Authoring Expert Knowledge Bases for Intelligent Tutors through Crowdsourcing

Mark Floryan[1], Beverly Park Woolf[1]

[1] Department of Computer Science, University of Massachusetts, Amherst
140 Governors Dr. Amherst, MA USA
{mfloryan, bev} @ cs.umass.edu

Abstract. We present a methodology for constructing domain-level expert knowledge bases automatically through crowdsourcing. This approach involves collecting and analyzing the work of numerous students within an intelligent tutor and using an intelligent algorithm to coalesce data from that student work to construct the domain model. This evolving expert knowledge base (EEKB) is then utilized to provide expert coaching and tutoring with future students. We compared the knowledge created in human crafted expert knowledge bases (HEKB) with knowledge resulting from our knowledge acquisition algorithm to judge quality. We find that our EEKB models have qualities that rival that of the human crafted knowledge bases and can be generated in significantly less time. We have built four unique knowledge bases using this methodology.

Keywords: Expert knowledge bases, crowd-sourcing authoring tools, ill-defined domains, collaboration

1 Introduction

This research investigates whether the process of authoring domain models for intelligent tutoring systems (ITS) can be simplified and automated, while at the same time maintaining the characteristics that make tutors effective and support large-scale tutor development. Specifically, this project tests whether authoring tools based on crowdsourcing can support development of large-scale, real-world tutors.

Intelligent tutors improve learning in a range of domains, using a variety of different approaches [2][14][12]. They promote learning in well-defined domains such as K-12 math [2][9]; however, it is still an open question how tutors can support learning in ill-defined problem spaces in which learners use multiple solution paths and opportunistically explore the subject matter at hand [10][3][12].

Our tutor Rashi (described in more detail in section 2) provides support in ill-defined domains by leveraging an expert knowledge base (EKB). One clear expense in building intelligent tutors like Rashi is the development of these domain models: computational models of problem solving or case-based reasoning in the domain. The domain model serves as a source of expert knowledge and a standard for evaluating the student's performance or for detecting errors [13]. It may also contain a detailed set of concepts, rules, facts, or hypotheses. Extensive interviews or "think aloud" protocols with subject matter experts (SMEs) are required to develop domain models while researchers try to take into account all the possible steps required to solve problems [1]. These interviews require extensive human input. Thus, the primary goal of our research is to provide techniques for improving the development time of these domain models.

One long-term goal of this research is to energize the development and use of intelligent tutors. Because their costs are high, tutors will only be built if they reduce overall teaching costs by reducing the need for human instructors or sufficiently boosting overall productivity [1]. Authoring tools provide strong advances in the ability to quickly create new domain models [12]. However, our goals are to make tutor development both easier and faster for researchers, trainers, and educators who are not experts in cognitive psychology or artificial intelligence.

This paper presents an approach to domain knowledge base construction that leverages the large corpus of data available when students work within tutors. We present the concept of an evolving expert knowledge base (EEKB), which is structurally equivalent to an expert knowledge base (EKB), but is generated by crowd-sourcing the actions of students using the tutor.

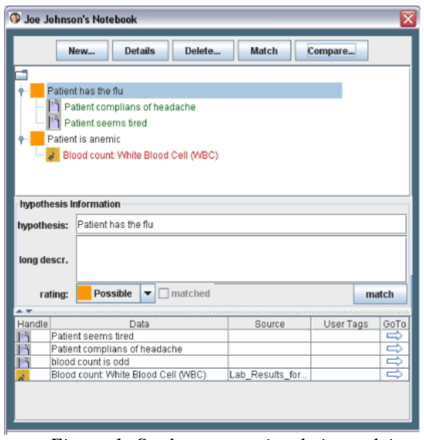
This approach is an application of the recently popularized field of crowdsourcing [11], in which anonymous participants are asked to contribute to a given body of work by each accomplishing a micro-task (a small spliced portion of the total workload). Crowdsourcing applications use various forms of motivation that include money [11] and fun [8]. Our approach explores a novel form of crowdsourcing and knowledge generation in which an EEKB is created by the amalgamation of student data, submitted through an ITS. Section 2 briefly describes our ITS Rashi, and the algorithm used to coalesce student data. Section 3 defines our method, focusing on the metrics for judging the quality of the generated EEKB. Section 4 describes our results and Section 5 concludes with a brief discussion.

2 Knowledge Acquisition Algorithm

This research attempts to validate the approach of using automated algorithms to generate an evolving expert knowledge base (EEKB). Our past research has shown that this model, once created, is useful for supporting students in ill-defined domains [5][6]. This section begins by briefly describing our tutor Rashi and the domain model that supports intelligent feedback in Rashi. Then we outline our algorithm for generating this domain model automatically.

Rashi is an existing well-vetted inquiry learning system that has been used by several thousand students. The system provides case descriptions for students to investigate problems, along with information about how to approach problems [4]. Rashi is domain independent and applications in several domains have been created and tested (e.g., biology, forestry and art history). This research focuses primarily on the Human Biology domain.

The system contains two components: a *procedural core* that supplies data collection mechanisms (e.g., interactive images, interview interfaces, video and



Patient has the flu
Patient complians of headache
Patient seems tired
Patient is anemic
Blood count: White Blood Cell (WBC)

hypothesis Information

hypothesis: Patient has the flu

long descr.

rating: Possible ☐ matched

Handle	Data	Source	User Tags	GoTo
	Patient seems tired			⇨
	Patient complians of headache			⇨
	blood count is odd			⇨
	Blood count: White Blood Cell (WBC)	Lab_Results_for...		⇨

Figure 1: Students organize their work in Rashi using the notebook.

dynamic maps) and a *content knowledge base* (i.e., an expert system) with knowledge of individual cases. In the Human Biology Tutor, students evaluate virtual patients and generate hypotheses about their medical condition. Patients' complaints form an initial set of data from which students begin the diagnostic process. Virtual patients are interviewed about symptoms. Students create hypotheses and establish relationships between observable data and hypotheses in their notebook (figure 1). Students move opportunistically from one inquiry phase to another as they sort, filter, and categorize data in order to form hypotheses about the patient's illness.

In addition, Rashi contains an algorithm for generating the domain model automatically. This algorithm works by accepting input from students that represents student actions within the tutor, e.g., providing evidence of a potential domain concept or relationship. The algorithm contains two handler methods, one that deals with analyzing evidence of incoming nodes (concept, hypothesis, data, etc.) and another that does the same for edges (relationships, etc.). These methods accept generic data in order to keep our algorithm generalizable, domain independent, and re-usable.

```
HandleNodeEvidence(nodeData:String)
HandleRelationshipEvidence(fromNodeId:int, toNodeId:int, relData:String)
```

These methods work in a very similar fashion, the pseudo-code for which is presented below:

```
Search EEKB for potential matches to this new item
IF: node / edge is not found already present in EEKB
        Then it is added with low default confidence of 10 percent
ELSE:
        The confidence of this information is increased
        A record of this addition is added
```

Thus as students work within Rashi, they generate hypotheses and data in their notebook (see Figure 1). Students establish relationships, discuss the problem via instant chat messaging, and type notes into a scratch pad. As students do this, the evidence of potential domain concepts and relationships is fed into our algorithm and a domain model is constructed. In addition, our algorithm exists on the server side, allowing it to collect this information invisibly among an array of students. The algorithm analyzes the evidence provided across students, merging new items into the domain model where appropriate.

3 Methodology

The evaluations presented in this paper address two goals. The first is to validate the quality of the evolving expert knowledge bases that are generated by our algorithm. We do this by defining two metrics: precision and recall. The second goal is to provide evidence that this approach decreases the build time of domain models for intelligent tutors.

We tested our approach by utilizing five years worth of Rashi data, garnered from four unique classroom settings (table 1). These settings provide a strong randomization of student age (middle school through college), student background (private and public school students, etc.), and pedagogical intervention (the large urban university used the tutor without pedagogical support while the large rural university included large amounts of researcher designed pedagogical interventions that described the goals of inquiry learning).

School	*Semester*	*Case(s) Used*
Small Rural Liberal Arts College Introduction to Biology	Fall 2007	Rene, Janet
	Fall 2008	Rene, Janet
	Fall 2009	Rene, Janet
College Summer Camp for Middle School Children	Summer 2007	Rene
	Summer 2008	Angela
	Summer 2009	Counselor, Janet
	Summer 2010	Counselor, Angela
Large Rural University Biology 101	Spring 2009	Janet, Freshman
	Spring 2010	Janet, Freshman
	Fall 2010	Janet, Angela
	Spring 2011	Janet, Angela
	Spring 2012	Janet
	Fall 2012	Janet
Large Urban University Biology 101	Fall 2012	Janet

Table 1. Schools from which data was gathered, along with the semester and medical cases used. Medical cases represented various medical diagnoses, e.g., thyroid diseases, allergy, insect bites.

Four of the medical cases from the table above provided sufficient data to analyze our approach. Our goal was to have at least 1000 data points per case sample for proper analysis. The four cases we used, along with the amount of usable student data per case, is summarized in table 2 below.

Because this data existed before our algorithm was implemented, it was fed through a specialized application that provided a virtual simulation of students using Rashi simultaneously. As this simulation progresses, the EEKB was generated and evolved over time.

Case	***Hypothesis Evidence***	***Num Data / Relationships***
Janet	4144	8108
Angela	1133	2424
Rene	3400	5978
Freshman	329	706

Table 2: Amounts of evidence provided by all students over the four cases from which our EEKB models are built.

To judge the quality of the evolving expert knowledge base (EEKB), we compared it directly to a human created expert knowledge base (HEKB). We utilize two distinct but related metrics: precision and recall.

We define precision as the information in the EEKB that is 'true' according to an HEKB built in the same domain. Thus, the precision of a generated model will be 100 percent if the EEKB is any subset of the HEKB. The formula for assessing precision of an EEKB compared to an HEKB is:

$$Precision\ (EEKB, HEKB) = |\ EEKB \cap HEKB\ |\ /\ |\ EEKB\ |$$

Where EEKB is the automatically generated graph and HEKB is the human generated graph

Recall is the percentage of the HEKB that we have successfully generated. Thus, this is a measurement of how well our algorithm has included the breadth of knowledge created by a human expert.

$$Recall\ (EEKB, HEKB) = |\ EEKB \cap HEKB\ |\ /\ |\ HEKB\ |$$

Where EEKB is the automatically generated graph and HEKB is the human generated graph

We observe how the generated model changes over time. Because of the nature of our simulation, we were able to take snapshots of the state of the EEKB at any time and calculate precision and recall. We decided to take such a measurement after every 100 student inputs were considered.

To judge build time efficiency, we first calculate the total number of hours necessary to build each model. For the HEKB, our subject matter estimated a required time of 60 hours for each of two team members in order to build 100 components within the domain model. Our HEKB models store 1000 components on average, which yields approximately 600 hours of work (as a team) or 1200 total human hours of work. We can divide these values to obtain various estimations of the number of hours needed per knowledge component generated.

Estimating efficiency of EEKB construction is slightly more difficult. We predicted (and eventually confirmed) that EEKB recall over time would be logarithmic in nature and thus level off. We examined this curve and determined a reasonable estimate for the amount of student data necessary to obtain a saturated EEKB. We defined an EEKB as being saturated when 1000 additional pieces of student evidence yield on average only one additional percentage point of recall. We simply divide this value by our observation of student data created per hour in Rashi. This data was obtained from our database and is summarized in Table 3.

Data Type	***Data per Hour***
Hypothesis	3
Data	8
Relationship	4
Total:	**15**

Table 3. Average amount student data created per hour in Rashi, these values are rounded for simplicity

4 Results

The table below provides a summary of the EEKB size totals generated by the respective amounts of student data.

Case	***~Student Data Considered***	***Nodes***	***Relations***	***Total***
Janet	12000	234	76	**310**
Angela	3500	158	8	**166**
Rene	9300	266	35	**301**
Freshman	1000	54	15	**69**

Table 4: Summary of the EEKB sizes generated for each Rashi case.

If we analyze precision over time, we see that this metric consistently hovers around 90 percent. This implies that our algorithm, although it creates relatively small knowledge bases, is precise in the knowledge it discovers and chooses to maintain.

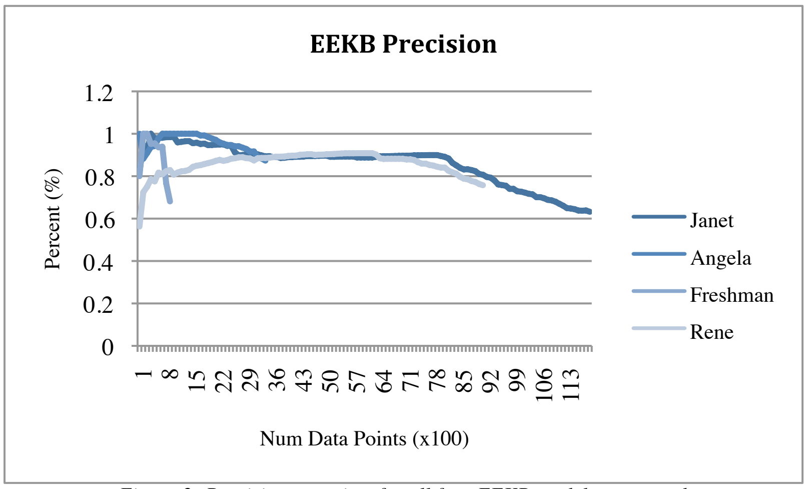


Figure 2: Precision over time for all four EEKB models generated

As the knowledge base begins to reach a saturation point, we see that precision begins to decline slightly. EEKB saturation occurs when the generated EEKB is large enough that substantial amounts of student data only lead to small increases in discovered knowledge.

EEKB recall over time is slightly different in that the raw numbers seem quite low (figure 3). In particular, our generated EEKB models only reached a maximum of 23 percent recall (meaning we generated 23 percent of what the expert generated). Additionally, analysis of recall over time shows a logarithmic curve, suggesting that saturation occurs around 20 percent recall.

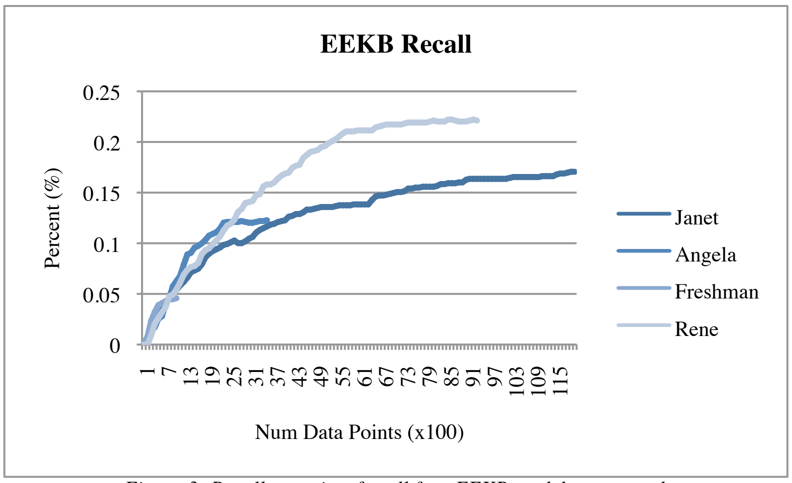


Figure 3: Recall over time for all four EEKB models generated

It is important to note now that we do not consider 20 percent recall to be a poor result. In fact, it is quite expected given results of past experiments and teaches us more about why automatic knowledge generation may actually be superior to hand crafted expert knowledge. We discuss this concept in detail in section 5.

Lastly, we attempt to calculate the efficiency with which our EEKB can be generated in comparison to a respective HEKB. We leverage the logarithmic property of recall over time to discover the saturation points of the four cases we analyzed. We do this by simply plotting a logarithmic fit to the recall over time curve for each Rashi case (see Figure 4).

We find the EEKB saturation point by calculating the amount of student data necessary to decrease the slope of the fitted curve to 0.001, which implies 1000 pieces of student data only lead to 1 additional recall percentage point. At this point in time, the EEKB is saturated because additional student tutor usage is not leading to more information, and instead our knowledge acquisition algorithm is observing a majority of duplicate entries. Thus, when this point is reached, it is sufficient to stop gathering student data and to consider the EEKB optimal. We perform this analysis to determine the minimum amount of data necessary on average from students in order to reach EEKB saturation.

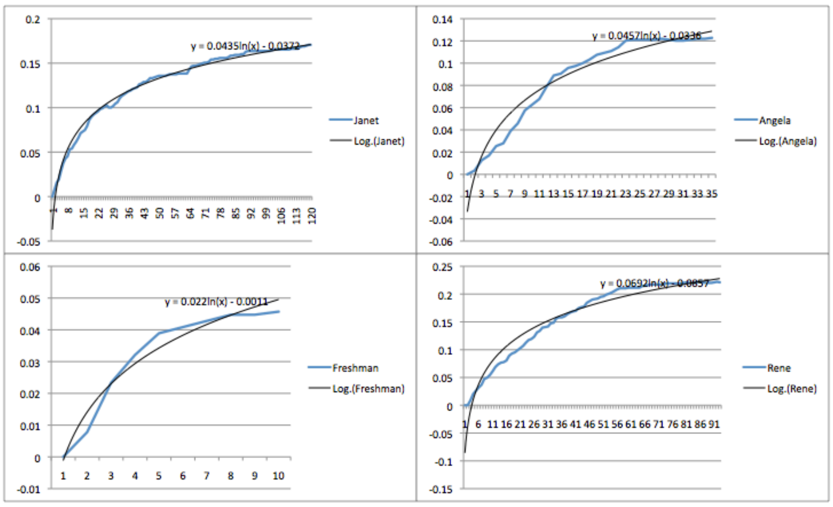


Figure 4: Logarithmic fits for recall curve of each case. The flattening of the curve represents the saturation of the EEKB that is generated

Once we have these values (left side of table 5), we use the average amount of data students create per hour in Rashi (right side table 5) to calculate the number of hours necessary to saturate an EEKB.

EEKB Case	*Num Data Points Until Saturation*
Janet	4350
Angela	4570
Rene	6920
Freshman	2200
AVG:	***4510***

Avg Student Hypos / Hour	3
Avg Student Data / Hour	8
Avg Relations / Hour	4
Avg. Stud. Data / Hour	***15***

Table 5: Amount of student evidence necessary to saturate the EEKB for each case (left) along with the average amount of student evidence created per hour on average (right)

The final calculation is shown below for automatic production of a knowledge base. We produced an average estimated value based on the disparate values for each of the four cases we considered. We found that it takes approximately 300.67 human hours to generate a saturated EEKB, which yields a generation rate of about 1 hour per component.

EEKB Saturation	*Stud. Data / Hour*	*Num. Hours*
4510	15	**300.67**

Table 6: Final calculation of student hours necessary to saturate an EEKB. Because students are generally available in groups, this time commitment has great potential to be parallelized

Our HEKB required 1200 total human hours to produce 1000 components, representing a build rate of 1.2 hours per component. The EEKB construction time reflects a 20 percent decrease in build time. In practice, student work is heavily parallelizable, and thus a class of 300 participants would create a saturated EEKB in merely one hour of Rashi usage. Table 7 summarizes the approach to building an HEKB versus an EEKB and the efficiencies of building the models. We assume a classroom size of 30 students using Rashi in parallel on average.

	Components Built	*Hours*	*Num Contributors*	*Hours per Component (non parallel)*	*Hours per Component (parallel)*
HEKB	1010	1200	2	1.188	0.594
EEKB	310	300.67	30	**0.970**	**0.032**
			% difference:	***18.37***	***94.56***

Table 7: A summary of build times necessary for an HEKB and respective EEKB. EEKB models can be built more efficiently using a combination of available students and our knowledge acquisition algorithm.

5 Conclusions

In conclusion, we present a novel approach for efficiently creating domain models within intelligent tutors without requiring extensive programming or tedious human interviews. In addition, we provide evidence that automated approaches to knowledge generation are efficient. Students, especially when sampled in mass quantities, are capable of creating precise knowledge. Our experiments calculate that students (along with our algorithm) can create knowledge about 20 percent faster than experts. In addition, because the student to teacher ratio is high, students have more capability for parallelization without being asked to perform additional work (i.e., the students were going to use Rashi as a learning activity anyway). This provides opportunities to create knowledge at rates that are up to 94 percent faster than when human experts perform this task. Related work, outside the scope of this paper, has shown evidence that these efficiency improvements can be elevated by applying game mechanics to the intelligent tutor from which the student source data is obtained [7].

Future work will involve testing this method to introduce more medical cases and then designing tools to evaluate how well our method transfers to cases in other domains. The models generated for this study exhibited high precision (meaning their content was reflective of 'true' knowledge). Our recall values may seem low, but previous results [6] have shown that students only explore approximately 20 percent of our human generated knowledge base when using Rashi and thus the result of 20 percent recall should be considered a positive one because it is consistent with the exploration patterns of students. If anything, our result provides evidence that expert crafted knowledge bases may lead to extraneous work (i.e., 70-80 percent of the human generated knowledge is rarely explored by students in practice).

Lastly, generating EEKB models is useful because they can be used to provide dynamic feedback to future students that use Rashi. Previous studies have demonstrated the benefits made available by a well-crafted knowledge base [4][5][6]. Thus, our approach aims to move forward towards system designs that both generate knowledge automatically and utilize that knowledge to benefit student education, while simultaneously relieving designers and programmers of the need to hand craft large domain models.

Acknowledgements

This research was funded by an award from the National Science Foundation, NSF 0632769, IIS CSE, Effective Collaborative Role-playing Environments, (PI) Beverly Woolf, with Merle Bruno and Daniel Suthers. Any opinions, findings, conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the funding agencies.

References

1. Aleven, V., McLaren, B., Sewall, J., Koedinger, K., (2009) A New Paradigm for Intelligent Tutoring Systems: Example-Tracing Tutors, International Journal of Artificial Intelligence in Education, Volume 19, Number 2 / 2009, 105-154
2. Arroyo, Ivon; Royer, James M. Woolf, Beverly Park . *"Using an Intelligent Tutor and Math Fluency Training to Improve Math Performance."* International Journal of Artificial Intelligence in Education (2011).
3. Dragon, T., Woolf, B. P., Marshall, D. and T. Murray. *"Coaching within a domain independent inquiry environment."* Proceedings of the Fifth International Conference on Intelligent Tutoring Systems. Jhongli, Taiwan, Springer 4053, 144-153 (2006)
4. Dragon, T., Floryan, M., Woolf, B.P., Murray, T. *"Recognizing Dialogue Content in Student Collaborative Conversation."* In Proceedings of the 10th International Conference for Intelligent Tutoring Systems, ITS 2010. Pittsburgh, PA (2010).
5. Dragon, T., Woolf, B.P. *"Understanding and Advising Students from within Inquiry Tutors."* Proceedings of the Workshop for Inquiry Learning of the 13th International Conference of Artificial Intelligence in Education. AIED 2007. Marina Del Rey, CA.
6. Floryan, M., Dragon, T., Woolf, B.P. *"When Less is More: Focused Pruning of Knowledge Bases to Improve Recognition of Student Conversation "* In Proceedings of the 11th International Conference for Intelligent Tutoring Systems. Chania, Greece (2012).
7. Floryan, M., Woolf, B.P. *"Improving the Efficiency of Automatic Knowledge Generation through Games and Simulations"*. In Submission.
8. Games with a Purpose. www.gwap.com/gwap Last accessed on 2 February 2012.
9. Koedinger, Kenneth R. Aleven, Vincent. Heffernan, Neil. McLaren, Bruce. Hockenberry, Matthew. *"Opening the Door to Non-Programmers: Authoring Intelligent Tutor Behavior by Demonstration"*. Intelligent Tutoring Systems, Vol. 3220 (2004).
10. Lynch, C. Ashley, K. D. Pinkwart, N., Aleven, V. (2009) "*Concepts, Structures, and Goals: Redefining Ill-Definedness"* In Aleven, V., Lynch, C. Pinkwart, N., Ashley, K. (Eds). *International Journal of AI in Education; Special Issue on Ill-Defined Domains*. Volume 19. Number 3. pp. 253-266
11. Kittur, Aniket; Chi, Ed H. Suh, Bongwon. *"Crowdsourcing User Studies With Mechanical Turk"*. Proceedings of the ACM CHI Conference. Florence, Italy. 2008.
12. Mitrovic, A.; Suraweera, P.; Martin, B.; Zakharov, K.; Milik, N.; Holland, J. *"Authoring Constraint-Based Tutors in ASPIRE"* Intelligent Tutoring Systems, Lecture Notes in Computer Science Volume 4053, 2006, pp 41-50.
13. Nkambou, Roger; Bourdeau, Jacqueline; Mizoguchi, Riichiro, eds. (2010). *Advances in Intelligent Tutoring Systems*. Springer. ISBN 3-642-14362-8.
14. Van Lehn, K., Lynch, C., Schulze, K., Shapiro, J. A., Shelby, R. H., Taylor, L., Treacy, D. J., Weinstein, A., and Wintersgill, M. C. (2005). The Andes physics tutoring system: Five years of evaluations. In: G. I. McCalla and C.-K. Looi (Eds.), *Proceedings of the Artificial Intelligence in Education Conference*. Amsterdam: IOS.