Automatic Aggregation of Qualitative Reasoning Networks

Kees de Koning and Bert Bredeweg and Joost Breuker
Department of SWI, University of Amsterdam
Roetersstraat 15, 1018 WB Amsterdam, the Netherlands
{kees,bert,breuker}@swi.psy.uva.nl

Abstract

Understanding and reasoning about the physical world is a task that is implied in large portions of both practical and theoretical education and training. In this paper we present techniques for the automated generation of domain knowledge models that support ITS functions for coaching these tasks. On the basis of a qualitative simulator (GARP), a complete reasoning network is generated for each exercise in a domain. However, a serious problem with such a network is that it easily becomes huge: it contains all necessary and grounded reasoning steps. In particular, complex coaching tasks like cognitive diagnosis (CD) soon become intractable. A solution can be found in 'summarising' the network into a hierarchical one by applying aggregation methods. The results these methods produce show similarities to the outcomes of learning mechanisms such as compiling out and chunking in human skill acquisition and in machine learning. The aggregated models present a more abstract view on the domain, in a way comparable to the result of a simulation using more abstract models. The major advantage of our approach, however, is that all different abstraction levels, *plus* their interconnections (*i.e.*, the abstraction operations) are available to the tutor. The hierarchical reasoning networks are thus not only relevant for making CD more tractable, but also, and in particular, for enabling an ITS to communicate with the student about the reasoning at different grain size levels. A worked-out example is presented.

Introduction

Reasoning in a qualitative way about the behaviour of a physical system involves making inferences about quantities on the basis of given or earlier derived quantities. The dependencies between all possible inference steps that enable the prediction of the behaviour of the system form a network (directed graph). This network thus represents both the domain knowledge (a behavioural model of a physical system) and the required inferences to derive this behaviour. In an intelligent coaching system such a domain representation is a model *par excellence* for interpreting the reasoning of a student in solving such problems.

Qualitative simulation can be used to construct such a model automatically on the basis of a model of the system (scenario) and model fragments that reflect the behavioural principles of the system involved. Since SOPHIE-III (Brown *et al.*, 1982), there seemed a bright future for using qualitative reasoning in ITSs. However, thus far only a few ITSs have been constructed using qualitative models. An important reason is that the construction of a correct qualitative simulation at the right level of detail is a laborious task (*e.g.*, Schut, 1996). In addition, there are two more principle problems in applying qualitative reasoning in teaching systems.

1. There is no one to one correspondence between the way a qualitative simulator generates its derivations and the way students do (de Koning and Bredeweg, 1996). A simulator usually generates all derivations in a breadth-first way, while students may reason backwards from otherwise predictable marked states.

2. The derivation models as resulting from qualitative simulation are relatively large, and contain too little internal structure to be directly applicable in a teaching environment. For instance, our original motivation for developing these models was to use them for model-based diagnosis: by conceiving the model (network) as consisting of components (inferences) and connections (input-output relations), one can use model based diagnosis techniques for the cognitive diagnosis (CD) of student's errors (Self, 1992; Bredeweg and Breuker, 1993; de Koning *et al.*, 1995). However, even for small physical systems such models become easily intractable for model-based diagnosis.

Both problems are addressed in this paper. A solution for the first problem is to have a qualitative simulator (GARP, Bredeweg, 1992) generate for each problem

situation the behavioural descriptions and then *reconstruct* derivation chains from these data on the basis of human interaction protocols. This way, we exploit the (correct) declarative knowledge generated by the simulator, but avoid its cognitively less plausible reasoning process. This is in line with earlier research on the cognitive plausibility of representations for qualitative prediction: we found that the declarative knowledge representation sufficiently covers the vocabulary that humans use in qualitative prediction of behaviour, but that indeed the reasoning patterns employed by human reasoners differs substantially from the simulator's (de Koning and Bredeweg, 1996). In the next section, we describe the generation of this model, referred to as the *base model*, from the simulator's output.

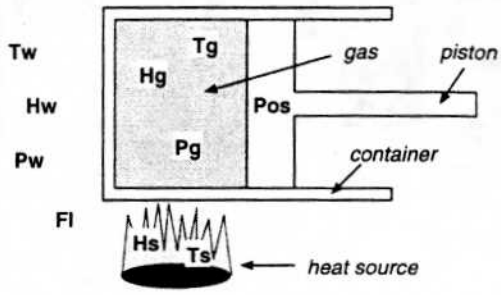
The tractability problem is tackled by the use of *aggregation*. This way, the base model becomes a hierarchical one, thus drastically reducing the complexity. We describe the techniques employed to dynamically generate such hierarchical models. The aggregation procedures implemented are not only beneficial for model-based diagnosis, but also facilitate advanced knowledge communication by structuring the knowledge at different abstraction levels. We present an example of the procedures implemented, as well as the hierarchical set of models that is generated for a simple physics system.

The Base Model

Quantity inferences (such as "because this pressure difference is zero, the resulting force is also zero") are shown to cover a major part of the reasoning reported in interactions between students and teachers (de Koning and Bredeweg, 1996). So, if we want a computer to teach about device behaviour, modelling quantity inferences is a good place to start. By focussing on quantity manipulation, we do not address another important aspect in qualitative reasoning, namely the conceptualisation of a domain: the first step in qualitative prediction is always the construction of a model of the system at hand.

The quantity inferences modelled are dependent upon one another. The input for an inference is some quantity expression that is either given in the problem description or depiction, or derived by a preceding inference. These dependencies form a graph, which is automatically generated on the basis of the simulator output using types of inference steps as observed in interaction protocols of student's solving qualitative prediction problems. As an example domain, consider the piston system shown in Figure 1.

The piston system consists of a movable piston in a container. In the container, there is some gas, and under the container there is a heater which heats the gas. The student's task is to predict what will happen to the piston system. Some (but not all) of the quantities needed in predicting the behaviour of the piston system are presented in Figure 1.



Tw = temperature of the surrounding world
Hw = heat of the surrounding world
Pw = pressure of the surrounding world
Tg = temperature of the gas
Hg = heat of the gas
Pg = pressure of the gas
Ts = temperature of the heat source
Hs = heat of the heat source
Fl = heat flow between heat source and gas
Pos = position of the piston

Figure 1: The Piston System

At the initial state of the system, the temperature of the gas (T_g) and that of the outside world (T_w) are given to be equal ($T_g = T_w$), and the heater is just turned on, expressed by the fact that its temperature (T_s) is higher than that of the gas ($T_s > T_g$). Furthermore, the temperature of the outside world is given not to change ($\delta T_w = 0$), and the piston is in its starting position ($Pos = s$).

A small part of the network for the behaviour prediction of this system is depicted in Figure 2. All the given expressions are printed in bold face. A rectangle represents an inference, and the arrows at the left, top, and right represent respectively the dynamic input, the generic knowledge, and the output of the inference. In line with the main motivation for building the models, namely (model-based) diagnosis, we conceive the network as consisting of *components* and *connections*. A reasoning step is modelled as a component, whereas the data transported over the connections are the quantity values, derivatives, and relations that are manipulated by the reasoning steps. From the given information (in bold face), the student should be able to derive $T_g > T_w$: the temperature in the container will become higher than the temperature of the surrounding world. A verbalisation of the deduction of $T_g > T_w$ by a student could be as follows:

> "Because the temperature of the heater is higher than the temperature of the gas, there will be a heat flow from the heater to the gas, which will

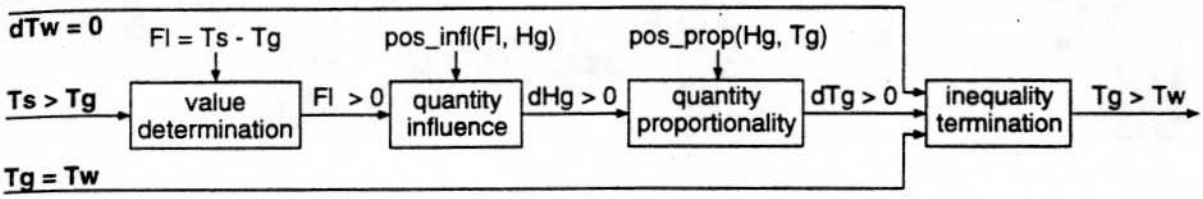


Figure 2: An Example Model

increase the temperature of the gas. Therefore, the temperature of the gas will no longer be equal to the temperature in the outside world, but will become higher."

The four component types mentioned in Figure 2 (`value determination`, `quantity influence`, `quantity proportionality`, and `inequality termination`) are examples of the total of 16 different types we defined. This set is based on experimental research on how students and (human) teachers communicate about prediction problems (de Koning and Bredeweg, 1996). The model is automatically generated from the *output* of GARP. An important aspect of the transformation is its 'reverse-engineering' nature. We do not translate the reasoning steps made by the simulator, but instead only use the bare *results* (facts) of the simulation. Then, we start from the set of 16 model component types and insert all components that make a valid (in terms of the component's behavioural rules) derivation. In other words, we start with the set of all facts that should be derived, and add all inference steps connecting them, where the set of possible inference steps (components) is not based on the simulator's reasoning, but on observed human reasoning. Figure 3 illustrates the process, including the hierarchical aggregation discussed in the next section.

Hierarchical models

A complete model of a behaviour prediction for systems like the contained piston in Figure 1 contains 824 components (representing reasoning steps) and 802 points (holding domain expressions to be communicated). For our first application of the models, *i.e.* in model-based diagnosis, this number is already too large to handle in reasonable time. Therefore, we need some kind of focussing of the diagnostic process. The solution we choose is based on the ideas of *hierarchical diagnosis* (*e.g.*, Mozetič, 1991): by defining abstract components that represent a set of components at a lower level of abstraction, we reduce the initial size of the model. We start diagnosing at the set of highest-level models, and only when a diagnosis has been found there, we decompose the abstract component into a lower level. Now we only have to diagnose within the model of the decomposed abstract component, hence drastically reducing the search space.

The initial applications of hierarchical diagnosis were in the field of electronics, for which the choice of the higher-level, abstract components was mostly guided by the function of a set of components (*e.g.*, a set of electronical components comprise an amplifier) and/or by the structural grouping on individual circuit boards (*i.e.* a mother board in a PC). In the case of a reasoning network there is no such thing as a blue print or a physical device to guide the hierarchical organisation. Hence, if we want to generate hierarchical models automatically, we need a set of operational criteria of which sets of components can be abstracted to a higher-level component. An important requirement is that these abstractions should result in models for which the points are still 'measurable', *i.e.*, the remaining domain knowledge indeed facilitates a communication with the student at a higher abstraction level.

We employ three different principles for combining different components, which are based both on the structural properties of the reasoning network and on theories about human learning (*cf.* the next section).

Hiding of Irrelevant Details results in some components and points of the model to be discarded at a higher level, because they are less important for the main prediction at hand.

A first form of hiding consists of leaving out all inference steps that do not strictly contribute to the right solution: although needed in the reasoning process to ensure completeness, they do not belong to the shortest path in the network from the givens to the final behavioural state. For instance, if the container is heated for some time, a second heat flow from the gas to the outside world will emerge. When we are reasoning about a state in which this effect is smaller than the flow from the heat source (*i.e.*, the effect is 'qualitatively negligible'), we can also leave out the components representing this effect in the model. We also hide all termination inferences that do not actively contribute to the actual transition(s) executed. Reasons may for instance be that a termination is preceded by another one, and hence will not be active yet.

Hiding also abstracts from *fully-corresponding*

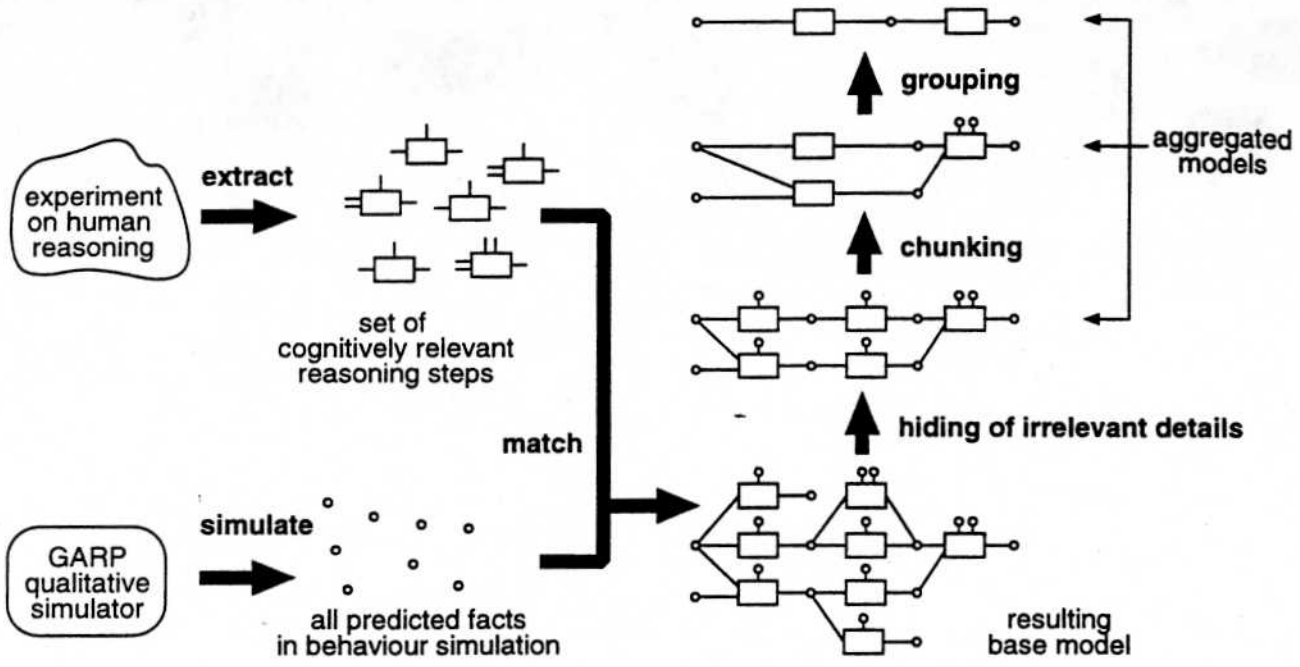


Figure 3: Building the Models

quantities, like *volume* and *amount* of the gas in the piston example: if two quantities are completely interchangeable in the model, then one of them can be left out (a definition of full correspondence is given below). At the higher level, one quantity [*volume, amount*] remains.

Chunking amounts to compiling subsequent transitive reasoning steps into one abstract inference. In natural language, this means that an inference like "the heat is increasing, therefore the temperature is increasing, and the pressure as well" can be replaced by "the heat is increasing, therefore the pressure increases as well". First, only chains of components of equal type are chunked. However, also different types can be chunked. For instance, the chain formed by the **value determination**, **quantity influence**, and **quantity proportionality** in Figure 2 can in principle be compiled to one inference component deriving $\delta T_g > 0$ from $T_s > T_g$ immediately (called a **combined influence**). Important to note is that chunking does not reflect *generic* abstractions: the resulting reasoning leap may be only applicable in that specific situation.

Grouping composes the different states and transitions between these states. Actually, grouping results in the top-level view on the state transition graph generated by the simulator. All components belonging to the specification of a state are grouped, as well as all components belonging to a transition between two states.

These three principles form the basis of the abstraction algorithm. First, hiding is applied to the base model, removing fully-corresponding quantities and 'irrelevant' branches in the reasoning paths. Then, different forms of chunking are applied, starting with equal-type chunking, and then combining pairs and triples of different types. Finally, the model is grouped into states and transitions. The aggregation algorithms, as well as the algorithm for generation of the base model from GARP's output, are implemented in SWI-Prolog (Wielemaker, 1994). A detailed example of the whole process is presented in a later section.

Roles of aggregated knowledge models in ITS

Hierarchical structuring of the base model, a reasoning network, provides shortcuts and fly-overs that have a number of general functions:

Compilation

The reasoning trajectories are shortened and can be performed with less effort. This role is the same as to what purpose reasoning trajectories are compiled out in knowledge compilation (Anderson, 1983), in chunking (Laird *et al.*, 1986), or in explanation based generalisation (EBG, Mitchell, 1982). The mechanisms (simplification and chunking) are the same as applied in these *learning* procedures, but there are two important differences:

- There is no *generalisation* over cases. Per case (problem state) the aggregation is constructed.

Of course, it is possible to use the same (EBG) mechanisms to generalise over cases, in particular when all possible cases can be generated, or all cases that are used in teaching are known beforehand.

- The aggregation covers *all possible* correct reasoning paths of a case. In EBG and knowledge compilation the chunking is only over one specific reasoning trace per case.

Hierarchical decomposition
is a means for efficient control of complex problem solving. This was also our original motivation to come to grips with the complex reasoning 'components' network. However, in our case, the network is not a structural model of a device (*e.g.*, the piston and container), but a network of (instantiated) inference steps. Therefore, the hierarchical decomposition is rather to be viewed as a problem decomposition, *i.e.* an instantiated *problem solving method* (PSM) or task structure (Breuker, 1994). In this context, CD-by-MBD means finding out which of the (set of) primitive reasoning steps has gone wrong to explain some error of the student.

Focussing and context in communication
In discourse, aggregation of 'reasoning' plays an important role too. When we understand discourse, we construct a *macro structure* (van Dijk, 1980). Macro structures are used to retrieve more specific information, or to orientate partners in a dialogue onto the more specific topics of discourse. The aggregated components of the reasoning network may perform this role when the student and the system engage in a diagnostic or explanatory dialogue. The aggregate components can serve to provide the context for diagnostic probes, or for detailed explanations to remedy the student's problem.

The two latter roles of hierarchically organized domain knowledge in an ITS support the two major functions in coaching: performance interpretation and communication (explanation, remedial) (Salles *et al.*, 1997). The first role, knowledge compilation, reflects the acquisition of a problem solving skill (Anderson, 1983).

However, the question is whether our aggregation procedures correspond to what happens to a student when becoming experienced. The inferences in the base model are based upon empirical data, but this does not guarantee cognitive plausibility of the *aggregations* constructed. This is difficult to assess. In the protocols we found many statements that reflect higher level, aggregated (correct) inference, but they look incidental rather than systematic. Indeed, one may expect that chunking is not an all or nothing process, but occurs by bits and pieces. This is not a problem because these bits and pieces can easily be recognized in performance interpretation or discourse. The question is whether the bits and pieces are the same as we have constructed by our procedures. The data in our protocols are too scarce to come to a conclusion: at least they do not contradict our automatically generated chunks. It should be noted that cognitive plausibility is more important for the communication role than for the hierarchical decomposition required for diagnosing, because this role is in the first place for computational efficiency.

An Example

To exemplify the ideas from the previous sections, recall the piston system from Figure 2. The qualitative model we built correctly predicts the different behavioural states of the system, assuming there is no friction, and no heat path between the gas in the container and the outside world.[1]

The Piston Base Model

An excerpt of the model, representing part of the initial behavioural state plus the transition to the next state (there is only one successor state from the first state), is depicted in Figure 4. The model is an extension of the part depicted in Figure 2, covering the same derivation in more detail. Even within one state, the amount of knowledge involved is considerable, and we need structuring and filtering mechanisms to guide communication with a student. This is exactly what the aggregation algorithms aim at.

The reasoning traces represented are roughly as follows. At the left side, the bold face expressions like **Pos = s** (the position of the piston is in the starting position) represent the inputs of the model, *i.e.* the information that is given to the student, and hence can be assumed known. The movement of the piston is modelled as influenced by a 'move force' F_m, which is the difference between the outward force F_o and the inward force F_i. Because no friction is modelled, these two forces are equal to the pressure of the gas and the pressure of the surrounding world, respectively. What happens in the model part shown is that, because the temperature difference between the heat source and the gas, $T_s > T_g$, there is a heat flow $Fl > 0$, which ultimately causes the outward force to increase ($\delta F_o > 0$). Furthermore, because there is no pressure difference between the inside and the outside of the container ($P_g = P_w$), the position of the piston is stable ($\delta Pos = 0$), and hence the volume of the container-piston assembly and the gas do not change ($\delta V_c = 0$ and $\delta V_g = 0$).[2]

[1] The latter assumption was originally not made in the model, but is only used here to reduce the complexity for reasons of presentation.

[2] For reasons of presentation, not all derivations for the initial state are depicted; most notably missing are derivations about quantity *values* that remain constant over the whole prediction.

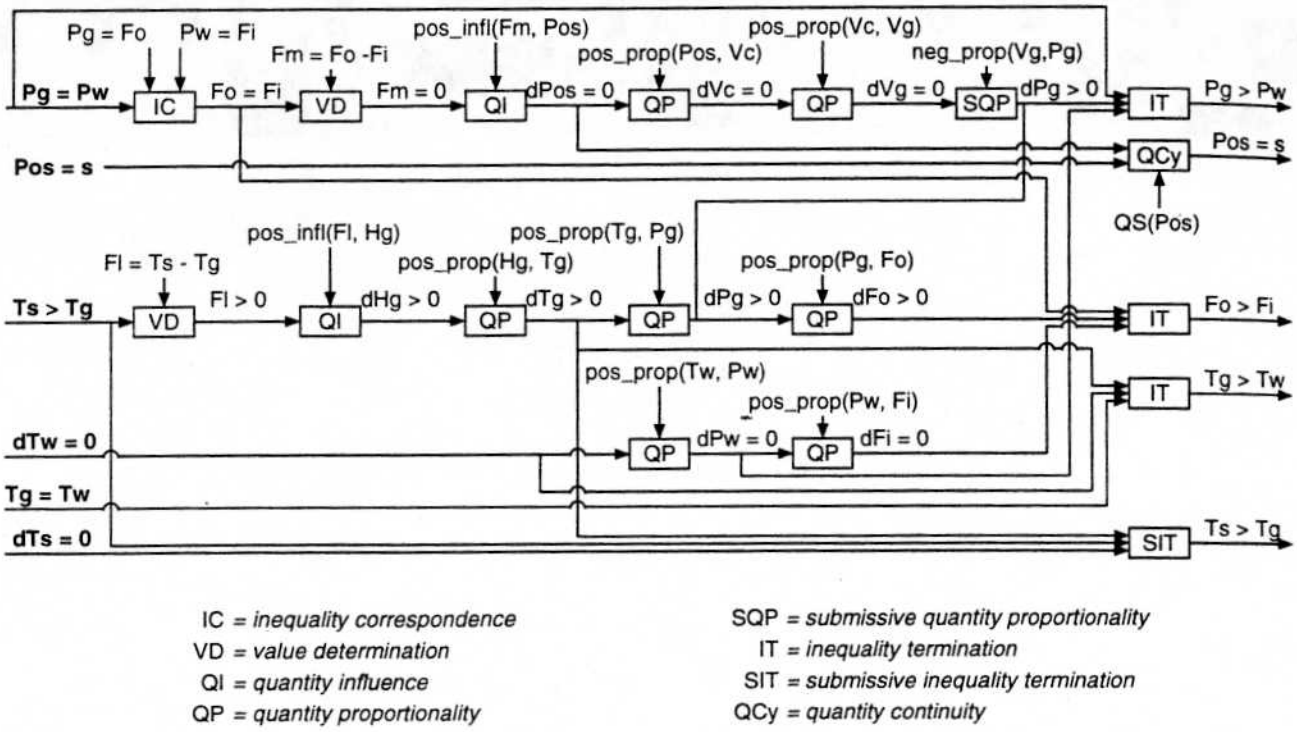


Figure 4: Base Model: First State and Transition

As described earlier, the base model was generated by mapping inferences on the set of facts derived by the simulator. That is, an inference is added if there is some (set of) input(s) A, some (set of) relation(s) $A \rightarrow B$, and some output B. This method works as long as relations $A \rightarrow B$ that hold in a state indeed produce the expected result B in that state. However, this is not always the case: for instance, in Figure 4, the termination component (labeled SIT) at the bottom does not apply. Although $T_s > T_g$, $\delta T_s = 0$, and $\delta T_g > 0$ hold, the termination is not effectuated because it is preceded by the termination $P_g = P_w \rightarrow P_g > P_w$ (*cf.* the ϵ-ordering rule (de Kleer and Brown, 1984)). The same goes for the second component from the upper right (labeled SQP): a proportionality between the volume and the pressure of the gas holds (*neg_prop*(V_g, P_g)), but both derivatives are different.

To represent such non-effective relations, we introduce *submissive* component types. A submissive inequality termination component thus represents a termination inference that does not deliver an output according to its behavioural rule, because it was overruled. In the case of competing causal relations (influences or proportionalities), the situation is more complex. There are three possible reasons for a causal relation from A to B to have no effect:

I the derivative of A is zero. In this case, any other causal effect on B will overrule this one, as was shown by the second example above;

II the effect of A is blocked by an explicit constraint. This is the case when another quantity C is also influencing B in the opposite direction, and a constraint $C > B$ holds;

III the effect is blocked by an implicit constraint. When two or more causal relations are competing, and no explicit constraints are available to resolve the ambiguity, then different behavioural states are generated that represent the different possible outcomes. In such a state, an effect can be blocked without an explicit justification within that state.

Figure 5 illustrates the base model representation of all three possibilities. In the second and third case, inference components for competitive relations have an extra knowledge input, representing the (set of) constraint(s) that was needed to resolve the conflict. When no explicit constraint is available (case III), we generate one, but also label it as being in fact implicit. This way, the teaching system has available all information needed for explaining the competitive effects. In the case of explicit constraints, an arbitrarily complex set of constraints may be applied to resolve the ambiguity. Therefore, the additional input can carry any set of constraints. We chose not to model the actual knowledge needed for applying this set of constraints to this situation—pure mathematical reasoning falls outside the scope of our model.

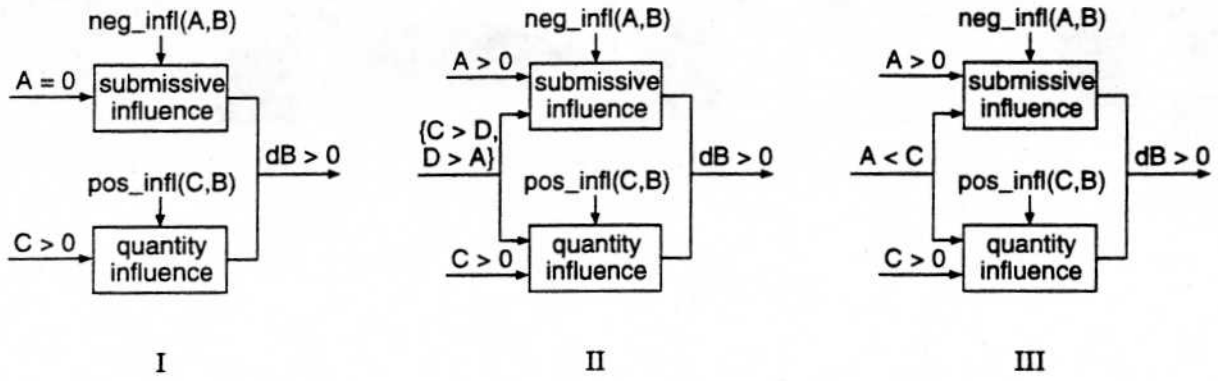


Figure 5: Three Variants on Competitive Relations

Aggregating the Base Model

Before discussing the different aggregation steps in more detail, the resulting numbers of components for the complete model after each successive step are shown in Table 1. The table shows that grouping has the biggest effect; this is because the network is reduced to one component for each behavioural state and one for each transition (in this example both eight components), no matter how big a state or transition is.

technique	nr. of comps	nr. of points
(base model)	824	802
hiding	403	517
chunking	220	408
grouping	16	9

Table 1: Numbers of Components after Different Hierarchical Techniques

Hiding of Irrelevant Details The first principle applied to the base model is that of hiding 'irrelevant' details. For hiding, the effect is particularly strong in this example because there are a lot of quantity values involved that never change value in the whole simulation.[3] *E.g.*, both the temperature and the heat of the heat source (T_s and H_s), as well as those of the surrounding world (T_w and H_w), have the value *plus* in every behavioural state. At the base level, this kind of 'staying the same' from state to state is modelled by **continuity** components, which are abstracted from by the hiding algorithm. As an example, consider the **quantity continuity** component assigned to the quantity *Pos*: because its derivative is zero, its value is the same in the next state. The same procedure is used for all *submissive* component types: both submissive termination and submissive causal relation components are abstracted from.

A rather separate part of the hiding algorithm is concerned with simplifying the model by merging quantities that are fully-corresponding. In model terms, two quantities A and B are fully-corresponding iff two opposite proportionality relations as well as an undirected correspondence relation exists between A and B. From a qualitative reasoning point of view, this definition is too strong: especially the double proportionality can be circumvented by adding additional constraints on the role of the quantities in the problem solving process (see for example Schut and Bredeweg, 1993; Falkenhainer and Forbus, 1991; Levy *et al.*, 1992; Subramanian and Genesereth, 1987). For our purposes, however, we merely want to apply data simplification as a mechanism for abstracting from what one could call *qualitative synonyms*: those quantities that are, in the context of this specific simulation (teaching situation), behaving identically.[4] We could weaken the requirements for full correspondence by adding more complex criteria. However, this would make the process of explaining this to the student too complicated and confusing for data simplification to be worthwhile (see (Schut and Bredeweg, 1993) for more details). In our example, the outward force and the pressure of the gas are modelled as fully corresponding, but also the volume of the gas and the volume of the container-piston assembly.

In Figure 6, the same model part is shown after application of the hiding algorithm. The merged force and pressure quantities are replaced by the pressure quantity for reasons of readability; however, actually a merge of A and B delivers a new quantity named

[3] At the base level, these values are indeed derived, but they are left out of Figure 4 for clarity of presentation.

[4] In fact, we would like another definition of full correspondence in between equality and the current one, which would be strong enough to derive $P_g = P_w \rightarrow F_o = F_i$ without modelling the pressure and force as being *quantitatively* equal, as is now required. However, this is not possible in current qualitative reasoning frameworks. See (Bredeweg *et al.*, 1995) for more details.

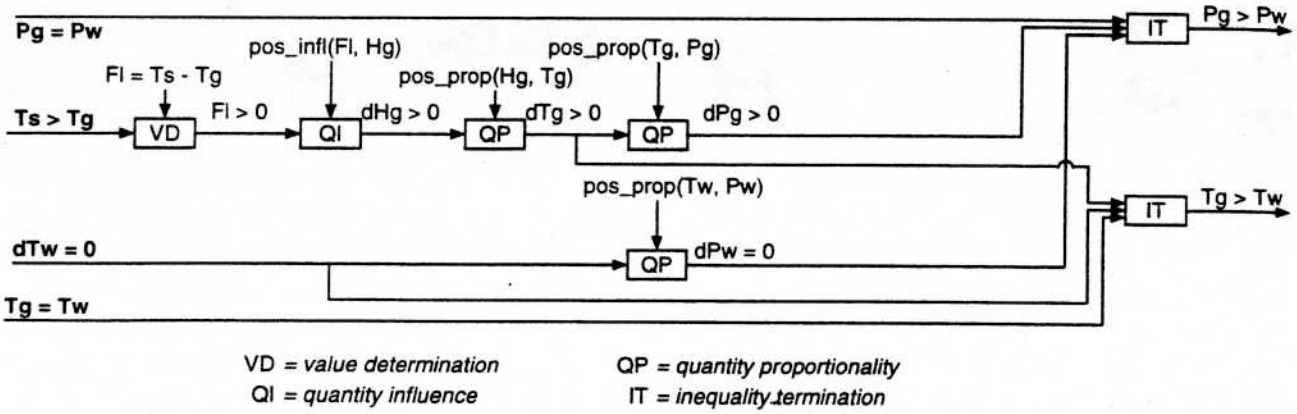


Figure 6: Same Model Part after Hiding of Irrelevant Knowledge

$[A, B]$. This way, both names can still be used—only their (similar) roles are no longer separately modelled. Note also that the proportionality between the volume and the pressure of the gas (*neg_prop*(V_g, P_g)) has disappeared, because it does not matter for the reasoning in this state: the volume does not change, hence it has no effect on the pressure. Therefore, the derivation of the volume (which is actually merged with the volume of the container-piston assembly) is removed, because as a 'dead end' it is not vital for the prediction. However, in later states, when the volume is increasing, its effect on the pressure becomes relevant, and hence the derivation of $[V_g, V_c]$ is *not* abstracted from. This is a clear example of the first difference with typical machine learning algorithms: there is no generalisation over cases (states).

Chunking The second technique applied is *chunking*. This is first done based on transitivity of proportionality and correspondence relations, resulting in combinations of components of the same type. The idea is that these kind of chunks of similar components, just as fully-corresponding quantities, are most easily abstracted from. Secondly, we combine those inferences that are considered more central to the prediction, namely computation of influences and determination of values from sums or differences, with their predecessors and successors. This last step reduces the chains within each state to a minimum. In the implementation, these two steps (first chunking similar components, and then chunking different ones) are done separately, and hence result in two separate levels of abstraction; however, Figure 7 shows the result after both steps in the chunking algorithm. The chunking algorithm also incorporates a procedure for cleaning up 'dead ends' in the model after the actual chunking. This is exemplified by the disappearance of the derivation of $T_g > T_w$: although this is a valid termination in this state, its result is not used in the next state, and

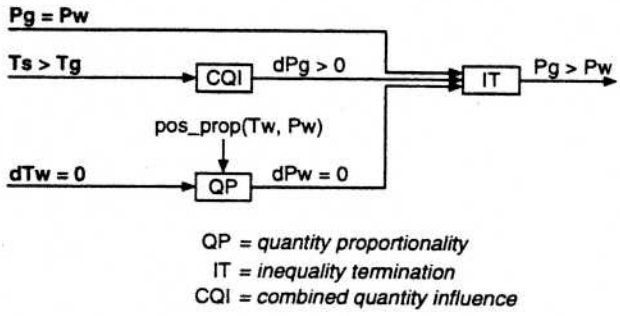


Figure 7: Same Model Part after Chunking

hence it is considered irrelevant to the main reasoning trace.

Grouping Finally, grouping selects the different states and transitions, and creates one component for each. As a result, the format of the data flow between the components changes from single expressions to *sets of expressions*, representing complete state descriptions at the previous hierarchical level. However, because we already abstracted from a large part of the quantities, here the expression sets are small, and contain only those expressions that are essential to that specific state or transition.

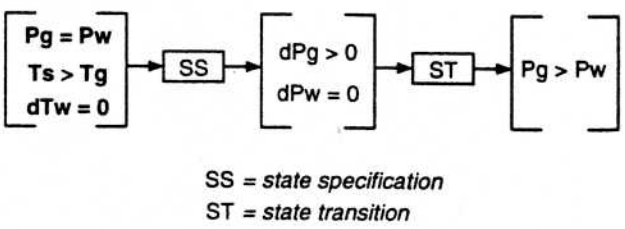


Figure 8: Same Model Part after Grouping

Employing the Models

Having built the models bottom-up, starting from the output of GARP, the important question is whether the result is useful in a teaching environment. For the mere purpose of model-based diagnosis of student behaviour, it is obvious that the complexity can be reduced drastically by focussing the diagnostic process using the higher-level components. But also for supporting the communication with the student about his or her reasoning on different grain size levels, the hierarchy of models may prove useful. As is indicated by the example model presented above, the abstractions hide less relevant information from the model, and hence delivers the central pieces of the prediction task at the higher levels.

As a first example, consider the model after grouping (Figure 8). The remaining expressions are exactly those needed to derive that the pressure difference between the gas and the outside world will change. Although other things happen, such as the changing temperature difference between the gas and outside world, this is the most important derivation because it is the trigger for the movement of the piston in the next state. In fact, this is not a coincidence: the algorithm takes into account the role an expression plays in a successor state S by looking at whether the derivation stops in a dead end in S or not. Important to note is that a central issue like *heat* is not present in the most abstract model: this is caused by the fact that the heat is not at the inputs, nor at the outputs of the model. That is, it is not a given (because these are adapted to more common sense knowledge to avoid confusing the student), nor an important 'end product' of the reasoning like the position of the piston. This way, we can use the top level model to communicate about the core behaviour of the system in common sense terms like temperatures and pressure, and hide the concept of heat. This is not to say that the concept of heat is not a key notion in understanding what is going on—only that we have the choice of abstracting from it.

As a second example, consider the model after application of the hiding algorithm (Figure 6). The 'additional' knowledge compared to the highest hierarchical level is that it shows the complete *correct* path that is needed for calculating all changes: the changing pressure ratio as well as the changing temperature ratio between the gas and the outside world. Here, we can also talk about the heat flow, the increasing heat of the gas, etcetera. 'Irrelevant' reasoning, like deriving that the inequality $T_s > T_g$ does not change at the moment, is still abstracted from.

On the basis of one example model, we can not claim that the aggregation algorithms described will always yield cognitively plausible abstractions. However, the example shows that the main principles on which the algorithm is based allows for starting with only the main aspects of the correct simulation, thereby decreasing the size and complexity of the network dramatically. If the teaching process requires so, we can gradually zoom in on the underlying details.

Related Research

Within the field of qualitative reasoning, some research has been done on summarising the results of qualitative simulations. In (Gautier and Gruber, 1996), an approach is described to collapse chains in a causal order graph of quantity relations for the purpose of simplifying explanations. This approach uses two *salience heuristics* to collapse chains. The first one collapses a chain equalities of the form $V_1 = V_2, V_2 = V_3, \ldots, V_{n-1} = V_n$ into one equality $V_1 = V_n$. Collapsing equality chains shows some resemblance to our notion of *fully-corresponding quantities*, because in both cases the idea is to dispose of sets of quantities (quantity values) that do not play distinctive roles in the simulation. In addition, chunking correspondence components is also a form of combining related quantity values, although this does not apply only to equalities: correspondence relations can also be defined between different values in different quantity space. The second heuristic is to collapse paths of indirect influences into single ones, comparable to the process of chunking proportionalities.

The major difference between our approach and the one described in (Gautier and Gruber, 1996) lies in the main application of the technique: while they focus on explanation, we focus on teaching a *reasoning process*. As such, we do not aggregate the relations in the causal order graph itself, but a model that represents the reasoning steps that can be made with these relations. This way, we can locate those reasoning steps that a student did not yet master, and not only the causal relations they do not know.

Most other approaches that are aimed at simplifying simulation outputs are at the level of states. For instance, Mallory *et al.* abstract complex behaviour graphs by reducing the number of states to be considered (Mallory *et al.*, 1996). This approach is complementary to ours, where we take the number of states and state transitions as a given, and try to abstract from less relevant details in the reasoning that is done 'within' these states and transitions.

Conclusions

In this paper we have described a two step process to construct a cognitively plausible domain representation for coaching problem solving in qualitative physics in a semi-automatic way. All procedures have been implemented in SWI-Prolog (Wielemaker, 1994) and tested. These procedures are fully generic for all types

of domains that lend themselves to qualitative modelling. For instance, we have applied the procedures to physics (like the piston domain), but also to ecological behaviour (succession in the Brazilian **cerrado** vegetation, Salles, 1997). The tools developed, a qualitative simulator *GARP* and the aggregation machinery, allow the semi-automatic generation of fully detailed and hierarchical reasoning networks that can be used as very efficient domain model(s) in an ITS. Of course, not all knowledge acquisition 'by hand' has been abandoned. The model fragments (behavioural description of the parts of a physical system) have to be specified. This is not trivial as modelling is still an important issue in QR research (Schut and Bredeweg, 1996). However, once the generic knowledge has been modelled, specifying the other inputs to the qualitative simulator, *i.e.*, the problem descriptions, is indeed trivial, and may also be automated. The generation of the base model and aggregation process are fully automatic.

The most important results from this work can be summarized as follows:

- The use of qualitative reasoning models in learning environments, including cognitively plausible reasoning steps resulted from 'reverse engineering' on the basis of data from human reasoning protocols;
- The automatic generation of hierarchical structure in the domain model (a reasoning network) by hiding less relevant knowledge, chunking and grouping, and use of this hierarchical model for communication purposes;
- The availability of reasoning traces at different levels of abstraction, *plus* the decomposition relations between these levels. This is a major advantage over abstraction of the qualitative knowledge *before* the simulation, which would result in separate models at fixed levels of abstraction.

Results from two domains (balances, pistons) are currently evaluated in a stripped version of an ITS. The ITS consists mainly of an adapted model based diagnoser, that has been described earlier (de Koning *et al.*, 1995; de Koning *et al.*, 1996). To this diagnoser, a simple communication component (discourse generator) is added to enable the diagnoser to ask questions (probes) to the student.

References

Anderson, J.R. 1983. Acquisition of proof skills in geometry. In Michalski, R.S.; Carbonell, J.G.; and Mitchell, T.M., editors 1983, *Machine Learning: An Artificial Intelligence Approach*. Morgan Kaufmann.

Bredeweg, B. and Breuker, J.A. 1993. 'Device Models' for model-based diagnosis of student behaviour. In Brna, P.; Ohlsson, S.; and Pain, H., editors 1993, *Proceedings of the World Conference on Artificial Intelligence in Education*, Charlottesville, USA. AACE. 441–448.

Bredeweg, B.; Koning, K. de; and Schut, C. 1995. Modelling the influence of non-changing quantities. In Wainer, J. and Carvalho, A., editors 1995, *Advances in Artificial Intelligence*. Springer-Verlag. 131–140.

Bredeweg, B. 1992. *Expertise in qualitative prediction of behaviour*. Ph.D. Dissertation, University of Amsterdam, Amsterdam, The Netherlands.

Breuker, J.A. 1994. Components of problem solving. In Steels, L.; Schreiber, G. Th.; and Velde, W. van de, editors 1994, *A Future for Knowledge Acquisition: Proceedings of the European Knowledge Acquisition Workshop '94*, Berlin. Springer Verlag. 118–136.

Brown, J.S.; Burton, R.R.; and Kleer, J. de 1982. Pedagogical, natural language and knowledge engineering techniques in SOPHIE I, II and III. In Sleeman, D. and Brown, J.S., editors 1982, *Intelligent Tutoring Sytems*. Academic Press, New York. 227–282.

Kleer, J. de and Brown, J.S. 1984. A qualitative physics based on confluences. *Artificial Intelligence* 24:7–83.

Koning, K. de and Bredeweg, B. 1996. Qualitative reasoning in tutoring interactions. *Journal of Interactive Learning Environments* 5. In press.

Koning, K. de; Breuker, J.A.; and Bredeweg, B. 1995. Cognitive diagnosis revisited. In Greer, J., editor 1995, *Proceedings of the World Conference on AI and Education*, Charlottesville, USA. AACE. 115–122.

Koning, K. de; Bredeweg, B.; and Breuker, J.A. 1996. Interpreting student answers: More than diagnosis alone. In Brna, P.; Paiva, A.; and Self, J., editors 1996, *Proceedings of the European Conference on Artificial Intelligence and Education*, Lissabon. Colibri. 233–239.

Falkenhainer, B.C. and Forbus, K.D. 1991. Compositional modeling: Finding the right model for the job. *Artificial Intelligence* 51:95–143.

Gautier, P.O. and Gruber, T.R. 1996. Generating explanations of device behavior using compositional modeling causal ordering. In *Proceedings of the National Conference on Artificial Intelligence*. AAAI, MIT Press. 264–270.

Laird, J.E.; Rosenbloom, P.S.; and Newell, A. 1986. Chunking in SOAR: The anatomy of a general learning mechanism. *Machine Learning* 1:11–46.

Levy, A.Y.; Iwasaki, Y.; and Motoda, H. 1992. Relevance reasoning to guide compositional modelling. In *Proceedings of the 6th International Workshop on Qualitative Reasoning about Physical Systems*, Edinburgh, Scotland. Heriot-Watt University. 7–21.

Mallory, R.S.; Porter, B.W.; and Kuipers, B.J. 1996. Comprehending complex behavior graphs through abstraction. In Iwasaki, Y. and Farquhar, A., editors 1996, *Proceedings of the Tenth International Workshop on Qualitative Reasoning*, Menlo Park, CA. AAAI Press. 137–146. AAAI Technical Report WS-96-01.

Mitchell, T.M. 1982. Generalization as search. *Artificial Intelligence* 18:203–226.

Mozetič, I. 1991. Hierarchical model-based diagnosis. *International Journal of Man-Machine Studies* 35(3):329–362.

Salles, P.; Bredeweg, B.; and Winkels, R. 1997. Deriving explanations from qualitative models. submitted to the World Conference on AI and Education.

Schut, C. and Bredeweg, B. 1993. Automatic enhancement of model parsimony. In Weld, D.S., editor 1993, *Proceedings of the 7th International Workshop on Qualitative Reasoning about Physical Systems*, Seattle Washington, U.S.A. University of Washington. 194–203.

Schut, C. and Bredeweg, B. 1996. An overview of approaches to qualitative model construction. *Knowledge Engineering Review* 11(1):1–25.

Self, J. 1992. Cognitive diagnosis for tutoring systems. In *Proceedings of the European Conference on Artificial Intelligence*, Vienna. 699–703.

Subramanian, D. and Genesereth, M.R. 1987. The relevance of irrelevance. In *Proceedings of the International Joint Conference on Artificial Intelligence*. 416–422.

Dijk, T.A. van 1980. *Macrostructures: an Interdisciplinary Study of Global Structures in Discourse, Interaction and Cognition*. Lawrence Erlbaum Associates, Hillsdale, NJ.

Wielemaker, J. 1994. *SWI-Prolog 1.9: Reference Manual*. University of Amsterdam, Social Science Informatics, Roetersstraat 15, 1018 WB Amsterdam, The Netherlands. E-mail: jan@swi.psy.uva.nl.