Representation and Reasoning for Deeper Natural Language Understanding in a Physics Tutoring System*

Maxim Makatchev, Kurt VanLehn, Pamela W. Jordan and **Umarani Pappuswamy**
Learning Research and Development Center
University of Pittsburgh
Pittsburgh, PA 15260
{maxim,vanlehn,pjordan,umarani}@pitt.edu

Abstract

Students' natural language (NL) explanations in the domain of qualitative mechanics lie in-between unrestricted NL and the constrained NL of "proper" domain statements. Analyzing such input and providing appropriate tutorial feedback requires extracting information relevant to the physics domain and diagnosing this information for possible errors and gaps in reasoning. In this paper we will describe two approaches to solving the diagnosis problem: weighted abductive reasoning and assumption-based truth maintenance system (ATMS). We also outline the features of knowledge representation (KR) designed to capture relevant semantics and to facilitate computational feasibility.

Introduction

One of the hypotheses behind the creation of NL-based intelligent tutoring systems is that allowing students to provide unrestricted input to a system would trigger meta-cognitive processes that support learning (i.e. self-explaining (Chi *et al.* 1994)) and help expose misconceptions (Slotta, Chi, & Joram 1995). WHY2-ATLAS is such a tutoring system. It is designed to elicit explanations in the domain of qualitative physics (VanLehn *et al.* 2002).

A typical problem and a student explanation are shown in Figure 1. An example of an incorrect explanation is shown in Figure 2. As can be seen from the examples, a student's explanation about a formal domain such as qualitative physics may involve a number of phenomena: algebraic formulas, NL renderings of formulas, various degrees of formality, and conveying the logical structure of an argument (Makatchev *et al.* 2005). Tutoring goals involve eliciting correct statements of the appropriate degree of formality and their justifications to address possible gaps and errors in the explanation. To achieve these goals the NL understanding is required to answer the following questions:

- Does the student explanation contain errors? If yes, what are the likely buggy assumptions that have led the student to these errors?

*This research was supported by ONR Grant No. N00014-00-1-0600 and by NSF Grant No. 9720359. We would like to thank all members of the WHY2-ATLAS team.

> Problem: A heavy clay ball and a light clay ball are released in a vacuum from the same height at the same time. Which reaches the ground first? Explain.
> *Explanation:* Both balls will hit at the same time. The only force acting on them is gravity because nothing touches them. The net force, then, is equal to the gravitational force. They have the same acceleration, g, because gravitational force=mass*g and f=ma, despite having different masses and net forces. If they have the same acceleration and same initial velocity of 0, they have the same final velocity because acceleration=(final-initial velocity)/elapsed time. If they have the same acceleration, final, and initial velocities, they have the same average velocity. They have the same displacement because average velocity=displacement/time. The balls will travel together until the reach the ground.

Figure 1: The statement of the problem and a verbatim student explanation.

- What required statements have not been covered by the student? Does the explanation contain statements that are logically close to the required statements?

These requirements imply that a logical structure needs to be imposed on the space of possible domain statements. Considering such a structure to be a model of the student's reasoning about the domain, the two requirements correspond to a solution of a model-based diagnosis problem (Forbus & de Kleer 1993).

How does one build such a model? A desire to make the process scalable and feasible necessitates an automated procedure. The difficulty is that this automated reasoner would have to deal with the NL phenomena that are relevant for our application. In turn, this means that the KR would have to be able to express these phenomena, that as we mentioned above include: algebraic formulas, NL renderings of formulas, various degrees of formality, logical structure. The reasoner would have to account for common reasoning fallacies, have flexible consistency constraints and perform within the tight requirements of a real-time dialogue application.

In the next section we will describe a KR that attempts to satisfy the expressivity and efficiency requirements. In the section on reasoning we will present two approaches for

The heavy clay ball and light clay ball will never reach the ground in a vacuum. A vacuum has no air or gravity so neither ball will ever touch the ground.

Figure 2: Another verbatim student explanation.

building models of student's reasoning: first, an on-the-fly approach based on weighted abductive theorem proving, and second, an approach based on a precomputed ATMS.

Knowledge representation

We have chosen an order-sorted first-order predicate logic (FOPL) as a base KR for our domain since it is expressive enough to reflect the hierarchy of concepts from the qualitative mechanics ontology (Ploetzner & VanLehn 1997) and has a straightforward proof theory (Walther 1987). Following the representation used in the abductive reasoner Tacitus-lite+ (Thomason, Hobbs, & Moore 1996), our KR is function-free, does not have quantifiers, Skolem constants or explicit negation. Instead all variables in facts or goals are assumed to be existentially quantified, and all variables in rules are either universally quantified (if they appear in premises) or existentially quantified (if they appear in conclusions only).

The first argument of a predicate defines the arity and constraints on the sort of each argument. Possible values for the first argument are: one-body vectors, i.e. `position`, `displacement`, `velocity`, `acceleration`; a two-body vector `force`; `distance`; `state`; etc. Since the first argument defines the syntax of a predicate, in the rest of this paper we will call the predicate and the respective atom by the name of its first argument (and will omit a predicate symbol preceding the list of arguments). The second argument of a predicate is normally a unique atom identifier used for cross-referencing. Most of the predicates end with two time arguments specifying time points and open intervals.

Although our KR has no explicit negation, some types of negative statements are represented by using (a) complimentary sorts, for example `constant` and `nonconstant`; (b) a relative position predicate, namely `(rel-position rp1 nonequal ...)`; or (c) a comparison of quantities predicate `(compare c1 ?var1 ?var2 ... ?diff)`, where the `?diff` argument is of sort `nonzero` to denote inequality.

Below we describe the representations for each of the relevant NL phenomena mentioned in the introduction.

Algebraic formulas and NL rendering of formulas

Most of the NL expressions that the system must process do not contain direct references to formulas (e.g. Figure 2). However, those utterances that do use algebraic expressions are usually highly informative for the tutor and must be identified. Examples of such expressions include "acceleration is final velocity minus initial velocity over elapsed time", "9.8 m/s^2", "a = 9.8 m/s^2", and "the equation <net force = m * a>". Instead of parsing arbitrary algebraic expressions, an equation identifier attempts shallow parsing of equation candidates and maps them into a finite set of anticipated equation labels (Makatchev *et al.* 2005), producing a representation of the form `(math-form mf1` <equation label>`)`.

In addition to the anticipated equations, there are representations for particular relationships between two variables:

- dependency: `(dependency d1 ?var1 ?var2 ?relation ?t1 ?t2)`, where `d1` is an atom identifier, `?var1` and `?var2` are arbitrary variables; `?relation` argument can be of sorts `dependent` and `independent`, where `dependent` sort has values `proportional` and `inversely-proportional`; and `?t1` and `?t2` are time interval variables.
- comparison: `(compare c1 ?var1 ?var2 ?order ?ratio ?diff)`, where `c1` is an atom identifier, `?var1` and `?var2` are arbitrary variables, `?order` specifies the order of the variables in the binary relation, the `?ratio` argument has possible values relevant to ratios (`one`, `greater-than-one`, `two`), and `?diff` is an argument used for difference comparisons.

Various degrees of formality

NL understanding needs to distinguish formal versus informal physics expressions so that the tutoring system can coach on proper use of terminology. Many qualitative mechanics phenomena may be described in a relatively informal language, for example "speed up" instead of "accelerate" and "push" instead of "apply a force." The relevant informal expressions fall into the following categories:

- relative position: "keys are behind (in front of, above, under, close, far from, etc.) man"
- motion: "move slower," "slow down," "moves along a straight line"
- dependency: "horizontal speed will not depend on the force"
- direction: "the force is downward"
- interaction: "the man pushes the pumpkin," "the gravity pulls the ball"

Each of these categories (except for the last one) have a dedicated representation:

- `(rel-position rp1 ?rel-location ?body1 ?body2 ?t1 ?t2)`, where the `?rel-location` argument assumes values of `in-front-of`, `behind`, `above`, `below`, `at`, `closer`, `father`, `close`, `far`, etc.
- `(motion m1 ?body ?component ?traj-shape ?traj-speed ?d-mag ... ?t1 ?t2)`, where `?component` specifies the `vertical` or `horizontal` component of a vector (or `none`); `?traj-shape` argument can be of sort `linear` or assume values of sort `curvilinear`, namely `parabolic`, `circle`, `ellipse`; `?traj-speed` specifies the speed as either `uniform` or `nonuniform`; `?d-mag` (the derivative of a magnitude) specifies whether the commonsense rate of the motion is `increasing` ("moves faster") or `decreasing` ("slows down").
- `(dependency ... ?relation ...)`, where `?relation` argument can be of sorts `dependent` or `independent`.

Step	Statement	Justification
1	Both balls are near earth	Unless the problem says otherwise, assume objects are near earth
2	Both balls have a gravitational force on them due to the earth	If an object is near earth, it has a gravitational force on it due to the earth
...	...	...
6	**Gravitational force is w = m*g for each ball**	**The force of gravity on an object has a magnitude of its mass times g, where g is the gravitational acceleration**
...	...	...
18	**The balls have the same initial vertical position**	given
19	The balls have the same vertical position at all times	[Displacement = difference in position], so if the initial positions of two objects are the same and their displacements are the same, then so is their final position
20	**The balls reach the ground at the same time**	

Figure 3: A fragment of an informal proof for the Clay Balls problem. The required points are in bold.

- Qualitative direction constants `up`, `down`, `left`, `right` can be put in correspondence with otherwise quantitative direction variables in vector predicates via reasoner rules that convert formal directions into informal ones.
- While representing push and pull expressions via a dedicated predicate seems straightforward, we are still assessing the utility of distinguishing "man pushes the pumpkin" and "man applies a force on the pumpkin" for our tutoring application and currently represent both expressions as a nonzero force applied by the man to the pumpkin.

These representations are generated by a combination of symbolic and statistical NLP (Jordan, Makatchev, & VanLehn 2004).

Logical structure

One of the tutoring objectives of WHY2-ATLAS is to encourage students to provide argumentative support for their conclusions. This requires recognizing and representing the justification-conclusion clauses in student explanations. Recognizing such clauses is a challenging NLP problem due to the issue of quantifier and causality scoping. It is also difficult to achieve a compromise between two competing requirements for a suitable representation. First, the KR should be flexible enough to account for a variable number of justifications. Second, reasoning with the KR should be computationally feasible.

The second requirement eliminates the representation of relation between N premises and 1 conclusion via N atoms (`relation c1 premise`<i> `conclusion1`), $i = 1, \ldots, N$. Indeed, cross-referencing between atoms via shared variables is a necessary but expensive feature. The best known algorithm for matching cross-referenced atoms has time complexity $O(2^n n^3)$, where n is the number of input atoms (Shearer, Bunke, & Venkatesh 2001).

Another possibility is using variable arity predicates of the form (`cause c1 cause1 cause2 cause3 ...causeN effect1`). This would require customizing our reasoners to allow soft matching between variable arity predicates, which leads to an increase in search space, similar to the case with N "short" atoms described in the previous paragraph.

Another nuance of justification-conclusion representation is *asserting* versus *non-asserting* conditions. Consider the following examples, "if there was air resistance, the larger ball would fall faster," and "since there is no air resistance, the balls fall at the same speed." Clearly there is a difference in the speaker's belief about whether the condition actually holds or not. The logical structures of these sentences can be represented as $A \rightarrow B$ and $C \wedge (C \rightarrow D)$ respectively. Due to the combined difficulties of NLP, KR and reasoning, currently we largely ignore the justification-conclusion clues found in the student's input and represent both cases as conjunctions of atoms: $A \wedge B$ and $C \wedge D$ respectively.

Rule base

The rules of qualitative mechanics, likely buggy student inferences and conversions between formal and informal statements are represented as *extended Horn clauses*, namely the head of the rule is an atom or a conjunction of multiple atoms.

Two different approaches to diagnosing student input have been implemented during different stages of the project: abductive reasoning and ATMS.

Abductive diagnosis

In the first prototype of the WHY2-ATLAS tutoring system the task of diagnosing student input was performed on-the-fly by the Tacitus-lite+ abductive theorem prover (Makatchev, Jordan, & VanLehn 2004). The advantages of computing the diagnosis on-the-fly as an abductively generated proof that minimizes the total cost of assumptions are as follows:

- An abductive proof of an observed student input expression can be generated even if the rule base is incomplete and the assumptions are not present as givens (everything is assumable).
- The real-time process of proof generation can be halted at any time and the cheapest proof generated so far can be taken as the best current approximation of the diagnosis.

Despite these advantages, this approach had two major drawbacks:

- The inherent unsoundness of abductive inference requires a costly procedure to ensure that a proof is consistent (no two mutually exclusive subgoals are produced).
- Lack of an estimate on the time necessary to find a proof below an acceptable cost threshold.

As a result, the average time required by the abductive reasoner for processing a student's explanation was about 170 seconds, a painful delay for a real-time dialogue application (Makatchev, Jordan, & VanLehn 2004).

ATMS-based diagnosis

The desire to reduce the amount of on-the-fly computation by computing the proofs offline led us to adopt a powerful tool from model-based diagnosis, an ATMS (Forbus & de Kleer 1993). ATMS's have been used for tasks that are closer to the front end of the NLP pipeline such as for parsers that perform reference resolution (e.g. (Nishida *et al.* 1988)), but there are few systems that utilize an ATMS at deeper levels of NL understanding (e.g. (Zernik & Brown 1988)). Our implementation of the ATMS includes a subset of the deductive closure of givens and correct and buggy assumptions, so that each derived proposition (a *node*) carries a list of assumptions (*labels*) that were made while deriving it. The subset of the deductive closure can be computed and checked for completeness off-line, thus improving on-the-fly efficiency and facilitating knowledge engineering.

Completeness and correctness analyzer

All domain statements that are potentially required to be recognized in the student's explanation or utterances are divided into principles and facts. The principles are versions of general physics (and "buggy physics") principles that are either of a vector form (for example, "F=ma") or of a qualitative form (for example,"if total force is zero then acceleration is zero"), while facts correspond to concrete instantiations of the principles (for example, "since there is no horizontal force on the ball its horizontal acceleration is zero") or to derived conclusions (for example, "the horizontal acceleration of the ball is zero"). The nodes of the ATMS correspond to facts derived from the problem givens, so the ATMS can only provide an analysis of utterances about facts, but not principles. Representations of input utterances are first matched against a set of manually created representations of relevant general principles, important facts and misconceptions. If there is a match, the input utterance is said to contain an *explicit correct statement* or an *explicit error.* Otherwise the statement is matched against the ATMS facts to determine whether it refers to an *implicit correct statement* or an *implicit misconception.*

If some nodes of the ATMS match the representation of the input utterance, they are analyzed for correctness by checking whether their labels contain only *environments* (consistent sets of assumptions that are sufficient to infer a node) with buggy assumptions. In the case when there are no environments that are free of buggy assumptions in the label of the node, the node can only be derived using one of the buggy assumptions and therefore represents an implicit misconception. These buggy assumptions are then reported to the tutoring-system strategist for possible remediation. Additionally, a neighborhood of radius N (in terms of a graph distance) of the matched nodes can be analyzed for whether it contains any of the required facts to get an estimate of the proximity of a student's utterance to a required point.[1] A few examples of the utterance analysis are presented in the next section.

Examples

In this section we will give example analyses of four types of utterances: an explicit correct statement (a fact or a principle), an explicit error, an implicit correct statement, and an implicit misconception.

Explicit correct statement Consider the following student utterance "Since average velocity is vf+vi/2, the balls will hit the ground at the same time." The sentence gets the following FOPL representation by the system (as before, we omit the representation of variable sorts for brevity):

```
(math-form mf1 avgv-is-vf-plus-vi-over-2)
(become b1 contact big-ball ?body2 detached
    attached ?t1 ?t2)
(become b2 contact small-ball ?body3
    detached attached ?t3 ?t4)
```

The first atom of the representation is generated by the equation identifier, described in (Makatchev *et al.* 2005). The following two atoms represent that the small ball and the big ball come in contact with some body at some time. Ideally, variables `?body2` and `?body3` should both be bound to the constant `earth`, and time points should be equal between these two predicates, to show that the events of changing the contact state from detached to attached happen at the same time. However, due to imperfection of NL to FOPL conversion, a typical representation has underconstrained variables.

At first, the analyzer is called to detect a direct match of the utterance with stored principles, facts and misconceptions. As it happens, the representation above contains two atoms that are close to the stored representation

```
(become b3 contact big-ball earth detached
    attached ?t1 ?t2)
(become b4 contact small-ball earth detached
    attached ?t1 ?t2)
```

of the fact which is a correct answer to the problem, namely "the balls hit the ground at the same time." The matcher would consider such partial match a success. Although the first atom of the input representation above is a part of the stored representation

```
(math-form mf2 avgv-is-vf-plus-vi-over-2)
(acceleration a0 ?body0 ... constant ...
    ?t5 ?t6)
```

[1] The value of N depends upon the tutoring strategy selected.

of the principle "If acceleration is constant, then average velocity = (vf+vi)/2," the matcher would consider the overlap insufficient to call this a match.

After comparing the input representation with representations of all relevant statements the analyzer returns the result: The utterance matches the fact "the balls hit the ground at the same time."

Explicit error The student utterance "the big ball would hit the ground before the small ball," being an anticipated error, has a matching hand-coded representation

```
(become b5 contact big-ball earth detached
    attached ?t1 ?t2)
(become b6 contact small-ball earth detached
    attached ?t3 ?t4)
(before ?t2 ?t4)
```

as a buggy statement that would be matched via a process similar to the one described in the previous section.

Implicit correct statement A more sophisticated processing has to be done when the student utterance does not directly match any of the stored representations. This can happen due to two reasons: first, the utterance is not a valid or relevant statement in the context of the problem, second the statement is relevant in the context of the problem but it is not considered important for tutoring goals and therefore it does not have a corresponding hand-coded stored representation. Deploying an ATMS aims at validating the second type of statements and alleviating the respective knowledge-engineering bottleneck. Ideally, ATMS should have all facts inferable from the givens (deductive closure). In reality, we settle for an incomplete ATMS to allow for efficient real-time matching of input representations with its nodes.

Consider the following utterance: "The large ball will have a greater force due to gravity." Although this is a correct fact in the context of the problem it is not deemed necessary for the solution shown in Figure 3. Consequently, there is no stored representation of this fact in the system. However, being unable to evaluate its correctness, the system would not only miss the opportunity to provide a positive feedback to the student, but arguably more importantly, if the statement turns out to be incorrect (as in the next section), the system would miss the student error.

This input statement is, however, inferable from the givens of the problem, and therefore is part of the deductive closure of the givens. If the ATMS is complete enough, the statement is represented as a set of nodes of the ATMS. In this case, the analyzer, after returning NIL as a result of direct match of the input utterance, would be called to match the input with the ATMS and would find a set of matching nodes in ATMS. The nodes of ATMS, however, contain not only facts inferable from the givens via correct rules, but also facts inferable via buggy rules and buggy assumptions. Therefore having a match with the ATMS does not guarantee correctness of the statement right away. Correctness can be easily checked by examining the environments (sets of assumptions) for which the matching nodes hold true. The statement "the large ball will have a greater force due to gravity" may hold in multiple environments, some of which may include buggy assumptions. However, since it can be inferred from the givens only, it also must hold in the environment that does not contain any buggy assumptions. Existence of such bug-free environment proves correctness of the statement.

Once the correctness of the utterance is confirmed, the chain of inferences recovered from the ATMS can be used to lead the student back on track of the required solution to the problem.

Implicit misconception Similar processing occurs when the statement is an implicit misconception. Consider the utterance "The balls will have the same force due to gravity." As with the example of the implicit correct statement above, this is not an anticipated statement and thus doesn't not have a hand-coded stored representation. However it indicates that student has a misconception that, ideally, should be remediated by the tutoring system.

Again, since the statement does not produce any direct matches, it is matched against the ATMS. The ATMS includes a number of buggy assumptions and their consequences. One of the buggy assumptions is that the student believes that the force of gravity is the same for all objects. While the representation of this assumption is close to the student utterance, it does not result in a direct match. Instead, the student utterance matches a statement inferred using this buggy assumption.

There are no bug-free environments for which the statement "Gravitational force is the same for both balls" holds, which suggests that the statement is wrong (this would be guaranteed, had our ATMS contained the complete deductive closure of the givens). In fact, one of the environments for which the statement holds contains a buggy assumption "The force due to gravity is the same for all objects". Given the limited number of anticipated buggy assumptions (which can nevertheless be used to infer a large number of erroneous facts) each of them can have a hand-coded remediation dialogue. Thus leveraging on the features of ATMS helps to reduce not only knowledge-engineering effort required for building formal representations of potentially relevant statements, but also the effort required for hand-coding knowledge-intensive structures in other parts of the system.

Preliminary evaluation

The completeness and correctness analyzer has been deployed in an evaluation of the full WHY2-ATLAS tutoring system. This evaluation, however, did not use essential features of the ATMS. Instead, we evaluated the performance of the direct matching procedure. Figure 4 shows results of classifying 62 student utterances for one physics problem with respect to 46 stored statement representations using only direct matching. To generate these results, the data is manually divided into 7 groups based on the quality of conversion of NL to FOPL, such that group 7 consists only of perfectly formalized entries, and for $1 \leq n \leq 6$ group n includes entries of group $n+1$ and additionally entries of somewhat lesser representation quality, so that group 1 includes all the entries of the data set. The flexibility of

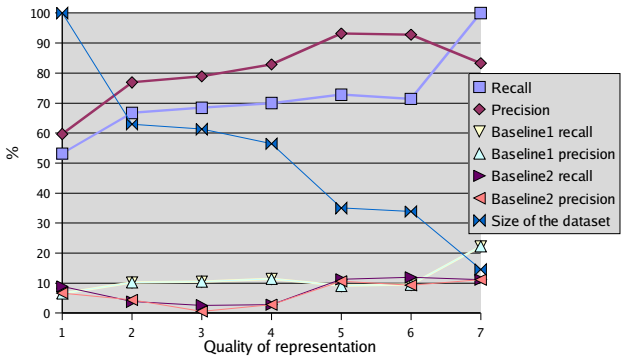

Figure 4: Average recall and precision of utterance classification. The size of a group of entries is shown relative to the size of the overall data set. Average processing time is 0.011 seconds per entry on a 1.8 GHz Pentium 4 machine with 2Gb of RAM.

the matching algorithm allows classification even of utterances that have mediocre representations, resulting in 70% average recall and 82.9% average precision for 56.5% of all entries (group 4). However, large numbers of inadequately represented utterances (at least 38.7% of entries that did not make into group 4) result in 53.2% average recall and 59.7% average precision for the whole data set (group 1). These results are still significantly better compared to the two baseline classifiers best of which peaks at 22.2% average recall and precision. The first baseline classifier always assigns the single label that is dominant in the training set (average number of labels per entry of the training set is 1.36). The second baseline classifier independently and randomly picks labels according to their distributions in the training set. The most frequent label in the training set corresponds to the answer to the problem. Since in the test set the answer always appears as a separate utterance (sentence), recall and precision rates of the first baseline classifier are the same.

Although the current evaluation did not involve matching against the ATMS, we did evaluate the time required for such a match to get a rough comparison with the earlier on-the-fly approach. Matching a 12 atom input representation against a 128 node ATMS that covers 55% of relevant problem facts takes around 30 seconds, which is a considerable improvement.

Conclusions and Future Work

Analyzing NL with a formal logical framework, while providing a number of benefits, is a task with long-standing challenges. In this paper we presented how a choice of KR and reasoning procedures can help solve the problems of expressiveness, improve performance, and reduce the knowledge-engineering effort. We described two reasoning frameworks that were implemented in the working system prototypes. Replacing an on-the-fly abductive reasoning procedure with a precomputed off-line ATMS has led to significant reduction of processing time. A more detailed evaluation of the ATMS for NL understanding will be conducted in a future study.

References

Chi, M. T. H.; de Leeuw, N.; Chiu, M.-H.; and LaVancher, C. 1994. Eliciting self-explanations improves understanding. *Cognitive Science* 18:439–477.

Forbus, K. D., and de Kleer, J. 1993. *Building Problem Solvers*. Cambridge, Massachusetts; London, England: MIT Press.

Jordan, P. W.; Makatchev, M.; and VanLehn, K. 2004. Combining competing language understanding approaches in an intelligent tutoring system. In *Proceedings of Intelligent Tutoring Systems Conference*, volume 3220 of *LNCS*, 346–357. Maceió, Alagoas, Brazil: Springer.

Makatchev, M.; Hall, B. S.; Jordan, P. W.; Pappuswamy, U.; and VanLehn, K. 2005. Mixed language processing in the Why2-Atlas tutoring system. In *Proceedings of the Workshop on Mixed Language Explanations in Learning Environments, AIED2005*, 35–42.

Makatchev, M.; Jordan, P. W.; and VanLehn, K. 2004. Abductive theorem proving for analyzing student explanations to guide feedback in intelligent tutoring systems. *Journal of Automated Reasoning, Special issue on Automated Reasoning and Theorem Proving in Education* 32:187–226.

Nishida, T.; Liu, X.; Doshita, S.; and Yamada, A. 1988. Maintaining consistency and plausibility in integrated natural language understanding. In *Proceedings of COLING-88*, volume 2, 482–487.

Ploetzner, R., and VanLehn, K. 1997. The acquisition of qualitative physics knowledge during textbook-based physics training. *Cognition and Instruction* 15(2):169–205.

Shearer, K.; Bunke, H.; and Venkatesh, S. 2001. Video indexing and similarity retrieval by largest common subgraph detection using decision trees. *Pattern Recognition* 34(5):1075–1091.

Slotta, J. D.; Chi, M. T.; and Joram, E. 1995. Assessing students' misclassifications of physics concepts: An ontological basis for conceptual change. *Cognition and Instruction* 13(3):373–400.

Thomason, R. H.; Hobbs, J.; and Moore, J. D. 1996. Communicative goals. In Jokinen, K.; Maybury, M.; Zock, M.; and Zukerman, I., eds., *Proceedings of the ECAI 96 Workshop Gaps and Bridges: New Directions in Planning and Natural Language Generation*.

VanLehn, K.; Jordan, P.; Rosé, C.; Bhembe, D.; Böttner, M.; Gaydos, A.; Makatchev, M.; Pappuswamy, U.; Ringenberg, M.; Roque, A.; Siler, S.; and Srivastava, R. 2002. The architecture of Why2-Atlas: A coach for qualitative physics essay writing. In *Proceedings of Intelligent Tutoring Systems Conference*, volume 2363 of *LNCS*, 158–167. Springer.

Walther, C. 1987. *A many-sorted calculus based on resolution and paramodulation*. Los Altos, California: Morgan Kaufmann.

Zernik, U., and Brown, A. 1988. Default reasoning in natural language processing. In *Proceedings of COLING-88*, volume 2, 801–805.