# From Qualitative Reasoning Models to Bayesian-based Learner Modeling

**Uroš Milošević** and **Bert Bredeweg**
Informatics Institute, Faculty of Science
University of Amsterdam, The Netherlands
E-mail: {B.Bredeweg}@uva.nl

**Abstract**

Assessing the knowledge of a student is a fundamental part of intelligent learning environments. We present a Bayesian network based approach to dealing with uncertainty when estimating a learner's state of knowledge in the context of Qualitative Reasoning (QR). A proposal for a global architecture is given. The essentials of the belief network structure for individual scenarios are described, while paying special attention to knowledge aggregation and some design issues that are specific for the domain of QR.

## Introduction

The DynaLearn project[1] is an effort to develop an Interactive Learning Environment that relies on Qualitative Reasoning (QR) methods to assist students in constructing their conceptual system knowledge. However, many key aspects of DynaLearn, such as deciding which problems to focus on while tutoring, are reliant upon accurate estimations of the student's knowledge state at any point in time. A well known framework that deals with quantification and manipulation of uncertainty is a *Bayesian network* (BN). This paper presents a BN approach to learner modeling for learners interacting with QR models.

**Qualitative Reasoning** An important power of QR lies in the ability to capture both structural and behavioral information, including the notion of causality. Garp3 provides a workbench that allows users to build, simulate, and inspect qualitative knowledge (Bredeweg et al. 2009). An important ingredient of a Garp3 model is the *Entity*, which can be arranged in a subtype hierarchy, and is used to represent the physical objects or abstract concepts the system itself is made of. The relevant properties that can change under the influence of processes are represented as *quantities*. *Agents* represent external entities that may influence the system's behavior, referred to as *exogenous quantities*. To indicate that certain conditions are presumed to be true, the modeler can use *Assumptions*. Their main use is to constrain the modeled system's behavior. Finally, *Configurations* represent the structural relationships between entities and agents.

[1]The research presented here is co-funded by the EC within FP7 (2009-2012) (project DynaLearn, 231526, www.DynaLearn.eu).

Key aggregates are *Scenarios* and *Model fragments (MFs)*. Scenarios are used to represent a specific state the simulation may start from. MFs represent individual parts of knowledge about the system. When running a simulation for a given scenario, the QR engine searches the MF library to find the fragments that match the conditions and generate a state graph.

**Bayesian networks as models of learners** *Probabilistic graphical models*, diagrammatic representations of probability distributions, are useful in analyzing and solving complicated probabilistic models. They consist of nodes connected by edges, where each node represents a random variable, and each edge denotes a probabilistic relationship between the variables it connects. BNs are represented in terms of a directed graph. An important restriction that applies to such directed graphs is that they must be kept *acyclic*. In other words, there must be no closed paths within the graph that would allow us to start at one node, follow a path and come back to the point of origin. Therefore, BNs are essentially *directed acyclic graphs* (DAGs).

Knowledge tracing was an early approach to student modeling that exploited the power of Bayesian inference, initially used by Corbett and Anderson in their ACT Programming Tutor (APT; 1995), but was also proven to be useful for designing tutors for mathematics (Koedinger 2002), and improving reading skills (Beck and Chang 2007), and is statistically equivalent to Reye's two-node dynamic BN used in many other learning environments (Reye 2004). Corbett and Anderson proposed a two-state model: each topic is either *learned* or *unlearned* (i.e. *known* or *unknown*). The model doesn't implement the idea of *forgetting*, but it does take the possibility of *guesses* and *slips* into consideration. Their knowledge tracing approach uses four initial parameters:

1. $G$ – the *guess* parameter, i.e. the probability of the student's correct answer being a "lucky guess" (when they don't know the actual correct answer);
2. $S$ – the *slip* parameter, i.e. the probability of the student accidentally giving an incorrect answer (when they actually know the correct one);
3. $L_0$ – the probability of knowing a skill at the beginning of a tutoring session;
4. $T$ – the probability of learning an "unknown" skill, at any

given point during a tutoring session.

Therefore, the probability of knowing a certain topic after an action $n$ is calculated using that action as evidence for the posterior probability (Equation 1). In the APT, the student is given exercises until their mastery level reached the probability of 95%.

$$\begin{aligned} p(L_n|evidence) &= p(L_{n-1}|evidence) \\ &+ ((1 - p(L_{n-1}|evidence))p(T)) \end{aligned}$$

Beck (2007) pointed out a deficiency of most methods for developing Bayesian Knowledge Tracing models for specific skills – the *identifiability* problem, where models with equally good statistical fit to performance data may make very different predictions about a learner's knowledge state, which could result in different numbers of problems to be assigned to a student. Consequently, the problem could result in under- or over-practice.

Baker, Corbett, and Aleven (2008) proposed a new method for finding two of the four basic parameters, that differs in estimating the guess and slip parameters for individual actions (i.e. related to the context), instead of holding them constant for all situations.

## Bayesian learner model for Garp3

In DynaLearn, the information flow starts with the user who constructs and simulates QR models. The information from these models and simulations is used to construct a BN. Based on the information in the network, i.e. the probabilities of knowing each concept, a question focus is determined, and a question request is sent to QUAGS (Goddijn, Bouwer, and Bredeweg 2003). The question generator uses this request, the information about the models, question templates and a number of criteria to output a list of possible questions. Then, a question is selected and forwarded to the learner. After the learner provides an answer, the information about the question and the response is stored in the dialog history, and the belief network is updated.

In Brielmann's (2009) learner model, each Garp3 primitive is represented as a node in the model BN, where magnitudes, derivatives, dependencies and correspondences are set to be the root nodes. We've already discussed the *knowledge aggregation* methods proposed by Reye (2004) and Millán, Pérez-de-la Cruz, and Suárez (2000), and Brielmann adopts the same approach in her model. We can observe the low level concepts as the concept nodes, the quantities as topic nodes, and entities as subject nodes found in Millán, Pérez-de-la Cruz, and Suárez's paper. That is, in order for a student to understand an entity, they must know all of its quantities first. In order to know a quantity, on the other hand, a learner must understand all of the root concepts *directly related to it*. The idea is shown in Fig. 1 for a simple tree and shade model (Bredeweg et al. 2006). It is supposed to describe the relation between a growing tree and the shade that's being cast on the ground by the tree. We see that the proportionality and influence nodes both contribute to two upper level nodes, as each of them carries knowledge about two concepts.

Each root node is connected to an extra child node representing an observation from a system-learner interaction. This approach makes it possible to include the possibility of guesses and slips, as proposed by Reye (2004). As the learner's knowledge changes over time, after each interaction, the network is extended by another time slice. This lets us collect more than one piece of evidence for a piece of knowledge and provides us with the means to add the possibility of the learner learning or forgetting something between two points in time/interactions, as discussed in the earlier sections.

There is a number of key issues not covered by Brielmann's approach. If a model has multiple scenarios, the system would see each scenario as a completely separate domain. Moreover, it doesn't cover recurring concepts or substructures within the same scenario. Learning about one instance of a MF should imply we have also learned something about any other instance of the same MF.

## Global architecture

A single scenario (and simulation) within a QR model holds only partial information about the domain covered by the model. Similarly, a single QR model could be only a small part of a bigger domain a student's supposed to cover. Therefore, our learner model should not only be able to handle individual scenarios, as per single examination session provided by the question generator, but also support moving between scenarios, or even models.

Two possibilities seem appropriate. We could either update the information inside the existing BN, or create a new (global) one, based on the information contained in the previously created network(s). We know, however, that we don't want to simply move the knowledge directly from one scenario to another and aggregate unnecessary information as we move on. Also, trapping the acquired information in a single external network is definitely not the way to move forward. Therefore, an ideal approach should store acquired knowledge in an external repository after each examination session, but also be able to retrieve the *necessary* information from that source when switching to a new scenario.

When loading a new simulation, the system would compare the concepts occurring in it with the ones contained in the model knowledge base. It would then create the belief network for the given scenario and update the information about the concepts the student has already seen earlier accordingly. The same approach could be easily extended to the global/domain level, so that the information would flow between a domain knowledge base (KB) and the individual scenario KBs.

## Individual scenario with simulation

What needs to be learned? We should pay special attention to the low level (system) view, as it holds the essential information stored in each QR model, as suggested by Brielmann (2009). To determine exactly what low-level concepts need to be represented in the network structure, we need to see first what questions can be asked, and what information the answers to questions hold.

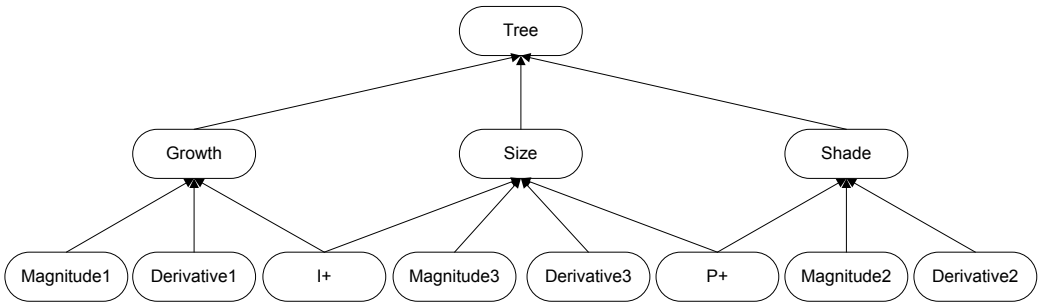


Figure 1: Tree and shade: Knowledge aggregation

An analysis of all QUAGS question types has produced the following list of low level concepts (to be explained below) the answers to such questions can hold information about: (1) Magnitudes, (2) Derivatives, (3) Quantity spaces, (4) Influences and proportionalities, (5) Inequalities, (6) Calculations, and (7) Correspondences.

Each of the items in the above list can be addressed *directly* by a QUAGS question. We consider quantities, entities and MFs high level concepts, and the question generator doesn't ask any direct questions about any of them. Keep in mind, though, that if the current focus of the question generator was a certain quantity, we would like to track the student's knowledge about that quantity. Therefore, we're not saying high level nodes should be left out from the network structure.

**Recurring concepts** Imagine the following setup – in an additional MF of the *Tree and shade* model, instead of one, we have two trees, where one grows in the shade of the other (bigger) one. Therefore, the more the big tree grows, the slower will the small tree grow. Keep in mind that this third MF would become active only if the scenario setup allowed for it. Now, suppose in one scenario, this new MF activates. In terms of low level concepts seen above, same forces are at work for both trees. This is due to the fact that they are merely instances of the same entity, that serves as a condition for two other MFs, that specify the dynamics of the tree and shade growth process. Going back to the original question, i.e. "What needs to be learned?", we realize that what we want to learn about is the *Tree* entity, and any instances of this entity should be treated as such. That is, every instance of a concept carries a certain amount of information about the generic entity, but the global (model/domain) knowledge base should keep track only of the generic concepts, as this is what we want to know about.

**Recurring substructures** The example with two trees given above served another purpose by bringing up the following question: if a student learns something about one of the trees, haven't they learned something about the other one as well? The problem at hand is a peculiar one, as it suggest the information in such cases should flow both ways. We already know that loops are not allowed in BNs. That means that a direct connection from one instance to another, as shown in Fig. 2, wouldn't work. Another approach, preserving the guess and slip factor for instance-specific question nodes, would also result in an illegal BN structure (Fig. 3). Many other options also fail.

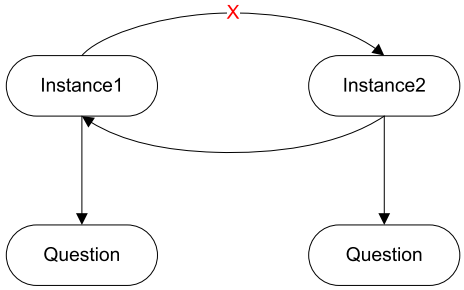


Figure 2: Illegal graph - Direct connection

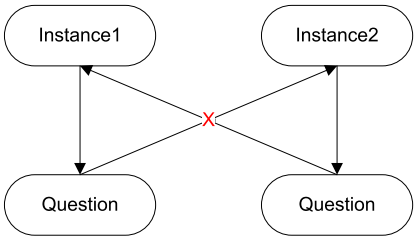


Figure 3: Illegal graph - Via question nodes

Therefore, we avoid the cycle trap through the idea of *answer collectors*. This approach adds an additional layer to the network, between the low level concepts and the question (outcome) nodes. The structure in Fig. 4 shows a low level concept called *PositiveProportionality_AH* and three of its instances. Each answer collector, as its name implies, *collects answers* coming from the outcome nodes and distributes it among all of the instance nodes. Moreover, the idea of guesses and slips is preserved.

**Soft evidential updating** As Valtorta, gyun Kim, and Vomlel (2000) explain, evidence is a collection of hard or

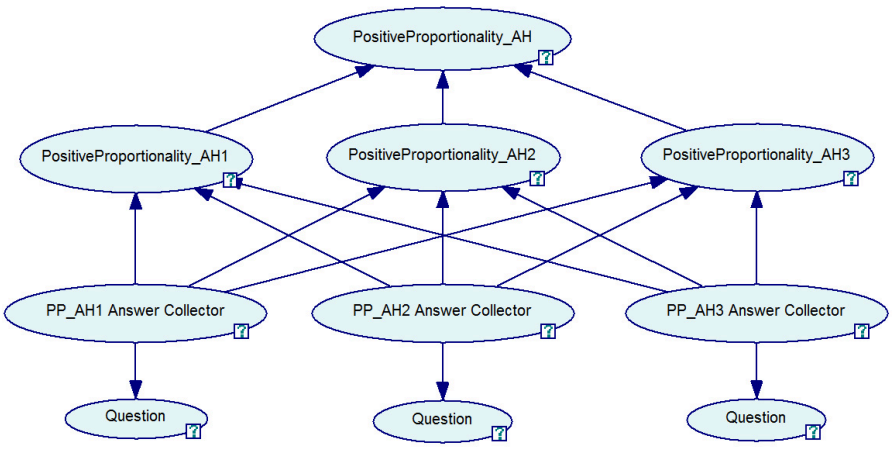


Figure 4: Recurring concept knowledge distribution via answer collectors

soft findings on variables. Whereas a hard finding specifies which value a variable is in, a soft one specifies a probability distribution of a variable. Soft evidence is a collection of soft findings. In other words, instead of saying a concept is simply *known* or *unknown*, we could directly involve uncertainty using soft evidence and, after an interaction, say something is, for instance, *65% known*. Then, by leaving only the question node dynamic, we could mimic Reye's (2004) approach by directly setting the probabilities for each time slice, with a controlled level of uncertainty.

**No temporal nodes** The last option is to abandon the idea of time entirely (this doesn't mean we should ignore the concept of interaction history, though). One way to do this is to assign multiple question nodes to each concept (i.e. one node for each question). However, in order to preserve the idea of guesses and slips, and merge the new approach with the rest of the architecture seen so far, we would need to add another, intermediate layer (as the direction of the edges pointing to question nodes would have to be reversed). Fig. 5 shows what this new substructure would look like for an answer collector with three questions.

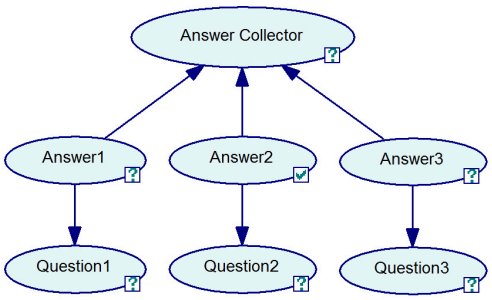


Figure 5: Multiple questions, no time slices

**How many questions?** An obvious question that remains to be answered is how many questions we need to ask for each concept in order to mark it as "known".

*Concept types* – First of all, we need to be aware of the number of possible question types the question generator can ask for each concept. This can also serve as a relevance measure for each concept type. For instance, QUAGS can ask only one question about a quantity space, but multiple questions about magnitudes, derivatives, (in)direct influences etc.

*Domain difficulty* – When it comes to QR models, domain difficulty could be measured in terms of model, or even simulation *complexity*. However, an appropriate complexity measure would have to be a relative one, taking into consideration a large library of models, scaling from simple domains to complex ones. We're saying "simple" and "complex", instead of "small" and "large", as a large model is not necessarily a complex one, as it may produce a simpler state graph than a considerably smaller model.

*Quantity spaces vs. network size* – As only a single question can be asked about a quantity space, not taking quantity spaces into consideration as separate concepts is also an option. Each quantity node (both generic and scenario specific) requires adding an additional node to represent its quantity space. Therefore, by excluding such nodes, we would be reducing the size of the network by a number corresponding to the total number of quantity nodes. As a substitute solution, we could attribute quantity space specific questions to magnitudes.

**Low level concept impact** In the simplest setup, we assume each of the low level concept nodes contributes to the higher level nodes equally. In the example seen in Fig. 1, for instance, that would mean that each of the four nodes connected to the *Size* node would "weigh" 25% in terms of knowledge. However, one could disagree that the concepts are equally important, and say that, for example, the knowledge about a magnitude is more important than the one about an influence. This could be regarded as a matter of preference.

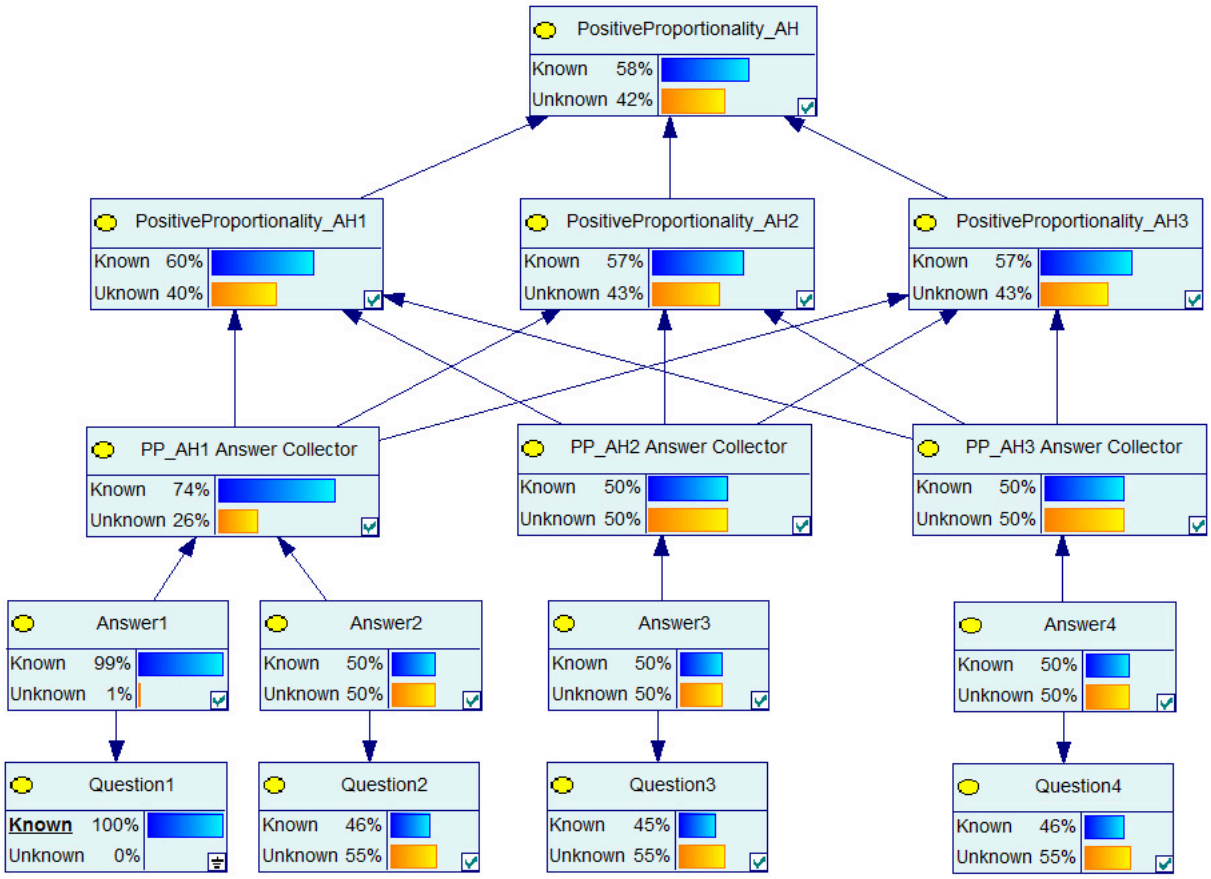


Figure 6: Answer collector impact with instance specific weights

**Answer collector impact** One might argue that an instance specific answer collector should "share" more knowledge with the instance it belongs to, i.e. have a little more impact on it than the other collector nodes. We can try to assign a weight, e.g. $X$, that will help us distribute the probabilities less "fairly". For example, in the above mentioned example with 3 instances, with a fair probability distribution, the definition table for PositiveProportionality_AH1 looks as given in Table 1 (K = Known, U = Unknown).

Therefore, if the instance specific answer collector is known, the algorithm should boost the probabilities by the amount assigned to the weight variable, and if it is unknown, the probabilities should be decreased by the same amount. To be more specific, currently, the probability of a node being known ($K$) is calculated as follows:

$$K = \frac{k}{N} \tag{1}$$

where $N$ is the number of instances and $k$ the number of known instances. By altering the distribution process as explained above we get:

1. if the instance specific answer collector is known:

$$K = \frac{k}{N} + \frac{X}{N} \tag{2}$$

2. if the instance specific answer collector is not known:

$$K = \frac{k}{N} - \frac{X}{N}. \tag{3}$$

We normalize the weight ($X/N$), so the probabilities get boosted/decreased by a normalized "fraction of knowledge" rather than by a fixed number (we don't want to add/subtract the same amount, e.g. 5%, to an instance with 1 and 10 duplicates). The modified CPD for $X = 0.06$ can be seen in Table 2.

Fig. 6 depicts this idea. The displayed structure uses the non-dynamic approach (i.e. a fixed number of questions) shown in Fig. 5 for the sake of clarity. We can see that one of question nodes was marked as known. Here, we can also see the guess factor at work – the reason the answer node right above the question one shows the probability of 99% instead of 100%, as one may expect, is the fact that the probability of a guess is set to 0.01 (i.e. 1%). This further boosts the probability of the instance-specific answer collector to

| PositiveProportionality_AH1 | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **PP_AH1_Answer Collector** | K | | | | U | | | |
| **PP_AH2_Answer Collector** | K | | U | | K | | U | |
| **PP_AH3_Answer Collector** | K | U | K | U | K | U | K | U |
| **Known** | 1 | 0.67 | 0.67 | 0.33 | 0.67 | 0.33 | 0.33 | 0 |
| **Unknown** | 0 | 0.33 | 0.33 | 0.67 | 0.33 | 0.67 | 0.67 | 1 |

Table 1: CPD table for the PositiveProportionality_AH1 node

| PositiveProportionality_AH1 | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **PP_AH1_Answer Collector** | K | | | | U | | | |
| **PP_AH2_Answer Collector** | K | | U | | K | | U | |
| **PP_AH3_Answer Collector** | K | U | K | U | K | U | K | U |
| **Known** | 1 | 0.73 | 0.73 | 0.39 | 0.61 | 0.27 | 0.27 | 0 |
| **Unknown** | 0 | 0.27 | 0.27 | 0.61 | 0.39 | 0.73 | 0.73 | 1 |

Table 2: Revised CPD table for the PositiveProportionality_AH1 node

74% and the probability of knowing that particular instance (*PositiveProportionality_AH1*) to 60%. At the same time, the other instances also receive an increase in probability, but the impact of the external answer collector is slightly less, as expected, so the new value for each of the duplicates is 57%. Each of the instances contributes equally to the generic concept, so the probability of knowing *PositiveProportionality_AH* is now 58%.

## Discussion

Brielmann's approach was considerably improved, as many fundamental issues related to constructing a BN from a QR model were successfully solved. The new approach advances this work. It provides the system with the means to successfully gather information about a student's understanding of a domain. Still, it appears some obstacles need more work. For instance, selecting an appropriate number of questions per concept type beforehand needs to be further investigated. The lack of temporal nodes makes it impossible to see the learner's knowledge of any concept go above a limit set/constrained by the guess and slip parameters.

Further improvement concerns alternative methods for updating a user's knowledge (both temporal nodes and soft evidence). This would solve the above mentioned problems regarding the maximum probability that can be achieved for any node. Also, the idea of different concept types carrying different amounts of information should be given more attention.

Another topic that should be investigated further is the idea of "landmarks". For instance, certain MFs have conditions, such as a specific quantity value that needs to be reached in order for the MF to get activated. This information appears to be crucial, and, as such, should be properly represented in the learner model.

## References


Baker, R. S.; Corbett, A. T.; and Aleven, V. 2008. More accurate student modeling through contextual estimation of slip and guess probabilities in bayesian knowledge tracing. In *Proc. of ITS 2008.*, 406–415. Springer-Verlag.

Beck, J. E., and Chang, K.-M. 2007. Identifiability: A fundamental problem of student modeling. In *Proc. of UM 2007.*, 137–146. Springer-Verlag.

Beck, J. E. 2007. Difficulties in inferring student knowledge from observations (and why you should care). In *Educational Data Mining: Supplementary proc. of AIED 2007.*, 21–30.

Bredeweg, B.; Bouwer, A.; Liem, J.; and Salles, P. 2006. Curriculum for learning about qr modelling. *Naturnet-Redime, STREP EU FP6 project no. 004074, D6.9.1*.

Bredeweg, B.; Linnebank, F.; Bouwer, A.; and Liem, J. 2009. Garp3 - workbench for qualitative modelling and simulation. *Ecological Informatics* 4(5–6):263–281.

Brielmann, M. 2009. A learner model based on a bayesian network for garp3. Master's thesis, AMSTEL Institute, University of Amsterdam.

Corbett, A. T., and Anderson, J. R. 1995. Knowledge tracing: Modelling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction* 4(4):253–278.

Goddijn, F.; Bouwer, A.; and Bredeweg, B. 2003. Automatically generating tutoring questions for qualitative simulations. In *Proc. of QR 2003*, 87–94.

Koedinger, K. R. 2002. Toward evidence for instructional design principles: Examples from cognitive tutor math 6. invited paper. In *Proc. of PME-NA XXXIII 2002.*, 21–49.

Millán, E.; Pérez-de-la Cruz, J.-L.; and Suárez, E. 2000. Adaptive bayesian networks for multilevel student modelling. In *Proc. of ITS 2000.*, 534–543. Springer-Verlag.

Reye, J. 2004. Student modelling based on belief networks. *International Journal of Artificial Intelligence in Education* 14(1):63–96.

Valtorta, M.; gyun Kim, Y.; and Vomlel, J. 2000. Soft evidential update for probabilistic multiagent systems. *International Journal of Approximate Reasoning* 29:71–106.