International Journal on Artificial Intelligence Tools
© World Scientific Publishing Company

DETERMINING THE DIMENSIONS OF VARIABLES IN PHYSICS ALGEBRAIC EQUATIONS

C.W. Liew

Department of Computer Science
Lafayette College
Easton, PA 18042
liew@cs.lafayette.edu

Joel A. Shapiro

Department of Physics and Astronomy
Rutgers University
Piscataway, NJ 08854-8019
shapiro@physics.rutgers.edu

D.E. Smith

Department of Computer Science
Rutgers University
Piscataway, NJ 08854-8019
dsmith@cs.rutgers.edu

Received 12 July 2004
Revised (15 December 2004)
Accepted (Day Month Year)

This paper describes work on methods that evaluate algebraic solutions to word problems in physics. Many current tutoring systems rely on substantial scaffolding and consequently require students to *completely* describe every variable used in the solution. A heuristic, based on constraint propagation, capable of inferring the description of variables (*i.e.*, the possible dimensions and physics concepts) is shown to be highly reliable on three real world data sets, one covering a few problems with a small number of student answers and two others covering a large class of problems (∼100) with a large number of student answers (∼11,000). The heuristic uniquely determines the dimensions of all the variables in 91–92% of the equation sets. By asking the student for dimension information about one variable, an additional 3% of the sets can be determined. An ITS tutoring system can use this heuristic to reason about a student's answers even when the scaffolding and context are removed.

Keywords: Intelligent tutoring systems; constraint propagation; units

1. Introduction

In teaching problem solving, Intelligent Tutoring Systems (ITS) often employ a rigid and explicit framework to guide the student along a predetermined sequence of steps. This mechanism, called scaffolding[16,2,9], is pedagogically sound and beneficial to beginning students in the subject, because it helps them learn how to systematically analyze a complex problem. After some experience, students internalize these steps, and the best pedagogy changes from requiring explicit demonstration of each and every step to allowing the most basic of these steps to be performed implicitly. In fact to continue to require explicit demonstration of these basic steps often frustrates the students, making the task more tedious than instructional. At some point, the scaffolding should be relaxed by the tutoring system.

Removing the scaffolding puts a greater burden on both the student and the tutoring system. The student must do more on his own without feedback from the tutor and the system must now interpret answers that may be in a different sequence or may have incorporated some basic assumptions. This is especially true when there are many ways to specify the solution, as when there are multiple equation sets that correctly describe the physics of the problem, represented in many equivalent forms with differing numbers of equations and variables. In addition, students can use any one of many different variable names to refer to a single physical property. Tutoring systems must be able to infer properties that are referred to by the variables in a set of equations before they can evaluate the correctness of the equations.

This paper describes our continuing work on developing approaches that reduce an ITS's reliance on scaffolding. In particular, we examine issues of identifying the meaning of variables in equation sets that solve college level introductory physics problems. Our initial techniques worked for a small set of problems and on a small number of students. Subsequent analyzes of a larger corpus showed that improvements were needed and the matching techniques were extended as described in section 4.1.1. The improved technique uniquely determine the dimensions of all the variables in 91–92% of the sets of equations. By asking for dimension information about one variable, an additional 3% of the sets could be determined. Earlier descriptions of this work can be found in [6,7,10,11,8,9]. Some of the results have been reported in [6,11,9].

These analyses show that a physics tutoring system can relax scaffolding and still reliably and robustly determine the dimensions of the variables used in the equations. This knowledge can in turn be used to identify the physical quantities corresponding to student-chosen variables used in their equations, and to associate them with a canonical solution set of variables and equations. We plan to build on these results to identify the physical concepts employed by students to solve a problem and to provide effective feedback when irrelevant concepts are used, or relevant concepts are omitted or used incorrectly.

2. Algebraic Physics Problems

Physics uses sets of algebraic equations to specify the interrelations of a set of physical quantities. One of the main differences between *generic* algebraic equations and algebraic equations describing a relationship in physics is that the latter must be dimensionally consistent. Two algebraic equations in physics are shown below.

$$T - m_1 * g = m_1 * a_1 \tag{1}$$

$$a_1 = -a_2 \tag{2}$$

Algebraically speaking, these equations could be added to one another to form a new equation. Physically speaking, equations describe the constraints between quantities given by the laws of physics while variables represent physical properties of an object or a system of objects. Consequently each of the variables, constants, terms, expressions, and even equations has specific dimensions and can only be combined using dimensionally consistent operations. For example, equation 1 is likely to have the dimensions of force (*i.e.*, $\mathrm{kg} \cdot \mathrm{m/s^2}$) while equation 2 would have dimensions of acceleration (*i.e.*, $\mathrm{m/s^2}$). It would be incorrect to add these equations, since that operation would violate dimensional consistency.

As a first step in deciding if an algebraic equation in physics is correct, a system can check if the equation is *dimensionally consistent*. This is analogous to verifying that the syntax of a program is correct by verifying that the type of each variable is consistent with the operations on that variable. This is a straightforward check if the meanings of the variables are known; if not, it provides constraints on what they can signify.

2.1. *Issues in Removing the Scaffolding*

Removing the scaffolding imposes an additional computational requirement on tutoring systems. We illustrate this with an example problem based on Atwood's machine, a pulley with two masses, m_1 and m_2 hanging at either end, as shown in Figure 1.

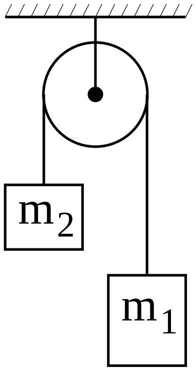


Fig. 1. Atwoods Machine

A common problem based on Atwood's machine asks the student for the equation(s) that would determine the acceleration of the mass m_1, assuming that m_1

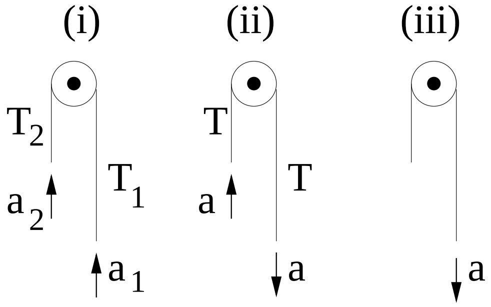


Fig. 2. Different variable sets describing the solution.

and m_2 are not equal. Equations 3 through 6 represent one solution to the problem using variable set (i) in Figure 2.

$$Forces/Acceleration\ on\ Block1: \quad T_1 - m_1 * g = m_1 * a_1 \quad (3)$$

$$Forces/Acceleration\ on\ Block2: \quad T_2 - m_2 * g = m_2 * a_2 \quad (4)$$

$$Tension\ in\ rope: \quad T_1 = T_2 \quad (5)$$

$$Acceleration\ of\ connected\ blocks: \quad a_1 = -a_2 \quad (6)$$

From a pedagogical standpoint, physics instructors teach beginning students that the steps involved in solving problems of this type are:

(1) *variable definition:* Each variable is defined with the object(s) and properties to which it refers. In some cases, the time when this variable is applicable is also defined.
(2) *identification of physics laws:* Each applicable physics law, *e.g.*, force balance or conservation of momentum, must be identified and the objects to which they apply must be specified.
(3) *instantiation of physics laws:* The general physics laws are stated as equations with "textbook" variables. Each variable specified from the first step is substituted as appropriate for the textbook variables. The result is an equation or system of equations sufficient to solve for all unknowns in the current problem.
(4) *solving the equation set:* The algebraic manipulations are performed to solve for the required variables. Our tutor does not address this step.

As students become accustomed to the vocabulary of the domain, they start using problem solving "shortcuts". Instead of defining each variable explicitly, the students select from a dictionary of well-known physics variables to represent the properties that they desire. For example in Newtonian mechanics, variables beginning with m typically represent masses, variables beginning with a typically represent accelerations, and variables beginning with T may represent tensions (*i.e.*, forces). Thus the naming of a variable implicitly specifies possible dimensions or properties and the subscripts of each variable specifies the object(s) to which the variable refers. For example, m_1 and a_1 would refer to the mass and acceleration of the same object while $p_{1,t1}$ might refer to the momentum of object 1 at time $t1$.

When the scaffolding is removed, the tutoring system must be able to determine the context of the system of equations. For example, rather than describe the Atwoods problem using equations 3 through 6, the student might choose to use a single variable a to represent acceleration, and a single T for the tension, implicitly using the principle that equates T_1 and T_2, and the constraint $a_1 = -a_2$, which comes from the fixed length of the cord. Variable set (*ii*) in figure 2 identifies the variables used with such an approach. The resulting equations are shown below.

$$T - m_1 * g = m_1 * a \tag{7}$$

$$T - m_2 * g = -m_2 * a \tag{8}$$

The tutor must determine that (1) the variable a has the dimensions of acceleration ($\mathrm{kg} \cdot \mathrm{m/s}^2$), (2) the single variable is mapped to the acceleration of object 1 and that (3) the acceleration of the other object is replaced by an algebraic substitution using Eq. 6. The system must make similar determinations for the tensions.

The two issues that a tutoring system must address when scaffolding is removed are (1) identification of the dimensions and therefore the properties of each variable and (2) identification of the object(s) that the variables refer to. In this paper, we focus on the first issue, that of determining the dimensions of each variable. Our preliminary work in addressing the second issue, that of mapping the variables to objects is described in Ref. [11].

3. Prior Work

Checking for dimensional consistency is an important first step for a physics tutoring system as it can then focus on reasoning about dimensionally correct equations only. Existing systems, *e.g.*, ANDES[3,14,4] and PHYSICS-TUTOR[6], require that the dimensions of each variable and constant be known *a priori* either through a knowledge base of variables and constants or by having the student define them. Once these dimensions are known, it is a fairly simple step to determine if the equation is dimensionally consistent by using standard dimensional analysis.

There are many systems that use constraint propagation to ensure consistency of values of variables. Examples of such systems include VEXED[15] and OPIS[13]. Their use of constraint propagation is similar to our use except that they are propagating values and not dimensions.

There has also been some work done on adding dimension specifications to programming languages to support compile-time[12,5] and run-time[1] detection of dimension errors. These systems are similar to strongly typed programming languages where every variable has to be defined and has a type. Our system is analogous to a weakly typed language where variables are partially defined on first use and their types are inferred from the context.

4. Dimension Check Algorithm

In an earlier paper[10], we described an approach for determining the dimensions of every variable in an algebraic equation. The earlier version of the technique combined the use of a knowledge base of commonly used physics variables and constants with constraint propagation.

A constraint graph is built in which variables in the equation are instantiated as leaf nodes, and operators (*e.g.*, $+, -, *, /, =$) and functions (*e.g.*, $\cos, \sin, \tan$) are instantiated as internal nodes. The value at each node represents the set of possible dimensions for that node. The algorithm uses a knowledge base of commonly used physics variables and constants with constraint propagation. The knowledge base determines the possible dimensions of each variable. There are usually multiple possible dimensions for a variable. For example, p can properly be used to represent a momentum [$\mathrm{kg} \cdot \mathrm{m/s}$], an object distance [m] in optics, a pressure [$\mathrm{kg/m/s^2}$], or an electric dipole moment [$\mathrm{C} \cdot \mathrm{m}$]. It might also be a variable which should have been called P, representing a power [$\mathrm{kg} \cdot \mathrm{m^2/s^3}$], a probability [dimensionless], or a probability per unit distance or volume [$\mathrm{m^{-1}}$ or $\mathrm{m^{-3}}$]. Constraint propagation is used to propagate dimension information to other terms and literals to (1) infer dimension information and (2) determine dimensional consistency. The algorithm can take partial information about the dimensions of a variable and combine that with knowledge of operators and functions (which are just operators) to completely determine dimensions. In essence knowledge, even incomplete knowledge, propagates from one part of the equation to another. This permits the algorithm to reason about dimensional consistency when the variables are not explicitly defined.

This section describes how the algorithm checks for dimensional consistency in equations. The checks are performed in a series of steps as described below:

(1) set up the constraint tree for each equation.
(2) using established naming conventions determine the possible dimensions of the variables and thus the dimensions of the leaf nodes in the constraint tree.
(3) propagate values in the constraint tree and determine the consistency of each tree.
(4) enforce consistency constraints on the dimensions of all leaf nodes corresponding to a single variable, and propagate the constraints throughout the resulting overall graph.
(5) generate feedback to the user.

4.1. *Setting Up The Graph of Constraints*

The system establishes the constraint graph by setting up a binary constraint tree for each equation and represents the dimensional possibilities for each node. Each interior node in the graph represents an operator (*e.g.,* $=, +, -, *, /$) or a function (*e.g.,* $\sin, \cos$). The leaves of the tree represent each instance of a variable or a constant in the equation. If a variable occurs twice in one equation or in two separate equations, there will be two separate nodes in the constraint network labeled with that variable. This allows our system to work with multiple instantiations of a variable, and if the equation set is found to be inconsistent, to pinpoint the specific instance that is at fault. To maintain consistency for each variable within the system of equations, an **identity** constraint is added. These constraints connect all nodes that are instances of the same variable, even if they occur in different equations, restricting these nodes to have the same set of dimensions.

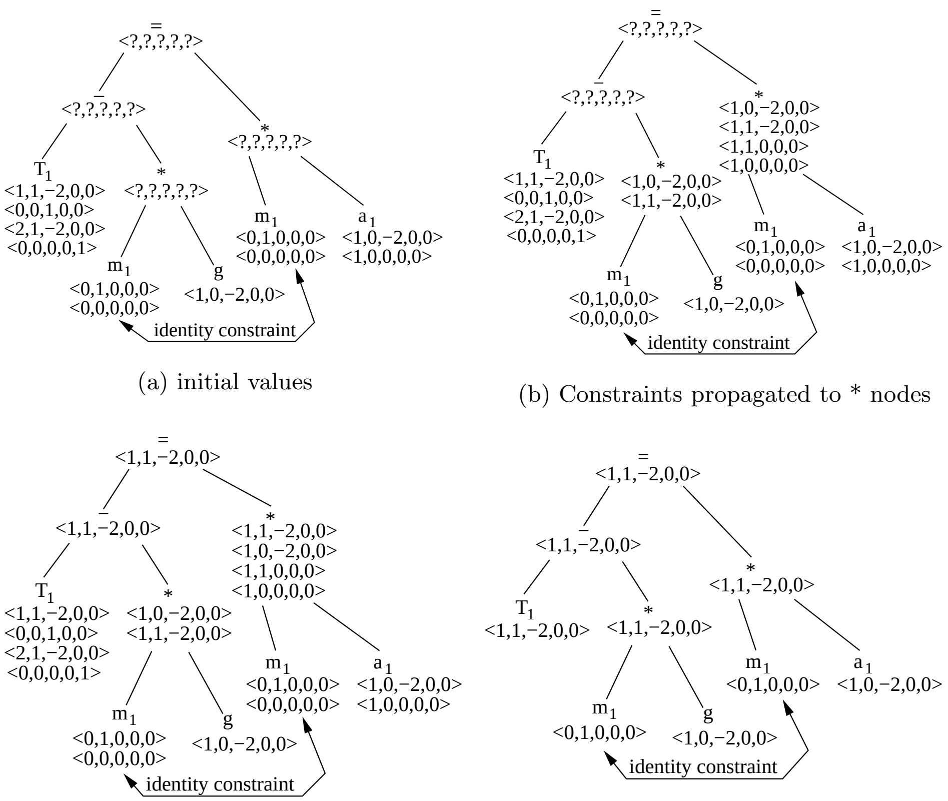
The edges of the constraint graph connect nodes that affect each other directly. The value at each node represents the set of dimensional values that are consistent with the values of the nodes connected to it. Each member of the set is a five tuple specifying the exponents of each dimension. The tuple is ordered as $< distance, mass, time, charge, temperature >$. For example, a dimensionless variable has a value $< 0, 0, 0, 0, 0 >$, a variable m for mass will have a value of $< 0, 1, 0, 0, 0 >$, and a variable a for acceleration (*i.e.*, $\mathrm{m/s^2}$) will have a value of $< 1, 0, -2, 0, 0 >$.

4.1.1. *Initializing Values*

Once the constraint trees have been constructed, the system attempts to obtain initial dimension values for the leaf nodes, *i.e.* the variables and constants. The algorithm uses a knowledge base of commonly used variables and constants along with their dimensions. It may be that there is either no mapping available or that there is more than one mapping. In these cases, the system will either leave the specific dimension blank or set the node to reflect that more than one dimension is possible. Figure 3(a) shows an example of a constraint tree after the leaf nodes have been initialized with possible values. The information in the knowledge base shows that T typically represents tension with dimensions $[\mathrm{kg \cdot m/s^2}]$, or time with dimension [s], or kinetic energy with dimension $[\mathrm{kg \cdot m^2/s^2}]$, or temperature with dimension [°K], and therefore assigns a set of these four dimension possibilities to the variable T_1.

The knowledge base supports three types of matches on variable names. Each entry into the knowledge base consists of (1) a string, (2) a set of dimensions, (3) a category, and (4) a type of match. The three types of matches are:

- *prefix match:* Any variable name whose prefix matches the string of an entry in the prefix knowledge base will have the associated set of dimensions as a possibility. For example, the variable alp will prefix match with the entry a and will have dimensions associated with acceleration as one of the valid possibilities.
- *pre-emptive match:* Any variable name whose prefix matches the string of an entry in the pre-emptive knowledge base will pre-empt any prefix matches. For example, the variable $alpha1$ will pre-emptively match with $alpha$ and



(a) initial values

(b) Constraints propagated to * nodes

(c) Constraints propagated upwards

(d) fully resolved

Fig. 3. Constraints Propagating through the Tree

have radians as one of the possible dimensions. This match will also remove acceleration (and any other prefix matches) from the list of possibilities.

- *exact match:* Any variable name that matches exactly with the string of an entry in the exact-match knowledge base will have the associated set of dimensions. This match overrides and excludes all other matches. For example, the variable G will have the dimensions of the universal gravitational constant and the match will remove all prefix or pre-emptive matches with G. The variable $G1$ however will not be an exact match.

The extended matching capability provided us with ways to specify preferences amongst the different possible matches for a variable and proved reliable and robust. Students could employ *non-standard* variables and make discovery of dimensions difficult or impossible; however, our experience has shown that this is seldom the case and that students almost always follow the conventions established in the field and used in texts and by instructors.

4.2. *Propagating Constraints*

Once the initial dimension possibilities for the leaf nodes have been established, dimension information is propagated to determine if each equation is dimensionally correct. The system uses a few simple rules to propagate and infer dimensions. The rules for reasoning about dimensions are listed below:

(1) If a node represents an additive operator $(+, -, =)$, then the dimension of this node and its children must be the same.

(2) If a node represents a trigonometric function $(\sin, \cos)$, then the node and its child are dimensionless.

(3) If a node represents a multiplication operator $(*)$, then the dimension of this node is the component-by-component sum of the dimensions of its children. Figure 3(b) shows the effect of propagating dimensions at leaf nodes to the two interior nodes representing multiplication (*i.e.*, labeled with a *).

(4) If a node represents a division operator $(/)$, then the dimension of this node is the result of subtracting the dimension (component by component) of the right child (*i.e.*, the denominator) from the dimension of the left child (*i.e.*, the numerator).

(5) The dimension possibilities of each node are repeatedly checked to assure that all possible dimensions are consistent with the possible dimensions of their parents and children. If one of the possible dimensions is not consistent it is removed. This process is repeated until a fixed point is found or until an inconsistency has been revealed.

(6) Nodes with unknown dimensions acquire them as necessary to maintain dimensional consistency.

Figure 3(c) shows the first step of propagating constraints to the nodes labeled $-$ and $=$. The constraints associated with these nodes require that the node and its children have the same dimension and therefore tightly constrain the overall tree. Figure 3(d) shows the final state after all constraints have been propagated to a fixed point. When fully resolved there is a single assignment of dimensions to variables consistent with the equation, and T_1 is identified as a tension $[\mathrm{kg} \cdot \mathrm{m}/\mathrm{s}^2]$.

Once all constraints due to mathematical operators have been satisfied, the system proceeds to impose identity constraints. If an inconsistency is found while processing mathematical operators, this can be brought to the student's attention. Identity constraints are introduced only after the other constraints have been satisfied. Delaying the application of identity constraints guarantees that local sources of inconsistency will be identified early in the evaluation. Consistency constraints are iteratively applied until the entire graph is stable or an inconsistency found.

5. Evaluation

The algorithm was evaluated on three data sets. Each data set has slightly different characteristics and can be described as follows:

- The Lafayette Data Set:

  The data set consists of approximately 350 answers to four physics problems from 88 different students in an introductory physics course for engineers and science majors at Lafayette college. The problems varied in difficulty from the example discussed in this paper (Section 2.1) to a problem involving an accelerating pulley. There were no restrictions on the types of variables that the students could use although they were discouraged from performing algebraic simplification of their equations. The students were given the questions and asked to write their answers on sheets of paper that were later transcribed into an electronic form. In addition, the students were not required to define or explain any of the variables that they chose to use.
- The ANDES 2000 Data Set:

  The ANDES system is also a tutoring system for introductory college level physics. It has a large database of problem types and is in current use at the United States Naval Academy. Logs of student answers and tutor responses have been maintained since the initial introduction of the ANDES system. We extracted the student answers from one semester (Fall 2000) and used it to evaluate our system. The key features of this data set (and of the ANDES system) are:

  - *large database of problems and problem types*: The ANDES system has a repository of approximately one hundred problems. These problems are much more diverse than the ones used to generate the Lafayette dataset.
  - *large number of equation sets:* The analyzed ANDES data contained 9,865 equation sets in 6,000 logs. These logs were created by many students, each of whom worked on many problems. Our analysis does not group equations sets by either student or problem but rather treats all 9,865 equations sets as a single corpus. The system recorded answers, including partial answers, making the number of equation sets larger than the number of logs. Many of these equation sets contain incomplete answers, *i.e.*, the student did not enter all the equations needed to solve the problem. We accepted any student equation that was correct, that is, consistent with the problem as stated, and otherwise JaSL8 imposed no constraints on the equations accepted.
  - *variables are explicitly defined before use.* The ANDES framework requires that the students define all variables before they can be used in equations and provides a graphical interface to help them with this step. Our analysis does not use this information, but the fact that the student was required to give it may have affected the inputs.
  - *use of numeric values:* The questions in ANDES are given in terms of explicit numerical quantities and require numeric answers. While students were strongly encouraged to generate complete algebraic solutions before substituting numeric values to arrive at the answer, students frequently

use numeric values in place of variables at earlier stages. In this data set, units were not required, so all numbers were treated as having unknown dimensions.

- The ANDES 2001 Data Set

  ANDES 2001 was an enhanced version of ANDES that allowed the specification of dimensions for constants. The ANDES 2001 data set is very similar to the ANDES 2000 data set, but it contains dimension information on student supplied constants.

The different properties of the three data sets allowed us to evaluate the performance with (1) unconstrained user input (Lafayette), (2) a large class of problems (Andes 2000, Andes 2001) and (3) hints from the student (Andes 2001).

5.1. *Experimental Results*

We used the data from the experiments to evaluate the technique along several directions.

- *Correctness:* Since our technique is based on a heuristic match using a knowledge base, one important question is "How often does the technique return an incorrect answer?"
- *Effectiveness:* Our initial goal was to remove the need for students to explicitly identify every variable that they used. The effectiveness is determined by the number of equation sets where the technique could uniquely determine the dimensions of all variables. We also measured in how many cases one clarification from the student would have sufficed.
- *Generality:* Our earlier work was evaluated on a small number of problems. The later set of experiments uses a much larger set of problems and hence tests a much wider set of variable types. The experiments should also determine what types of problems are problematic for the technique.
- *Robustness:* How well will the technique perform on incomplete sets of equations? The data from both the Lafayette and ANDES experiments includes incomplete equations submitted by the students. If the technique does not work well on incomplete sets of equations, then the system would not be able to provide feedback to a student who needed help to generate the remaining equations.

5.1.1. *Results from the Lafayette Data Set*

Dimensional inconsistencies occurred in approximately 15% of the students' answers and the errors were all detected by our algorithm. Our original algorithm failed to disambiguate only 5% of the submitted answers (two to three answers for each problem) A tutor would need to ask the student a question about the meaning of a variable to disambiguate. The evaluation of this dataset showed that the technique was correct, effective and robust for this small sample of answers. It was robust in that the students were not constrained in the type of answers they could write down.[a]

5.1.2. *Results from the ANDES 2000 Data Set*

An initial evaluation showed problems that were not revealed with the smaller Lafayette data set. Many equation sets had more than one set of possible dimension assignments for the set of variables. We observed that because we were using possible concepts from all of physics, including electricity and magnetism and modern physics (which were not covered in the Andes problems) the range of choices of dimensionality were often very large and the constraints were often insufficient to uniquely determine the correct choice.

This problem was addressed by (1) splitting the knowledge base into broad subfields of relevance and (2) adding a more powerful three-level matching capability, as described in section 4.1.1, to the knowledge base. The knowledge base was partitioned into major subfields, such as Newtonian mechanics, electricity and magnetism, and modern physics, and the ANDES problems were annotated to specify that they were problems in Newtonian mechanics. The results are shown in Table 1[b]. We found that in 80.5% of the equation sets the dimensionality of all variables were uniquely determined. In 3.2% of the cases we found that exactly one variable was ambiguous, so that with at most one clarifying question to the student we could uniquely determine the dimension of all variables in 83.8% of the cases.

Of the remaining 16.2% of the cases, 13.9% had more than one ambiguous variable and 2.4% were found to be dimensionally inconsistent. The variable-matching knowledge base that we used had 109 entries and contained information covering all of Newtonian mechanics, the area from which the analyzed corpus was obtained.

Table 1. Evaluation of the ANDES 2000 data.

Equation Set Property	Number in Corpus	Percent of Corpus
No ambiguous variables	8022	80.5%
One ambiguous variable	320	3.2%
Two or more ambiguous variables	1381	13.9%
Inconsistent Dimensions	237	2.4%

As described earlier, the ANDES system permits the students to use numeric

[a] Students were free to use any naming convention for variables and not asked to define them. If a definition was given it was ignored in our analysis. Standard naming conventions were employed on all student submissions even though this was not required.

[b] These results differ from those reported in Ref. [9], because in the interim we discovered several acceptable variable assignments, such as A for area and V for volume, which had been inadvertently left out of our knowledge base. Because Andes is not case sensitive, these possibilities, though little or never used by the Andes students, introduce additional ambiguities when students used v and a for velocity and acceleration.

values in place of variables. In 2000, Andes did not check that correct dimensional information was included. Thus a student might enter 9.8 instead of g for the acceleration of gravity. Consequently, constants can sometimes have unstated dimensions and the system has to treat each constant initially as having all dimension possibilities instead of as dimensionless constants. This proved to be a source of much of the ambiguity.

Thus we see that our technique was correct, reasonably effective, general and robust on a large set of problems over a large group of students. But we expected that we could do better if we could make use of dimensional information provided by the students, such as in an equation like $g = 9.8\ \mathrm{m/s}^2$. We tested our method on equation sets which contained only correct equations, as we were not prepared to examine by hand whether a failure to find the dimensionality of the variables was due to our in inadequacy or to student mistakes. Andes in 2000 did not check units, but Andes in the fall of 2001 did. Thus we turned to the 2001 Andes corpus.

5.1.3. *Evaluation of Fall 2001 Data Set*

The experimental results from the ANDES 2001 data set were very informative. The main difference between this data set and the one from fall 2000 was Andes' additional capability of analyzing dimension specifications. Students were asked to specify the dimensions of constants in their equations. For this analysis, we used the students' dimensions whenever provided. The summary results (Table 2) shows that using the dimensions provided by the student did improve the success at unambiguous determinations from 80.5% to 92.0%, and over 95% could be resolved with at most one question.

Table 2. Evaluation of the ANDES 2001 data.

Equation Set Property	Number in Corpus	Percent of Corpus
No ambiguous variables	9737	92.0%
One Ambiguous variable	319	3.0%
Two or more Ambiguous variables	325	3.1%
Inconsistent Dimensions	200	1.7%

To check that this improvement over 2000 was due to student-supplied dimensions, we ran our method on the 2001 logs in the 2000 mode, ignoring user-supplied dimensions and ignoring all variables which occur only in statements giving their numerical value, such as $L = 2.3$ m. (If we ignore user-supplied units, we can extract no information about L from such a statement, so if L appears nowhere else in the equations, it is irrelevant and indeterminate.)

We evaluated the performance of our methods based on whether (1) user specified dimensions were used or ignored and (2) the system used a knowledge base that mapped variables to all physics quantities or only to concepts from Newtonian mechanics. The results are shown in Table 3.

Table 3. Breakdown of results from the ANDES 2001 data.

KB	User Dimensions	No amb var	1 amb var	$\geq$ 2 amb var	Inconsistent
full	no	6998 (70.9%)	189 (1.9%)	2629 (26.6%)	52 (0.5%)
newton	no	7958 (80.7%)	213 (2.2%)	1612 (16.2%)	83 (0.8%)
full	yes	9663 (91.3%)	339 (3.2%)	426 (4.0%)	152 (1.4%)
newton	yes	9737 (92.0%)	319 (3.0%)	325 (3.1%)	200 (1.9%)

Without user specified dimensions, the technique performed nearly identically on the 2001 data set and on the 2000 set. Not surprisingly, we see that when students are required to include dimensions or when the class of problems are restricted to a specific domain of physics (*i.e.*, Newtonian mechanics) our methods give unique dimensional identifications more often. Further we see that user supplied dimensions are much more effective at disambiguating than is restricting the knowledge base to one domain of physics. From the third column in table 3, giving the fractions which our methods unambiguously resolve, we see that only 70.9% of the corpus is unambiguous when the knowledge base for all of physics is used and no dimension information is provided by the student. When the knowledge base is restricted to only consider Newtonian Mechanics, and user dimensions are accepted, this increases by 21.1% to 92.0%. Almost all this increase (*i.e.*, 20.4% out of a total of 21.1%) can be obtained using only user supplied dimensions. This indicates that the dimensionality of variables can be inferred without knowing the domain of physics to which the problem belongs, and that user supplied dimensions are a key component of an ITS working with algebraic physics equations.

It is also instructive to examine the 200 equation sets (2% of the corpus) which Andes accepted as correct but which our heuristic methods declared inconsistent. We examined these by hand and found they can be partitioned into four groups.

58 inconsistencies were caused by equations that Andes accepted though instructors would have marked as wrong. Two such examples are $v = 0\ \mathrm{m/s}^2$ and $4.5\ \mathrm{J} = 0.5\ \mathrm{kg} * 3\ \mathrm{m/s}^2$. Our system rejects the first equation since v cannot represent a concept with dimensions m/s^2 (acceleration). Andes ignores any dimension information when the numeric value is zero, therefore ignores the dimension information following the 0, and accepts this equation. Our system rejects the second equation since energy(*i.e.*, J) does not have the dimensions of $\mathrm{kg} \cdot \mathrm{m/s}^2$. Andes interprets the input as $(0.5\ \mathrm{kg}) \times (3\ \mathrm{m/s})^2$ (notice the the location of the additional parentheses) and accepts this equation as well.

60 inconsistencies were caused by equation sets with ambiguous inputs for which our parser found a correct, but unintended, parse. Our parser is a standard deterministic parser and when an input is ambiguous (*i.e.*, has more than one parse) it only considers one of the possible parses. Andes found a different, dimensionally consistent, parse than our system. Had our parser found the parse Andes used the equations sets would have been determined as consistent. This indicates an inadequacy in our parser, but not in our method of determining the meaning of variables.

68 inconsistent sets are attributable to decisions made by our method regarding students' choice of variable names for angles. Many of these may have been caused by the awkward handling of Greek in the Andes interface.

The remaining 14 inconsistencies of the 9727 sets in the corpus were caused by inexplicable choices, such as the student who used j for distance and g for speed.

From our analysis a tutor based on our methods would resolve a student's intent on 99% of the submissions and require a dialogue on less that 1% of the submissions. Of course such dialogs would discourage students from using unconventional name choices, which is, overall, a good thing.

6. Discussion and Future Work

We see that dimensional consistency allows a tutor to recognize the types of student variables in a large fraction of equation sets. A tutor, however, needs to provide useful feedback when presented with erroneous responses, and in particular, to point to the source of dimensional inconsistency. As the student answers are **sets of equations**, this is not always straightforward. For example, consider the student solution

$$L = I\omega; \qquad I = \frac{1}{6}mL^2.$$

The first equation has only one consistent dimensional interpretation, with L an angular momentum [kg·m^2/s]. If this conclusion is inserted into the second equation, it will be declared dimensionally inconsistent, even though that equation is correct if the L represents instead the length of the side of the rotating square. We would like a tutor to point to the inconsistent use of L rather than to say either equation is inconsistent.

The solution is to propagate information in two steps, first only within each equation until the system is quiescent and then secondly between equations. This allows the system to isolate errors within equations before errors of inconsistencies between equations. There are similar problems with inconsistencies of usage between different terms of a single equation. We intend to evaluate the effectiveness of these heuristics and to develop additional ones as necessary to ensure that users are pointed in the right direction when they make mistakes.

7. Conclusion

This paper has shown how domain knowledge combined with heuristic constraint propagation can be used to determine the context and implicit information contained in student answers, specifically the dimensions of variables in systems of equations. This approach has been tested and evaluated on answers from students at two institutions. The results show that the technique uniquely determined the dimensions of all the variables in 91–92% of the sets of equations. By asking for dimension information about one variable, an additional 3% of the sets can be determined.

Scaffolding is a technique that is useful and helpful to beginning students. After some experience, students would benefit from having the scaffolding removed. The experiments validate the hypothesis that our technique allows us to remove the scaffolding from a physics tutoring system and still determine the dimensions of the variables used in the equations. A tutorial system that accepts equations from students without requiring them to explicitly define each variable used can make very effective use of physical dimensionality constraints and standard variable naming conventions. These can help identify the physical quantities corresponding to each student-chosen variable name. The heuristics rarely lead to mistaken assumptions, and in most cases completely determine the dimensionality subclass to which each variable belongs. This encourages us to continue to the next step: given a problem statement and thereby a canonical set of variables relevant to the problem, to associate each student variable with the physics concept it represents first, and then to try to determine the corresponding member of the canonical set.

8. Acknowledgments

We are grateful to Kurt VanLehn and the Andes group for making the ANDES logs available to us, and to Anders Weinstein for answering questions about the ANDES implementation. Jim Appenzeller and Dave Santin provided invaluable help in the implementation and evaluation of the system.

References

1. Cunis, R. A package for handling units of measure in lisp. *ACM Lisp Pointers 5*, 2 (1992).
2. Fretz, E. B., Wu, H.-K., Zhang, B., Krajcik, J. S., and Soloway, E. A further investigation of scaffolding design and use in a dynamic modeling tool. In *Proceedings of the American Education Research Association Conference* (2002).
3. Gertner, A., and VanLehn, K. Andes: A coached problem solving environment for physics. In *Proceedings, 5th International Conference, ITS 2000* (Montreal Canada, June 2000), Springer.
4. Gertner, A. S. Providing feedback to equation entries in an intelligent tutoring system for physics. In *Proceedings of the 4th International Conference on Intelligent Tutoring Systems* (1998).
5. Hilfinger, P. N. An ada package for dimensional analysis. *ACM Transactions on Programming Languages and Systems 10*, 2 (1988), 189–203.
6. Liew, C., Shapiro, J. A., and Smith, D. Reasoning about algebraic answers in physics. In *Proceedings of Twelfth International Florida AI Research SocietyConference* (1999), pp. 167–171.
7. Liew, C., Shapiro, J. A., and Smith, D. What is wrong with this equation? error detection and feedback with physics equations. In *Proceedings of Thirteenth International Florida AI Research SocietyConference* (2000).
8. Liew, C., Shapiro, J. A., and Smith, D. Identification of variables in model tracing tutors. In *Proceedings of 11th International Conference on AI in Education 2003* (July 2003), IOS Press, pp. 464 – 466.

9. Liew, C., Shapiro, J. A., and Smith, D. Inferring the context for evaluating physics algebraic equations when the scaffolding is removed. In *Proceedings of Seventeenth International Florida AI Research Society Conference* (2004).
10. Liew, C., and Smith, D. Checking for dimensional correctness in physics equations. In *Proceedings of Fifteenth International Florida AI Research Society Conference* (2002).
11. Liew, C. W., and Smith, D. E. Reasoning about systems of physics equations. In *Intelligent Tutoring Systems: ITS 2002* (2002), Cerri, Gouarderes, and Paraguacu, Eds., Lecture Notes in Computer Science, Springer-Verlag. (LNCS 2363).
12. Novak, G. S. Conversion of units of measurement. *IEEE Transactions on Software Engineering 21*, 8 (August 1995), 651–661.
13. Ow, P. S., and Smith, S. F. Towards an opportunistic scheduling system. In *Proceedings of 19th Hawaii International Conference on System Sciences* (1986).
14. Schulze, K., Shelby, R., Treacy, D., Wintersgill, M., VanLehn, K., and Gertner, A. Andes: An intelligent tutor for classical physics. *The Journal of Electronic Publishing 6*, 1 (2000).
15. Steinberg, L. Design as refinement plus constraint propagation: The VEXED experience. In *Proceedings of AAAI-87* (1987).
16. VanLehn, K., Freedman, R., Jordan, P., C. Murray, C., Osan, R., Ringenberg, M., Ros, C., Schulze, K., Shelby, R., Treacy, D., Weinstein, A., and Wintersgill, M. Fading and deepening: The next steps for andes and other model-tracing tutors. In *Proceedings of Fifth International Conference on Intelligent Tutoring Systems (ITS)* (2000), G. Gauthier, C. Frasson, and K. VanLehn, Eds.