Model Minimization in Qualitative Reasoning

Andreas van Cranenburgh
0440949
acranenb@science.uva.nl

Hanne Nijhuis
0364568
hnijhuis@science.uva.nl

June 2, 2010

Abstract

Qualitative Process theory provides a way of modeling causality in a system on a qualitative level. Previous work has attempted to automatically induce such qualitative models from behavior. We extend this work by turning the resulting monolithic models into minimized, modular models, which makes the induced models re-usable. The resulting minimization algorithm is general enough to apply to user created models as well. We also sketch other possible improvements for model induction, based on the exploitation of human intuitions about systems rather than the current reliance on complete and error-free data.

Contents

1 Introduction

2 Literature Review

In this section we will review a few of the papers that have contributed greatly to the definition of Qualitative Reasoning. We will give a short summary of the papers and will conclude with an overview of the current field of research and possible future work.

2.1 Qualitative Reasoning

The following papers describe the foundations of qualitative reasoning.

2.1.1 A qualitative Physics based on confluences (de Kleer & Brown)

[6] present a framework for formalizing physics in a qualitative rather than quantitative manner. Whereas physicists describe the world in terms of continuous differential equations, the rest of the world uses an implicit psychological model of naive physics. The framework of [6] falls between these two extremes of precision and informality. Although their framework is completely qualitative, as opposed to standard physics, on the other hand their framework is formal in the sense of explicitly defining causal and structural relations, as opposed to the psychological models of humans (presumably). Moreover, their approach is based on deriving qualitative physics from first principles, like normal physics, as opposed to the contextual and tacit knowledge of mere mortals.

This approach is presented as being useful for physics education, expert systems and even physics, in that it explicitly deals with causality. The latter is in contrast with standard physics, in which laws are merely exceptionless correlations with predictive value.

2.1.2 Qualitative Process Theory (Forbus)

Around the same time as de Kleer & Brown, another approach was introduced by Forbus [4]. The important points that set his work appart were, among others, the *quantity-spaces* and the *sole mechanism assumption*.

The principle of quantity spaces represents values by a set of ordinal relationships. This enables the system to easily compare the different quantities in terms of smaller, equal or larger. If such comparisons would only occur between two values, then sign values would suffice (-,0,+), but for many cases, the ordinal representation is more natural.

Physical processes are viewed as the mechanisms through which change occurs. To this end Forbus introduced the *sole mechanism assumption* which states that any behavior must be explainable by the direct or indirect effects of some collection of physical processes.

Forbus has reviewed his own work and the impact of his work in [5].

2.1.3 Garp3 - Workbench for Qualitative Modelling and Simulation (Bredeweg et al.)

Garp is an implementation of Qualitative Process Theory with a focus on educational uses.

2.1.4 The Danuba river (Cioca et al.)

A Garp model of a system in the ecological domain, demonstrating its applicability to the formalization of real-world problems.

2.1.5 Discussion

The idea that Qualitative Reasoning can formalize common sense knowledge and understanding of physics is by now outdated. It is an instance of the "Good Old Fashioned Artificial Intelligence" (GOFAI) approach, and in line with the Symbol System hypothesis of Newell and Simon [1]. Both of these have either failed spectucularly or faded silently, having been replaced by stochastic and data-driven approaches. Claiming that Qualitative Reasoning can provide a way to model actual common sense knowledge or mental models is ostensively wrong – the list of mismatches with human intuitions is endless, and among these mismatches are the parts where Folk Physics is ostensively *wrong*. Therefore we submit that it is no longer useful to present it as a part of Artificial Intelligence or having anything to do with common sense; while QR models are certainly artificial, they are not in the least intelligent, nor are they common or do they make sense to non-experts. They are simply an explicit formalization of a certain set of behaviors of interest. This does not mean that Qualitative Reasoning itself is a fruitless enterprise – it can be used as an educational tool, where there is no need for bold philosophical claims.

With this educational focus in mind it makes sense to try to facilitate the modeling process for non-experts for which QR is useful. To this end we turn to the topic of model induction.

2.2 Automated Modeling

To introduce the recent developments in the field of automated modeling we will now review a few papers on this topic.

2.2.1 Automated modeling in process-based qualitative reasoning (Buisman)

The concept of automated modeling for qualitative process theory was introduced by Buisman [8] in 2007. He motivates the research with the goal to relieve the strain placed on experts and beginners, and to speed up the modelling process. The approach he took however, does not attain these goals yet, but is more of an exploration of the possibilities of using Artificial Intelligence to induce models based on behavior graphs. We will give a short overview of the algorithm he developed.

The input for the algorithm consists of:

- Behavior graph (states and state transitions) of a full envisionment
- Scenario (partial information about structural relations between entities and their quantities)
- ISA-hierarchy (full description of entity type hierarchy and quantities)

With this input, the algorithm produces models which are ready for simulation. For each scenario they should give the same output as the original model.

There are certain constraints on the input. The *full envisionment* requirement means that the behavior is known for all possible initial values for all quantities in the system. Also, the derivatives and amounts have to be defined for every state. Finally, the input data cannot contain any noise, ie., it should be perfect.

The algorithm starts with searching for so called *naive dependencies*, which are dependencies that are consistent with the entire state-graph. They are found by applying consistency rules on all pairs of quantities. These consistency rules consider the amount and derivative values of the quantities for all states, so as to induce possible dependencies. For example, a positive influence between Q_1 and Q_2 could exist if for every state $A_s(Q_1) = D_s(Q_2)$. There will be redundancy in the set of naive dependencies which has to be pruned by filtering out substitutionary groups, which are defined as mutually exclusive possible disambiguations.

2.2.2 Jochem

Actuators

2.2.3 Carsten

Causal groups

2.3 Overview

3 Theory

In the field of QR, models consist of several smaller fragments which can be roughly divided in three categories: static, process and agent. This categorization is not strict, but gives a certain view on how things work. Static fragments are used to describe the structure of a model, as well as proportionalities between quantities. A static fragment cannot have any agents or influences in it. A *process* fragment should have at least one direct influence and is not restricted in terms of dependencies. Finally, *agent* fragments should be used to describe any influences on a system that are external, hence not part of the system itself.

Fragments can be useful to make the complete system more comprehensible – ie., they are a form of chunking. Also, while building these complex models, fragments allow the user to focus on what happens in a small part of the model, without having to worry about the other 'components'. For example in a model describing a population, seperate fragments could describe processes like birth, death, emigration, etc.

Recent work on automated modeling [8, 9, 10] has focused on generating a correct model based on a full-envisionment behavior graph. The current algorithms always produce a single model fragment which describes the complete system. The implicit compositionality of a system is not revealed and comprehending the output could become quite difficult with large models. We therefore introduce a way of splitting and minimizing such monolithic models into smaller fragments while preserving the exact same behavior.

3.1 Interdisciplinary parallels

- Finite-State automaton miniminization, compression

  In automata theory there are well-known results regarding the minimization of grammars. For example there exists an algorithm to obtain a minimized version of a Finite-State automaton (FSA) which generates the exact same language with a minimum amount of states.

  Naturally this problem gets more difficult (no general purpose algorithms exist) higher up in the Chomsky hierarchy, so it would be an important result if it turned out that (a subset of) qualitative model simulation can be encoded as a Finite-State automaton that generates a regular language containing state-strings corresponding to behavior paths.

  A related but less hopeful result is that of Kolmogorov complexity, which states that given an arbitrary string, it is undecidable to verify whether a given program generating that string is the shortest one to do so (ie., highest compression), let alone to produce such a program given data.

- Minimum Description Length principle

  In Machine Learning there is the Minimum Description Length (MDL) principle, which provides an inductive bias to choose among possible hypotheses for data, given a certain representation (the latter is crucial to success). This provides a powerful heuristic because whenever

a hypothesis is found that fits the data, all larger hypotheses can be discarded.

If we apply this principle to qualitative model induction, then minimizing a model that fits the data will yield a better hypothesis according to the MDL principle.

- Economy of Representation (Generative Linguistics)

  This principle states that representations are non-redundant. The abstract representation of a surface form like "the boys walk" will only contain the feature PLURAL once, because agreement in number between the subject and verb is mandatory, and hence there is no point in storing the feature twice.

  Similarly when a scenario in a qualitative model contains two trees, both should rely on one and the same dependency stating the relation between size and shade, not on two dependencies specific to their instances.

4 Approach

In this section we describe our approach on minimizing, generalizing and splitting single model fragments into modular and compositional fragments.

4.1 Problem statement

We take a monolithic model and divide it into several smaller fragments. To do so, we need to comply to the following ranked list of constraints:[1]

1. The model should be equivalent, that is, the resulting behavior should be exactly the same as the original given the same input (a scenario).
2. The output should be as parsimonious as possible, that is, fragments should be minimal.
3. General fragments are preferred over specific fragments; this entails a bias for smaller fragments and modularity.

We make no claim as to the didactic and cognitive suitability of these constraints, but it is not hard to imagine that these constraints will provide an improvement when applied to any model lacking in modularity. It is well known that modularity is a sound engineering approach, and there is no reason to assume Qualitative Reasoning to be an exception. However, there are subtle choices to be made as to the exact granularity of fragments, because these constraints underdetermine the space of candidates, and because cognitive optimality would demand empirical studies about chunking and comprehension etc. We will gloss over these issues and apply the aforementioned constraints without further claims of optimality.

[1] Note that the terminology in this section of ranked constraints, candidates and optimality is of course a nod to Optimality Theory, FIXME cite smolensky.

4.2 Algorithm

Overview:

- Find a list of pivots, conditions on which the model fragments are based. Currently these are single structural relations.
- For each pivot, find a set of dependencies which apply whenever the pivot is present, without exception
- Generalize these dependencies into a single model fragment, several dependencies among instances may be collapsed into one

Sketch of the algorithm:

1. Input: take the set of dependencies, M, (as generated by eg. model induction or from a monolithic model fragment created by a user).
2. Partition this set into equivalence classes according to the equivalence relation of the conditions under which each dependency holds.
3. Minimize each equivalence class by substituting $Q_1 \overset{I+}{\rightarrow} Q_2$ for a set such as $\{Q_1^a \overset{I+}{\rightarrow} Q_2^b, Q_1^c \overset{I+}{\rightarrow} Q_2^d, ...\}$, yielding elements of the set MFs
4. Dump the remaining dependencies in a single fragment, UF. This set should be empty if the model is properly designed, ie., it should contain no dependencies that apply to some instances but not to others.
5. Output: minimized model = $MFs \cup \{UF\}$

This sketch can be formalized and instantiated into a definition with set builder notation.[2] First two helper definitions. The first definition defines a predicate for instances of structural relations and related quantities:

$$\begin{aligned} instance(R, Q_1, Q_2, q_1, q_2) := \exists e_1, e_2 \; [&struct\_rel(R, e_1, e_2) \\ &\wedge has(e_1, q_1) \wedge has(e_2, q_2) \\ &\wedge isa(q_1, Q_1) \wedge isa(q_2, Q_2)] \end{aligned}$$

In this definition q_n refers to an instance of a quantity Q_n, similarly e_n refers to an instance of an entity E_n; such pairs are related by the $isa(a, b)$ predicate denoting that a is an instance of b. Quantities and entities are related by $has(e_n, q_m)$ denoting that e_n has the quantity q_m. Finally $struct\_rel(R, e_n, e_m)$ denotes the configuration that e_n is structurally related to e_m with relation R; for practical purposes we assume that each entity has a reflexive relation called "self," which allows the isolation of dependencies between the quantities of a single entity.[3]

[2] Note that the following definitions closely follow our Prolog implementation, as such they include certain choices and simplifications limiting the generality of the approach as previously sketched.

[3] see the communicating vessels for an example with amount, height and pressure of a container

The next definition defines an equivalence relation on these instances, stating that their dependencies hold universally given their condition (which is here restricted to a single structural relation):

$$\begin{aligned} samedeps(R, Q_1, Q_2) := \ \forall d, q_1, q_2 \ [\ & (dependency(d, q_1, q_2) \in M \\ & \wedge isa(q_1, Q_2) \wedge isa(q_2, Q_2) \\ & \wedge has(q_1, e_1) \wedge has(q_2, e_2) \\ & \wedge isa(e_1, E_1) \wedge isa(e_2, E_2)) \\ & \quad \rightarrow \forall e_1', e_2' \ [\ (isa(e_1', E_1) \wedge isa(e_2', E_2) \\ & \quad \wedge struct\_rel(r, e_1', e_2') \\ & \quad \wedge has(e_1', q_1') \wedge has(e_2', q_2')) \\ & \quad\quad \rightarrow dependency(d, q_1', q_2') \in M\ ]\ ] \end{aligned}$$

Here $dependency(d, a, b)$ denotes a directed relation d between a and b (undirected relations can be expressed using two directed relations). Possible values for d correspond to influences, proportionalities, correspondences and equalities. While a and b usually denote quantities, note that they can also refer to an arithmetic operation between two quantities, eg. $min(q_n, q_m)$.

These two predicates combined give us the set of pivots for a given monolithic model:

$$pivots := \{\ \langle R, Q_1, Q_2 \rangle \ | instance(R, Q_1, Q_2, q_1, q_2) \wedge samedeps(R, Q_1, Q_2) \}$$

After defining the pivots we simply map them to their dependencies to obtain minimized, modular fragments:

$$\begin{aligned} fragments := \{\ \{\ & dependency(d, Q_1, Q_2)\ | \\ & dependency(d, q_1, q_2) \in M \\ & \wedge isa(q_1, Q_1) \wedge isa(q_2, Q_2)\ \} \\ & |\ \langle R, Q_1, Q_2 \rangle \in pivots \vee \langle R, Q_2, Q_1 \rangle \in pivots\ \} \end{aligned}$$

Note that while in this definiton pivots are restricted to single triples of a relation and two quantities, the definition should be generalized so that pivots can be merged into multiple triples when they are applicable under the same conditions.

5 Results

The evaluation is based on a few well known example Garp models. The output of the automatic modeling implementations of [8] and [10] is not easily accessible in a single prolog list, but rather through a poorly documented Prolog database; the current implementation [10] does not return equalities and arithmetic relations among its output. The scenario and

entity hierarchy is accessible through the Garp API, but this is also less straightforward than it should be.

To avoid having to deal with these implementation matters we have based our current proof of concept implementation on our own Prolog representation. Our implementation expects as its input a flat list of dependencies, and as background knowledge the scenario and entity hierarchy. Because of this the evaluation had to be done manually.

The implementation of the algorithm previously described has certain limitations which mean that the number of fragments is not as minimal as it could be. This boils down to:

- not considering other conditions than structural relations as pivots
- not considering multiple conditions for a single fragment
- not considering merging the dependencies of multiple quantities belonging to the same entity in a single fragment

However, this seems to be largely an implemantion matter. The implementation provides a proof of concept of the algorithm, more sophisticated implementations can easily achieve more results.

5.1 Tree and shade growth

Input in figure 1 and output in figure 2.

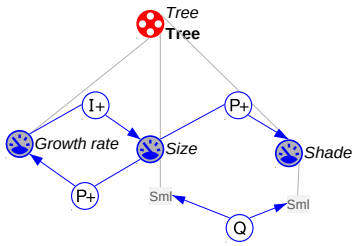

Figure 1: Monolithic model as input

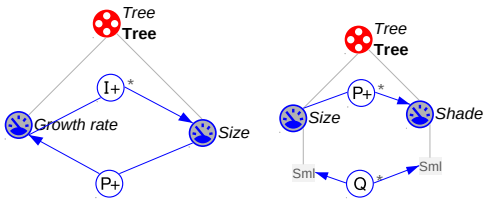

Figure 2: After minimization: two model fragments for growth and size of shade

5.2 Stacked bath tubs

For the stacked bath tub model we changed the quantity space of flow to ZPM (instead of ZP) because then we could replace the value-correspondences by a single Q-correspondence.

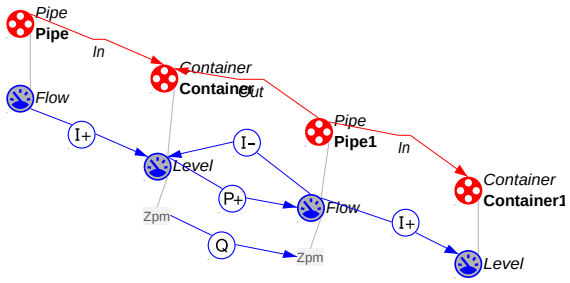

Figure 3: Monolithic bath tub model as input

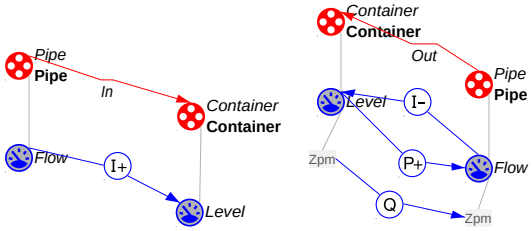

Figure 4: After minimization: two model fragments for flows going in and out respectively.

5.3 Communicating vessels

Inducing the communicating vessels model almost works, but the last two equalities are not found, so they have been added manually. Also, the order of arguments for proportionalities has been tweaked.

Two communicating vessels:

Three communicating vessels.

```
Input:
[dependency(inf_pos, flow3, amount5),
dependency(inf_neg, flow3, amount4),
dependency(prop_neg, flow3, pressure5),
dependency(prop_pos, flow3, pressure4),
dependency(prop_pos, height4, amount4),
dependency(prop_pos, height5, amount5),
dependency(prop_pos, pressure5, height5),
dependency(prop_pos, pressure4, height4),
dependency(q_correspondence, height5, amount5),
```

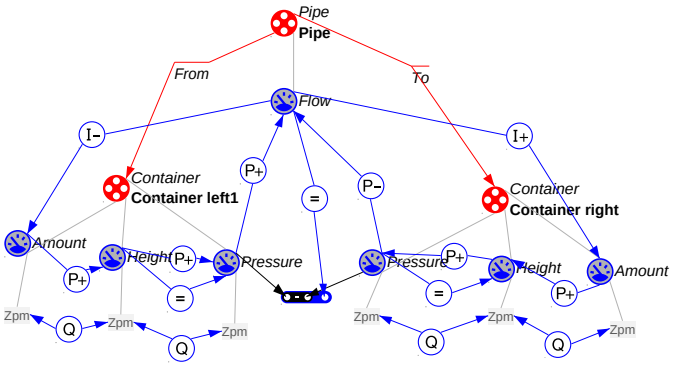

Figure 5: Monolithic model for two communicating vessels as input

```
dependency(q_correspondence, pressure4, height4),
dependency(q_correspondence, pressure5, height5),
dependency(q_correspondence, height4, amount4),
dependency(equals, flow3, min(pressure4, pressure5)),
dependency(equals, height4, pressure4),
dependency(equals, height5, pressure5),
dependency(inf_pos, flow4, amount6),
dependency(inf_neg, flow4, amount5),
dependency(prop_neg, flow4, pressure6),
dependency(prop_pos, flow4, pressure5),
dependency(prop_pos, height6, amount6),
dependency(prop_pos, pressure6, height6),
dependency(q_correspondence, height6, amount6),
dependency(q_correspondence, pressure6, height6),
dependency(equals, flow4, min(pressure5, pressure6)),
dependency(equals, height6, pressure6)]

Set of struct rels:
[ (self, flow, flow), (self, amount, amount), (self, amount, height),
(self, amount, pressure), (self, height, height), (self, height, pressure),
(self, pressure, pressure), (from, amount, flow), (from, height, flow),
(from, pressure, flow), (to, flow, amount), (to, flow, height),
(to, flow, pressure)]

Fragments (6):
[ (self, amount, height),
dependency(prop_pos, height, amount),
dependency(q_correspondence, height, amount)]
[ (self, height, pressure),
dependency(prop_pos, pressure, height),
dependency(q_correspondence, pressure, height),
dependency(equals, height, pressure)]
[ (from, amount, flow), dependency(inf_neg, flow, amount)]
```

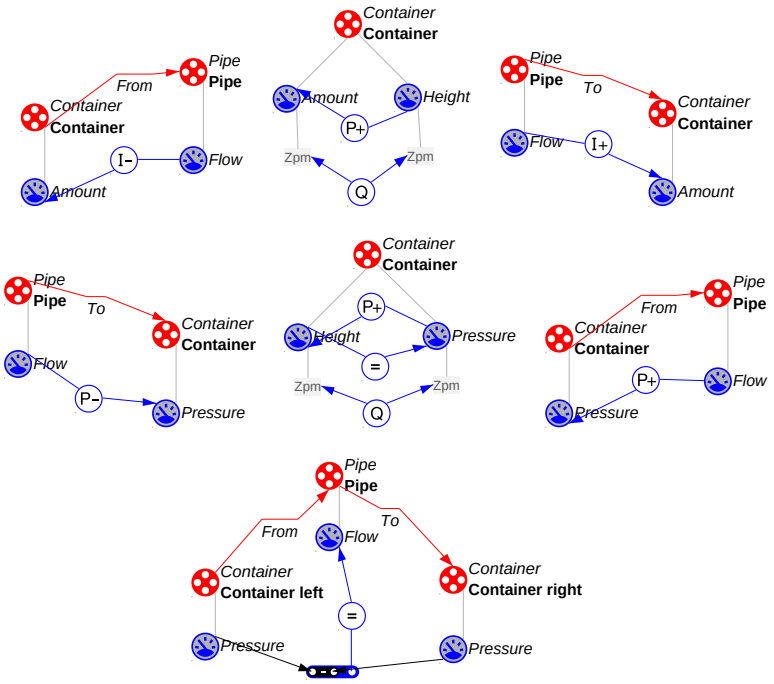

Figure 6: After minimization: seven model fragments for two communicating vessels.

```
[ (from, pressure, flow), dependency(prop_pos, flow, pressure)]
[ (to, flow, amount), dependency(inf_pos, flow, amount)]
[ (to, flow, pressure), dependency(prop_neg, flow, pressure)]

Unfragment:
[dependency(equals, flow3, min(pressure4, pressure5)),
dependency(equals, flow4, min(pressure5, pressure6))]
```

5.4 Population dynamics

TBD.

6 Discussion

We will now discuss several possible extensions both to model induction in general and to model minimization in particular.

6.1 Model minimization

6.1.1 Modular fragments as input

Currently the algorithm expects a single monolithic model as input, since that is the output as given by the automated modeling algorithm. However, this requirement could be lifted to allow partially modular fragments as input, which is especially useful when the minimization algorithm is applied as part of an interactive modeling assistant.

This could be implemented by finding a way to merge these fragments into a monolithic fragment, while avoiding undergeneration by exploding all dependencies to specific dependencies between instances. Such an extension is fairly trivial but has not been implemented yet.

6.1.2 Inheritance: hierarchy of fragments in output

Currently the output consists of a flat list of model fragments, but a more advanced method would turn this into an inheritance hierarchy, to avoid duplication

6.2 Model induction

6.2.1 Conditions

The induction algorithm currently has no support for conditions, except for copying structural relations between entities as found in the scenario. This severly limits the scope of models that can be induced. However, adding the induction of conditions increases the complexity greatly. This is demonstrated by an analogy to the Chomsky Hierarchy: with every addition of context-sensitivity, the complexity increases, until it reaches Turing equivalence in the unrestricted case. The Garp models that can currently be induced are the subset of possible models such that only structural relations can serve as conditions.

On top of that, it only makes sense to add conditions to fragments, which implies that splitting into fragments (but not minimizing) would need to be integrated into the model induction algorithm itself instead of as a transformation as has been presented in the current project.

6.2.2 Interactivity

The current requirement for the induction to have access to a perfect behavior graph makes the algorithm virtually useless for practical purposes, because such a behavior graph is very difficult to specify without already having a qualitative model, which makes for a vicious circle. An alternative would be to make the induction interactive. The algorithm would formulate a series of maximally informative questions (comparable to automatic diagnosis [11]), employing the user as an informant. A series of iterations can be performed, presenting the resulting behavior graph to the user who can then highlight errors.

6.2.3 Negative exemplars

This brings us to the related problem of negative exemplars. Humans probably have an idea not only of what a system does, but also what it does not do (constraints). Negative exemplars are of course the very reason for the problem of induction in philosophy (Hume, Popper, etc.). However, since qualitative models are a formalization of human knowledge, not induced from naturalistic data, negative exemplars are a resource that should be exploited. Negative exemplars could be presented by the user in the form of (in)equalities and as a list of impossible combinations of values.

7 Conclusion

Model minimization has been fruitfully applied to qualitative models. Our approach is general enough that it applies both to Automatic Modeling and to models made by beginners. From now on, minimizing qualitative models in Garp can be done automatically, at least for the subset of models supported by the proof of concept implementation. Further work on inducing models lies ahead and given the results on inducing constraint models this work would seem to have ample room for success.

References

[1] Allen Newell and Herbert A. Simon, "Computer Science as Empirical Inquiry: Symbols and Search," Communications of the ACM. vol. 19, No. 3, pp. 113-126, March, 1976. http://www.rci.rutgers.edu/~cfs/472html/AISEARCH/PSS/PSSH1.html

[2] Bredeweg, B. and Salles, P. (2009), Handbook of Ecological Modelling, Chapter 19 - Mediating conceptual knowledge using qualitative reasoning. In: Jorgen, S.V., Chon, T-S., Recknagel, F.A. (Eds.), Handbook of Ecological Modelling and Informatics. Wit Press, Southampton, UK, pp. 351.398.

[3] Bredeweg, B., Linnebank, F., Bouwer, A. and Liem, J. (2009) 02 Bredeweg EtAl ECOINF 2009.pdf (598.325 Kb) Garp3 - Workbench for Qualitative Modelling and Simulation. Ecological Informatics 4(5-6), 263-281.

[4] Forbus, K.D. (1984) Qualitative process theory. Artificial Intelligence, 24:85-168.

[5] Forbus, K.D. (1993) Qualitative process theory: twelve years after. Artificial Intelligence, 59:115-123.

[6] Kleer, J. de and Brown J.S. (1984) deKleerBrown1984.pdf (4.005 Mb) A qualitative Physics based on confluences, Artificial Intelligence, 24:7-83

[7] Cioaca, Linnebank, Bredeweg, Salles 2009 Cioaca EtAlECOINF 2009.pdf (1,001.442 Kb) A qualitative reasoning model of algal bloom in the Danube Delta Biosphere Resere (DDBR) in: *ecological informatics 4* (2009) p282-298

[8] Buisman, H., "Automated modeling in process-based qualitative reasoning" http://staff.science.uva.nl/~bredeweg/pdf/BSc/20062007/Buisman.pdf

[9] Liem, J., Buisman, H. and Bredeweg, B., "Supporting Conceptual Knowledge Capture Through Automatic Modelling"

[10] van Weelden, C., "Automated modeling of conceptual knowledge" http://staff.science.uva.nl/~bredeweg/pdf/BSc/20082009/vanWeelden.pdf

[11] De Kleer, J. and Williams, B. C. 1989. Diagnosis with behavioral modes. In *Proceedings of the 11th international Joint Conference on Artificial intelligence - Volume 2* (Detroit, Michigan, August 20 - 25, 1989). International Joint Conference On Artificial Intelligence. Morgan Kaufmann Publishers, San Francisco, CA, 1324-1330.