors students commit in their free form attempts to analyze trusses. The tutor should grant the user the latitude to commit most of the same errors. The following sample of student errors in solving truss problems, taken from students in a mechanical engineering

statics class, is not meant to be exhaustive; rather it provides guidance as to constraints the tutor can place on solving.

In Figure 1, the student has not drawn a subsystem that would reveal internal forces (a joint or a section), but still writes equations involving internal forces ( $F_{GF}$ ).

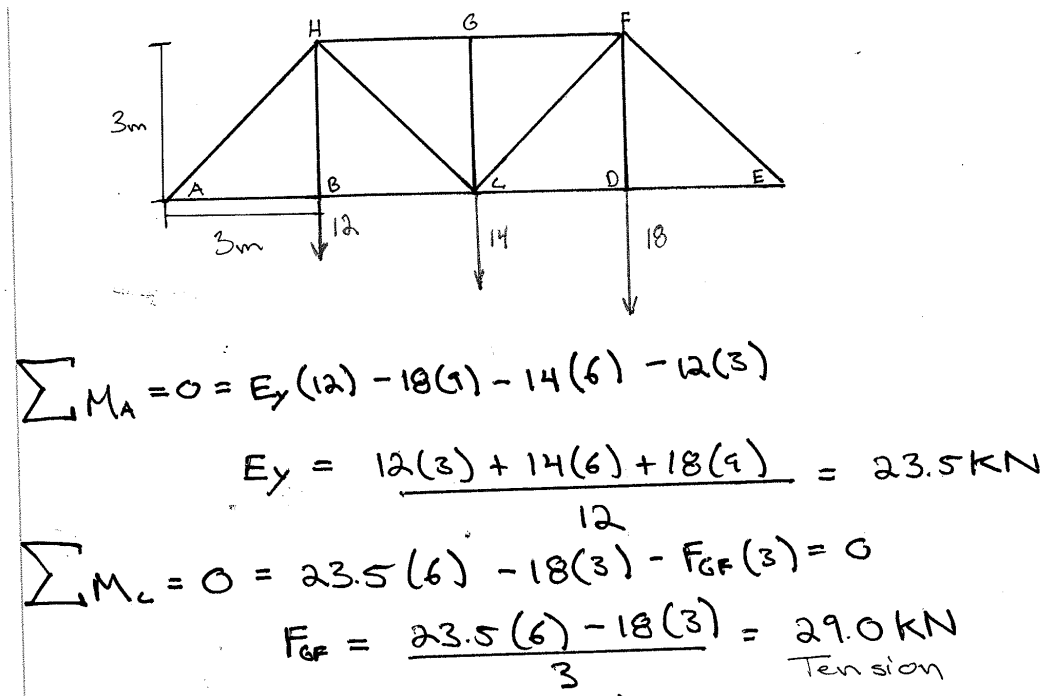


Figure 1. Equilibrium equation contains internal force ( $F_{GF}$ ) not revealed in drawn subsystem.

In Figure 2, the student has drawn a subsystem (sectioned at the right), but has not drawn internal forces, so their assumed senses are unclear. The equilibrium equations that contain the internal forces are more likely to contain sign errors (and do).

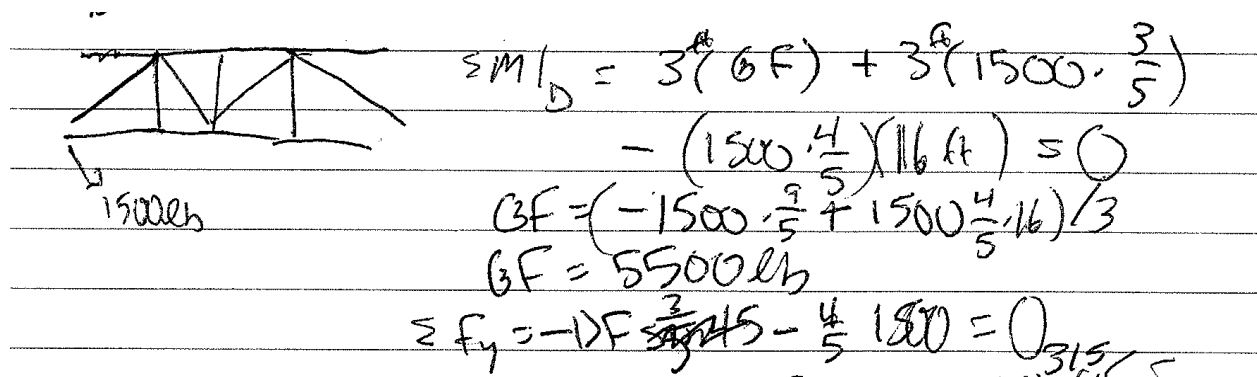


Figure 2. Internal forces, which are not drawn on section, appear in equilibrium equations with sign errors.

In Figure 3, the student is attempting to draw a subsystem, but it is unclear which bars and partial bars are included, and which forces are acting.

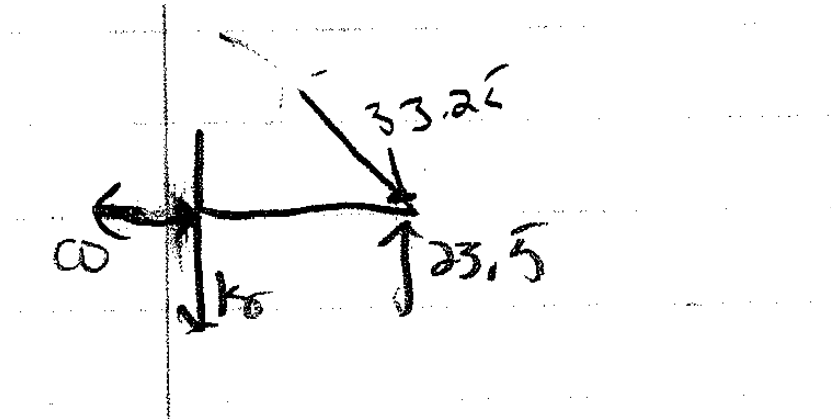


Figure 3. FBD in which included members and acting forces are unclear.

In Figure 4, the student is attempting to draw a subsystem, but internal forces are drawn on certain bars even though the entire bar is included in the subsystem.

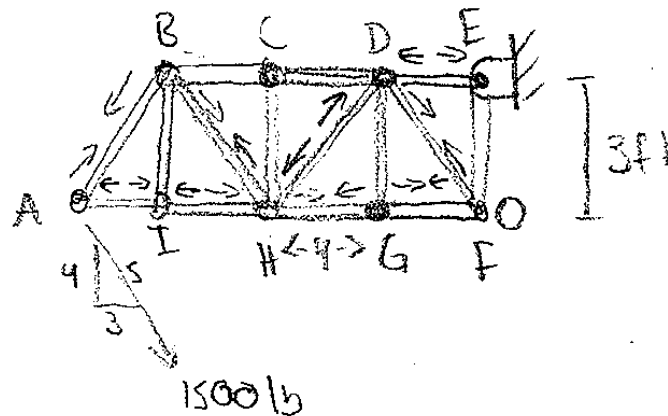


Figure 4. Internal forces drawn on bars that are fully included in the subsystem.

Despite all the errors, students do form subsystems by selecting pins, bars and partial bars, and other extraneous objects. In drawing free body diagrams, students do tend to draw forces on the pins or at the ends of partial members, but not in the middle of a member or at some random external point. Further, students tend to draw forces at those points, rather than concentrated moments or couples. (This is not to say that students fully understand why an idealized pin cannot exert a couple, or why the connections between truss members are idealized as pins.) The forces tend to be drawn parallel to x-y axes, or parallel or perpendicular to members present, but not in random directions. In writing down equations of equilibrium, students tend to write

summations of forces in x and y directions, and sometimes on some inclined axes, and they tend to write summations of moments about points coinciding with pins.

The above observations of student work inform how the tutor should constrain users in solving truss problems. We argue that a tutor constraining user choice as follows will capture most student work (correct and incorrect) on truss problems:

- Each subsystem can be any collection of pins, members and half members (there can be multiple such subsystems analyzed)
- In free body diagrams, only forces can be drawn, either at pins or at the free ends of half members. Forces are confined to lie along x-y directions or parallel or perpendicular to bars, but can act in either sense.
- Equations of force equilibrium along x-y, and equations of moment equilibrium about any joint, can be written. (Future developments will enable the user to define x'-y' axes and to write equilibrium along those axes.)

While students are free to carry out the actions just described, in devising the tutor we have made some non-obvious choices regarding: what constitutes a correct subsystem and what must be done to fully specify a free body diagram. These choices, which serve largely to make his or her thinking more visible to the student and to the tutor; are described in the next section.

In summary, we seek to develop a computer tutor with a simple, easily learnable user interface that gives students reasonably wide latitude to solve truss problems using method of joints and method of sections with minimal distractions and unnecessary effort. Students using the tutor are expected to have learned about truss analysis through other means, such as lecture and textbook; the tutor focuses helping student practice truss analysis. The tutor should allow students to make the errors commonly committed by students when solving with pencil and paper. The tutor should capture all student work in ways that enable it to: (i) highlight errors in steps so students can correct them and (ii) track which skills students have mastered and which errors are committed. From the tracking of student work, the system can provide the instructor with student-specific and class-averaged measures of skill mastery and error commission. In addition, the tracking can also enable the tutor to offer subsequent problems to each student in a way that focuses on to-be-mastered skills.

### **3. Description Of Tutor**

A screen shot of the tutor overall, with a problem partially solved, is shown in Figure 5. The left half of the display contains a menu bar at the top and the problem diagram and statement. The problem diagram can be toggled to display the solution diagram, where support reactions and bar forces that have been determined are registered by the student, as described below. The user chooses a subsystem for analysis by clicking on a set of pins, members and partial members, and then clicking on the draw (pencil) icon from the menu bar. The selected group of parts is added as another subsystem to the right half of the display, and would appear as one of the thumbnails

depicted in Figure 5. Clicking on a thumbnail expands that subsystem, allowing the user to draw its FBD and write its associated equilibrium equations. Figure 6 shows one subsystem, selected by the student, on which the forces are being drawn for the FBD.

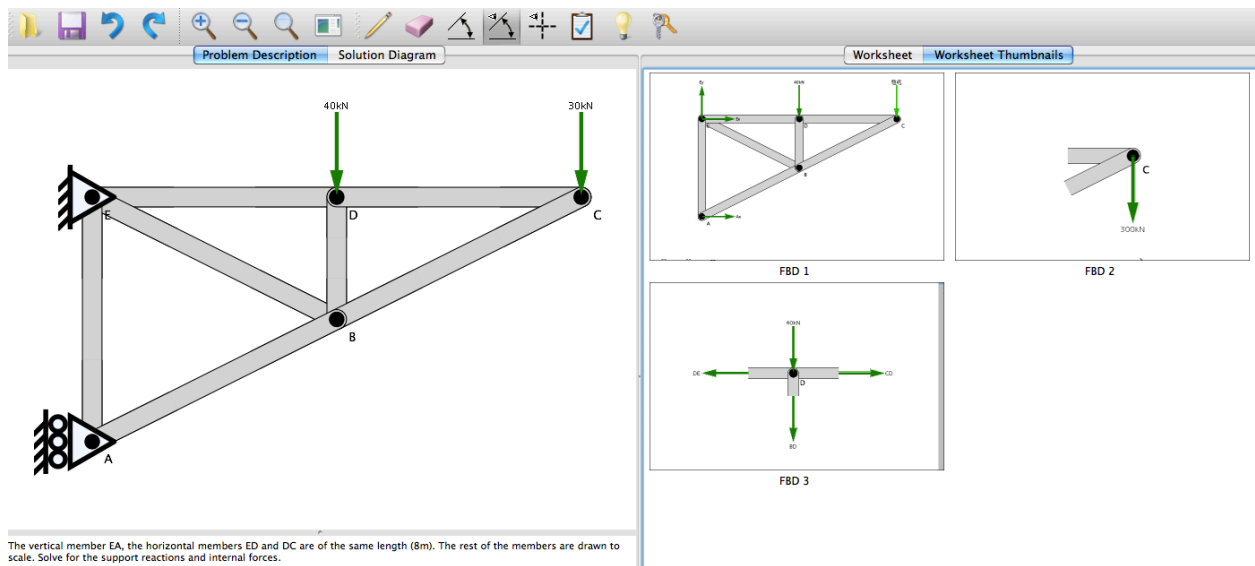


Figure 5. Screen shot of full display of truss tutor.

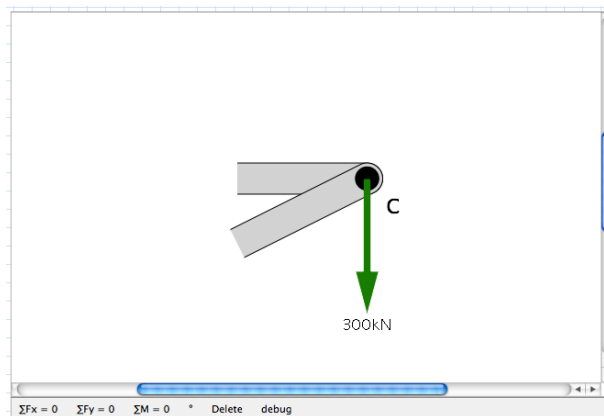


Figure 6. Screen shot of right half of display of truss tutor, focusing on free body diagram of a single subsystem.

Note that the user can select any combinations of pins, bars, and half bars to form a system; most combinations would not be valid. To explain what the tutor treats as a correct or valid subsystem, it is useful to note the practice in textbooks. With the method of sections, it is common to draw the partial members that have been cut, along with the whole members and pins to one side of the chosen section. With the method of joints, it is common to draw just the joint, perhaps as a dot, although some books draw the pin along with the partial members that connect to it. With the tutor we have chosen to insist on the latter method: a pin alone is not viewed as a valid subsystem, but must include the partial bars that connect to it. There are two rationales for this. First, the jumble of forces near just a pin would be difficult for the tutor to interpret. Second,

showing partial bars, together with the method of specifying forces described below, is intended to help students develop a unified picture, relevant to both methods (joints and sections), as to the representation of the internal force in a member.

In Figure 7, we show the same partial free body diagram from Figure 6, now with a new force being added to a half bar. In free form solving of trusses, students draw arrows (for forces) and label those arrows with variables or numbers. With the tutor, we have chosen to require the user to categorize each force being drawn; the choices, shown in the window labeled “Defining a force”, include: known applied force, support reaction (unknown or determined), and internal force (unknown or determined). Depending on the force category chosen, a variable label or number is required. Even though a student in free form solving may not be thinking in terms of these categories, an expert, such as instructor, is surely clear when drawing a force which of these is being represented.

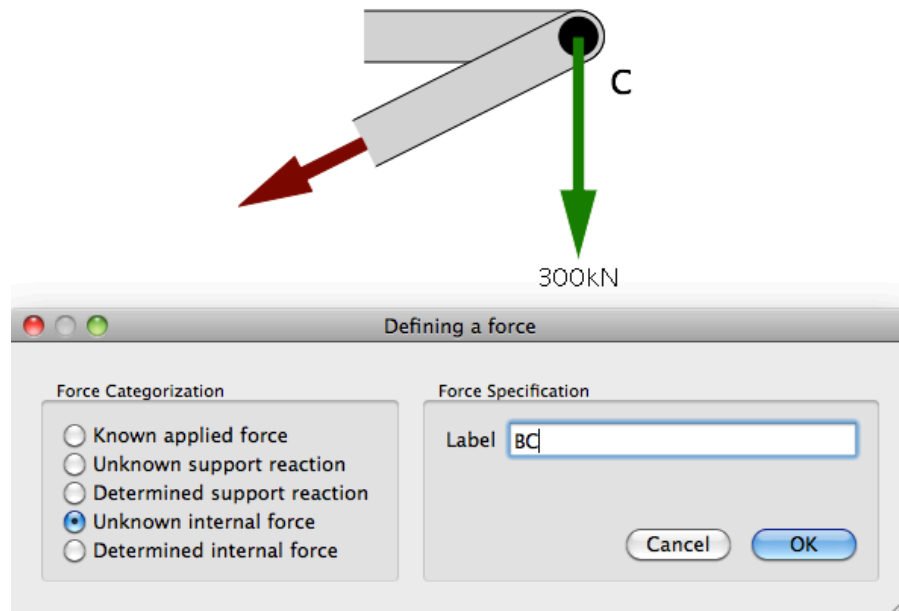


Figure 7. Screen shot of force being added to free body diagram, showing force categorization.

Requiring force categorization, together with the insistence on including partial members and pins in a subsystem, provides two benefits: (i) it helps students organize their thinking about the various forces in a way that can carry over to paper-and-pencil problem solving after tutor use and (ii) it establishes some clear bases for the tutor to recognize errors in student work, namely that applied and support forces can only act at pins, and internal forces can only act at the ends of partial bars. This requirement of categorizing forces also addresses in a partial way a general conceptual difficulty in statics: being clear which body exerts a given drawn force. It can also be seen that the requirement of selecting pins, as well as bars and half-bars, when forming a subsystem serves a similar purpose. The applied and support forces act on the pins, which have been explicitly selected to be part of the subsystem. Pins in the FBDs also serve as convenient anchors for drawing these forces, again avoiding the jumble of forces that is difficult to interpret.





When the user has written down an equation with one variable (always a linear equation in truss analysis), the tutor can solve the equation for that variable. This eliminates the need to use a calculator. The user can substitute such a solved variable into another equation that has more than one variable. But tutor does not permit the simultaneous solution of multiple equations with multiple variables. The restriction on the solving capability promotes the practice of seeking to find an equation with a single variable, which can be determined and then used in subsequent equations; such a practice is often wise when solving with pencil and paper. Admittedly, some instructors might think that packages for solving simultaneous equations are common and even available on electronic calculators, and thus students should not be forced to choose equations in this way.

Once a variable such as a support reaction or a bar internal force has been determined, the user needs to “register” that force in the solution diagram to get credit for completing that part of the problem. Registration serves to declare a force has having been determined, so it can be categorized as a determined force in a subsequent FBD). Registration is also an important opportunity for the student to signal the meaning of what has been solved. Unknown support forces can be drawn on FBD’s in any direction, and the associated variables can be positive or negative. But in the solution diagram the support force must be drawn in its actual sense and given a positive magnitude. Likewise, when the internal force of a bar is registered, the user gives it a magnitude and describes it as in tension or compression.

#### **4. Learning To Use Tutor**

We anticipate that a typical student would use the tutor for several hours over a period of a week or at most a few weeks, depending on the class. Therefore, the tutor must be easy to learn to use. Utilizing several rounds of user testing, we have sought to make its design as an intuitive and simple as possible. However, some instruction in its use will inevitably be necessary. When students first start the program, the tutor appears with an example problem loaded, and on top of the tutor window there appears a window with voice-over instruction video that addresses how to solve the example problem. A screen shot of one video tutorial at one instant is shown in Figure 9, which describes where to click and drag to add a new force to a FBD. The instruction video contains four phases, which deal with success features of using the tutor. The video pauses after each phase, and prompts the user to go to the tutor window and carryout the portion of the solution just described in the video.

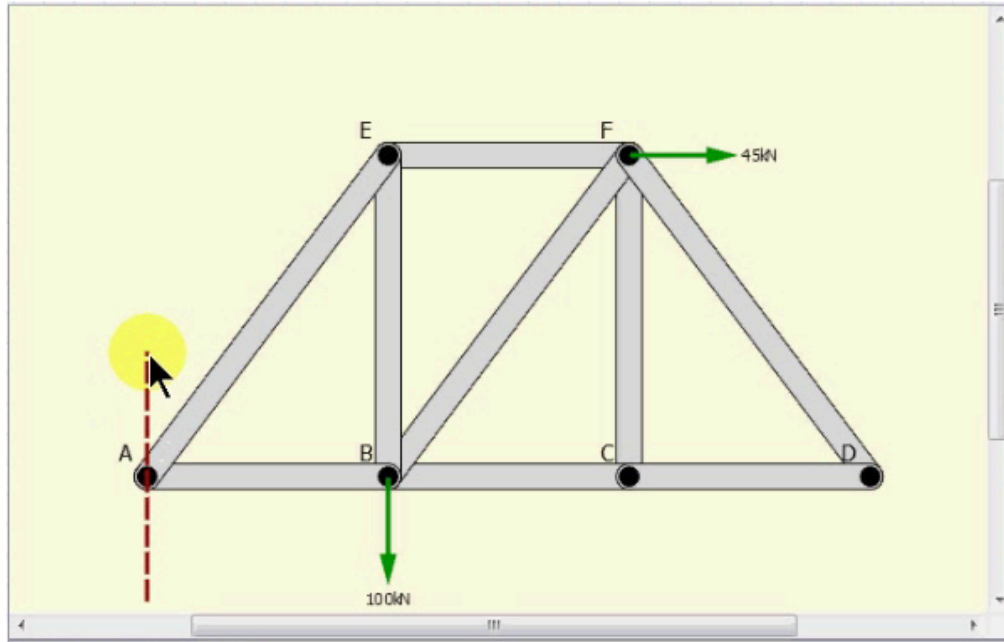


Figure 9. Instant from video tutorial when drawing of new forces is described.

## 5. Judging student work and giving feedback

A key component of the tutor will be to ascertain, judge, and give feedback on what students have done. When to offer feedback is a critical part of the tutor's design. On the one hand, we don't want to interrupt a student who is still formulating the current portion of the solution. On the other hand, we don't want to wait so long that the student builds new portions of the solution on others that are as yet unjudged and may be incorrect. In the latter situation we can find ourselves needing to indicate that the built-on portion is correct in and of itself, but that it must be redone to arrive at the correct answer because it was based on incorrect prior material.

We met this challenge in truss tutor by judging student work just after the completion of each of the major phases of the solution; namely, after: selecting a subsystem, drawing a FBD of that subsystem, writing equations of equilibrium for that FBD, and solving and registering results of those equations. Each of those has a natural breakpoint at which the student can be viewed as done: upon selection of the parts for a subsystem, it is judged; upon choosing the first equation to be written (e.g.,  $\Sigma F_x$ ), the FBD of the subsystem is judged; upon hitting return at the end of writing an equation, or choosing a next equation, the equation is judged; and, upon registering a result in the solution diagram, the registered result is judged. (The tutor avoids all errors associated with algebraic manipulation and solving the equations – it solves one equation for one unknown upon request and the user must use this facility to obtain numerical solutions.)

Provided the user does not make an error, the judging is invisible and the user can work without interruption. Upon making an error, the user receives an unmistakable error message. The message points out what is in error, with additional information to enable the user to fix the error and to learn why it is in error so it might not be repeated. The user can alter the indicated part

and proceed with the solution; the judging occurs at the same junctures so if the error is not fixed the error message will be sent again. There is not currently a hierarchy of hints from general to very direct. While they don't give the right answer, the error messages are sufficiently pointed that students manage to fix errors and move on. Whether the error is repeated in the future is the subject of the next section on modeling of student learning.

## 6. Modeling Student Learning

### *Challenge of Modeling and General Approach Taken*

Students must execute steps or actions of different types when solving truss problems; students will tend to err to varying degrees, depending on the type of action. A tutor is successful if the frequency of errors decreases with practice. Not only will different types of actions be more or less prone to error, but those actions may be learned, that is, come to be executed with fewer errors, at different rates. While, two distinct actions taken by a student in truss tutor rarely involve identical circumstances, we may sometimes consider them to involve the same knowledge, and thus viewable as distinct instances of the same type of action. At any instant in a student's learning process, those two steps are equally prone to error, and we expect that those actions will be two instances in a series of actions for which the error rate falls, ideally steadily and smoothly, with practice. The challenge is to decide how the overall solution process is to be divided into discrete actions, and which actions are to be grouped into the same type. This can be based on reasoned analysis of the knowledge that is likely to be called upon, but also upon the empirical grounds. For a given division into different types of actions, we can ask how well learning – reduction in error rate with practice - is predicted based on that division.

To analyze the progression of learning quantitatively, we adopt as far as possible the terminology, methodology, and tools from Learning Factors Analysis<sup>26</sup> and the associated tools from the PSLC Datashop<sup>24</sup>. Discrete bits of knowledge needed to solve problems in a domain are referred to as Knowledge Components (KCs). Datashop was designed based on long experience with computer tutors – where the student solves problems that are composed of predefined steps. Carrying out any particular step is assumed to require one or more KCs. In truss tutor, because students can execute a problem in distinct ways, there are no predefined steps. However, because the interface in truss tutor constrains the user to only a limited set of actions, we can extract on-the-fly from the solution actions which can be judged as correct or incorrect, and which we take as requiring one KC. (Nothing would prevent us in the future from recognizing that some actions require more than one KC.)

Log files from students using truss tutor are saved as the student works. From the log files, we can reconstruct each instance in which an action and corresponding KC is called for and whether it is applied properly. A learning curve typically pertains to a KC, and in it the fraction of students making an error using the KC is plotted as a function of the opportunity (first, second, third). The Datashop tools also fit a statistical model, which reflects the power law of learning, to the data. More specifically, for our case in which each action is dependent on a single KC, the fit predicts performance or error rate as follows:

$$\log \left[ \frac{p_{ij}}{1 - p_{ij}} \right] = \theta_i + a_j + b_j T_j$$

where  $p_{ij}$  is the probability of a correct answer by the  $i$ th student on opportunity  $T_j$  for using the  $j$ th KC. (In the more general Additive Factors Model, there is a sum of terms  $a_j$  and  $b_j T_j$  for each of the KC's that contributes to being able to execute the action.)

The fit yields a student-specific parameter, an overall initial skill  $\theta_i$ , which is independent of the KC. The fit also produces coefficients  $a_j$  and  $b_j$ , both KC-dependent. The coefficient  $a_j$ , the intercept, corresponds to the initial ease of correctly applying the KC. The coefficient  $b_j$ , referred to as the slope, corresponds to how quickly successive opportunities to use a skill reduces errors in using it. Thus, values for  $b_j$  are one measure of the tutor's effectiveness. In particular, more effective error messages or hints may lead to higher slopes.

### *Initial KC Model for Truss Tutor*

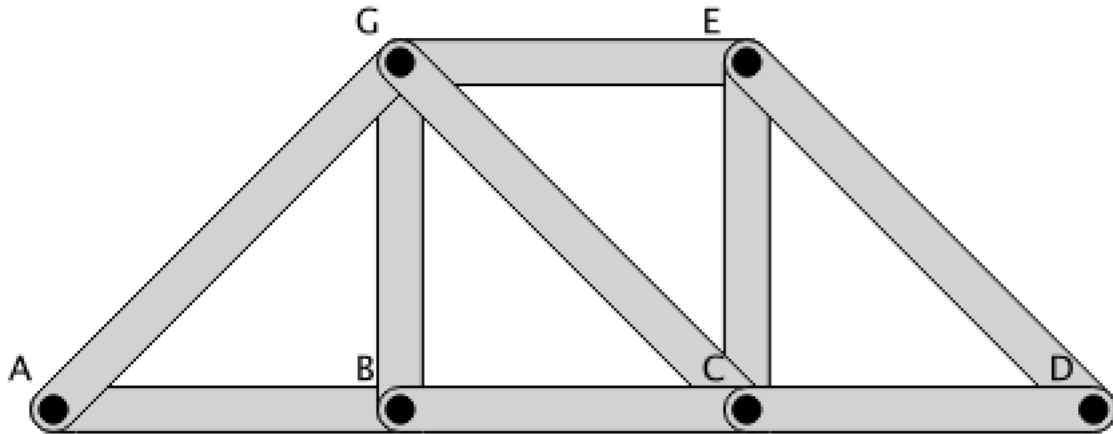
The collection of Knowledge Components chosen their association with specific student actions is referred to as the KC Model. Many different KC models are possible; in this section we describe the initial KC model chosen for Truss tutor. Each KC corresponds to an action that falls into one of the four phases of the solution process: (i) selecting a subsystem, (ii) drawing a free body diagram, (iii) writing an equation of equilibrium, and (iv) registering a result derived from an equation of equilibrium in the solution diagram. In every case, once an attempt at a KC has been made and is incorrect with feedback given, correction of that error does not constitute a new attempt at the KC.

#### Selecting a subsystem

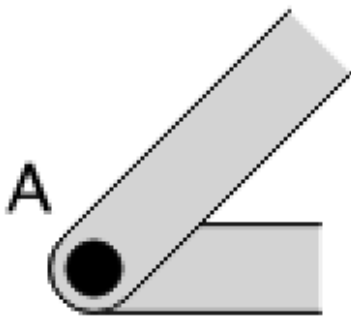
Legitimate subsystems are either the entire truss, or a portion of the truss consisting of pins, connected members, and partial members. For a portion of the truss, every included pin must also have each bar or have bar connected to it. (The rationale for our choice of insisting on partial bars connected to pins, and not just pins alone, was described earlier.)

We define three distinct KCs: whole truss as subsystem, joint as subsystem, and section as subsystem. There is a particular challenge in implementing judging the correctness of KCs associated with subsystem: there are many technically correct subsystems that are irrelevant to the problem or not useful. For example in a MOJ problem, which explicitly demands that the solution be arrived at from a series of joints, a section could be selected. Or for a MOS problem, which explicitly demands that the internal forces in a few selected bars be determined from a single section, a subsystem with too many unknowns or a section not exposing the desired bars, may be selected. In those cases, the user is told that the subsystem is validity but that the problem cannot be solved with it and why. This prevents the student from wasting time analyzing a subsystem which is will not ultimately lead to problem completion. That selection of a

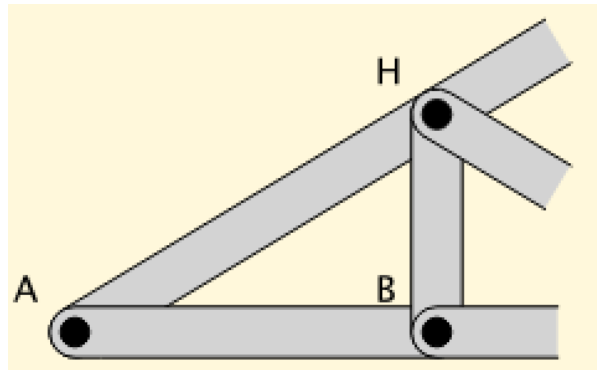
subsystem is not counted as an opportunity to practice the KC. Subsystems are shown in Figure 9.



full\_truss\_as\_subsystem



joint\_as\_subsystem



section\_as\_subsystem

Figure 9. Knowledge Components (KCs) associated with defining subsystems.

### Drawing a FBD

When drawing a force on a FBD, the user chooses from five choices: applied force, unknown support reaction, determined support reaction, unknown internal force, and determined internal force. In the case of unknown forces, the user must supply a label, and in the case of applied forces or determined forces, the user must supply a value. Applied forces and support reactions can only be legitimately applied to pins, and internal forces can only be legitimately applied at the ends of partial bars (see Figures 10a and 10b). Note that the KC related to a free pin enables one to have a KC, incorrectly applied, when a force is added at a free pin – when the pin is left blank it is correct. In practice the error rate on free pins is extremely low.

The system gives the user additional freedom, which is comparable to that exercised in paper and pencil. Even if a user has determined a support reaction and registered it in the solution diagram, when that support reaction appears on a subsequent subsystem, the user could choose to declare it undetermined (Figure 10b). In that case, though, the user would need to give the forces variable labels that are consistent with those originally drawn, and the unknown forces must be drawn in directions consistent with first definition of those unknowns.

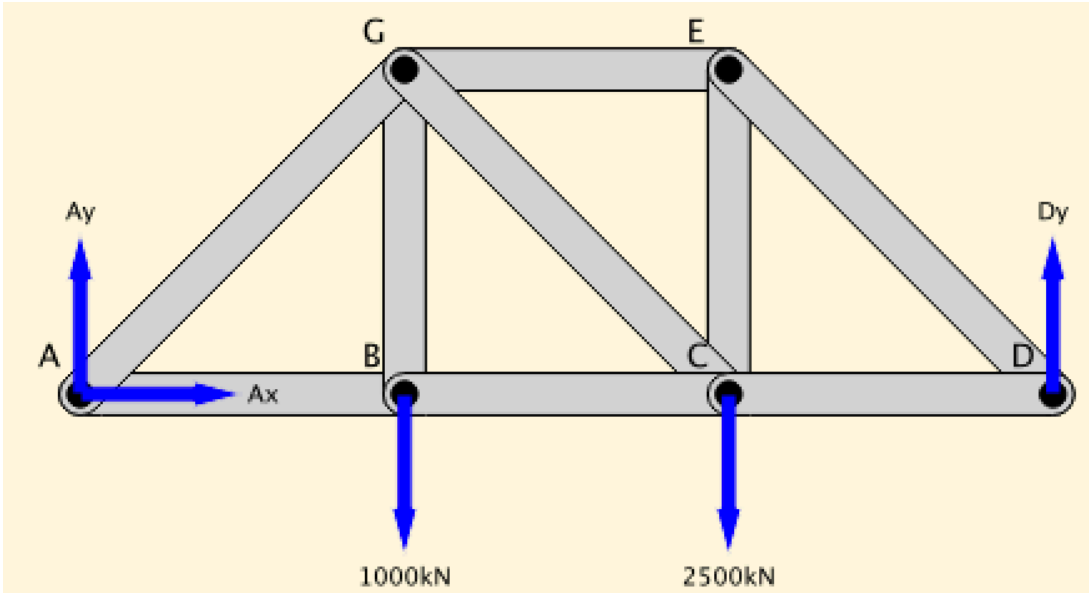


Figure 10a. Unknown support reactions (pin,  $A_x$  and  $A_y$ , roller:  $D_y$ ), applied forces (at B and C), and free pins (E and G).

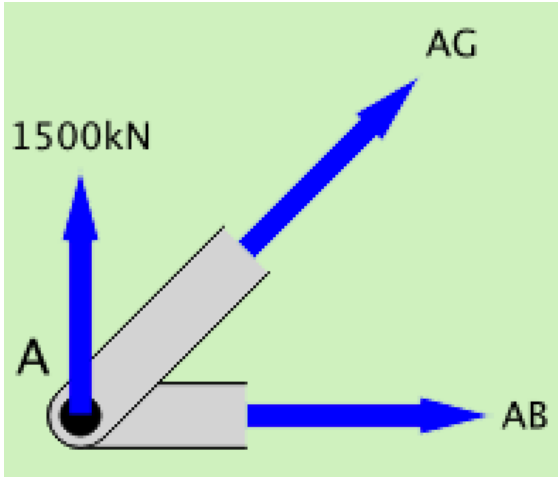


Figure 10b. Unknown internal forces (AB and AG), and determine support reaction at A; note the reactions at A could

In like manner, an internal force can be declared as unknown even if it had determined previously, but the label must again be consistent. Or the determined internal force, with correct value and sense, could be used. Such KCs are shown in Figure 11.

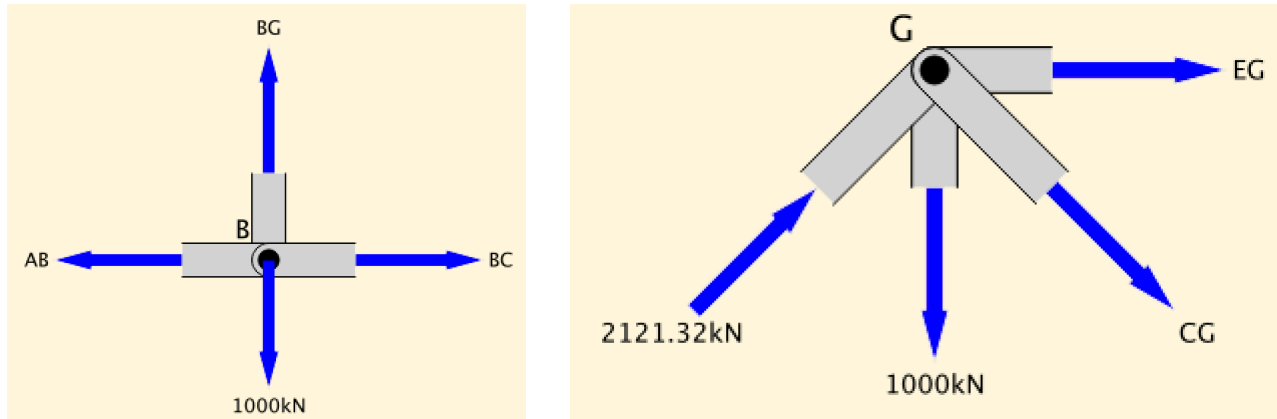


Figure 11. Unknown internal forces consistent (AB to be consistent with FBD of joint A), or determined internal force (2121.32 kN, found from AG, determined by previously analyzing joint A).

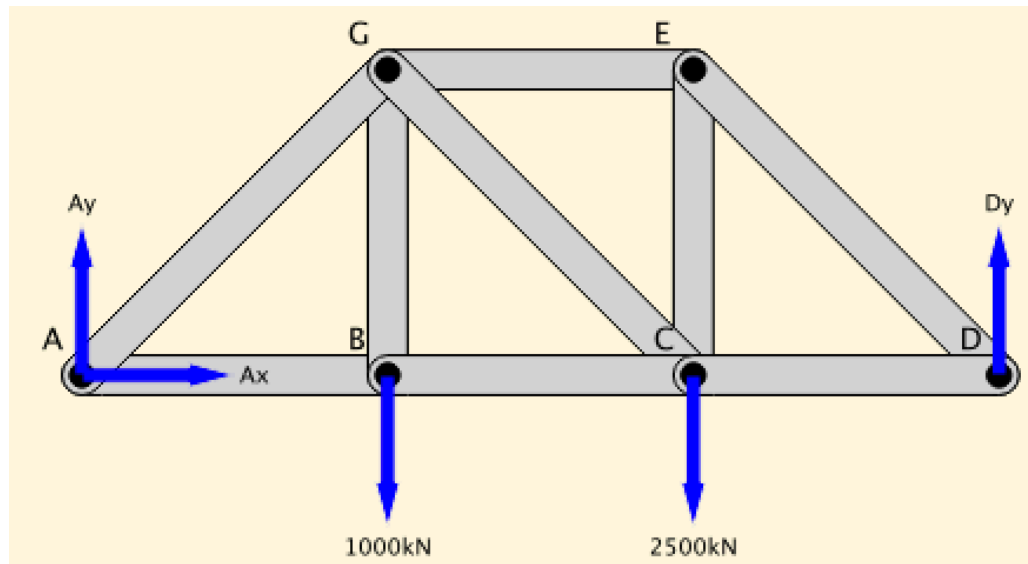
### Writing equations of equilibrium

Because of the truss tutor interface, equations are typed in spaced underneath the FBD, so it is clear to which FBD equations pertain (something that might not always be clear in hand written solutions). For each equation written, users declare whether it is a force summation in x, a force summation in y, or a moment about a selected point. This enables the program to determine whether the equation written is indeed correct. Of all the phases in solving truss problems, it is probably least obvious how to divide up equations into evaluable portions and to which KCs they should be ascribed. There is also question of what can feasibly be pulled apart from the equations. Terms in an equation include those involving constants and those involving variables. Forces in a given FBD are constants or involve variables, which must be unique. Thus, terms in an equation that involve variables can be unambiguously associated with forces and can be judged. Constant terms cannot be separately judged: if there were two constant forces that contributed to the equation, they could be added up in the mind of the user and the total entered. Or, a user could, perversely, take one force and write it as the sum of two forces in the equation, and it would still be correct. In the case of moments, there is even more potential for ambiguity, since the moment is the product of force times distance, which could be expressed in many ways. Thus, only the total of constant terms can be judged with certainty, not the distinct entries which sum to the total.

KCs involving variables may be more or less error prone than those involving constants, and KCs involving moments are likely to be more error prone than those involving just forces,

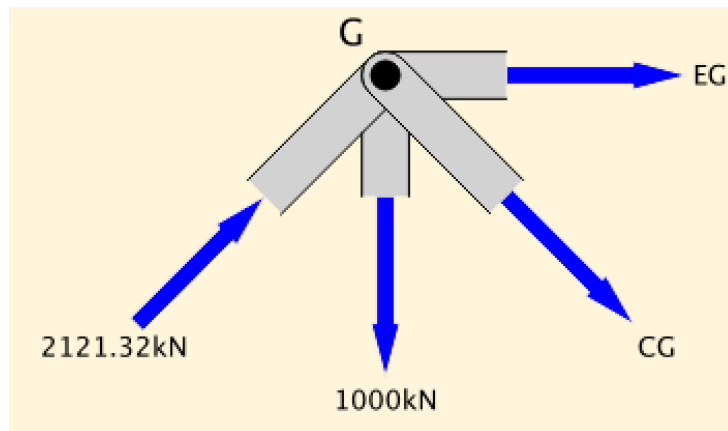


because errors can be made both on the force and the distance. It is also likely that force contributions that necessitate a force to be resolved is more error prone than a force oriented along the x or y axes. Errors associated with computing moments are likely to be more likely when the force acts at a point that is neither along simply the x or y direction from the moment center. In light of the above potential variation in error rates, the KCs for writing equations have been chosen as in Figures 12 and 13.



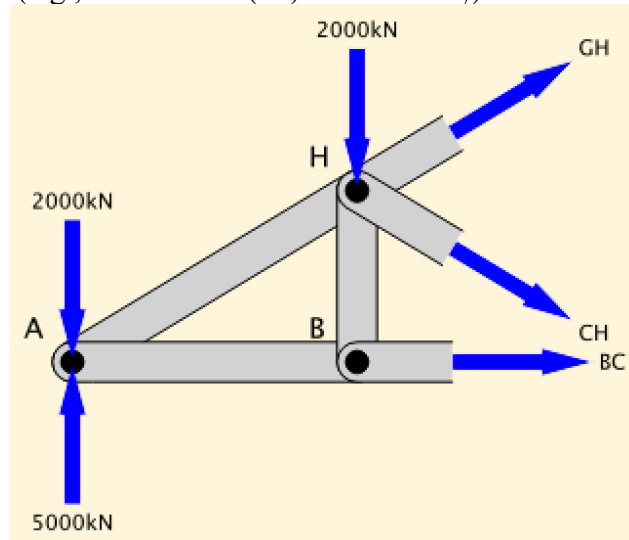
$$\begin{array}{l}
 \sum F_x = 0 \text{ (kN)} \quad \mathbf{Ax} = 0 \\
 \sum F_y = 0 \text{ (kN)} \quad \mathbf{Ay} - 1000 - 2500 + \mathbf{Dy} = 0 \\
 \sum M|_A = 0 \text{ (kN-m)} \quad -1000 * 10 - 2500 * 20 + \mathbf{Dy} * 30 = 0
 \end{array}$$

Figure 12a. KCs associated with variable force ( $A_x$ ,  $A_y$ ,  $D_y$ , in  $\Sigma F_x$  and  $\Sigma F_y$ ), constant force (-1000-2500 in  $\Sigma F_y$ ), variable moment ( $D_y * 30$  in moment equation), constant moment (-1000\*10-2500\*20 in  $\Sigma M|_A$ ).



$$\begin{array}{l}
 \sum F_x = 0 \text{ (kN)} \quad 2121.32 * \cos(45^\circ) + \mathbf{CG} * \cos(45^\circ) + \mathbf{EG} = 0 \\
 \sum F_y = 0 \text{ (kN)} \quad 2121.32 * \sin(45^\circ) - 1000 - \mathbf{CG} * \sin(45^\circ) = 0
 \end{array}$$

Figure 12b. KCs associated with resolving variable force (e.g.,  $CG \cdot \cos(45)$  in  $\Sigma F_x$ ), and resolving constant force (e.g.,  $2121.32 \cdot \sin(45) - 1000$  in  $\Sigma F_y$ ).



$$\sum M|_C = 0 \text{ (kN-m)} \quad -5000 \cdot 20 + 2000 \cdot 20 + 2000 \cdot 10 - GH \cdot \cos(30.9638^\circ) \cdot 6 - GH \cdot \sin(30.9638^\circ) \cdot 10 = 0$$

Figure 13. KCs associated with resolving variable moments (e.g., terms involving GH), and resolving and combining constant moments (constant terms because moment center is taken about point C).

While not shown, another skill is substituting the value of a determined variable into an equation. More fine-grained KCs models for writing equations, which may better fit the observed variation in error rates, are being considered.

### Registering Variables

Once a support reaction is determined it must be registered in the solution diagram. This includes using the assumed direction in the FBD, and sign of the value found for the variable, to determine the correct direction of the reaction. Likewise, when the internal force is computed from equilibrium equations, the sign is used to infer whether the bar is in tension (if positive) or compression (if negative). The registration of support reactions and internal forces are tracked as separate KCs (Figure 14).

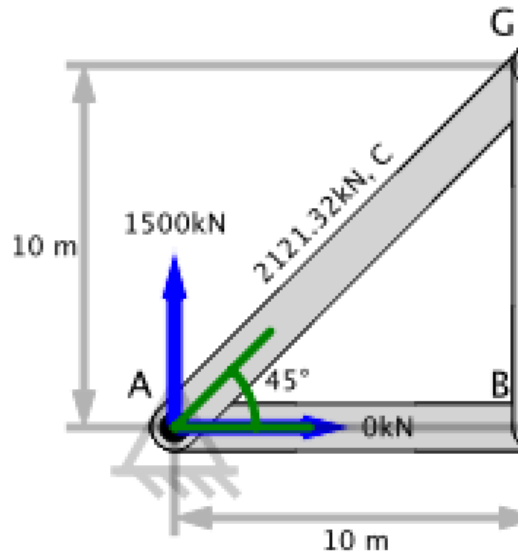


Figure 14. KCs associated with registering determined support reactions (1500 kN and 0 kN at A) and internal forces (2121.32 KN in compression).

### 7. Learning Curves and KC modeling results for an Initial data set

A first dataset from students, who were first learning trusses and using only Truss Tutor as their problem-solving mode, was obtained from cohort of students in a 3 credit-hour statics course at a community college. A total of 21 students were in the class. Students had received a lecture on trusses, covering the method of joints and method sections, and the instructor worked through an example of each for the whole class. Thereafter, student practiced solving trusses exclusively using Truss tutor. Of those students 18 registered to participate in using Truss Tutor and submitted completed problems.

The Learning Curves, which plot (in red) the fraction in error for each KC as a function of opportunity, are shown in Figure 15. The dashed blue curve shows the model prediction of the error rate once the parameters are fit.

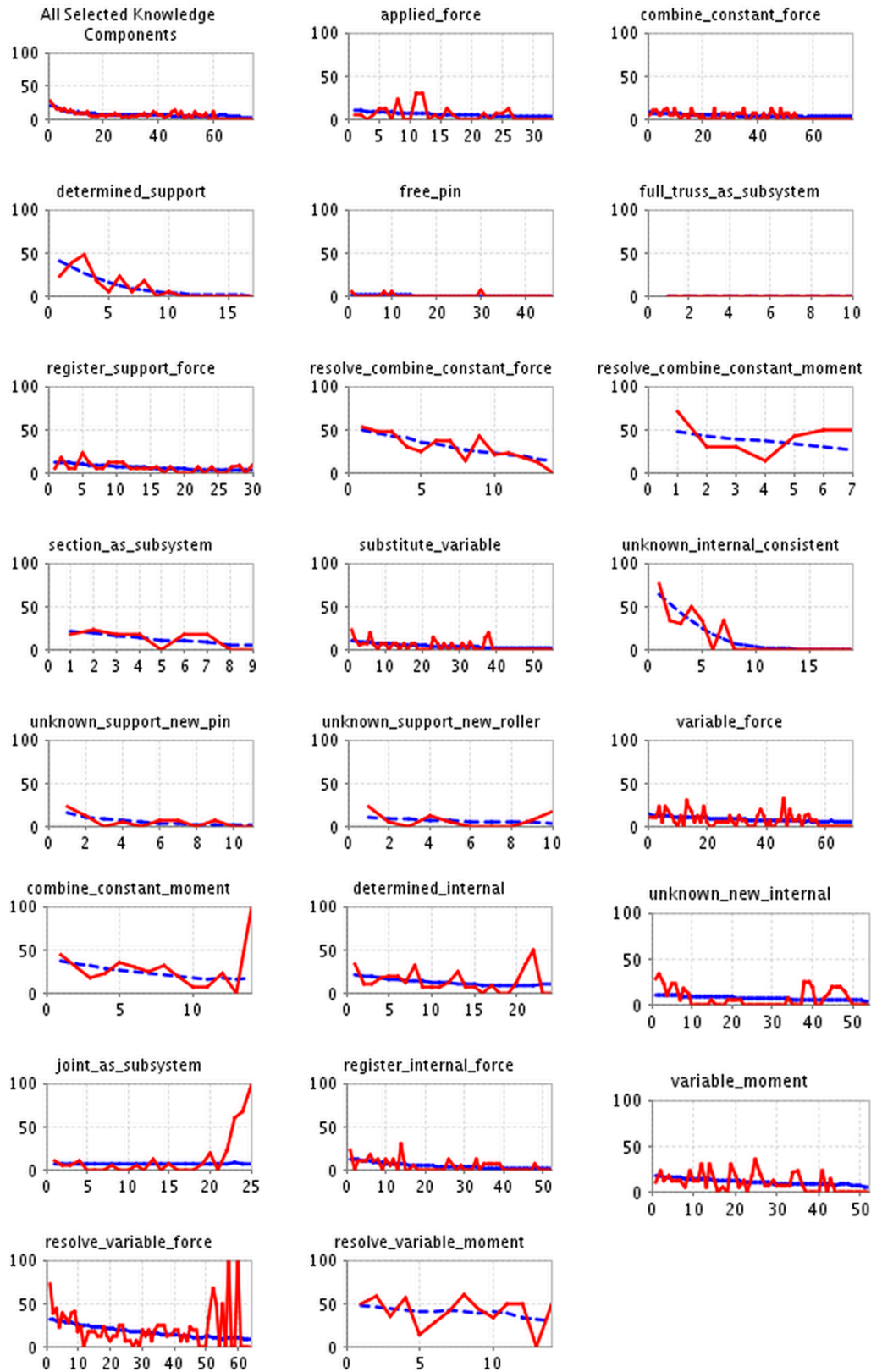


Figure 15. Learning Curves for KC model based on 18 student cohort.

The results of the model fits are shown in Table 1. The KCs have been divided into the four major groups, and then have been sorted from highest to lowest intercept  $a$  (fewest errors to most errors). Ideally, the KCs would either be: relatively free of errors, with a high intercept, in which case significant improvement (high slope) is not expected, or have a lower intercept but with a high slope. In the latter case, students are committing errors, but they are learning with the tutor, that is, making fewer errors with practice. The model fit also includes student specific parameter, which is usually not of interest, but it does vary significantly over students.

TABLE 1. Fit Model parameters  $a$  and  $b$  for different Knowledge Components.

	Intercept ( $a$ )	Slope ( $b$ )
<b>DEFINING SUBSYSTEM</b>		
full_truss_as_subsystem	1.00	4.54
joint_as_subsystem	0.94	0.00
section_as_subsystem	0.74	0.20
<b>FBD</b>		
unknown_support_consistent	1.00	0.11
free_pin	0.98	0.07
applied_force	0.90	0.04
unknown_new_internal	0.89	0.02
unknown_support_new_roller	0.89	0.12
unknown_support_new_pin	0.82	0.28
determined_internal	0.79	0.06
determined_support	0.51	0.32
unknown_internal_consistent	0.26	0.46
<b>EQUILIBRIUM EQUATIONS</b>		
combine_constant_force	0.93	0.02
substitute_variable	0.90	0.04
variable_force	0.87	0.02
variable_moment	0.83	0.02
resolve_variable_force	0.69	0.03
combine_constant_moment	0.61	0.11
resolve_variable_moment	0.50	0.06
resolve_combine_constant_moment	0.47	0.17
resolve_combine_constant_force	0.47	0.13
<b>INTERPRET-REGISTER RESULTS</b>		
register_internal_force	0.88	0.05
register_support_force	0.87	0.06

To provide context for the results found here, slopes of 0.05 to 0.15 are common across data sets in other learning contexts. So, the fit suggests that relatively high learning, even in comparison with other tutoring systems, is occurring for a number of KCs in the truss tutor.

Within Selecting Subsystems and drawing FBDs, the lower intercept skills have comparatively high slopes, with the one exception of determined internal. This corresponds drawing a force on an FBD for a bar whose internal force has already been determined, and using the correct value and the correct sense, depending on whether the bar was found to be in tension or compression. From the particular errors associated with this KC, which are tracked but not displayed on learning curves, it is found that the most frequent error is in the sense. Further work needs to be done to determine if error messages could be improved. Within Registering Results, which involves interpretation, the error rates are relatively low and there is some slow improvement with practice.

Within writing Equilibrium Equations, the KC model does not give evidence of learning: the slopes for a number of the skills with higher intercepts are still reasonably low. Writing equilibrium equations for a given FBD is not a new skill developed in truss analysis, but has been practiced by students repeatedly in earlier statics problems. Improving student ability to write equations was not viewed as likely attained goal in truss tutor. In the case of writing equilibrium equations, error messages only point to the term or terms in an equation that are in error. It was not viewed as feasible to offer more explanation as to why a force was entered or resolved wrong, or why a moment was computed incorrectly. Interestingly, the KCs that deal with using variables tend to have lower slopes than those associated with constant terms. Perhaps students find it more difficult to manipulate terms involving variables than numbers, a result that may not be surprising. More fine grained KC models of writing equilibrium equations, which are currently being developed, may enable us to determine more precisely which aspects of writing equilibrium equations students find the most difficult.

## **8. Summary and Conclusions**

Statics is an important course for many engineering majors, and learning to solve problems is a key goal of statics. Particularly with more realistic topics in statics, such as structures, problems often involve multiple analyses, which can be carried out independently; thus, there are many pathways to solution. Feedback is important to learning, particularly when students are engaged in problem solving. To the extent possible, one wishes to capitalize on the potential of the computer to provide such feedback to students when instructors or human tutors are not present. But, such feedback can be challenging for the computer to offer in the case of complex problems, when students are allowed to pursue different pathways to solutions. To follow the steps of students solving such problems, the computer needs a cognitive model for the multitude of different steps a user could take and still be able to compare the user's steps with correct ones pertinent to a chosen solution pathway. An additional challenge is for the computer to recognize the complex graphical and textual input in mechanics.

Our goal has been to develop a computer tutor that can offer students feedback as they practice analyzing trusses. The tutor should efficiently improve the student's ability to analyze trusses,

with the heightened skills applicable to solving truss problems with pencil and paper. Such a tutor, if successful, may point the way to new computer tutors appropriate to student solving of complex multi-path problems in other subjects.

A first version of the Truss Tutor has been described. The design of the tutor has sought to strike a balance between (1) allowing students wide latitude to solve truss problems correctly and to commit errors typical of novices and (2) constraining student actions so as to be unambiguously interpretable by the tutor. In a few instances, users of the tutor must take actions or make choices that do not have precise counterparts to pencil and paper solving. These serve either to help students organize their thinking and/or to make interpretation by the tutor more straightforward. The tutor offers feedback to users at convenient points, seeking a balance between not interrupting work in progress, but not allowing incorrect work to accumulate and be the basis for subsequent work. The feedback is sufficient for nearly all students to complete work correctly on all problems.

To track whether students are learning, namely that errors are becoming fewer with practice, we have adopted the approach of Learning Factors Analysis. We seek, based on analyzing the task of solving truss problems, to identify discrete skills or Knowledge Components (KCs). The tutor recognizes the exercise of each such KC as the student practices and records the outcome, correct or incorrect, which is plotted on a so-called Learning Curve. This sequence of opportunities to exercise KC is then analyzed via a statistical model, which assumes that the probability of error for a student depends on the particular KC, on the number of times the KC has been exercised, and on the student. The statistical model fits a limited set of parameters to the data; the fit parameters gives insight into how different skills are mastered over time. It is found that the some of the KCs involve few errors right from the start, other KCs are error prone, but the error rate does improve markedly with practice, and for other KCs, the reduction in error is less than desired. Results from such Learning Curve analysis enable one to reconsider the choice of KCs and their mapping to the various actions students take. The results also enable one to identify potential improvements to the tutor, such as improved error messages, which may lead to greater reductions in error rate with practice.

In sum, at its current stage of development Truss Tutor is successful in enabling student to solve truss problems relatively freely on the computer, and in offering feedback that enables nearly most students to complete problems correctly. Analysis based on Knowledge Components and a statistical fit of the Knowledge Component model to data both indicate that the rate at which many types of errors occur decreases with time, and provide a quantitative basis on which further improvements to the tutor should be possible.

### **Acknowledgement**

We thank Jackie Yang, Jeremy Jiang, and Rebecca Piston for their assistance in development and initial testing of Truss Tutor. We thank Ken Koedinger for insights into the learning curve analyses. Support by the National Science Foundation under grant DUE-1043241 is gratefully acknowledged.

## Bibliography

1. Harris, T.A. and Jacobs, H.R., 1995, *J. Eng. Educ.*, Vol. 84, p.343.
2. Bangert-Drowns, R. L., Kulik, C.-L., Kulik, J. A., & Morgan, M. (1991). The instructional effect of feedback in test-like events. *Review of Educational Research*, 61, 213-238.
3. Corbett, A.T. and Anderson, J.R. (2001). Locus of feedback control in computer-based tutoring: Impact on learning rate, achievement and attitudes. *Proceedings of ACM CHI'2001 Conference on Human Factors in Computing Systems*, 245-252.
4. Lewis, M. W., & Anderson, J. R. (1985). Discrimination of operator schemata in problem solving: Learning from examples. *Cognitive Psychology*, 17, 26-65.
5. Lee, A. Y. (1992). Using tutoring systems to study learning: An application of HyperCard. *Behavior Research Methods, Instruments, & Computers*, 24(2), 205-212.
6. Schmidt, R. A., & Bjork, R. A. (1992). New conceptualizations of practice: Common principles in three paradigms suggest new concepts for training. *Psychological Science*, 3(4), 207-217.
7. Schooler, L. J., & Anderson, J. R. (1990). The disruptive potential of immediate feedback. In M. Piattelli-Palmarini (Ed.), *Proceedings of the Annual Conference of the Cognitive Science Society* (pp. 702-708). Hillsdale, NJ: Erlbaum.
8. Roselli, RJ, Howard, L and Brophy, S. (2006) A computer-based free body diagram assistant. *Computer Applications in Engineering Education* 14: 281-290.
9. Dannenhoffer, J. & Dannenhoffer, J. (2009). An online system to help students successfully solve statics problems. *Proceedings of the 2009 American Society for Engineering Education Annual Conference & Exposition*, Austin, Texas, June 2009.
10. Peschel, J., and Hammond, T. STRAT: A Sketched-Truss Recognition and Analysis Tool. *International Workshop on Visual Languages and Computing (VLC 2008)*, Boston, Massachusetts, September 4-6, 2008, pp. 282--287.
11. Newton's Pen - A Pen-based Tutoring System for Statics WeeSan Lee, Ruwanee de Silva, Eric J. Peterson, Robert C. Calfee, and Thomas F. Stahovich, *Computers & Graphics*, 32(5):511-524, October 2008.
12. Sleeman, D. H., & Brown, J. S. (1982). *Intelligent Tutoring Systems*. New York, NY: Academic Press.
13. Wenger, E. (1987). *Artificial Intelligence and Tutoring Systems: Computational and Cognitive Approaches to the Communication of Knowledge*. Los Altos, CA: Morgan Kaufmann.
14. David J. Gunaratnam, A knowledge-based expert system for tutoring in structural engineering, *Computers & Structures*, Volume 30, Issue 3, 1988, Pages 767-773
15. Ana Lilia Laureano-Cruces and Fernando De Arriaga-Gomez, Multi-Agent Architecture for Intelligent Tutoring Systems, *Interactive Learning Environments*, Volume 6, Issue 3 December 1998 , 225 - 250
16. Ana Lilia Laureano Cruces and Fernando De Arriaga, REACTIVE AGENT DESIGN FOR INTELLIGENT TUTORING SYSTEMS, *Cybernetics and Systems*, Volume 31, Issue 1 January 2000, 1 – 47
17. Kulik, C. C. & Kulik, J. A. (1991). Effectiveness of Computer-Based Instruction: An Updated Analysis. *Computers in Human Behavior*, 7, 75-95.
18. Anderson, J. R., Boyle, C. F., & Reiser, B. J. (1985). *Intelligent tutoring systems*. Science, 228, 456-468.
19. Newell, A., & Simon, H. A. (1972). *Human Problem Solving*. Englewood Cliffs, NJ: Prentice-Hall.
20. P.S. Steif, 2004, An Articulation of the Concepts and Skills Which Underlie Engineering Statics, 34rd ASEE/IEEE Frontiers in Education Conference, Savannah, GA.
21. P.S. Steif and J. A. Dantzler, "A Statics Concept Inventory: Development And Psychometric Analysis", *Journal of Engineering Education*, Vol. 94, pp. 363-371 (2005).



22. Steif, P.S. and Hansen, M.A, "Comparisons Between Performances In A Statics Concept Inventory and Course Examinations," *International Journal of Engineering Education*, Vol. 22, pp.1070-1076 (2006)
23. P.S. Steif and M. A. Hansen, "New Practices for Administering and Analyzing the Results of Concept Inventories", *Journal of Engineering Education*, Vol. 96, pp. 205-212, (2007).
24. Koedinger, K.R., Baker, R.S.J.d., Cunningham, K., Skogsholm, A., Leber, B., Stamper, J., (2011) A Data Repository for the EDM community: The PSLC DataShop. In Romero, C., Ventura, S., Pechenizkiy, M., Baker, R.S.J.d. (Eds.) *Handbook of Educational Data Mining*. Boca Raton, FL: CRC Press.
25. Dollár, A. and Steif, P.S., "An Interactive, Cognitively Informed, Web-Based Statics Course," *International Journal of Engineering Education*, Vol. 24, No. 6, pp. 1229–41 (2008).
26. Cen, H., Koedinger, K. R., & Junker, B. (2006). Learning Factors Analysis: A general method for cognitive model evaluation and improvement. In M. Ikeda, K.D. Ashley, T.-W. Chan (Eds.) *Proceedings of the 8th International Conference on Intelligent Tutoring Systems*, 164-175. Berlin: Springer-Verlag.